ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE CIENCIAS

TÉCNICAS DE MACHINE LEARNING APLICADAS EN LA IMPLEMENTACIÓN DE UN MODELO DE CREDIT SCORING DE TIPO ORIGINACIÓN EN UNA ENTIDAD BANCARIA DEL ECUADOR

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO MATEMÁTICO

PROYECTO DE INVESTIGACIÓN

LUIS EDUARDO OÑA GUALLICHICO
lsdrdhs@hotmail.com

Director: MIGUEL ALFONSO FLORES, PHD
miguel.flores@epn.edu.ec

QUITO, ENERO 2022

DECLARACIÓN

Yo LUIS EDUARDO OÑA GUALLICHICO, declaro bajo juramento que el trabajo aquí escrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual, correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

Luis Eduardo Oña Guallichico

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por LUIS EDUARDO OÑA GUALLICHICO, bajo mi supervisión.

_____

Miguel Alfonso Flores, PHD
Director del Proyecto

AGRADECIMIENTOS

A Miguel Flores por haberme ayudado a plasmar y concluir este tema de investigación, apoyarme con su paciencia, motivación y conocimiento.

A mis amigos Alexander, Cristian, Jhosselyne y Miguel por su compañia y ayuda a lo largo de la carrera.

DEDICATORIA

A mis padres y hermanos por estar pendientes en todo momento de las cosas que me hacían falta para poder finalizar mis estudios.

A Leonor por ser una parte importante de mi vida además de ser la inspiración para poder concluir este proyecto.

Índice general

Resumen xv

Abstract xvi

1. Introducción 1

1.1. Metodologías analíticas para el Credit Score 5

1.1.1. Aprendizaje Estadístico 6

1.1.2. Aprendizaje Automático 7

1.1.3. Interpretabilidad de Modelos de Aprendizaje Automático . . 8

1.2. Objetivos 10

1.2.1. Objetivo General 10

1.2.2. Objetivos Específicos 10

2. Marco Teórico 12

2.1. Definiciones y conceptos básicos del sistema financiero 13

2.1.1. ¿Qué es y para qué sirve un banco? 13

2.1.2. Qué es un crédito 14

2.1.3. Riesgos Financieros 15

2.1.4. Administración del Riesgo 20

2.1.5. Comité de Basilea 21

2.1.6. Modelos Credit Scoring 23

2.2. Problemas de Clasificación 25

2.2.1. Árboles de decisión 26

2.3. Conjuntos Clasificadores 32

2.3.1. Bagging . . . 33
2.3.2. Boosting . . . 35
2.3.3. Árboles Boosting . . . 39
2.3.4. Optimización numérica a través del Gradient Boosting . . . 42
2.3.5. Implementación por Gradient Boosting . . . 45
2.3.6. Importancia de variables . . . 46
2.3.7. Algoritmo XGBoost . . . 47
2.4. Regresión logística con estimación a través del método de Firth . . . 54
2.4.1. Regresión logística Múltiple (RLM) . . . 54
2.4.2. Estimación de los parámetros . . . 55
2.4.3. Método de Firth . . . 57
2.5. Evaluación y selección de modelos . . . 58
2.5.1. Sesgo, varianza y complejidad del modelo . . . 58
2.5.2. Optimismo de la tasa de error de entrenamiento . . . 63
2.5.3. Estimación del error de predicción en la muestra . . . 65
2.5.4. Validación cruzada . . . 68
2.6. Evaluación y desempeño de modelos de clasificación . . . 77
2.6.1. Matriz de confusión . . . 77
2.6.2. AUC y Curva ROC . . . 78
2.6.3. Estadístico de Kolmogorov - Smirnov (KS) . . . 80
2.6.4. Coeficiente Gini . . . 81
2.6.5. Tabla de ODDS . . . 82
2.7. Interpretable Machine Learning Models . . . 83
2.7.1. LIME . . . 85

3. Metodología Analítica y Resultados **89**
3.1. Procesamiento de la información . . . 90
3.1.1. Descripción de la Base de Datos . . . 90
3.1.2. Selección de la ventana de muestreo . . . 91
3.1.3. Definición de la variable dependiente . . . 93

3.1.4. Análisis Descriptivo de la Base de Datos 95
3.1.5. Análisis Bivariado 99
3.1.6. Preprocesado de datos 102
3.1.7. Muestra de entrenamiento y prueba 105
3.2. Construcción del modelo predictivo XGBoost 106
3.2.1. Selección de variables 106
3.2.2. El problema de muestras desproporcionadas 108
3.2.3. Implementación del Algoritmo 109
3.2.4. Resultados del Algoritmo 111
3.3. Evaluación estadística del modelo Xgboost 115
3.3.1. Estadísticos de rendimiento 115
3.3.2. Tabla de ODDS 117
3.4. Construcción del Modelo LIME 120

4. Comparación del modelo XGBoost con técnicas tradicionales(precisión e interpretabilidad) 122

4.1. Modelo de regresión logística con estimación a través del método de Firth 123
4.1.1. Estimación del modelo 123
4.1.2. Evaluación del modelo 127
4.2. XGBoost vs regresión logística con estimación a través del método de Firth 129
4.3. Comparación en precisión 129
4.4. Comparación de variables usadas en el modelo 130

5. Implementación del modelo en la entidad bancaria 133

5.1. Forward Testing 134
5.1.1. Evaluación estadística 135
5.1.2. Estabilidad de la población 135
5.2. Construcción de Perfiles de Riesgo 137
5.2.1. Determinación del Punto de Corte 138

5.2.2. Cálculo del Score 140
5.2.3. Construcción de Perfiles utilizando la cartera vencida como indicador de corte 141
5.3. Esquema de Pilotos 143

6. Conclusiones y Recomendaciones **145**
6.1. Conclusiones 145
6.2. Recomendaciones 147

A. ANEXO 1: Definiciones y algoritmos **150**
A.1. Función de pérdida de Huber 150
A.2. Función de pérdida de desviación 150
A.3. Función Sofmax 151
A.4. Clasificador de Bayes 151
A.5. Algoritmo Impulso progresivo por etapas (Forward stagewise boosting) 151

B. ANEXO 2: Código para la creación del modelo **153**
B.1. Funciones para EDA 153
B.1.1. Variables Numéricas 153
B.1.2. Variables Categóricas 154
B.2. Funciones para Análisis Bivariado 155
B.2.1. Variables Numéricas vs Dependiente 155
B.2.2. Variables Categóricas vs Dependiente 156
B.3. Búsqueda de hiperparámetros 156
B.4. Búsqueda de hiperparámetros 157
B.5. Estimación del modelo XGBoost 159
B.5.1. Estimación del modelo 159
B.5.2. Iteraciones del modelo 159
B.5.3. Métricas de evaluación del modelo 160
B.5.4. Importancia de variables 161
B.6. Código LIME 162

B.7. Implementación algoritmo Regresión logística con el método de Firh 163

C. ANEXO 3: Resultados del modelo **165**

C.1. Estadístico KS para variables numéricas 165

C.2. Importancia de Variables 166

Bibliografía **171**

Índice de figuras

1.1. Tipos de modelos de Credit Score en función del tipo de información disponible. 3
1.2. Esquema de generación de información 8
1.3. Panorama general del aprendizaje automático explicable 9

2.1. Ejemplo de clasificación binaria (Operador AND) 26
2.2. Medidas de impurezas de los nodos para la clasificación de dos categorías 31
2.3. Comportamiento del error de las muestras de prueba y entrenamiento a medida que varía la complejidad del modelo. 59
2.4. AIC utilizado para la selección de modelos para el reconocimiento de fonemas. 67
2.5. Curva de aprendizaje hipotética para un clasificador en una tarea determinada: un gráfico de $1 - Err$ frente al tamaño del conjunto de entrenamiento N. 69
2.6. Error de predicción esperado (naranja), sesgo al cuadrado (verde) y varianza (azul) para un ejemplo simulado. La fila superior es una regresión con pérdida de error al cuadrado; la fila inferior es una clasificación con pérdida $0-1$. Los modelos son los vecinos más cercanos (izquierda) y el mejor subconjunto de regresión de tamaño p (derecha). Las curvas de varianza y sesgo son las mismas en la regresión y la clasificación, pero la curva de error de predicción es diferente. . . . 72
2.7. Error de predicción (naranja) y curva de validación cruzada de diez veces (azul) estimada a partir de un único conjunto de entrenamiento, del escenario del panel inferior derecho de la Figura 2.6 73

2.8. Validación cruzada a la manera incorrecta y correcta: los histogramas muestran la correlación de las etiquetas de clase, en 10 muestras elegidas al azar, con los 100 predictores elegidos utilizando las versiones incorrecta (rojo superior) y correcta (verde inferior) de la validación cruzada. 74
2.9. 76
2.10. Matriz de confusión (caso binario) 77
2.11. Caso1: Curva ROC 79
2.12. Caso2: Curva ROC 79
2.13. Caso3: Curva ROC 80
2.14. Ejemplo Índice de Gini 82
2.15. Tabla de Odds, aplicado a un modelo de fraude 83
2.16. Ejemplo básico para presentar la intuición de LIME. 87

3.1. Esquema de generación de información 91
3.2. Análisis Roll Rate 94
3.3. Tasa de malos vs FechaAnálisis 95
3.4. Entrenamiento Modelo XGBoost 112
3.5. Importancia de variables 114
3.6. Matriz de confusión. XGBoost 116
3.7. Odds del Modelo (Entrenamiento) 118
3.8. Odds del Modelo (Prueba) 120
3.9. Resultado modelo LIME para 1 caso aleatorio1 121

4.1. Odds del Regresión Logística (Prueba) 129

5.1. Componente de una solución de ML 133
5.2. Estabilidad de las predicciones 137
5.3. Definición del Cutoff con la curva ROC 140
5.4. Resultado Pilotos 144

Índice de cuadros

1.1. Tipo de información por tipo de modelo. 3

2.1. Gradientes para funciones de pérdida usadas comúnmente 45

3.1. Distribución del número de registros por fecha de análisis 92
3.2. Descripción de las variables en la base de datos 96
3.3. Análisis exploratorio de variables numéricas 97
3.4. Análisis exploratorio de variables categóricas 98
3.5. Estadístico KS para las variables numéricas 100
3.6. Reglas para el Information Value. 101
3.7. Information value para las variables categóricas 101
3.8. Comparativo variable 'CodigoProfesion' antes y después de la recategorización 104
3.9. Partición del conjunto de datos en entrenamiento y prueba 105
3.10. Hiperparámetros óptimos 110
3.11. Medidas de discriminación. XGBoost 115
3.12. Tabla de Odds (Entrenamiento) 117
3.13. Tabla de Odds (Prueba) 119

4.1. Modelo Regresión Logística Firth 124
4.2. VIF Primer modelo Regresión Logíistica 125
4.3. Modelo Final Regresión Logística Firth 126
4.4. VIF Modelo final Regresión Logística 127
4.5. Medidas de discriminación. Regresión Firth 127
4.6. Tabla de Odds Modelo Regresión Logística (Prueba) 128

4.7. Tabla de efectos marginales (Regresión Logística) 131

5.1. Distribución del número de registros por fecha de análisis 136
5.2. Perfiles por Ratio de Cartera Vencida 142

B.1. Hiperparámetros iniciales de búsqueda 156

C.1. Estadístico KS para las variables numéricas 165
C.2. Importancia de variables 166

Resumen

El presente estudio tiene como finalidad detallar la construcción, validación e implementación de un modelo de *Credit Scoring* en una entidad bancaria del Ecuador, a través del cual se estime la probabilidad de que un cliente no cumpla con sus obligaciones crediticias. El modelo está dirigido para el segmento de clientes que inician sus actividades financieras (no tienen historial crediticio), y que en muchas de las instituciones financieras todavía es calificado mediante el uso de técnicas tradicionales por no poseer la información suficiente. Bajo estas características el modelo es denominado: 'Modelo de Originación'.

Para la elaboración del modelo se contempla el uso de una metodología moderna basada en algoritmos de aprendizaje automático, tales como: XGBoost y LIME, cuyos resultados serán evaluados y validados con el fin de obtener un modelo óptimo capaz de trabajar correctamente sobre nuevos conjuntos de datos. Esto determinará la importancia de utilizar técnicas actuales que sean objetivas y capaces optimizar el nivel de riesgo al cual se encuentran expuestos los recursos de la institución bancaria.

La información empleada es proporcionada por una entidad bancaria del Ecuador. Además, la metodología utilizada para la construcción del modelo será implementada en el software estadístico R, permitiendo así obtener los resultados de manera automática y ahorrar una gran cantidad de tiempo de procesamiento de datos.

Palabras clave: Aprendizaje Automático, XGBoost, hiperparámetros, LIME, interpretabilidad, remuestreo, programación en R, clientes sin antecedentes crediticios, probabilidad de default (PD), Credit Scoring.

Abstract

The purpose of this study is to detail the construction, validation and implementation of a Credit Scoring model in a banking entity in Ecuador, through which the probability that a client will not meet their credit obligations is estimated. The model is aimed at the segment of clients who start their financial activities (they do not have a credit history), and that in many of the financial institutions is still qualified by using traditional techniques because they do not have enough information. Under these characteristics the model is called: 'Application Model'.

For the elaboration of the model, the use of a modern methodology based on machine learning algorithms is contemplated, such as: XGBoost and LIME, whose results will be evaluated and validated in order to obtain an optimal model capable of working correctly on new data sets. . This will determine the importance of using current techniques that are objective and capable of optimizing the level of risk to which the banking institution's resources are exposed.

The information used is provided by a bank in Ecuador. In addition, the methodology used for the construction of the model will be implemented in the R statistical software, thus allowing the results to be obtained automatically and saving a large amount of data processing time.

Keywords: Machine Learning, XGBoost, hyperparameter, LIME, interpretability, resampling, programming in R, clients with no credit history, probability of default.

Capítulo 1

Introducción

El sector financiero dentro de la economía de cualquier país juega un papel muy importante ya que es el encargado de captar el excedente de los ahorradores y canalizarlo hacia los prestatarios públicos o privados, pero al tener dicho rol está expuesto a grandes inestabilidades que según Restrepo M y Restrepo D (2009), de no existir una administración adecuada del riesgo se podrían generar problemas que acarreen altos índices de cartera vencida [1] que pueden desencadenar finalmente en una crisis financiera de grandes proporciones.

Al estar expuestos a estas inestabilidades, los bancos suelen asumir ciertos riesgos, los cuales deben ser tratados y monitoreados de la mejor manera con el fin de poder minimizarlos. De la gran variedad de riesgos que existen, a continuación se enumeran los más importantes en el sector bancario:

- Riesgo de Crédito
- Riesgo de Mercado
- Riesgo de Liquidez
- Riesgo Operativo

De todos estos riesgos, en el Ecuador el Riesgo de Crédito es el que más ha sido tratado y normado por la Superintendencia de Bancos, ya que para evitar que crisis económicas globales afecten la economía del país, tuvieron que establecerse normativas basados en los acuerdos de BASILEA. Se define el Riesgo de Crédito como: "La posibilidad de pérdida debido al incumplimiento del prestatario o la contraparte en operaciones directas, indirectas o de derivados que conlleva el no pago, el pago parcial o la falta de oportunidad en el pago de las obligaciones pactadas" (Su-

[1] Son las cuentas por cobrar que tiene una institución bancaria cuya fecha de pago ya venció y no se han cobrado.

perintendencia de Bancos Ecuador, 2019, p.1).

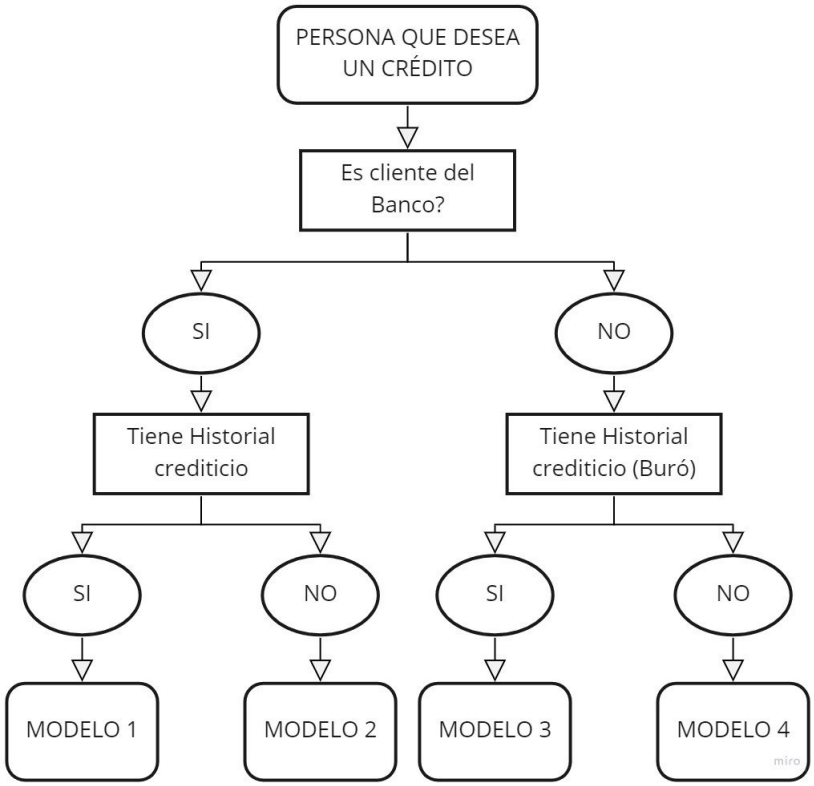
Las instituciones financieras, para administrar de manera adecuada el Riesgo de Crédito, han utilizado técnicas tradicionales como las 5C, las cuales se basaban en puntos como: Carácter, Capacidad de Pago, Colateral, Capital y Condiciones. Sin embargo, dado el gran incremento en los volúmenes de solicitudes de créditos, las instituciones se vieron en la necesidad de buscar otros mecanismos que ayuden a la toma de decisiones con respecto a la administración del riesgo de crédito. A partir de esto nace el *Credit Score* que se define como: "métodos estadísticos utilizados para clasificar a los solicitantes de crédito entre las clases de riesgo bueno o malo" (Hand y Henley, 1997, p. 535). Su uso comenzó a mediados de los 70's pero su auge fue a partir de 90's, esto debido a los avances computacionales y estadísticos, permitiendo que las instituciones financieras desarrollen sus metodologías de créditos a través de la probabilidad de incumplimiento [2] (PD, por sus siglas en inglés) de sus clientes, definida como: "la posibilidad de que ocurra el incumplimiento parcial o total de una obligación de pago o el rompimiento de un acuerdo del contrato de crédito, en un período determinado" (Superintendencia de Bancos Ecuador, 2019, p.1). El *Credit Score* ha sido avalado tanto por el Comité de Basilea como la Superintendencia de Bancos del Ecuador a inicios de siglo.

En la actualidad, gran parte de las instituciones financieras ecuatorianas han comenzado a dejar las técnicas tradicionales de calificación de crédito dando el paso hacia el *Credit Score*. Este paso juega un rol muy importante dentro de las entidades financieras, a tal punto que ha sido motivo de estudio en varias tesis de pregrado y postgrado para diferentes instituciones. De estas investigaciones se conoce que las variables mas importantes en un modelo de *Credit Score* vienen de información interna de la entidad financiera (comportamiento de pago en operaciones activas que posee la persona) y externa (buró de crédito), las cuales permiten crear modelos estadísticos robustos y con un alto rendimiento.

En la figura 1.1 podemos distinguir los distintos modelos de *credit scoring* que se pueden crear en las entidades financieras, esta clasificación está hecha por el tipo de información que se posea sobre el cliente. Es importante hacer esta distinción ya que dependiendo de la completitud de información con la que se cuente, se puede clasificar de mejor manera a un cliente con un modelo que con otro. En la figura 1.1 se

[2] Se utiliza con más frecuencia el término *Probabilidad de default*

Figura 1.1: Tipos de modelos de Credit Score en función del tipo de información disponible.



Fuente: Elaboración Propia

listan los principales modelos clasificados por el tipo de información que van a tener como entrada, esto no significa que sea la única clasificación que se puede realizar (es una de las más sencilla y completa) ni que no se pueda hacer mas distinciones basados en alguna otra característica importante como el tipo de cliente, el producto, etc. Como resumen en la tabla 1.1 se presenta el detalle del tipo de variables con las que cuentan cada uno de los modelos detallados en la figura 1.1

Cuadro 1.1: Tipo de información por tipo de modelo.

Modelo	Sociodemográficos	Pasivos	Activos	Buró
Modelo 1	SI	SI	SI	SI
Modelo 2	SI	SI	NO	NO
Modelo 3	SI	NO	NO	SI
Modelo 4	SI	NO	NO	NO

Fuente: Elaboración Propia

Los modelos tipo 1 son los que generalmente poseen un mejor performance ya que utilizan la información de todas las fuentes disponibles para predecir el com-

portamiento de pago de una persona, estos son muy importantes ya que califican aproximadamente al 60 % de los clientes de una entidad financiera, por otro lado clasificar a una persona que cae dentro de los modelos tipo 4 es más difícil, dado que estamos intentando obtener una predicción correcta del comportamiento de pago de un cliente a partir de variables netamente sociodemográficas; por lo general se usa este tipo de modelos para la calificación de clientes que entran a formar parte de la población económicamente activa y buscan un producto crediticio sin antes haber contado con una cuenta de ahorros.

En el caso ecuatoriano, la mayoría de los modelos de *Credit Score* se encuentran focalizados en la concesión del crédito, es decir, para clientes de las instituciones financieras que ya cuentan con un historial crediticio (modelo 1 de la figura 1.1). Sin embargo, existe un segmento de clientes que dentro de las instituciones financieras que a pesar de ya manejar cuentas de ahorro están intentando acceder a su primer crédito (modelo 2 de la figura 1.1), pero que, al no poseer un historial crediticio la mayoría de los bancos no está en la capacidad de determinar una *probabilidad de default*, dado que sus modelos de *Credit Scoring* no están dirigidos para este segmento de clientes, y por ende la responsabilidad del otorgamiento de crédito es designada a los analistas de las instituciones, conllevando a una serie riesgos, tales como:

- La decisión final puede cambiar dependiendo del analista, por lo general aquellos asesores con menor experiencia tienden a equivocarse más.
- El tiempo de respuesta es muy variable, produciendo una insatisfacción en los clientes; por ejemplo, si una solicitud tarda semanas en tener una respuesta, el cliente al final puede ya no estar interesado en recibir el crédito.
- Si no se logra determinar correctamente el perfil de pago de un cliente se pueden cometer los siguientes errores:
    - Otorgar un crédito a una persona que no va a pagar, generando pérdidas económicas para las instituciones financieras.
    - Negar un crédito a una persona que si va a pagar, produciendo la pérdida de una operación rentable y que de incrementarse en número podría generar una caída en la participación de mercado. Este punto es muy importante en la captación de clientes nuevos ya que la mayoría de las personas se "casan" con la institución que le ayudó en sus primeros pasos crediticios.

Con todos los antecedentes mencionados en el apartado anterior, se propone desarrollar un modelo de calificación de créditos de originación (modelo 2 en la figura 1.1), que le permita a una entidad bancaria privada del Ecuador manejar de mejor manera su portafolio de clientes nuevos y sin historial crediticio. El modelo de *scoring* que emane de este trabajo, se desarrollará tomando en cuenta la información de variables de pasivos bancarios, variables sociodemográficas y de características laborales. Para ello, se propone el uso de los algoritmos XGBOOST para la estimación del modelo y LIME para la interpretación del mismo.

A partir de este punto, cada vez que se mencione algún término relacionado al tema *credit score* se asumirá que es enfocado al segmento de clientes sin antecedentes crediticios al cual denominaremos *Credit Score* de Originación.

1.1. Metodologías analíticas para el Credit Score

Para poder abordar el problema de *credit scoring* de manera adecuada, es preciso utilizar técnicas analíticas que sean capaces de afrontar las características particulares del problema (clases significativamente desbalanceadas y gran volumen de datos). Hoy en día, las técnicas empleadas para calcular la probabilidad de incumplimiento de los clientes son técnicas analíticas, matemáticas o estadísticas, como por ejemplo: regresión logística, árboles de decisión, regresión logística con estimación a través del método de Firth (procedimientos descritos minuciosamente en el Capítulo 2), cuyo objetivo es pronosticar la probabilidad de no pago de una obligación que presenta un cliente.

No obstante, la aplicación de la regresión logística para modelos de *credit scoring* orientados a segmentos de clientes sin antecedentes crediticios no logra atacar de forma adecuada la inexistencia de variables importantes dentro de este tipo de problemas, ya que son modelos que se basan solamente en relaciones de tipo lineal, por ende la precisión que llegan a tener no es la óptima [3]. Los árboles de decisión; en cambio, son algoritmos con los que se obtienen resultados bastante buenos siempre y cuando la cantidad de datos que se maneje no sea excesivamente grande, según lo afirma la investigación de Hidalgo (2014); ya que en dicho caso el modelo siempre tiende a ajustarse demasiado a los datos de entrenamiento teniendo una capacidad

[3] No se dice que la técnica es mala, sino que no es aplicable en este tipo de problemas.

muy baja de generalización. Por otro lado, la regresión logística con estimación a través del método de Firth, introducida en el año 1993 es una técnica diseñada especialmente para eventos raros, es decir, se enfoca en el adecuado tratamiento del desequilibrio de clases; pero a pesar de ser buena manejando la desproporción de clases presenta el mismo problema que la regresión logística tradicional, en el apartado 2.1.6 se presenta a detalle los resultados de varias investigaciones sobre los modelos detallados.

Lo que se propone en este estudio, es el uso del algoritmo XGBoost, técnica relativamente nueva y poco usada en estos estudios para calcular la probabilidad de incumplimiento en las entidades bancarias, técnica no utilizada actualmente en Ecuador y que genera mejores y correctos resultados en comparación a las técnicas utilizadas habitualmente, de acuerdo a las investigaciones de Petropoulos, et al (2018). Dicha técnica no es del todo desconocida, pues se basa principalmente en una de las técnicas tradicionales, como lo es el árbol de decisión, método con el que nos encontramos bastante familiarizados.

Por ello, es importante destacar, que el objetivo de este proyecto no es construir un "super" modelo que clasifique perfectamente a los clientes de la entidad bancaria (aunque sí será robusto gracias la eficacia de técnica propuesta), sino más bien, exponer que el uso de una técnica estadística confiable y moderna, ayuda a solucionar las dificultades presentadas en el problema de *credit score*. Además, la exposición de esta nueva herramienta brindará a las entidades bancarias una solución a diversos problemas con características similares.

Con la finalidad de mostrar su gran poder predictivo se realizará una comparación con la regresión logística con estimación a través del método de Firth, la cual ha resultado ser mejor que cualquier otro método tradicional en este tipo de problemas.

1.1.1. Aprendizaje Estadístico

El campo de la estadística se ve constantemente desafiado por los problemas que la ciencia y la industria le plantean. En sus inicios, estos problemas a menudo provenían de experimentos agrícolas e industriales y tenían un alcance relativamente pequeño. Con el advenimiento de las computadoras y la era de la información, los problemas estadísticos se han disparado tanto en tamaño como en complejidad. Los desafíos en las áreas de almacenamiento, organización y búsqueda de datos han

llevado al nuevo campo de la "minería de datos" [4]; los problemas estadísticos y computacionales en biología y medicina han creado la "bioinformática".

Se están generando grandes cantidades de datos en muchos campos, y el trabajo del estadístico es darle sentido a todo: extraer patrones y tendencias importantes y comprender "lo que dicen los datos". A esto lo llamamos aprender de los datos. Los desafíos para aprender de los datos han llevado a una revolución en las ciencias estadísticas. Dado que la computación juega un papel tan importante, no es sorprendente que gran parte de este nuevo desarrollo haya sido realizado por investigadores en otros campos como la informática y la ingeniería dando paso al aprendizaje automático.

1.1.2. Aprendizaje Automático

El aprendizaje automático se basa en la teoría del aprendizaje estadístico, que todavía se basa en esta noción axiomática de espacios de probabilidad. Esta teoría se desarrolló en la década de 1960 y amplía las estadísticas tradicionales. Esto debería ser abiertamente obvio, ya que el aprendizaje automático involucra datos, y los datos deben describirse utilizando un marco estadístico.

El aprendizaje automático (en inglés *machine learning*), es una rama de la inteligencia artificial [5] encargada de desarrollar una serie de algoritmos, que les permite a las máquinas o programas de cómputo aprender de los datos (información que se les proporcione), es decir, averiguar patrones que están implícitos sobre ellos y conocer el tipo de estructura que poseen, para luego extraer, consolidar y crear un nuevo conocimiento que será aplicado para la toma de decisiones correcta sobre nuevos conjuntos de datos. En resumen, el aprendizaje automático es el motor que hace que las máquinas o programas de cómputo sean inteligentes artificialmente (Alpaydin, 2014). Existen tres tipos de aprendizaje automático: aprendizaje supervisado, aprendizaje no supervisado y reforzado. Nuestro estudio se enfoca en el aprendizaje supervisado, específicamente en un problema de clasificación binaria (distinguir clientes buenos y malos).

[4] Es un campo de la estadística y las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos

[5] La Inteligencia Artificial (IA) es la combinación de algoritmos planteados con el propósito de crear máquinas que presenten las mismas capacidades que el ser humano.

Black Box Model

La caja negra (*black box* traducida al inglés) en la aviación, también conocida como registrador de datos de vuelo, es un dispositivo extremadamente seguro diseñado para proporcionar a los investigadores información altamente objetiva sobre cualquier anomalía que pueda haber provocado incidentes o contratiempos durante un vuelo.

La caja negra en los programas de Inteligencia Artificial y Aprendizaje Automático ha adquirido un significado opuesto. Los desarrolladores reconocen que el funcionamiento interno de estas "máquinas de autoaprendizaje" agrega una capa adicional de complejidad y opacidad con respecto al comportamiento de las máquinas. Una vez que se entrena un algoritmo de aprendizaje automático, puede ser difícil entender por qué da una respuesta particular a un conjunto de entradas de datos, es por esta razón que se han ganado la denominación de *black box models* (ver Figura 1.2).

Figura 1.2: Esquema de generación de información

Fuente: Elaboración Propia

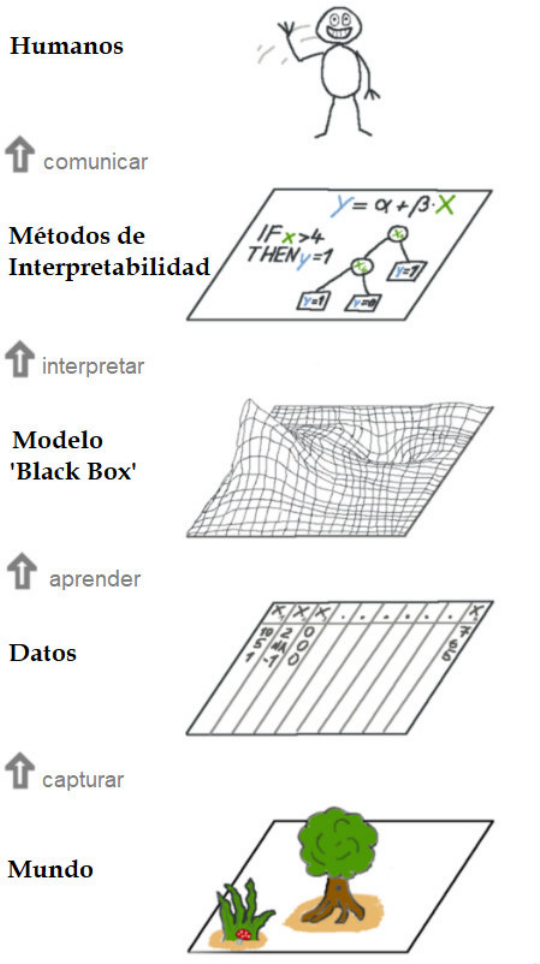
A medida que los algoritmos de aprendizaje automático se vuelven más inteligentes, también se vuelven más incomprensibles, y aunque el hecho de mejorar la precisión de los modelos parece tentador a la hora de la elección de un algoritmo, es importante también mencionar que existe una alta incertidumbre por determinar el por qué de una decisión. Es por esta razón que por lo menos en el ámbito bancario la adopción de este tipo de modelos ha sido lenta. Inspirados en esta controversia, en los últimos años ha aumentado el número investigadores que han trabajado fuertemente sobre temas de interpretabilidad de modelos de aprendizaje automático y su aplicación a problemas reales (Rothman, 2020).

1.1.3. Interpretabilidad de Modelos de Aprendizaje Automático

La pregunta que algunas personas se hacen a menudo es ¿por qué no estamos simplemente contentos con los resultados del modelo? y ¿por qué estamos tan em-

peñados en saber por qué se tomó una decisión en particular?. Mucho de esto tiene que ver con el impacto que un modelo podría tener en el mundo real. Porque los modelos que simplemente están destinados a recomendar películas tendrán un impacto mucho menor que los creados para predecir el resultado de una medicina o una inversión monetaria. La figura 1.3 nos muestra un esquema de cómo percibe una persona la información del mundo real y que es representada por un modelo, en este caso siempre es necesario tener un paso en donde se dé énfasis en la comunicación proveniente del modelo, es por eso la necesidad de contar con métodos de interpretabilidad de modelo de aprendizaje automático.

Figura 1.3: Panorama general del aprendizaje automático explicable



Fuente: Elaboración Propia

Contar con un modelo interpretable brinda algunos beneficios como:

- Fiabilidad
- Facilitar la depuración de errores

- Ayudar a la creación de nuevas variables
- Dirigir la recopilación de datos en el futuro
- Informar la toma de decisiones humana
- Generar confianza

Mientras el modelo no tenga un impacto significativo, su interpretabilidad no importa tanto, pero cuando hay implicaciones involucradas basadas en la predicción de un modelo, ya sea financiera o social, la interpretabilidad se vuelve relevante. Es por tal razón que al enfrentarse a la decisión de prestar dinero y tener la posibilidad de perderlo, a las instituciones financieras le interesa contar con modelos interpretables. Por tal motivo es importante entregar a la entidad bancaria para la cual se desarrolla este proyecto, un modelo con una capacidad predictiva muy alta e incluir un modelo que ayude a la interpretabilidad (detallado en el apartado 2.7) y toma de decisones, el modelo propuesto es el algoritmo LIME .

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un modelo predictivo que permita determinar la probabilidad de incumplimiento de un cliente sin antecedentes crediticios, a partir del uso de algoritmos de machine learning, con el fin de determinar el perfil de riesgo del cliente para mejorar la administración del portafolio de crédito de una entidad bancaria del Ecuador y adicionalmente complementar la metodología con la implementación de un algoritmo que facilite la interpretabilidad del modelo.

1.2.2. Objetivos Específicos

1. Demostrar empíricamente que los resultados del algoritmo de machine learning XGBOOST presentan una mejor precisión y/o poder predictivo que la técnica regresión logística con estimación a través del método de Firth.
2. Complementar la metodología de originación de crédito con la creación del algoritmo LIME, ayudando la interpretabilidad del modelo de Machine Learning para facilitar la aprobación e implementación del modelo en la entidad bancaria.

3. Generar Perfiles de Riesgo a partir de la probabilidad de incumplimiento de tal manera que se pueda clasificar a los clientes para tener una adecuada administración del portafolio.

Capítulo 2

Marco Teórico

En el presente capítulo se detallan conceptos relacionados al ámbito financiero a partir de los cuales se desarrolló el problema estudiado, además de los conceptos teóricos de diversas técnicas analíticas sobre las cuales se sustenta la metodología utilizada en la construcción y validación del modelo estadístico de *credit scoring*.

Se abordará temas concernientes a problemas de clasificación, árboles de decisión, conjuntos clasificadores (bagging, boosting, gradient boosting y XGBoost) y regresión logística con estimación a través del método de Firth (técnica utilizada para analizar la ocurrencia de eventos raros). Posteriormente, se describen los conceptos utilizados en la evaluación y selección de modelos, para finalmente abordar el tema de la interpretabilidad de modelos de aprendizaje automático.

Para comprender la utilidad de los aspectos metodológicos utilizados en el proceso de modelización y descritos a continuación, es necesario realizar las siguientes consideraciones iniciales:

- En primer lugar, el conjunto de datos utilizado para la construcción y validación del modelo se encuentra estructurado de forma matricial, cuyas filas corresponden a registros o casos y las columnas a variables o atributos observados en cada registro.
- Adicionalmente, al ser el *credit scoring* un problema de clasificación binaria, las nociones que se describen posteriormente se particularizan exclusivamente a este campo.

2.1. Definiciones y conceptos básicos del sistema financiero

2.1.1. ¿Qué es y para qué sirve un banco?

Se puede definir un banco como: "Una institución financiera que administra los recursos de sus accionistas y el de sus clientes, y que utiliza esos recursos para prestar a otras personas o empresas cobrándoles un interés. (Corporación Financiera Nacional, 2016, p.2)"

También se cataloga a los bancos como entes que se organizan de acuerdo a leyes especiales y que se dedican a trabajar con el dinero de otros, para lo cual reciben y tienen a su custodia depósitos hechos por las personas y las empresas, y otorgan préstamos usando esos mismos recursos, esta actividad es denominada intermediación financiera.

Al realizar la actividad de recibir dinero y luego darlo en préstamo, los bancos realizan estas actividades:

- Pagan intereses a quienes les entregan dinero en depósito (por la confianza depositada).
- Cobran intereses a quienes piden préstamos (lo necesitan).

La diferencia entre lo que se les paga y lo que ellos pagan, representa uno de los negocios que realiza el banco. Otras vías de negocio que llevan a cabo los bancos tienen que ver con comisiones por servicios realizados (uso de cajeros automáticos, transferencias, etc), actividades de tesorería y otros.

El rol principal de un banco recae en la intermediación de los fondos prestables. Según Merton (1993): "Un sistema financiero bien desarrollado y saludable facilita la alocación [1] eficiente de consumo de los individuos y del capital físico de las empresas en usos más productivos". Mientras más avanzado se encuentre el mercado de créditos de un país, este podría desarrollarse mucho más rápido. Es decir, las instituciones financieras contribuyen al avance económico del cliente y por ende del

[1] Término usado en el ámbito financiero para referirse a la distribución o colocación de dinero.

país mediante la oferta créditos y microcréditos[2] (Beckman, 1949).

2.1.2. Qué es un crédito

Según la Superintendencia de Bancos del Ecuador (2018) un crédito puede definirse como:

- El uso de un capital ajeno por un tiempo determinado a cambio del pago de una cantidad de dinero que se conoce como interés.
- Obtención de recursos financieros en el presente sin efectuar un pago inmediato, bajo la promesa de restituirlos en el futuro en condiciones previamente establecidas.

Además define el crédito bancario como: "es un contrato por el cual una entidad financiera pone a disposición del cliente cierta cantidad de dinero, el cual deberá de devolver con intereses y comisiones según los plazos pactados".

Las entidades bancarias obtienen sus ingresos principalmente de la colocación de créditos tanto a personas naturales como a empresas, siendo esta actividad la que ha apalancado el crecimiento de esta actividad económica. En este sentido las instituciones otorgan recursos financieros a un determinado cliente para que pueda cubrir sus necesidades ya sean eventuales o permanentes; cobrando una tasa de interés por el monto (Gobat, 2012). Existen créditos para cubrir necesidades específicas como por ejemplo: la compra de vehículos, adquisición de terrenos y viviendas, créditos educativos, de consumo, etc. De estas definiciones dependerán las condiciones del crédito, algunas de ellas son:

- Requisito de codeudor.
- Porcentaje de la tasa de interés.
- Garantías.

Dado que se presentan muchos factores que influyen en el pago o no pago de una obligación de crédito, esta actividad es considerada de alto riesgo, pues la institución financiera está expuesta a que el cliente no cumpla con sus obligaciones de

[2]El microcrédito es una modalidad de financiamiento que se caracteriza por prestar cantidades reducidas de capital para impulsar proyectos productivos de las pymes en los distintos sectores de la economía.

pago total o parcial en la devolución del crédito; a este riesgo se lo denomina riesgo de crédito.

Con el fin de explorar todos los riesgos a los que se encuentran expuestos los bancos, en el siguiente apartado, además de detallar la información concerniente al riesgo de crédito, mencionaremos algunos otros tipos de riesgos.

2.1.3. Riesgos Financieros

El riesgo financiero puede entenderse como la probabilidad de tener un resultado negativo e inesperado debido a los movimientos del mercado, estos riesgos pueden provocarse por una mala administración de los flujos de caja o por los riesgos relacionados con ingresos por debajo de lo esperado. Se cuenta con una extensa clasificación del riesgo, sin embargo este estudio estará enfocado en una clasificación que contiene cuatro tipos de riesgos, planteados en los Acuerdos de Capital de Basilea I y II [3] y de mayor impacto en las instituciones financieras. Los conceptos fueron tomados de los trabajos de Tapiero (2004), Iñiguez y Morales (2009) y Roncalli (2020). Se definen a continuación los principales riesgos:

Riesgo de Mercado

El riesgo de mercado se define como la pérdida potencial por cambios en los factores de riesgo que inciden sobre la valuación o sobre los resultados esperados, como tasas de interés, tipos de cambio, precios de mercado, índices y otros factores de riesgo en los mercados de dinero, cambios, y productos derivados a los que se encuentra expuesto. Su valuación correcta requiere de la oportunidad y calidad de la información sobre el valor de mercado actual de los activos, pasivos y elementos de cuentas de orden de una institución.

El riesgo de mercado puede, por tanto, subdividirse en:

- Riesgo de tipos de interés
- Riesgo de tipo de cambio

[3] Los acuerdos de Basilea son una serie de directrices elaboradas por el Comité de Basilea a finales de 1974, formado por los gobernadores de los bancos centrales del G-10, para evitar riesgos sistémicos en situaciones de pánico bancario, que tuvieron su origen en las turbulencias financieras registradas en los mercados de divisas.

- Riesgo de precios bursátiles
- Riesgo de precios de las mercancías

Las mediciones del riesgo de mercado pueden ser globales, relativas a todos estos subriesgos, o específicas, de cada una de estas categorías de riesgo. Sin embargo, estos factores no son independientes entre sí, sino que están relacionados de manera que no resulta posible la fragmentación del riesgo de mercado en cuatro diferentes.

La metodología de Valor en Riesgo (VaR, por su siglas en inglés Value at Risk) se ha consolidado hasta convertirse en el método de medición de riesgos de mercado más comúnmente utilizado, el VaR se suele definir como la pérdida máxima esperada en un horizonte de tiempo dado y con cierto nivel de confianza. El Valor en Riesgo está directamente relacionado con la volatilidad del valor del portafolio, el cual se ve afectado por los cambios en los factores que inciden en el valor de las posiciones que componen el portafolio. Una definición más completa indica que el VaR es un número que representa la caída de valor de la cartera, correspondiente a un percentil determinado de la variable aleatoria rentabilidad futura de dicha cartera, en un horizonte temporal determinado.

Riesgo de Liquidez

El riesgo de liquidez se define como la pérdida potencial por la imposibilidad o dificultad de renovar pasivos en condiciones normales para la institución, o por la venta anticipada o forzosa de activos a descuentos inusuales. Los riesgos de liquidez de una entidad financiera se derivan de desfases en los flujos de las operaciones de captación, crédito y negociación como son:

- Pasivos a la vista
- Vencimientos de depósitos a plazo
- Disposición de líneas de crédito
- Liquidación de operaciones con fines de negociación y con instrumentos derivados.
- Gastos operativos.

En la medida en que la institución tenga la capacidad de obtener recursos de fuentes de fondeo alternas que tengan un costo aceptable, el riesgo de liquidez se reduce. Entre los elementos que intervienen en la estrategia aplicada en la gestión de la liquidez están:

- Diferenciar el tratamiento de Activos, Pasivos e ítems fuera de balance.
- Controlar las brechas de vencimientos de activos y pasivos,
- Diversificar las fuentes de captación de fondos,
- Diversificar los vencimientos de activos y pasivos,
- Establecer límites prudentes y garantizar el acceso inmediato a los activos líquidos.

La metodología LaR (Liquidity at Risk) es la más frecuentemente utilizada para la medición del riesgo de liquidez, se determina por el nivel de bursatilidad de cada uno de los instrumentos que conforman la posición, obteniéndose una medida de VaR ajustado por liquidez, dicha metodología consiste en adicionar al VaR de mercado el costo que representaría no poder vender el instrumento por falta de liquidez en el mercado.

Riesgo Operativo

El riesgo operativo se define como el riesgo de pérdida debido a la inadecuación o fallos de los procesos, el personal y los sistemas internos o bien a causa de acontecimientos externos.

Entre las actividades generadoras de riesgo operativo se encuentran: el Outsourcing de procesos, la integración de sistemas por fusiones o adquisiciones, las prácticas comerciales agresivas y el crecimiento de servicios bancarios a través del internet [4]. Los eventos de riesgo operativo generan pérdidas indirectas y pérdidas directas, estas últimas se contabilizan como gasto. A continuación se presentan ejemplos de pérdidas indirectas por riesgo operativo:

- Transacciones no realizadas por falta de reemplazo de personal o por falta de documentación.

[4]Scalar Consulting, Aspectos Cuantitativos y Cualitativos de Riesgo Operativo.

- Horas – Hombre gastadas en resolver fallas diarias, realizar operaciones mecánicas, seguir procesos ineficientes.
- Tiempo consumido en reprocesos.
- Costos de oportunidad en general.

Para la identificación del riesgo operativo en una entidad se realiza un inventario de procesos, que permita establecer las etapas críticas de cada uno de los procesos seguidos en la institución, así como vincular los procesos críticos con sus respectivas áreas o departamentos. Es decir se requiere identificar líneas de negocio y tipos de eventos.

Riesgo de Crédito

El riesgo de crédito se refiere a la pérdida potencial en la que incurre quien otorga un crédito, debido a la posibilidad de que la contraparte no cumpla con sus obligaciones (probabilidad de no-pago). Esta definición simplificada esconde varios riesgos, la cantidad de riesgo es el saldo insoluto del crédito otorgado. La calidad resulta tanto de la probabilidad de que ocurra el incumplimiento, como de las garantías que reducen la pérdida, debido a la recuperación potencial que se puede hacer del crédito, lo que depende de cualquier elemento que mitigue el riesgo, tales como las garantías reales, los avales, la capacidad de negociación con el acreditado, entre otros que permiten identificar la pérdida en el evento del default.

El incumplimiento es un elemento incierto y por otro lado la exposición al riesgo de crédito al momento del incumplimiento no se conoce. Así mismo la recuperación que se pueda hacer de un crédito tampoco se conoce de antemano.

Los factores que influyen en el riesgo de crédito pueden resumirse en los que se detallan a continuación:

- La economía: un buen crecimiento económico implica menor desempleo y mejor índice de calidad de cartera.
- El segmento de mercado: No siempre sigue el mismo camino de la economía.
- La actividad económica del asociado o cliente: Factores socioeconómicos, por ejemplo cuando es empleado su situación económica está muy ligada a la salud financiera de la empresa donde labora.

La administración del riesgo de crédito incluye fijar límites de crédito, con el fin de restringir las pérdidas en caso de incumplimiento. Antes de la toma de cualquier decisión de crédito debe existir un proceso de evaluación, mismo que debe establecer el monto máximo en riesgo que se está dispuesto a asumir con un cliente actual o futuro.

Los principios para establecer límites persiguen los objetivos siguientes:

- Evitar que la pérdida en un solo crédito ponga en peligro a la institución.
- Diversificar los compromisos de otorgamiento de crédito en varias dimensiones (por cliente, por sector económico, por región o zona geográfica).
- Evitar otorgar crédito a cualquier persona o grupo por una cantidad tal que exceda su capacidad de endeudamiento.

Las técnicas desarrolladas para la medición del riesgo de crédito y la medición de la probabilidad de incumplimientos son muy variadas, a continuación se nombran brevemente algunas de ellas:

- Técnicas econométricas: análisis lineal y discriminante, regresiones múltiples, modelos binarios para estimar la probabilidad de incumplimiento como variable dependiente, cuya varianza es explicada por un conjunto de variables independientes. Las variables independientes deben estar relacionadas con el acreditado.
- Redes neuronales: utilizan los mismos datos que las técnicas econométricas pero crean un modelo de decisión a través de emular una red de neuronas (unidades de decisión) interconectadas.
- Modelos de Optimización: herramientas matemáticas de programación que buscan optimizar la relación entre el acreditado y los atributos del crédito para minimizar el incumplimiento y maximizar la utilidad de la institución.
- Sistemas expertos: Se utilizan para tratar de replicar de manera estructurada el proceso que un analista experto realiza para tomar una decisión de crédito, se caracterizan por establecer un grupo de reglas de decisión.
- Sistemas Híbridos (Sistemas de cómputo, Estimaciones y Simulaciones): Buscan relaciones directas causales de incumplimiento a través de la estimación de parámetros y la elaboración de matrices de probabilidad de migración para predecir la tendencia de un crédito a migrar a una mejor o peor condición.

2.1.4. Administración del Riesgo

Una vez que se ha definido el concepto de riesgo financiero es necesario mencionar las características de la administración de riesgos en el sistema financiero. Según De Lara Haro (2005), la gestión de riesgos: "...es en esencia un método racional y sistemático para entender los riesgos, medirlos y controlarlos en un entorno en el que prevalecen los instrumentos financieros sofisticados, mercados financieros que se mueven con gran rapidez y avances tecnológicos en los sistemas de información que marcan nuestra era".

El riesgo puede ser catalogado como la incertidumbre o aleatoriedad en la obtención de un resultado al desarrollar una determinada actividad, el concepto de riesgo tiene dos elementos:

- La probabilidad de que un evento ocurra
- Las consecuencias de tal evento

En las actividades financieras el riesgo es ineludible, ya que existe una relación directa entre el grado de riesgo asumido por una institución y el potencial rendimiento a ser generado, por ejemplo, es conocido que al conceder un crédito siempre existe un nivel de riesgo (no pago de las obligaciones), si la entidad no quisiera correr ningún tipo de riesgo simplemente no debería prestar dinero pero en este caso no tendría rendimientos, en el caso contrario si incrementa mucho la colocación de créditos los rendimientos aumentan pero también lo hace de manera descontrolada el riesgo asumido. Es por ello que en este tipo de instituciones se debe mantener un adecuado juicio del riesgo y la institución financiera tendrá que determinar el nivel de riesgo que está dispuesta a asumir.

El proceso de administración de riesgos tiene como objetivo los siguientes puntos:

1. Identificar los riesgos a los que se encuentra expuesta una institución.
2. Medirlos los riesgos, bajo metodologías implementadas dentro de la institución.
3. Hacer el debido seguimiento del impacto operacional y controlar los efectos sobre los rendimientos.

Todo esto, mediante la implementación de estrategias y mecanismos que permitan realizar las operaciones con niveles acordes con su respectivo capital global y capacidad operativa, integrando la cultura de riesgos en la operación diaria.

Según Iñiguez y Morales (2019): "Una adecuada gestión del riesgo tiene un impacto positivo en la rentabilidad de una institución, puesto que se controla la exposición a pérdidas, aunque en un inicio estas medidas provoquen la constitución de provisiones y otros cambios en el Estado de Situación General de la entidad, dichos cambios influyen en los indicadores financieros, principalmente en los de rentabilidad y eficiencia".

A través de los años las instituciones financieras han ido incorporando a su estructura organizativa las Unidades de Riesgos, las cuales son encargadas de la administración integral de riesgos, convirtiéndose en un pilar fundamental en el soporte a la toma de decisiones como:

- La identificación y valuación de los distintos tipos de riesgo.
- El establecimiento de políticas, procedimientos y límites de riesgo.
- Monitoreo y reporte del cumplimiento de los límites establecidos.
- Delineación del capital asignado y de la administración de la cartera.
- Guías para el desarrollo de nuevos productos y la inclusión de nuevas exposiciones al riesgo dentro de la estructura existente.
- Aplicación de nuevos métodos de medición a los productos existentes.

2.1.5. Comité de Basilea

Basados en la enorme preocupación que representan los riesgos en el sector financiero y con el fin de lograr una adecuada gestión de riesgos y, por ende, estabilidad en el sistema financiero, se creó un fórum internacional dedicado a la promoción de contactos, discusiones e intercambio entre los bancos centrales llamado *Comité de Supervisión Bancaria de Basilea* (o simplemente Comité de Basilea), en donde, se buscaron crear políticas de regulación que ayuden a una adecuada administración del riesgo. En este sentido, se dio inicio a los Acuerdos de Basilea de 1988, conocidos como Basilea I, lo cual significó un gran paso transformando así el sistema financiero. Estos acuerdos se han ido mejorando o agregando nuevas políticas que se ajusten

a las nuevas realidades y retos que enfrenta el sector. Posterior a Basilea I, vinieron dos grandes cambios más, a los cuales se denominó Basilea II y Basilea III, y actualmente se encuentra un proceso de implementación de nuevas políticas que se denomina Basilea IV.

Basilea I, se encuentra enfocada principalmente en el riesgo de crédito. Entre estas políticas se encontraba que al riesgo de crédito se lo agrupaba en cinco categorías y cada categoría se le asignaba una ponderación de riesgo.

Basilea II, se basa en tres pilares fundamentales:

- Pilar I: El cálculo de los requisitos mínimos de capital.
- Pilar II: El proceso de supervisión de la gestión de los fondos propios.
- Pilar III: La disciplina de mercado

El Pilar I , establece políticas y metodologías para la administración de riesgo de crédito. Se introducen los conceptos de probabilidad de incumplimiento(PD), pérdida dado el incumplimiento o severidad (LDG) y exposición al momento del incumplimiento (EAD). Para el cálculo del PD y LGD, se propone dos tipos de metodologías: el método estándar, el cual se basa en las calificaciones externas y la metodología de calificaciones internas, en la cual se da potestad a las entidades financieras a realizar sus propias evaluaciones internas. Esta metodología se puede a su vez subdividir en dos tipos: método básico (FIRB) y método avanzado (AIRB), con la única diferencia que en el método básico la entidad financiera calcula la PD, y el resto es determinado por el ente regulador, mientras que el método avanzado da potestad a que cada entidad estime todas las variables.

Un resumen completo de los acuerdos de Basilea los presenta Yépez (2019), en este trabajo se plantea usar la misma metodología de construcción de modelos de Credit Score recomendada por Basilea y se propone hacer un cambio en el uso de modelos clásicos como la regresión logística, al uso de algoritmos de machine learning.

2.1.6. Modelos Credit Scoring

Credit scoring es el conjunto de métodos de decisión que aplican los bancos para aprobar préstamos, estos utilizan puntajes para evaluar el riesgo potencial que representa prestar dinero a los consumidores, además de, ser útiles para mitigar las pérdidas debidas a deudas incobrables.

Los modelos de score o *credit scoring* son considerados un buen punto de partida para comenzar el análisis formal de las filosofías de clasificación; básicamente sirven para estimar el riesgo de que un cliente no pague, basados en fórmulas estadísticas para identificar buenos o malos clientes. La estimación se realiza por medio de la discriminación de los nuevos clientes con el uso de datos sobre el cumplimiento de clientes antiguos (Rodríguez, Becerra y Cardona, 2017), con el fin de determinar el comportamiento de pago de esos nuevos clientes.

Las técnicas del credit Scoring asignan el riesgo de crédito a un cliente en particular. No obstante, la técnica no puede asignar a un consumidor la categoría de 'sujeto de crédito', pues éste no es un atributo o característica inherente al individuo, como el peso, la altura o incluso el nivel de ingresos. La consideración 'sujeto de crédito' es una valoración del que acredita respecto al acreditado y refleja las circunstancias en las cuales ambos se encuentran, así como la percepción del primero sobre los futuros escenarios económicos. Por lo tanto, no es saludable considerar 'no sujeto de crédito' a un consumidor cuyo perfil de riesgo no se acopla al requerido por la institución financiera. Resulta menos agraviante y refleja mejor el estado de la realidad decir que la solicitud de crédito del prestatario representa un riesgo que no se está dispuesto a asumir (Thomas, Crook y Edelman, 2017).

Las entidades financieras deben tomar decisiones en cualquier etapa del ciclo de crédito [5], por ejemplo, se debe decidir si otorgarle o no un crédito a un cliente nuevo, y en una etapa más avanzada la institución financiera requiere decidir cómo tratar a clientes ya conocidos e incluso si debe incrementar o disminuir sus límites de crédito o cupos de crédito.

[5] se entiende por ciclo de crédito a las actividades secuenciales directamente relacionadas con la concesión de créditos

Ventajas y Desventajas de las distintas Metodologías del Credit Scoring

En este apartado se describirá algunos trabajos de credit scoring que usan distintas metodologías, y se mostrarán sus ventajas, desventajas y las características que identifican los diferentes casos.

Existen diversos tipos de métodos que podemos aplicar en la creación de modelos de Credit Scoring, entre las que se encuentran: la regresión lineal, regresión logística, redes neuronales, árboles de decisión, análisis del discriminante, cadenas de Markov, modelos logit y probit, algoritmos genéticos, etc. Para Gutiérrez (2007), todos estos métodos obtienen resultados similares, por lo que el uso de cada uno de ellos dependerá exclusivamente de las características particulares de cada caso.

La regresión lineal consiste en aproximar la dependencia lineal de una variable en función de otras variables. Este método es el menos usado debido al problema que en este tipo de modelos, las probabilidades de incumplimiento no necesariamente se encuentran entre 0 y 1, contradiciendo así la definición de probabilidad. Sin embargo, si las probabilidades se encuentran entre 0.2 y 0.8 , los resultados serán similares a los modelos probit, logit.

Bonilla, Puertas y Olmeda (2003) sugieren el uso de redes neuronales, concluyendo que el uso de estos algoritmos otorga mejores resultados. Además de esto, otro estudio desarrollado por Abdouab, Dongmo, Ntim y Barker (2016) en la cual comparaba la eficiencia de las diversas técnicas llegaron a la misma conclusión. Sin embargo, su uso aún es limitado, por la dificultad de interpretación y por ende difícil toma de decisiones que ayuden a mejorar el proceso de otorgamiento de crédito.

Se ha observado que la mayoría de entidades financieras prefieren la regresión logística. Aunque existan otras técnicas que mejoran los resultados, el uso de esta técnica se debe a la fácil interpretación de los resultados que facilitan la toma de decisiones que ayudan a mejorar el proceso de otorgamiento del crédito, aunque con el pasar del tiempo se han ido desarrollando investigaciones que promueven el uso de técnicas alternativas, a continuación se detallan algunas:

- Srinivasan y Kim (1987) realiza una comparación entre arboles de decisión y las regresiones logísticas y análisis del discriminante, llegando a la conclusión

que los árboles de decisión arrojan mejores resultados que las regresiones logísticas y estas a su vez, arrojan mejores resultados que el análisis del discriminante.

- Khandani et al. (2010) utilizan técnicas de machine learning para la construcción de modelos no paramétricos no lineales de riesgo de crédito de consumo obteniendo muy buenos resultados.

- Sun, Ramayya y Krishnan (2012) recurren a los modelos de supervivencia de Cox, para estimar la probabilidad de incumplimiento en tarjetas de crédito.

- Addo et al. (2018) concluye que las técnicas basadas en árboles son más estables que las técnicas basados en redes neuronales artificiales multicapas.

Sin embrago, uno de los principales artículos lo publica Petropoulos, et al (2018) en la Conferencia de Basilea, en el cual utilizan la técnica de machine learning XGBoost para la estimación de la probabilidad de incumplimiento de una fuente de datos de gran tamaño a nivel de préstamo y, en segundo lugar, crear un sistema útil, desde un ámbito regulatorio, de calificación crediticia que mida el riesgo crediticio en carteras de bancos supervisados. La conclusión de este artículo indica que la técnica XGBoost junto con las redes neuronales proporciona un mejor rendimiento en términos de precisión de clasificación y calibración del sistema de calificación crediticia en comparación con técnicas ampliamente empleadas en riesgo crediticio como la regresión logística y análisis discriminante lineal. Es por tal razón que es la técnica elegida para usar en este trabajo.

2.2. Problemas de Clasificación

Tanto en la estadística clásica como en el aprendizaje automático, la clasificación se refiere a un problema de modelado predictivo donde se predice una etiqueta de clase para un caso dado que cuenta con una serie de datos de entrada. Los ejemplos de problemas de clasificación incluyen:

- Dado un correo, clasifique si es spam o no.

- Dado la imagen de un tumor, clasificarlo como benigno o maligno.

- Dado el comportamiento de pago del usuario, clasifíquelo como buen o mal pagador.

Desde una perspectiva de modelado, la clasificación requiere un conjunto de datos de entrenamiento con muchos ejemplos de entrada y salida de los cuales aprender. Un modelo utilizará el conjunto de datos de entrenamiento y calculará la mejor manera de mapear ejemplos de datos de entrada a etiquetas de clase específicas.

Las etiquetas de clase son a menudo valores de cadena, por ejemplo: ("spam", "no spam"), y deben asignarse a valores numéricos antes de proporcionarse a un algoritmo para el modelado. Esto a menudo se denomina codificación de etiquetas, donde se asigna un número entero único a cada etiqueta de clase, para el ejemplo anterior sería: ("spam" = 0, "no spam" = 1).

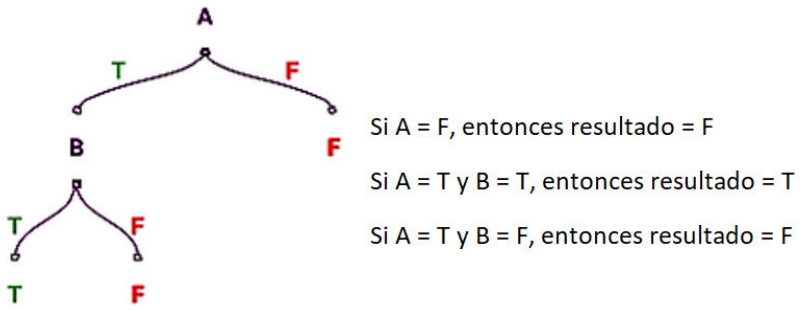
Para revolver estos problemas de clasificación existen varias técnicas de modelamiento, las cuales serán descritas a lo largo de esta sección y cuyas definiciones fueron tomadas de Breiman et al. (1984) y Hastie et al. (2009).

2.2.1. Árboles de decisión

Los árboles de decisión son algoritmos que dividen progresivamente conjuntos de datos en grupos de datos más pequeños en función de una característica descriptiva, hasta que alcanzan conjuntos que son lo suficientemente pequeños como para ser descritos por alguna etiqueta; son conceptualmente simples pero a la vez muy potentes.

Figura 2.1: Ejemplo de clasificación binaria (Operador AND)

A	B	A & B
F	F	F
F	T	F
T	F	F
T	T	T



Fuente: Elaboración Propia

Los algoritmos de árboles de decisión son perfectos para resolver problemas de clasificación (clasificar los datos en categorías) y de regresión (predecir valores). Los

árboles de regresión se utilizan cuando la variable dependiente es continua o cuantitativa (por ejemplo, si queremos estimar el precio de una casa a partir de sus características), y los árboles de clasificación se utilizan cuando la variable dependiente es categórica o cualitativa (por ejemplo, si queremos estimar el tipo de sangre de una persona).

Los árboles de decisión son extremadamente populares por una variedad de razones, siendo su interpretación probablemente su ventaja más importante. Pueden entrenarse muy rápido y son fáciles de entender, lo que abre sus posibilidades a fronteras mucho más allá de los muros científicos.

Árboles de decisión para regresión

El algoritmo necesita decidir automáticamente las variables de división y los puntos de división, y también qué topología (forma) debe tener el árbol. Supongamos primero que tenemos una partición en M regiones $R_1, R_2, ..., R_M$, y modelamos la respuesta como una constante c_m en cada región:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{2.1}$$

Utilizando el criterio de minimización de la suma de cuadrados $\sum_i (y_i - f(x_i))^2$, es fácil ver que el mejor $\hat{c}_m$ es el promedio de y_i en la región R_m

$$\hat{c}_m = mean(y_i | x_i \in R_m) \tag{2.2}$$

Ahora bien, encontrar la mejor partición binaria en términos de suma mínima de cuadrados es generalmente 'computacionalmente inviable'. Por lo tanto, procedemos con un algoritmo codicioso. Comenzando con todos los datos, considere una variable de división j y un punto de división s, y defina el par de semiplanos

$$R_1(j,s) = \{X | X_j <= s\} \quad \text{y} \quad R_2(j,s) = \{X | X_j > s\} \tag{2.3}$$

Luego buscamos la variable de división j y el punto de división s que resuelven

$$\min_{j,s} \left[\min_{c1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \tag{2.4}$$

Para cualquier elección de j y s, la minimización es resuelta por

$$\hat{c}_1 = mean(y_i | x_i \in R_1(j,s)) \quad \text{y} \quad \hat{c}_2 = mean(y_i | x_i \in R_2(j,s)) \tag{2.5}$$

Para cada variable de división, la determinación de los puntos de división s se puede hacer muy rápidamente y, por lo tanto, al escanear todas las entradas, es factible determinar el mejor par (j, s).

Habiendo encontrado la mejor división, dividimos los datos en los dos resultados regiones y repita el proceso de división en cada una de las dos regiones. Luego, este proceso se repite en todas las regiones resultantes.

¿Qué tan grande debemos hacer crecer el árbol? Claramente, un árbol muy grande podría sobreajustar los datos, mientras que un árbol pequeño podría no capturar la estructura importante.

El tamaño del árbol es un parámetro de ajuste que gobierna la complejidad del modelo, y el tamaño óptimo del árbol debe elegirse de forma adaptativa a partir de los datos. Un enfoque sería dividir los nodos de los árboles solo si la disminución en la suma de cuadrados debido a la división excede algún umbral. Sin embargo, esta estrategia es demasiado simple, ya que una división aparentemente sin valor podría llevar a una división muy buena por debajo de ella. La estrategia preferida es hacer crecer un árbol grande T_0, deteniendo el proceso de división solo cuando se alcanza un tamaño mínimo de nodo (digamos 5 registros). Luego, este gran árbol se poda mediante la *poda de costo-complejidad*, que se describirá a continuación. Definimos un subárbol $T \subset T_0$ como cualquier árbol que se pueda obtener podando T_0, es decir, colapsando cualquier número de sus nodos internos (no terminales). Indexamos los nodos terminales por m, y el nodo m representa la región R_m. Sea $|T|$ el número de nodos terminales en T. Teniendo

$$\begin{aligned} N_m &= \#\{x_i \in R_m\} \\ \hat{c}_m &= \frac{1}{N_m} \sum_{x_i \in R_m} y_i \\ Q_m(T) &= \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 \end{aligned} \tag{2.6}$$

definimos el criterio *costo-complejidad*

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T| \tag{2.7}$$

La idea es encontrar, para cada α, el subárbol $T_\alpha \subseteq T_0$ para minimizar $C_\alpha(T)$. El parámetro de ajuste $\alpha >= 0$ gobierna la compensación entre el tamaño del árbol y su capacidad de ajustar los datos. Valores grandes de α dan como resultado árboles T_α más pequeños y, a la inversa para valores más pequeños de α. Como sugiere la notación, con $\alpha = 0$ la solución es el árbol completo T_0.

Vamos a elegir α adaptativamente, se puede demostrar que para cada α existe un único subárbol T_α que minimiza $C_\alpha(T)$. Para encontrar T_α usamos *la poda de eslabones más débiles*: colapsamos sucesivamente el nodo interno que produce el menor aumento por nodo en $\sum_m N_m Q_m(T)$, y continuamos hasta que producimos el árbol de un solo nodo (raíz). Esto da una secuencia (finita) de subárboles, y se puede mostrar que esta secuencia debe contener T_α. Ver Breiman et al. (1984) o Ripley (1996) para más detalles.

La estimación de α se logra mediante validación cruzada [6] de cinco o diez veces: elegimos el valor $\hat{\alpha}$ para minimizar la suma de cuadrados con validación cruzada. Nuestro árbol final es $T_{\hat{\alpha}}$.

Árboles de decisión para clasificación

En la apartado anterior se presentó la forma de resolver un problema de regresión usando árboles de decisión, veamos ahora qué sucede si el objetivo es resolver un problema de clasificación, ahora se tiene una variable con k categorías: $1, 2, ..., K$; los únicos cambios necesarios en el algoritmo del árbol pertenecen a los criterios para dividir nodos y podar el árbol. Mientras que para el problema de regresión usamos el error cuadrático como medida de impureza del nodo $Q_m(T)$ definido en 2.6, este no es adecuado para problemas de clasificación.

[6] Es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba, se detallará en los siguientes apartados.

En un nodo m, que representa una región R_m con N_m observaciones, definimos

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

la proporción de observaciones de la clase k en el nodo m. Vamos a clasificar las observaciones en el nodo m en la clase $k(m) = arg\,\text{máx}_k\, \hat{p}_{mk}$, la clase mayoritaria en el nodo m. Se detallan a continuación algunas de las medidas de impureza del nodo m, $Q_m(T)$:

$$\begin{aligned} \text{Error de clasificación} &: \frac{1}{N_m} \sum_{i \in R_m} I(y_i \neq k(m)) = 1 - \hat{p}_{mk(m)} \\ \text{Índice de Gini} &: \sum_{k \neq k'} \hat{p}_{mk}\hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \\ \text{Entropía cruzada o desviación} &: -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \end{aligned} \tag{2.8}$$

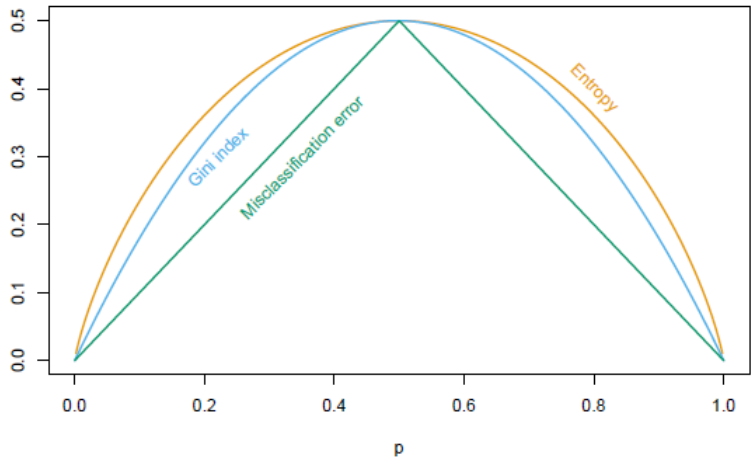
Para dos categorías, si p es la proporción en la segunda categoría, estas tres medidas son:

- $1 - \text{máx}(p, 1-p)$
- $2p(1-p)$
- $-p \log p - (1-p) \log (1-p)$

respectivamente, las cuales se pueden visualizar en la Figura 2.2. Se evidencia que las tres son similares, pero la entropía cruzada y el índice de Gini son diferenciables y, por lo tanto, más susceptibles de optimización numérica. Comparando 2.4 y 2.6, vemos que necesitamos ponderar las medidas de impureza del nodo por el número N_{mL} y N_{mR} de observaciones en los dos nodos secundarios creados al dividir el nodo m.

Además, la entropía cruzada y el índice de Gini son más sensibles a los cambios en las probabilidades de los nodos que la tasa de clasificación errónea. Por ejemplo, en un problema de dos clases con 400 observaciones en cada clase (denotada por $(400; 400)$), suponga que una división creó nodos $(300; 100)$ y $(100; 300)$, mientras que la otra creó nodos $(200; 400)$ y $(200; 0)$. Ambas divisiones producen una tasa de clasificación errónea de 0.25, pero la segunda división produce un nodo puro y probablemente sea preferible. Tanto el índice de Gini como la entropía cruzada son más

Figura 2.2: Medidas de impurezas de los nodos para la clasificación de dos categorías



Fuente: (Breiman et al.,1984)

bajos para la segunda división. Por esta razón, se debe usar el índice de Gini o la entropía cruzada al hacer crecer el árbol. Para orientar la poda de costo-complejidad, se puede usar cualquiera de las tres medidas, pero típicamente es la tasa de clasificación errónea.

El índice de Gini se puede interpretar de dos formas interesantes. En lugar de clasificar las observaciones en la clase mayoritaria del nodo, podríamos clasificarlas en la clase k con probabilidad $\hat{p}_{mk}$. Entonces, la tasa de error de entrenamiento de esta regla en el nodo es $\sum_{k \neq k'} \hat{p}_{mk}\hat{p}_{mk'}$ - el índice de Gini. De manera similar, si codificamos cada observación como 1 para la clase k y cero en caso contrario, la varianza sobre el nodo de esta respuesta $\{0/1\}$ es $\hat{p}_{mk}(1 - \hat{p}_{mk})$. La suma de las clases k da nuevamente el índice de Gini.

Predictores con Valores perdidos

Suponga que a nuestros datos les faltan algunos valores predictores en algunas o todas las variables. Podríamos descartar cualquier observación con algunos valores perdidos, pero esto podría conducir a una disminución significativa del conjunto de entrenamiento. Alternativamente, podríamos intentar completar (imputar) los valores perdidos, con la media de ese predictor sobre las observaciones no perdidas. Para los modelos basados en árboles, existen dos enfoques mejores. El primero es aplicable a los predictores categóricos: simplemente creamos una nueva categoría para "missing". A partir de esto, podríamos descubrir que las observaciones con va-

lores perdidos para alguna medición se comportan de manera diferente a aquellas con valores no perdidos. El segundo enfoque más general es la construcción de variables sustitutas. Cuando se considera un predictor para una división, usamos solo las observaciones para las cuales ese predictor no falta. Habiendo elegido el mejor predictor (primario) y el punto de división, formamos una lista de predictores sustitutos y puntos de división. El primer sustituto es el predictor y el punto de división correspondiente que mejor imita la división de los datos de entrenamiento lograda por la división primaria. El segundo sustituto es el predictor y el punto de división correspondiente que obtiene el segundo mejor resultado, y así sucesivamente. Al enviar observaciones hacia abajo del árbol, ya sea en el fase de entrenamiento o durante la predicción, usamos las divisiones sustitutas en orden, si falta el predictor primario. Las divisiones sustitutas aprovechan correlaciones entre predictores para tratar de aliviar el efecto de los datos faltantes. Cuanto mayor sea la correlación entre el predictor faltante y los otros predictores, menor será la pérdida de información debido al valor faltante.

Inestabilidad de los árboles

Un problema importante con los árboles es su alta varianza. A menudo, un pequeño cambio en los datos puede resultar en una serie de divisiones muy diferente, lo que hace que la interpretación sea algo precaria. La razón principal de esta inestabilidad es la naturaleza jerárquica del proceso: el efecto de un error en la división superior se propaga a todas las divisiones inferiores. Se puede aliviar esto hasta cierto punto tratando de utilizar un criterio de división más estable, pero la inestabilidad inherente no se elimina. Es el precio a pagar por estimar una estructura simple basada en árboles a partir de los datos. En el siguiente apartado vamos a revisar dos técnicas que utilizan el promedio de muchos árboles para reducir esta variabilidad, esto ayudará para obtener estimadores mas robustos que ayuden a mejorar la clasificación de clientes buenos y malos pagadores.

2.3. Conjuntos Clasificadores

Hidalgo (2014) asegura que los conjuntos de clasificadores son estructuras formadas por una serie de clasificadores individuales o base, cuya combinación logra significativamente un rendimiento superior (predicción más precisa) que cualquier clasificador individual bajo el cumplimiento de algunos requisitos, las condiciones

que se deben cumplir para que el conjunto de clasificadores sea más efectivo son:

- Los clasificadores individuales deben ser lo más diversos posibles pero, manteniendo su precisión, es decir, que no dependan entre sí, ya que de este modo elementos clasificados incorrectamente por un clasificador, pueden estar correctamente clasificados por otro. Conseguir diversidad en los clasificadores se logra mediante la introducción de procesos aleatorios en el conjunto de datos.

- Los clasificadores individuales no deben ser demasiado estables, dado que los conjuntos de clasificadores están justamente diseñados para mejorar el rendimiento de los clasificadores individuales y si aquellos son estables su desempeño no mejorará o incluso en otros casos empeorará.

Los conjuntos de clasificadores comúnmente usados por su simplicidad y su potente capacidad de clasificación se presentan a continuación.

2.3.1. Bagging

Bagging (de agregación bootstrap) es una técnica propuesta por Breiman (1996a, 1996b). Se puede utilizar para mejorar tanto la estabilidad como el poder predictivo de los árboles de clasificación y regresión, pero su uso no se limita a mejorar las predicciones basadas en árboles. Es una técnica general que se puede aplicar en una amplia variedad de entornos para mejorar las predicciones.

Método

Para comprender por qué funciona el método Bagging y para determinar en qué situaciones se puede esperar una mejora apreciable de precisión, es útil considerar el problema de predecir el valor de una variable respuesta numérica Y_x, que resultará de la ocurrencia de un conjunto de entradas x. Suponga que $\phi(x)$ es la predicción que resulta de usar un método particular, como CART [7], o regresión por MCO[8] con un método prescrito para la selección del modelo. Se define μ_ϕ como $E(\phi(x))$, donde la esperanza es con respecto a la distribución subyacente a la muestra de aprendizaje

[7] Classification and Regression Tree (CART) es un modelo predictivo que explica cómo se pueden predecir los valores de una variable de resultado en función de otros valores.

[8] Es el nombre de un método para encontrar los parámetros poblacionales en un modelo de regresión lineal. Este método minimiza la suma de las distancias verticales entre las respuestas observadas en la muestra y las respuestas del modelo

(dado que, visto como una variable aleatoria, $\phi(x)$ es una función de la muestra de aprendizaje, que puede verse como una variable aleatoria de alta dimensión) y no de x (que se considera fija), tenemos que

$$\begin{aligned}
&E\left([Y_x - \phi(x)]^2\right) \\
&= E\left([(Y_x - \mu_\phi) + (\mu_\phi - \phi(x))]^2\right) \\
&= E\left([Y_x - \mu_\phi]^2\right) + 2E\left(Y_x - \mu_\phi\right) E\left(\mu_\phi - \phi(x)\right) + E\left([\mu_\phi - \phi(x)]^2\right) \\
&= E\left([Y_x - \mu_\phi]^2\right) + E\left([\mu_\phi - \phi(x)]^2\right) \\
&= E\left([Y_x - \mu_\phi]^2\right) + Var(\phi(x)) \\
&\geq E\left([Y_x - \mu_\phi]^2\right)
\end{aligned} \tag{2.9}$$

(Arriba, se usa la independencia de la respuesta futura, Y_x, y el predictor basado en la muestra de aprendizaje, $\phi(x)$.) Dado que en situaciones no triviales, la varianza del predictor $\phi(x)$ es positiva (ya que normalmente no todas las muestras aleatorias que podrían ser la muestra de aprendizaje producen el valor de la muestra para la predicción), por lo que la desigualdad anterior es estricta, este resultado nos da que si $\mu_\phi = E(\phi(x))$ pudiera usarse como predictor, tendría un error de predicción cuadrático medio más pequeño que $\phi(x)$.

Por supuesto, en aplicaciones típicas, μ_ϕ no puede servir como predictor, ya que no se conoce la información necesaria para obtener el valor de $E(\phi(x))$. Para obtener lo que a veces se denomina la verdadera estimación *bagging* de $E(\phi(x))$, la esperanza se basa en la distribución empírica correspondiente a la muestra de aprendizaje. En principio, es posible obtener este valor, pero en la práctica suele ser demasiado difícil de obtener con sensatez, por lo que se considera que la predicción *bagging* de Y_x es

$$\frac{1}{B}\sum_{b=1}^{B} \phi_b^*(x) \tag{2.10}$$

donde $\phi_b^*(x)$ es la predicción obtenida cuando el método de regresión base (por ejemplo, CART) se aplica a la b-ésima muestra bootstrap extraída (con reemplazo) de la muestra de aprendizaje original. Es decir, para utilizar *bagging* para obtener una predicción de Y_x en una configuración de regresión, se elige un método de regresión (al que se hace referencia como método base) y se aplica el método a las B muestras bootstrap extraídas de la muestra de aprendizaje. Los valores de predic-

ción B obtenidos se promedian luego para producir la predicción final.

En los problemas de clasificación, las muestras de bootstrap B se extraen de la muestra de aprendizaje y se aplica un método de clasificación específico (por ejemplo, CART) a cada muestra de bootstrap para obtener una clase predicha para una entrada determinada, x. La predicción final (la que resulta de usar *bagging* con el método base especificado) es la clase que ocurre con mayor frecuencia en las predicciones B. Un esquema alternativo es empaquetar estimaciones de probabilidad de clase para cada clase y luego dejar que la clase pronosticada sea la que tenga la mayor probabilidad promedio estimada. Por ejemplo, con árboles de clasificación, uno tiene una clase predicha correspondiente a cada nodo terminal, pero también hay una estimación de la probabilidad de que un caso que tenga x correspondiente a un nodo terminal específico pertenezca a una clase particular. Existe una estimación de este tipo para cada clase, y para predecir la clase para x, estas probabilidades estimadas de los B árboles se pueden promediar y se puede elegir la clase correspondiente a la mayor probabilidad promedio estimada. Esto puede producir un resultado diferente al que se obtiene mediante una simple votación. Ninguno de los dos métodos funciona mejor en todos los casos. Hastie et al. (2009) sugieren que promediar las probabilidades tiende a ser mejor para un B pequeño, pero también incluye un ejemplo en el que el método de votación funciona ligeramente mejor con un valor grande de B.

2.3.2. Boosting

Boosting es un método para combinar clasificadores, que se crean iterativamente a partir de versiones ponderadas de la muestra de aprendizaje, con los pesos ajustados de forma adaptativa en cada paso para dar mayor peso a los casos que se clasificaron erróneamente en el paso anterior. Las predicciones finales se obtienen ponderando los resultados de los predictores producidos de forma iterativa.

El método *Boosting* se desarrolló originalmente para la clasificación y, por lo general, se aplica a aprendices débiles [9]. Para un clasificador de dos clases, un aprendiz débil es un clasificador que puede ser solo un poco mejor que la elección aleatoria.

[9] Mejor conocidos por su nombre en inglés *weak learners*, son algoritmos de clasificación o regresión que tienen un desempeño relativamente malo pero que al ser agrupados producen un clasificador fuerte que generalmente presenta un mejor rendimiento.

Dado que la elección aleatoria tiene una tasa de error de 0.5, un clasificador débil solo tiene que predecir correctamente, en promedio, un poco más del 50 % de los casos. Un ejemplo de clasificador débil es un árbol de dos nodos (stumps por su nombre en inglés). (En algunos entornos, incluso un clasificador tan simple puede tener una tasa de error bastante pequeña [10]. Pero en otros entornos, este árbol puede tener una tasa de error de casi 0.5, por lo que estos árboles son considerados como aprendices débiles). Sin embargo, el método *boosting* no se limita a ser utilizado con aprendices débiles. Se puede usar con clasificadores que son bastante precisos, como árboles cuidadosamente cultivados y podados, que sirven como aprendiz básico. Hastie et al. (2009) afirman que el uso de árboles con entre cuatro y ocho nodos terminales funciona bien en la mayoría de los casos, y que el rendimiento es bastante indiferente a la elección de este rango. Friedman et al. (2000) dan un ejemplo en el que usar muchos *stumps* en el método *boosting* es considerablemente peor que usar un solo árbol grande, pero utilizar el método con árboles de ocho nodos produjo una tasa de error que es menos del 25 por ciento de la tasa de error de un solo árbol grande, lo que demuestra que la elección de lo que se usa como clasificador base puede marcar una gran diferencia, y que la opción popular de aumentar los *stumps* puede estar lejos de ser la óptima. Sin embargo, también dan otro ejemplo en el que los *stumps* son superiores a los árboles más grandes. Como estrategia general, se pueden elegir *stumps* si se sospecha que los efectos son aditivos y elegir árboles más grandes si se anticipan interacciones para las que se deben hacer ajustes.

Algoritmo AdaBoost

AdaBoost es un algoritmo de *boosting* desarrollado por Freund y Schapire (1996) para ser utilizado con clasificadores. Hay dos versiones de AdaBoost.

AdaBoost.M1 primero pide que se cree un clasificador utilizando la muestra de aprendizaje (es decir, el aprendiz base se ajusta a la muestra de aprendizaje), y a cada caso se le da el mismo peso. Si la muestra de aprendizaje consta de N casos, el peso inicial de todos ellos será $1/N$. Las ponderaciones utilizadas para cada paso posterior dependen de la tasa de error ponderada del clasificador creado en el paso inmediatamente anterior, y los casos que se clasifican erróneamente en un paso dado reciben mayor peso en el siguiente paso.

Específicamente, si $I_{m,n}$ es igual a 1 si el n-ésimo caso está mal clasificado en el

[10] Esto sucede cuando las clases se pueden separar casi con una sola división

m-ésimo paso, e igual a 0 en caso contrario, y $w_{m,n}$ es el peso del n-ésimo caso para el m-ésimo paso, donde los pesos son positivos y suman 1, para la tasa de error ponderada para el m-ésimo paso, e_m, tenemos

$$e_m = \sum_{n=1}^{N} w_{m,n} I_{m,n} \tag{2.11}$$

Los pesos para el $(m+1)$-ésimo paso se obtienen a partir de los pesos del m-ésimo paso, multiplicando los pesos de los casos correctamente clasificados por e_m / $(1 - e_m)$, que debe ser menor que 1, y luego multiplicando todo el conjunto de valores por un número mayor que 1 de tal manera que la suma de los N valores de 1. Este procedimiento da menor peso a los casos que se clasificaron correctamente, lo que equivale a dar mayor peso a los casos que se clasificaron erróneamente. Para el segundo paso, solo se usarán dos valores para los pesos, pero a medida que continúan las iteraciones, los casos que a menudo se clasifican correctamente pueden tener pesos mucho más pequeños que los casos que son los más difíciles de clasificar correctamente. Si el método de clasificación del aprendiz débil no permite casos ponderados, a partir del segundo paso en adelante, se extrae una muestra aleatoria de los casos de muestra de aprendizaje originales para que sirva como muestra de aprendizaje para ese paso, con selecciones independientes realizadas utilizando las ponderaciones como las probabilidades de selección.

El hecho de que a los casos mal clasificados en un paso dado se les dé un mayor peso en el siguiente paso normalmente conduce a un ajuste diferente del aprendiz base, que clasifica correctamente algunos de los casos mal clasificados del paso anterior, lo que contribuye a una baja correlación de las predicciones de un paso y el siguiente. Amit et al. (1999) proporcionan algunos detalles que indican que el algoritmo AdaBoost intenta producir una baja correlación entre las predicciones de los clasificadores base ajustados, y algunos creen que este fenómeno es un factor importante en el éxito de AdaBoost (ver la discusión de Breiman en Friedman et al. (2000) y Breiman (2001, p. 20)).

Suponiendo que la tasa de error ponderada para cada paso no es mayor que 0.5, el procedimiento *boosting* crea M clasificadores utilizando los pesos ajustados iterativamente. Se han utilizado valores como 10 y 100 para M. Pero en muchos casos será mejor utilizar un valor mayor, como 200, ya que el rendimiento a menudo continúa mejorando a medida que M se acerca a 200 o 400, y es raro que el rendimiento mejore si M se hace demasiado grande. Los M clasificadores luego se combinan

para obtener un solo clasificador para usar. Si la tasa de error excede 0.5, lo que no debería ocurrir con solo dos clases, pero podría ocurrir si hay más de 2 clases, las iteraciones se terminan y solo se combinan los clasificadores que tienen tasas de error menores a 0.5. En cualquier caso, los clasificadores se combinan mediante el uso de una votación ponderada para asignar la clase predicha para cualquier entrada x, y el clasificador creado en el m-ésimo paso recibe el peso $\log((1 - e_m)/e_m)$. Esto da la mayor ponderación a los clasificadores que tienen las tasas de error de resustitución ponderadas más bajas. Aumentar el número de iteraciones para AdaBoost.M1 lleva a cero la tasa de error de entrenamiento del clasificador compuesto. No es necesario que la tasa de error de una muestra de prueba independiente se acerque a cero, pero en muchos casos puede acercarse mucho a la tasa de error más pequeña posible.

La discusión de Breiman en Friedman et al. (2000) señala que después de un gran número de iteraciones, si se examina la proporción ponderada de veces que cada uno de los casos de muestra de aprendizaje se ha clasificado erróneamente, los valores tienden a ser casi iguales. Entonces, en lugar de afirmar que AdaBoost se concentra en los casos difíciles de clasificar, como han hecho algunos, es más exacto afirmar que AdaBoost otorga (casi) la misma importancia a la clasificación correcta de cada uno de los casos en la muestra de aprendizaje. Hastie et al. (2009) muestran que AdaBoost.M1 es equivalente al modelado aditivo progresivo por etapas utilizando una función de pérdida exponencial.

AdaBoost.M2 es equivalente a AdaBoost.M1 si solo hay dos clases, pero parece ser mejor que AdaBoost.M1 (ver resultados en Freund y Schapire, 1996) si hay más de dos clases, porque para que AdaBoost.M1 funcione bien, los aprendices débiles deberían tener tasas de error de resustitución ponderadas inferiores a 0.5, y esto puede ser bastante difícil de lograr si hay muchas clases. Los estudiantes débiles usados con AdaBoost.M2 asignan a cada caso un conjunto de valores plausibles para las clases posibles. También hacen uso de una función de pérdida que puede asignar diferentes penalizaciones a diferentes tipos de clasificaciones erróneas. Los valores de la función de pérdida utilizados se actualizan para cada iteración para poner más énfasis en evitar los mismos tipos de clasificaciones erróneas que ocurrieron con mayor frecuencia en el paso anterior. Una vez completadas las iteraciones, para cada x de interés se utiliza la secuencia de clasificadores para producir un promedio ponderado de las plausibilidades para cada clase, y la clase correspondiente al mayor de estos valores se toma como la predicción de x. Se pueden obtener más detalles

relacionados con AdaBoost.M2 en Freund y Schapire (1996). En nuestro trabajo trabajaremos con **AdaBoost.M1** cuyo procedimiento de calculo se detalla en Algoritmo 1.

Algoritmo 1 AdaBoost.M1

1: Inicializar los pesos de cada observación $w_i = \frac{1}{N}, \quad i = 1, 2, ..., N$
2: **para** $m = 1$ hasta M **hacer**
3: Ajustar un clasificador débil $G_m(x)$ a los datos de entrenamiento usando pesos w_i
4: Calcular

$$e_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}$$

5: Calcular $\alpha_m = \log((1 - e_m)/e_m)$
6: Fijar $w_i \longleftarrow w_i \exp[\alpha_m I(y_i \neq G_m(x_i))] \quad$ para $i = 1, 2, ..., N$
7: **fin para**
8: **devolver** $G(x) = sign\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$

2.3.3. Árboles Boosting

Los árboles de regresión y clasificación analizados en el apartado 2.2.1. Dividen el espacio de todos los valores de las variables predictoras conjuntas en regiones disjuntas $R_j; j = 1, 2, ..., J$, representado por los nodos terminales del árbol. Se asigna una constante γ_j a cada una de estas regiones y la regla predictiva es

$$x \in R_j \Rightarrow f(x) = \gamma_j$$

Por tanto, un árbol puede expresarse formalmente como

$$T(x; \Theta) = \sum_{j=1}^{J} \gamma_j I(x \in R_j) \tag{2.12}$$

con parámetros $\Theta = \{R_j, \gamma_j\}_1^J$. Donde J es usualmente tratado como hiperparámetro. Los parámetros son encontrados al minimizar la función de costo

$$\hat{\Theta} = \arg\min_{\Theta} \sum_{j=1}^{J} \sum_{x_i \in R_j} L(y_i, \gamma_j) \tag{2.13}$$

Este es un formidable problema de optimización combinatoria, y generalmente

nos conformamos con soluciones subóptimas aproximadas. Es útil dividir el problema de optimización en dos partes:

- **Encontrar γ_j dado R_j:** Dado el valor de R_j, estimar el γ_j es típicamente trivial y, a menudo, $\hat{\gamma}_j = \bar{y}_j$, el promedio de los y_i que quedan en la región R_j. Para la función de pérdida de clasificación errónea, $\hat{\gamma}_j$ es la clase modal de las observaciones que caen en la región R_j.

- **Encontrar R_j:** Ésta es la parte difícil, para la que se encuentran soluciones aproximadas. Debemos tomar en cuenta que encontrar el R_j implica estimar el valor γ_j también. Una estrategia típica es utilizar un algoritmo codicioso de particionamiento recursivo de arriba hacia abajo para encontrar el R_j. Además, a veces es necesario aproximar 2.13 mediante un criterio más suave y conveniente para optimizar el R_j:

$$\tilde{\Theta} = \arg\min_{\Theta} \sum_{i=1}^{N} \tilde{L}\left(y_i, T(x_i;\Theta)\right) \tag{2.14}$$

  Entonces, dado el valor $\hat{R}_j = \tilde{R}_j$, γ_j puede ser estimado de una manera más precisa usando el criterio original.

En la sección 2.2.1 se explicó una estrategia de este tipo para árboles de clasificación. El índice de Gini reemplaza a la función de pérdida por clasificación errónea en el crecimiento del árbol (identificando el R_j). El modelo de árbol *boosting* es una suma de tales árboles,

$$f_M(x) = \sum_{m=1}^{M} T(x;\Theta_m) \tag{2.15}$$

inducido de manera progresiva por etapas (algoritmo 3, apéndice A). En cada paso del procedimiento progresivo por etapas, uno debe resolver

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} L\left(y_i, f_{m-1}(x_i) + T(x_i;\Theta_m)\right) \tag{2.16}$$

para el conjunto de regiones y constantes $\Theta_m = \{R_{jm}, \gamma_{jm}\}_1^{J_m}$ del siguiente árbol, dado el modelo actual $f_{m-1}(x)$. Dadas las regiones R_{jm}, encontrar las constantes

óptimas γ_{jm} en cada región es típicamente sencillo:

$$\hat{\gamma}_{jm} = \arg\min_{\gamma_{jm}} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma_{jm}\right) \tag{2.17}$$

Encontrar las regiones es difícil, e incluso más difícil que para un solo árbol. Para la función de pérdida de error al cuadrado, la solución de 2.16 no es más difícil que para un solo árbol. Es simplemente el árbol de regresión que mejor predice los residuos actuales $y_i - f_{m-1}(x_i)$, y γ_{jm} es la media de estos residuos en cada región correspondiente.

Para problemas de clasificación con dos clases y una función de pérdida exponencial, este enfoque por etapas da lugar al método *boosting* para los árboles de clasificación **AdaBoost** (algoritmo 1). En particular, si los árboles $T(x;\Theta_m)$ están restringidos a ser árboles de clasificación a escala, se tiene que la solución a 2.16 es el árbol que minimiza la tasa de error ponderada $\sum_{i=1}^{N} w_i^{(m)} I(y_i \neq T(x_i;\Theta_m))$ con los pesos $w_i^{(m)} = e^{-y_i f_{m-1}(x_i)}$. Como un árbol de clasificación a escala, nos referimos a $\beta_m T(x;\Theta_m)$, con la restricción de que $\gamma_{jm} \in \{-1, 1\}$. Sin esta restricción, 2.16 aun podría simplificarse para la pérdida exponencial a un criterio exponencial ponderado para el nuevo árbol:

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} w_i^{(m)} \exp\left[-y_i T(x_i;\Theta_m)\right] \tag{2.18}$$

Es sencillo implementar un algoritmo codicioso de partición recursiva utilizando esta función de pérdida exponencial ponderada como criterio de división. Dado el R_{jm}, se puede demostrar que la solución de 2.17 son las probabilidades logarítmicas ponderadas en cada región correspondiente

$$\hat{\gamma}_{jm} = \log \frac{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = 1)}{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = -1)} \tag{2.19}$$

Esto requiere un algoritmo especializado de crecimiento de árboles; en la práctica, se utiliza la aproximación que se presenta a continuación, que utiliza un árbol de regresión de mínimos cuadrados ponderados.

El uso de criterios de pérdida como el error absoluto o la función de pérdida

de Huber (apéndice A.1) en lugar de la pérdida por error al cuadrado para la regresión, y la desviación (apéndice A.2) en lugar de la pérdida exponencial para la clasificación, servirá para robustecer los árboles boosting. Desafortunadamente, a diferencia de sus contrapartes no robustas, estos criterios sólidos no dan lugar a algoritmos boosting simples y rápidos. Para criterios de pérdida más generales, la solución a 2.17, dado el R_{jm}, suele ser sencilla, ya que es una estimación simple de "localización". Para la pérdida absoluta, es solo la mediana de los residuos en cada región respectiva. Para los otros criterios, existen algoritmos iterativos rápidos para resolver 2.17, y por lo general sus aproximaciones más rápidas de "paso único" son adecuadas. El problema es la inducción de árboles. No existen algoritmos rápidos simples para resolver 2.16 para estos criterios de pérdida más generales, y aproximaciones como 2.14 se vuelven esenciales.

2.3.4. Optimización numérica a través del Gradient Boosting

Los algoritmos de aproximación rápida para resolver 2.16 con cualquier función de pérdida diferenciable pueden derivarse por analogía con la optimización numérica. La función de pérdida al usar $f(x)$ para predecir y en los datos de entrenamiento es

$$L(f) = \sum_{i=1}^{N} L\left(y_i, f(x_i)\right) \tag{2.20}$$

La meta es minimizar $L(f)$ con respecto a f, donde aquí $f(x)$ se limita a ser una suma de árboles 2.15. Ignorando esta restricción, minimizar 2.20 puede verse como una optimización numérica

$$\hat{f} = \arg\min_{f} L(f) \tag{2.21}$$

donde los 'parámetros' $f \in \mathbb{R}^N$ son los valores de la función de aproximación $f(x_i)$ en cada uno de los N puntos de datos x_i

$$f = \{f(x_1), f(x_2), , f(x_N)\}$$

Los procedimientos de optimización numérica resuelven 2.21 como una suma de los vectores componentes

$$f_M = \sum_{m=0}^{M} h_m, \quad h_m \in \mathbb{R}^N$$

donde $f_0 = h_0$ es una suposición inicial, y cada elemento de la sucesión f_m se induce basándose en el vector de parámetro actual f_{m-1}, que es la suma de las actualizaciones inducidas previamente. Los métodos de optimización numérica difieren en sus prescripciones para calcular cada vector de incremento h_m ("paso").

Método del descenso más profundo

El método del descenso más profundo elige $h_m = \rho_m g_m$ donde ρ_m es un escalar y $g_m \in \mathbb{R}^N$ es el gradiente de $L(f)$ evaluado en $f = f_{m-1}$. Los componentes del gradiente g_m son

$$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{m-1}(x_i)} \tag{2.22}$$

La longitud del paso ρ_m es la solución de

$$\rho_m = \arg\min_{\rho} L(f_{m-1} - \rho g_m) \tag{2.23}$$

A continuación, se actualiza la solución actual de la siguiente manera

$$fm = f_{m-1} - \rho_m g_m$$

y el proceso se repite en la siguiente iteración. El método del descenso más profundo puede verse como una estrategia muy ávida, ya que $-g_m$ es la dirección local en $\mathbb{R}^N$ para la cual $L(f)$ está disminuyendo más rápidamente en $f = f_{m-1}$.

Gradient Boosting

El impulso progresivo por etapas (Algoritmo 3) también es una estrategia muy codiciosa. En cada paso, el árbol de solución es el que reduce al máximo 2.16, dado el modelo actual f_{m-1} y su ajuste $f_{m-1}(xi)$. Por tanto, las predicciones del árbol $T(x_i; \Theta_m)$ son análogas a las componentes del gradiente negativo 2.22. La principal diferencia entre ellos es que los componentes del árbol $t_m = (T(x_1; \Theta_m), ..., T(x_N; \Theta_m))$ no son independientes. Están limitados a ser las predicciones de un nodo terminal

J_m de un árbol de decisión, mientras que el gradiente negativo es la dirección máxima de descenso sin restricciones.

La solución a 2.17 en la aproximación por etapas es análoga a la búsqueda lineal 2.23 en el método de descenso más profundo. La diferencia es que 2.17 realiza una búsqueda lineal separada para aquellos componentes de t_m que corresponden a cada región terminal separada $\{T(x_i;\Theta_m)\}_{x_i \in R_{jm}}$.

Si minimizar la función de pérdida en los datos de entrenamiento 2.20 fuera el único objetivo, el método del descenso más profundo sería la estrategia preferida. El gradiente 2.22 es trivial de calcular para cualquier función de pérdida diferenciable $L(y, f(x))$, mientras que resolver 2.16 es difícil. Desafortunadamente, el gradiente 2.22 se define solo en los puntos de datos de entrenamiento x_i, mientras que el objetivo final es generalizar $f_M(x)$ a nuevos datos no representados en el conjunto de entrenamiento.

Una posible solución a este dilema es inducir un árbol $T(x;\Theta_m)$ en la m-ésima iteración cuyas predicciones t_m estén lo más cerca posible del gradiente negativo. Usando el error al cuadrado para medir la cercanía, esto nos lleva a

$$\tilde{\Theta}_m = \arg\min_{\Theta} \sum_{i=1}^{N} \left(-g_{im} - T(x_i;\Theta)\right)^2 \tag{2.24}$$

Es decir, se ajusta el árbol T a los valores de gradiente negativos 2.22 por mínimos cuadrados. Como se señaló en el apartado 2.3.3, existen algoritmos rápidos para la inducción del árbol de decisión de mínimos cuadrados. Aunque las regiones de solución $\tilde{R}_{jm}$ a 2.24 no serán idénticas a las regiones R_{jm} que resuelven 2.16, generalmente es lo suficientemente similar para cumplir el mismo propósito. En cualquier caso, el método *boosting* progresivo por etapas y la inducción del árbol de decisión de arriba hacia abajo son en sí mismos procedimientos de aproximación. Después de construir el árbol 2.24, las constantes correspondientes en cada región vienen dadas por 2.17.

Cuadro 2.1: Gradientes para funciones de pérdida usadas comúnmente

Tipo	Función de pérdida	$\frac{\partial L(y_i, f(x_i))}{\partial F(x_i)}$
Regresión	$\frac{1}{2}[y_i - f(x_i)]^2$	$y_i - f(x_i)$
Regresión	$\lvert y_i - f(x_i)\rvert$	$sign[y_i - f(x_i)]$
Regresión	Huber (A.1)	$y_i - f(x_i)$ para $\lvert y_i - f(x_i)\rvert \leq \delta_m$ $\delta_m sign[y_i - f(x_i)]$ para $\lvert y_i - f(x_i)\rvert \geq \delta_m$ donde $\delta_m = \alpha$th cuantil$\{\lvert y_i - f(x_i)\rvert\}$
Clasificación	Desviación (A.2)	kth componente: $I(y_i = G_k) - p_k(x_i)$

Fuente: (Hastie, Tibshirani y Friedman, 2009)

La tabla 2.1 resume los gradientes para las funciones de pérdida de uso común. Para la pérdida de error al cuadrado, el gradiente negativo es solo el residuo ordinario $g_{im} = y_i - f_{m-1}(xi)$, de modo que 2.24 por sí solo equivale a un método *boosting* estándar de mínimos cuadrados. Con la pérdida de error absoluto, el gradiente negativo es el signo del residual, por lo que en cada iteración de 2.24 se ajusta el árbol al signo de los residuales actuales por mínimos cuadrados. Para la regresión M de Huber, el gradiente negativo es un compromiso entre estos dos (ver la tabla). Para la clasificación, la función de pérdida es la desviación multinomial A.2 y se construyen K árboles de mínimos cuadrados en cada iteración. Cada árbol T_{km} es ajustado a su respectivo vector de gradiente negativo g_{km},

$$\begin{aligned} -g_{ikm} &= \frac{\partial L\left(y_i, f_{1m}(x_i), ..., f_{1m}(x_i)\right)}{\partial f_{km(x_i)}} \\ &= I(Y_i = G_k) - p_k(x_i) \end{aligned} \tag{2.25}$$

con $p_k(x)$ dado por A.3. Aunque se construyen K árboles separados en cada iteración, están relacionados a través de A.3. Para clasificación binaria, solo se necesita un árbol.

2.3.5. Implementación por Gradient Boosting

El algoritmo 2 presenta el algoritmo genérico del gradiente para árboles de regresión con *boosting*. Los algoritmos específicos se obtienen insertando diferentes criterios de pérdida $L(y, f(x))$. La primera línea del algoritmo inicializa en el mo-

delo la constante óptima, que es solo un árbol de nodo terminal único. Los componentes del gradiente negativo calculados en la línea 4 se denominan generalizados o pseudo-residuos, r. Los gradientes de las funciones de pérdida de uso común se resumen en la tabla 2.1.

Algoritmo 2 Gradient Tree Boosting Algorithm

1: Inicializar $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$
2: **para** $m = 1$ hasta M **hacer**
3: **para** $i = 1, 2, ..., N$ **hacer**
4: calcular

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$

5: Ajustar un árbol de regresión para los objetivos r_{im} dadas las regiones terminales R_m, $j = 1, 2, ..., J_m$
6: **para** $j = 1, 2, ..., J_m$ **hacer**
7: calcular

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$

8: **fin para**
9: Actualizar $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$
10: **fin para**
11: **fin para**
12: **devolver** $\hat{f}(x) = F_M(x)$

El algoritmo para cada árbol de clasificación es similar. Desde la línea 3 hasta la 10 son repetidas K veces en cada iteración m, una por cada clase si se usa 2.25. El resultado de la línea 12 son K diferentes árboles de expansión $f_{kM}(x)$, $k = 1, 2, ..., K$. Esto produce probabilidades si se utiliza A.3 o hace clasificaciones en el caso de usar A.4. Dos parámetros de ajuste básicos son el número de iteraciones M y los tamaños de cada uno de los árboles constituyentes J_m, $m = 1, 2, ..., M$.

2.3.6. Importancia de variables

Los árboles de decisión únicos son muy interpretables. El modelo se puede representar completamente mediante un gráfico bidimensional simple (árbol binario) que se visualiza fácilmente. Las combinaciones lineales de árboles 2.15 pierden esta importante característica y, por lo tanto, deben interpretarse de manera diferente.

Importancia relativa de las variables predictoras

En las aplicaciones de minería de datos [11], las variables predictoras de entrada rara vez son igualmente relevantes. A menudo, solo algunas de ellas tienen una influencia sustancial en la respuesta; la gran mayoría son irrelevantes y bien podrían no haber sido incluidas. A menudo es útil aprender la importancia relativa o la contribución de cada variable de entrada para predecir la respuesta. Para un árbol de decisión simple T, Breiman et al. (1984) proponen

$$I_l^2(T) = \sum_{t=1}^{J-1} \hat{i}_t^2 I(v(t) = l) \tag{2.26}$$

como medida de relevancia para cada variable predictora X_l. La suma está definida sobre los $J-1$ nodos internos del árbol. En cada uno de estos nodos t, se utiliza una de las variables de entrada $X_{v(t)}$ para dividir la región asociada con ese nodo en dos subregiones; dentro de cada uno se ajusta una constante que separa a los valores respuesta. La variable particular elegida es la que da una máxima mejoría estimada $\hat{i}_i^2$ en la función de pérdida de errores al cuadrado sobre el error de toda la región. La importancia relativa al cuadrado de la variable X_l es la suma de dichas mejoras al cuadrado sobre todos los nodos internos para los que se eligió como variable de división.

Esta medida de importancia se generaliza fácilmente a las expansiones de árboles aditivos 2.15; simplemente se promedia sobre los árboles

$$I_l^2(T) = \frac{1}{M} \sum_{m=1}^{M} I_l^2(T_m) \tag{2.27}$$

Debido al efecto estabilizador de promediar, esta medida resulta ser más confiable que su contraparte 2.26 para un solo árbol.

2.3.7. Algoritmo XGBoost

XGBoost significa eXtreme Gradient Boosting, fue implementado por Tianqi Chen y hoy en día es soportado por una numerosa comunidad de desarrolladores.

[11] La minería de datos es un proceso de extracción y descubrimiento de patrones en grandes conjuntos de datos que involucran métodos en la intersección del aprendizaje automático, las estadísticas y los sistemas de bases de datos.

> The name xgboost, though, actually refers to the engineering goal to push the limit of computations resources for boosted tree algorithms. Which is the reason why many people use xgboost.
>
> (*Tianqi Chen* en Quora.com, 2019)

Es una implementación del algoritmo Gradient Boosting detallado en la sección 2.3.5 y pertenece a una colección más amplia de herramientas bajo el paraguas de la Comunidad Distribuida de Aprendizaje Automático o DMLC [12], que también son los creadores de la popular biblioteca de aprendizaje profundo *mxnet*. Tianqi Chen ofrece una breve e interesante historia de fondo sobre la creación de XGBoost en el tutorial Historia y lecciones detrás de la evolución de XGBoost2. Específicamente, XGBoost admite las siguientes interfaces principales:

- Command Line Interface (CLI).
- C++ (el lenguaje en el que está escrito la librería).
- Python
- *R*
- Julia
- Java y lenguajes JVM como Scala y plataformas como Hadoop.

Características del modelo

Incorpora tres formas principales de gradient boosting:

- El algoritmo de aumento de gradiente también se llama máquina de aumento de gradiente, incluida la tasa de aprendizaje.
- Stochastic Gradient Boosting con submuestreo en la fila, columna y columna por subniveles.
- Regularized Gradient Boosting con regularizaciones de tipo L^1 y L^2

[12] DMLC es una comunidad que aloja y desarrolla bibliotecas de código abierto portátiles, escalables y confiables para aprendizaje automático distribuido

Características del Sistema

La librería provee un sistema para su uso en una variedad de entornos informáticos, entre los que se incluyen:

- Paralelización de la construcción de árboles utilizando todos los núcleos de su CPU durante el entrenamiento.
- Computación distribuida para entrenar modelos muy grandes usando un grupo de máquinas.
- Computación fuera del núcleo para conjuntos de datos muy grandes que no se almacenan en la memoria.
- Optimización de caché de estructuras de datos y algoritmos para aprovechar al máximo el hardware.

Características del algoritmo

La implementación del algoritmo fue diseñada para la eficiencia del tiempo de cómputo y los recursos de memoria. Un objetivo de diseño era hacer el mejor uso de los recursos disponibles para entrenar al modelo. Algunas características clave de implementación de algoritmos incluyen:

- Implementación de *Sparse Aware* con manejo automático de valores de datos faltantes.
- Estructura de bloques para apoyar la paralelización de la construcción de árboles.
- Capacitación continua para que pueda impulsar aún más un modelo ya ajustado con nuevos datos.

¿Por qué usar XGBoost

Hay dos razones principales para usar este algoritmo, las cuales se detallan a continuación

1. **Ejecución rápida** En general, XGBoost es realmente rápido en comparación con otras implementaciones de *Gradient Boosting*. Szilard Pafka realizó algunos

puntos de referencia objetivos comparando el rendimiento de XGBoost con otras implementaciones de *Gradient Boosting* y *bagged decision trees.*. Escribió sus resultados en mayo de 2015 en el tutorial del blog titulado Benchmarking Random Forest Implementations[13]. También proporciona todo el código en GitHub [14] y un informe más extenso de resultados con números concretos. Sus resultados mostraron que XGBoost fue casi siempre más rápido que las otras implementaciones de R, Python Spark y H2O.

2. **Rendimiento del modelo** XGBoost domina los problemas de modelación predictiva con conjuntos de datos estructurados o tabulares sobre problemas de clasificación y regresión. La evidencia es que es el algoritmo de referencia para los ganadores de la competencia en la plataforma de ciencia de datos competitiva de Kaggle. Por ejemplo, hay una lista incompleta de ganadores de la competencia de primer, segundo y tercer lugar que se tituló: *Machine Learning Challenge Winning Solutions* [15]

XGBoost es un algoritmo que recientemente ha dominado el aprendizaje automático aplicado y las competencias de Kaggle[16] para datos estructurados o tabulares. XGBoost es una implementación de árboles de decisión impulsados por gradiente diseñados para la velocidad y el rendimiento.

XGBoost es un software gratuito de código abierto disponible para su uso bajo la licencia permisiva Apache-2 [17].

Optimización de hiperparámetros

En el aprendizaje automático, la optimización o el ajuste de hiperparámetros es el problema de elegir un conjunto o combinación de hiperparámetros óptimos para un algoritmo de aprendizaje. Un hiperparámetro es un parámetro cuyo valor se

[13] `http://datascience.la/benchmarking-random-forest-implementations/`

[14] `https://github.com/szilard/benchm-ml`

[15] `ttps://github.com/dmlc/xgboost/tree/master/demo`

[16] Kaggle es una plataforma para competiciones de modelado predictivo y análisis en la que empresas e investigadores publican datos y los estadísticos y mineros de datos compiten para producir los mejores modelos para predecir y describir los datos. Puedes competir en muchos problemas

[17] La misión de la Apache Software Foundation (ASF) es proporcionar software para el bien público y la versión 2.0 de la Licencia Apache, aprobada por la ASF en 2004, tiene el objetivo de proporcionar productos de software confiables y duraderos a través del desarrollo colaborativo de software de código abierto.

utiliza para controlar el proceso de aprendizaje. Por el contrario, se aprenden los valores de otros parámetros (normalmente ponderaciones de los nodos).

El mismo tipo de modelo de aprendizaje automático puede requerir diferentes restricciones, pesos o tasas de aprendizaje para generalizar diferentes patrones de datos. Estas medidas se denominan hiperparámetros y deben ajustarse para que el modelo pueda resolver de manera óptima el problema del aprendizaje automático. La optimización de hiperparámetros encuentra una tupla de hiperparámetros que produce un modelo óptimo que minimiza una función de pérdida predefinida en datos de entrenamiento. La función objetivo toma una tupla de hiperparámetros y devuelve la pérdida asociada. La validación cruzada se utiliza a menudo para estimar el rendimiento de esta generalización.

Gradient Boosting es una de las técnicas más poderosas para el aprendizaje automático aplicado y, como tal, se está convirtiendo rápidamente en una de las más populares. Pero, ¿cómo se debería configura el algoritmo a nuestro problema?. En el artículo de 1999 "Greedy Function Approximation: A Gradient Boosting Machine", Jerome Friedman comenta sobre la compensación entre el número de árboles (M) y la tasa de aprendizaje (v). Sugiere establecer primero un valor grande para el número de árboles y luego ajustar el parámetro de tasa de aprendizaje (*shrinkage*) para lograr los mejores resultados. Los estudios en el artículo prefirieron una tasa de aprendizaje de 0.1, un número de árboles en el rango de 100 a 500 y el número de nodos terminales en un árbol entre 2 y 8.

En el artículo de 1999 "Stochastic Gradient Boosting", Friedman reiteró la preferencia por la tasa de aprendizaje. Introduce e investiga empíricamente el modelo *stochastic gradient boosting* (submuestreo basado en filas). Encuentra que casi todos los porcentajes de submuestreo son mejores que el llamado refuerzo determinista y quizás 30 % a 50 % es un buen valor para elegir en algunos problemas y 50 % a 80 % en otros.

También estudió el efecto del número de nodos terminales en árboles y encontró valores como 3 y 6 mejores que valores más grandes como 11, 21 y 41.

El capítulo 10 titulado Impulso y árboles aditivos del libro "The elements of statistical learning: Data mining, inference, and prediction' está dedicado al impulso. En él, proporcionan tanto heurísticas para configurar el aumento de gradiente como algunos estudios empíricos. Comentan que un buen valor el número de nodos en el

árbol (J) es alrededor de 6, con valores generalmente buenos en el rango de 4 a 8.

Aunque en muchas aplicaciones J = 2 será insuficiente, es poco probable que se requiera J>10. La experiencia hasta ahora indica que 4 <J <8 funciona bien en el contexto del impulso, y los resultados son bastante insensibles a elecciones particulares en este rango.

Sugieren monitorear el rendimiento en un conjunto de datos de validación para calibrar el número de árboles y utilizar un procedimiento de detención temprana una vez que el rendimiento en el conjunto de datos de validación comience a degradarse. Como en el primer artículo de refuerzo de gradiente de Friedman, comentan sobre la compensación entre el número de árboles (M) y la tasa de aprendizaje (v) y recomiendan un valor pequeño para la tasa de aprendizaje <0,1.

Además, como en el artículo de refuerzo de gradiente estocástico de Friedman, recomiendan un porcentaje de submuestreo (n) sin reemplazo con un valor de alrededor del 50Un valor típico para n puede ser $\frac{1}{2}$, aunque para un valor de N grande, n puede ser sustancialmente menor $\frac{1}{2}$

La biblioteca XGBoost está dedicada completamente a la implementación del algoritmo *gradient boosting*. También especifica parámetros predeterminados que son interesantes de tener en cuenta, primero los parámetros XGBoost [18]:

- eta = 0.3 (también conocido como tasa de aprendizaje)
- max_depth = 6
- sumsample =1

En una charla con TechEd Europe [19] titulada *xgboost: Un paquete R para un aumento de gradiente rápido y preciso* [20], cuando se le preguntó cómo configurar XGBoost, Tong He sugirió que los tres parámetros más importantes para ajustar son: el número de árboles, la profundidad del árbol y la tasa de aprendizaje. También proporciona una estrategia de configuración concisa para nuevos problemas:

1. Ejecute la configuración predeterminada (¿y probablemente revise las curvas de aprendizaje?).
2. Si el sistema está sobreaprendiendo, ralentice el aprendizaje (¿utilizando contracción?).

[18] https://xgboost.readthedocs.io/en/latest/parameter.html

[19] TechEd es el principal evento de educación técnica de Microsoft que brinda la capacitación técnica más completa sobre el conjunto de productos, tecnologías, soluciones y servicios de Microsoft.

[20] `https://www.youtube.com/watch?v=0IhraqUVJ_E`

3. Si el sistema está subaprendiendo, acelere el aprendizaje para que sea más agresivo (¿utilizando la contracción?).

En la charla de Owen Zhang en la NYC Data Science Academy en 2015 titulada "Winning Data Science Competitions" [21], ofrece algunos consejos generales para configurar el *gradient boost* con XGBoost. Owen es un gran usuario del algoritmo *gradient boosting*. Proporciona algunos consejos para configurar el algoritmo:

- Apunte de 500 a 1000 árboles y luego ajuste la tasa de aprendizaje (n estimadores).
- Establezca el número de muestras en los nodos de las hojas en suficientes observaciones necesarias para hacer una buena estimación media (*min child weight*).
- Configure la profundidad de interacción en aproximadamente 10 o más (profundidad máxima).

En las diapositivas actualizadas para la misma charla [22], proporciona una tabla de parámetros comunes que usa para XGBoost, resumidos de la siguiente manera:

- **Number of Trees** (n_estimators) establecido en un valor fijo entre 100 y 1000, según el tamaño del conjunto de datos.
- **Learning Rate** (learning_rate) simplificada a la proporción: $\frac{2a10}{\text{árboles}}$, dependiendo del número de árboles.
- **Row Sampling** (subsample) buscó valores en el rango $[0.5, 0.75, 1.0]$.
- **Column Sampling**(colsample_bytree) valores de búsqueda de cuadrícula en el rango $[0.4, 0.6, 0.8, 1.0]$.
- **Min LeafWeight** (min_child_weight) simplificado a la proporción de $\frac{3}{eventos\_raros}$, donde los eventos raros es el porcentaje de observaciones de eventos raros en el conjunto de datos.
- **Tree Size** (max depth) busqueda de valores en $[4, 6, 8, 10]$.
- **Min Split Gain** (gamma) fijada con un valor de cero.

[21] https://www.youtube.com/watch?v=LgLcfZjNF44
[22] `http://goo.gl/0qIRIc`

2.4. Regresión logística con estimación a través del método de Firth

La regresión logística se trata de un tipo de análisis de regresión utilizado para predecir el resultado de una variable categórica (aquella que puede adoptar un número limitado de categorías) en función de las variables predictoras. Este modelo se enmarca dentro de los modelos denominados de predicción lineal generalizados o *glm* como son conocidos por sus siglas en inglés. Como se mencionó al inicio de este capítulo, nos encontramos con un problema de clasificación binaria (paga, no paga)

2.4.1. Regresión logística Múltiple (RLM)

El modelo de regresión logística surge del deseo de modelar las probabilidades posteriores de las clases K mediante funciones lineales en x, mientras que al mismo tiempo se asegura que sumen uno y se encuentren en $[0, 1]$. El modelo tiene la forma:

$$\begin{aligned}
\log \frac{Pr(G=1|X=x)}{Pr(G=K|X=x)} &= \beta_{10} + \beta_1^T x \\
\log \frac{Pr(G=2|X=x)}{Pr(G=K|X=x)} &= \beta_{20} + \beta_2^T x \\
\log \frac{Pr(G=3|X=x)}{Pr(G=K|X=x)} &= \beta_{30} + \beta_3^T x \\
&\dots \\
\log \frac{Pr(G=K-1|X=x)}{Pr(G=K|X=x)} &= \beta_{K-1} + \beta_{k-1}^T x
\end{aligned} \tag{2.28}$$

El modelo se especifica en términos de K - 1 *log-odds* o transformaciones logit (lo que refleja la restricción de que las probabilidades suman uno). Aunque el modelo utiliza la última clase como denominador en las razones de probabilidad, la elección del denominador es arbitraria en el sentido de que las estimaciones son equivariantes en esta elección. Un simple cálculo muestra que:

$$\begin{aligned}
Pr(G=k|X=x) &= \frac{\exp(\beta_{k0} + \beta_k^T x)}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \beta_l^T x)} \quad k = 1, 2, ..., K-1 \\
Pr(G=K|X=x) &= \frac{1}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \beta_l^T x)}
\end{aligned} \tag{2.29}$$

que claramente suman 1. Para enfatizar la dependencia de todo el conjunto de parámetros se denota

$$\theta = \{\beta_{10}, \beta_1^T, \beta_{20}, \beta_2^T, ..., \beta_{(K-1)0}, \beta_{K-1}^T\}$$

y también

$$Pr(G = k|X = x) = p_k(x; \theta)$$

Antes de proceder con la estimación de parámetros, recordemos que tenemos un problema de clasificación binaria, por lo tanto, fijamos $K = 2$, y continuamos con el ajuste del modelo.

2.4.2. Estimación de los parámetros

Los modelos de regresión logística generalmente se ajustan mediante el método de máxima verosimilitud [23], utilizando la probabilidad condicional de G dada X. Dado que $Pr(G|X)$ especifica completamente la distribución condicional, la distribución multinomial es apropiada. La función *log-likehood* [24] para N observaciones es

$$l(\theta) = \sum_{i=1}^{N} \log p_{g_i}(x_i; \theta) \tag{2.30}$$

donde $p_k(x_i; \theta) = Pr(G = k|X = x_i; \theta)$.

Dado que en nuestro caso $K = 2$, es conveniente codificar las 2 clases g_i por 0 o 1, donde $y_i = 1$ cuando $g_i = 1$, y $y_i = 0$ cuando $g_i = 2$. Sea $p_1(x; \theta) = p(x; \theta)$ y $p_2(x; \theta) = 1 - p(x; \theta)$. La función *log-likehood* puede ser escrita de la siguiente manera:

$$\begin{aligned} l(\beta) &= \sum_{i=1}^{N} \{y_i \log p(x_i; \beta) + (1 - y_i) \log(1 - p(x_i; \beta))\} \\ &= \sum_{i=1}^{N} \left\{ y_i \beta^T x_i - \log(1 + e^{\beta^T x_i}) \right\} \end{aligned} \tag{2.31}$$

Donde $\beta = \{\beta_{10}, \beta_1\}$ y suponemos que el vector de entradas x_i incluye el término constante 1 para acomodar la intersección.

[23] La estimación por máxima verosimilitud (conocida también como EMV, o MLE por sus siglas en inglés) es un método habitual para ajustar un modelo y estimar sus parámetros.

[24] Es una transformación logarítmica de la función de verosimilitud

Para maximizar la función *log-likehood*, establecemos sus derivadas en cero, Obteniendo las siguientes ecuaciones de puntuación

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i(y_i - p(x_i;\beta)) = 0 \tag{2.32}$$

que son $p+1$ ecuaciones no lineales en β. Note que dado que el primer componente de x_i es 1, la primera ecuación de puntuación especifica que $\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} p(x_i;\beta)$; el número esperado de clases uno coincide con el número observado (y por lo tanto también clase dos).

Para resolver las ecuaciones de puntuación 2.32, utilizamos el algoritmo de Newton-Raphson [25], que requiere la segunda derivada o matriz de Hessiana

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i x_i^T p(x_i;\beta)(1 - p(x_i;\beta)) = 0 \tag{2.33}$$

Comenzando con β^{old}, un paso de actualización de Newton es

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T}\right)^{-1} \frac{\partial(\beta)}{\partial \beta} \tag{2.34}$$

donde las derivadas son evaluadas en β^{old}.

Es conveniente escribir la puntuación y la hessiana en notación matricial. Sea $\mathbf{y}$ el vector de valores y_i, $\mathbf{X}$ la matriz $N * (p+1)$ de los valores x_i, $\mathbf{p}$ el vector de probabilidades ajustadas con el i-ésimo elemento $p(x_i;\beta^{old})$ y $\mathbf{W}$ una matriz diagonal de dimensiones $N * N$ de pesos, con el i-ésimo elemento $p(x_i;\beta old)(1 - p(x_i;\beta^{old}))$. Entonces tenemos

$$\frac{\partial l(\beta)}{\partial \beta} = X^T(y - p) \tag{2.35}$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -X^T W X \tag{2.36}$$

El paso de Newton es entonces

$$\begin{aligned} \beta^{new} &= \beta^{old} + (X^T W X)^{-1} X^T (y - p) \\ &= (X^T W X)^{-1} X^T W (X\beta^{old} + W^{-1}(y - p)) \\ &= (X^T W X)^{-1} X^T W z \end{aligned} \tag{2.37}$$

[25] Es una forma de encontrar rápidamente una buena aproximación para la raíz de una función de valor real $f(x) = 0$

En la segunda y tercera línea hemos reexpresado el paso de Newton como un paso de mínimos cuadrados ponderados, con la respuesta

$$z = X\beta^{old} + W^{-1}(y - p) \tag{2.38}$$

a veces conocido como la respuesta ajustada. Estas ecuaciones se resuelven repetidamente, ya que en cada iteración p cambia, por lo tanto, también lo hacen W y z. Este algoritmo se conoce como *mínimos cuadrados reponderados iterativamente* o IRLS por sus siglas en inglés, ya que cada iteración resuelve el problema de mínimos cuadrados ponderados:

$$\beta^{new} \longleftarrow \arg\min_{\beta}(z - X\beta)^T W(z - X\beta) \tag{2.39}$$

Parece que $\beta = 0$ es un buen valor inicial para el procedimiento iterativo, aunque la convergencia nunca está garantizada. Por lo general, el algoritmo converge, ya que la función *log-likehood* es cóncava, pero puede ocurrir un exceso. En los raros casos en que la probabilidad logarítmica disminuye, la reducción a la mitad del tamaño del paso garantizará la convergencia.

2.4.3. Método de Firth

Como se revisó en el apartado anterior, la estimación de los parámetros $\beta = \{\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_n\}$ del modelo logit se realiza mediante el método de estimación de máxima verosimilitud, obteniendo que el sistema de ecuaciones que se forma al maximizar la verosimilitud es no lineal, por lo que para su resolución es necesario hacer intervenir métodos iterativos usualmente el de *Newton-Raphson*, la solución de este sistema de ecuaciones es lo que se denomina el estimador de máxima verosimilitud para los coeficientes de la regresión logística.

Cuando la variable respuesta (dicotómica) del modelo logit presenta un gran desequilibrio entre sus clases (evento raro), el estimador de máxima verosimilitud empieza a deteriorarse y a tomar un sesgo bastante significativo, esto implica que las probabilidades que se estimen a partir del modelo se encuentren subestimadas, y por ende arrojen resultados erróneos y no aptos para la solución del problema en cuestión (King and Zeng, 2001).

Una forma de reducir el sesgo del estimador de máxima verosimilitud es me-

diante la utilización del método de prevención de sesgo de Firth introducido en el año 1993, el cual consiste básicamente en modificar las ecuaciones del sistema no lineal de tal forma que su solución resulte en un estimador insesgado. Esta corrección de la regresión logística múltiple tradicional constituye, lo que hoy se denomina regresión logística con estimación a través del método de Firth. Para más detalle de la implementación de este método revisar Firth (1993).

2.5. Evaluación y selección de modelos

Los siguientes conceptos fueron tomados del capítulo 7 de Hastie et al. (2009).

El rendimiento de generalización de un método de aprendizaje se relaciona con su capacidad de predicción sobre datos de prueba independientes. La evaluación de este desempeño es extremadamente importante en la práctica, ya que guía la elección del método o modelo de aprendizaje y nos da una medida de la calidad del modelo finalmente elegido.

En este capítulo describimos e ilustramos los métodos clave para la evaluación del desempeño y mostramos cómo se utilizan para seleccionar modelos. Comenzamos el capítulo con una discusión sobre la interacción entre el sesgo [26], la varianza [27] y la complejidad del modelo.

2.5.1. Sesgo, varianza y complejidad del modelo

Es muy importante antes de poner un modelo en producción, realizar varias pruebas y evaluar la capacidad de generalización de este modelo o método de aprendizaje, en este punto es casi obligatorio evaluar el sesgo y la varianza de un modelo ya que son dos conceptos que se utilizan a la hora de medir el error de un modelo. En la figura 2.3 se puede ver que las curvas de color azul claro muestran el error de entrenamiento $\overline{err}$, mientras que las curvas de rojo claro muestran el error de prueba condicional Err_τ para 100 conjuntos de entrenamiento de tamaño 50 cada uno, ya que la complejidad del modelo aumenta. Las curvas continuas muestran el error de prueba esperado Err y el error de entrenamiento esperado $E[\overline{err}]$.

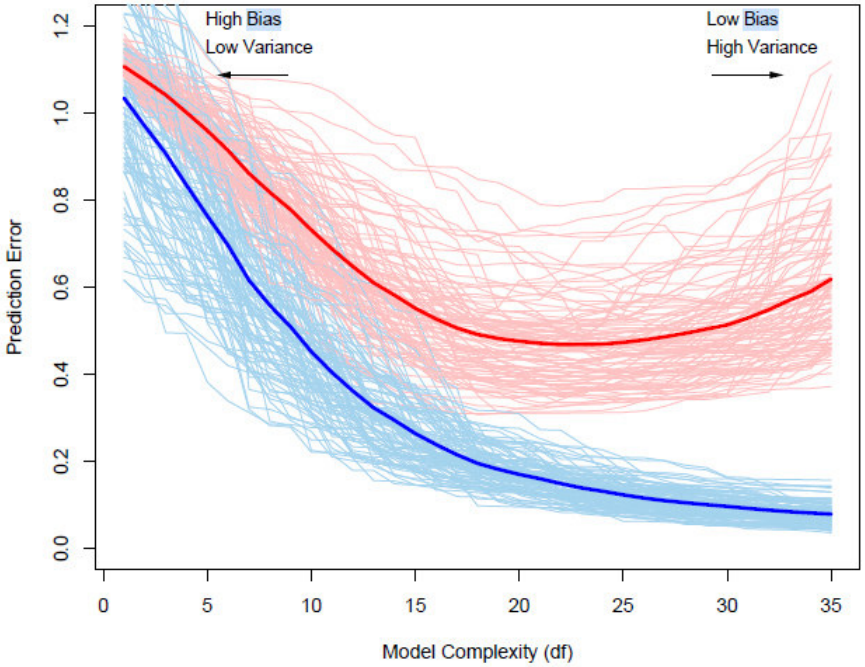
[26] El sesgo mide lo lejos que se encuentra el valor estimado respecto al real.

[27] La varianza se refiere a la cantidad que la estimación de la función objetivo cambiaría si se utiliza diferentes datos de entrenamiento

Consideremos primero el caso de una respuesta cuantitativa, tenemos una variable objetivo Y, un vector de características X, y un modelo predictivo $\hat{f}(X)$ que ha sido estimado a partir de un conjunto de entrenamiento τ. La función de costos para medir los errores entre Y y $\hat{f}(X)$ es denotada por $L(Y, \hat{f}(X))$. Típicamente se eligen:

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \text{errores al cuadrado} \\ |Y - \hat{f}(X))| & \text{errores absolutos} \end{cases} \tag{2.40}$$

Figura 2.3: Comportamiento del error de las muestras de prueba y entrenamiento a medida que varía la complejidad del modelo.



Fuente: (Hastie, Tibshirani y Friedman, 2009)

El *error de prueba*, también conocido como *error de generalización*, es el error de predicción sobre una muestra de prueba independiente.

$$Err_\tau = E[L(Y, \hat{f}(X))|\tau] \tag{2.41}$$

donde, tanto X como Y fueron obtenidas aleatoriamente a partir de su distribución de probabilidad conjunta (población). Aquí, el conjunto de entrenamiento τ está fijo y el error de prueba se refiere al error para este conjunto de entrenamiento específico. Una cantidad relacionada es el error de predicción esperado (o error de

prueba esperado)

$$Err = E[L(Y, \hat{f}(X))] = E[Err_\tau] \tag{2.42}$$

Tenga en cuenta que esta expectativa promedia todo lo que es aleatorio, incluida la aleatoriedad en el conjunto de entrenamiento que produjo $\hat{f}$. La Figura 2.3 muestra el error de predicción (curvas en rojo claro) Err_τ para 100 conjuntos de entrenamiento simulados cada uno de tamaño 50. Se utilizó una regresión lasso [28] para producir la secuencia de ajustes. La curva roja continua es el promedio y, por lo tanto, una estimación de Err.

La estimación de Err_τ será nuestro objetivo, aunque veremos que Err es más susceptible de análisis estadístico y la mayoría de los métodos estiman eficazmente el error esperado. No parece posible estimar el error condicional efectivamente, dada solo la información en el mismo conjunto de entrenamiento. El *error de entrenamiento* es la pérdida promedio sobre el conjunto de entrenamiento

$$\overline{err} = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{2.43}$$

Nos gustaría conocer el error de prueba esperado de nuestro modelo estimado $\hat{f}$. A medida que el modelo se vuelve cada vez más complejo, utiliza más los datos de entrenamiento y es capaz de adaptarse a estructuras subyacentes más complicadas. Por lo tanto, hay una disminución en el sesgo pero un aumento en la varianza. Existe cierta complejidad de modelo intermedio que da un error de prueba mínimo esperado.

Desafortunadamente, el error de entrenamiento no es una buena estimación del error de prueba, como se ve en la Figura 2.3. El error de entrenamiento disminuye constantemente con la complejidad del modelo, por lo general cae a cero si aumentamos la complejidad del modelo lo suficiente. Sin embargo, un modelo con un error de entrenamiento cero se ajusta demasiado a los datos de entrenamiento y, por lo general, se generalizará mal.

La historia es similar para una respuesta cualitativa o categórica G tomando uno

[28] La regresión de lazo es un tipo de regresión lineal que utiliza la contracción. La contracción es donde los valores de los datos se reducen hacia un punto central, como la media.

de los K valores en un conjunto G, etiquetado por conveniencia como $1, 2...., K$, Típicamente se modela las probabilidades $p_k(X) = Pr(G = k|X)$ (o alguna transformación monótona) $f_k(X)$, y entonces $\hat{G}(X) = \arg\max_k \hat{p}_k(X)$. Las típicas funciones de costo son:

$$L(G, \hat{G}(X)) = I(G \neq \hat{G}(X)) \quad \text{(0-1 p)} \tag{2.44}$$

$$\begin{aligned} L(G, \hat{p}(X)) &= -2 \sum_{k=1}^{K} I(G = k) \log \hat{p}_k(X) \\ &= -2 \log \hat{p}_G(X) \text{(-2 x log-verosimilitud)} \end{aligned} \tag{2.45}$$

La cantidad -2 x log-verosimilitud a veces se denomina desviación. Aquí el error de prueba es $Err_\tau = E[L(G, \hat{G}(X))|\tau]$, el error de clasificación errónea de la población del clasificador entrenado en τ, y Err es el error de clasificación erróneo esperado. El error de entrenamiento es análogo de la muestra, por ejemplo,

$$\overline{err} = -\frac{2}{N} \sum_{i=1}^{N} \log \hat{p}_{g_i}(x_i) \tag{2.46}$$

la log-verosimilitud de la muestra para el modelo.

La log-verosimilitud se puede utilizar como función de costo para densidades de respuesta general, como Poisson, gamma, exponencial, log-normal y otras. Si $Pr_{\theta(x)}(Y)$ es la densidad de Y, indexada por un parámetro $\theta(X)$ que depende del predictor X, entonces

$$L(Y, \theta(X)) = -2 \log Pr_{\theta(X)}(Y) \tag{2.47}$$

El "-2" en la definición hace que el costo de la log-verosimilitud para la distribución gaussiana [29] coincida con la función de costo de error al cuadrado.

Para facilitar la exposición, durante el resto de esta sección usaremos Y y $f(X)$ para representar todas las situaciones anteriores, ya que nos enfocamos principalmente en la configuración de respuesta cuantitativa (pérdida de error al cuadrado).

[29] En la teoría de la probabilidad, una distribución gaussiana o normal es un tipo de distribución de probabilidad continua para una variable aleatoria de valor real.

Para las otras situaciones, las traducciones apropiadas son obvias. En este sección describimos varios métodos para estimar el error de prueba esperado para un modelo. Normalmente, nuestro modelo tendrá un parámetro o parámetros de ajuste α, por lo que podemos escribir nuestras predicciones como $\hat{f}_\alpha(x)$. El parámetro de ajuste varía la complejidad de nuestro modelo, y deseamos encontrar el valor de α que minimiza el error, es decir, produce el mínimo de la curva del error de prueba promedio en la Figura 2.3. Dicho esto, por brevedad, a menudo suprimiremos la dependencia de $\hat{f}(x)$ de α.

Es importante señalar que, existen dos objetivos importantes que se estudiarán en este trabajo:

- **Selección del modelo.-** Estimar el desempeño de distintos modelos para elegir el mejor.
- **Evaluación del modelo.-** Una vez elegido el modelo, estimar el error de predicción (error de generalización) en un nuevo conjunto de datos.

Si nos encontramos en una situación con una gran cantidad de datos, el mejor enfoque para ambos problemas es dividir aleatoriamente el conjunto de datos en tres partes:

- **Un conjunto de entrenamiento** Utilizado para ajustar los modelos.
- **Un conjunto de validación** Se utiliza para estimar el error de predicción para la selección del modelo.
- **Un conjunto de prueba** Se utiliza para evaluar el error de generalización del modelo final elegido.

Idealmente, el conjunto de prueba debe guardarse en una "bóveda" y sacarse solo al final del análisis de datos. Supongamos en cambio que usamos el conjunto de prueba repetidamente, eligiendo el modelo con el menor error en el conjunto de prueba. Luego, el error de conjunto del modelo final elegido subestimará el verdadero error de prueba, a veces sustancialmente. Es difícil dar una regla general sobre cómo elegir el número de observaciones en cada una de las tres partes, ya que esto depende de la relación señal-ruido en los datos y el tamaño de la muestra de entrenamiento. Una división típica podría ser 50 % para entrenamiento y 25 % para validación y prueba:

2.5.2. Optimismo de la tasa de error de entrenamiento

Las discusiones sobre la estimación de la tasa de error pueden ser confusos, porque tenemos que dejar claro qué cantidades son fijas y cuáles son aleatorias. Antes de continuar, necesitamos unas cuantas definiciones. Dado un conjunto de entrenamiento $\tau = \{(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)\}$ el error de generalización de un modelo $\hat{f}$ es

$$Err_\tau = E_{X^0,Y^0}[L(Y^0, \hat{f}(X^0))|\tau]; \tag{2.48}$$

Obsérvese que el conjunto de entrenamiento τ es fijo en la expresión 2.48. El punto (X^0, Y^0) es un nuevo punto de datos de prueba, extraído de F, la distribución conjunta de los datos. Al promediar los conjuntos de entrenamiento τ se obtiene el error esperado

$$Err = E_\tau E_{X^0,Y^0}[L(Y^0, \hat{f}(X^0))|\tau], \tag{2.49}$$

que es más fácil de analizar estadísticamente. Como se ha mencionado anteriormente, resulta que la mayoría de los métodos estiman efectivamente el error esperado en lugar del E_τ. Ahora bien, normalmente, el error de entrenamiento

$$\overline{err} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{2.50}$$

será menor que el verdadero error Err_τ , porque se están utilizando los mismos datos para ajustar el método y evaluar su error. Un método de ajuste suele adaptarse a los datos de entrenamiento y, por tanto, el error aparente o de entrenamiento $\overline{err}$ será una estimación demasiado optimista del error de generalización Err_τ. Parte de la discrepancia se debe al lugar donde se encuentran los puntos de evaluación. La cantidad Err_τ puede considerarse como un error *extra-muestra*, ya que los vectores de entrada de la prueba de prueba no tienen por qué coincidir con los vectores de entrada de entrenamiento. La naturaleza del optimismo en $\overline{err}$ es más fácil de entender si nos centramos en cambio en el error *dentro de la muestra*

$$Err_{in} = \frac{1}{N}\sum_{i=1}^{N} E_{Y^0}[L(Y_i^0, \hat{f}(x_i))|\tau] \tag{2.51}$$

La notación Y^0 indica que observamos N nuevos valores de respuesta en cada

uno de los puntos de entrenamiento $x_i, i = 1, 2, \ldots, N$. Definimos el optimismo como la diferencia entre Err_{in} y el error de entrenamiento $\overline{err}$:

$$op \equiv Err_{in} - \overline{err}. \tag{2.52}$$

Suele ser positivo, ya que $\overline{err}$ suele estar sesgado a la baja como estimación del error de predicción. Por último, el optimismo medio es la expectativa del optimismo sobre los conjuntos de entrenamiento

$$\omega \equiv E_y(op). \tag{2.53}$$

Aquí los predictores del conjunto de entrenamiento son fijos, y la expectativa es sobre los valores de resultado del conjunto de entrenamiento; de ahí que hayamos utilizado la notación E_y en lugar de E_τ . Normalmente podemos estimar sólo el error esperado ω en lugar de op, de la misma manera que podemos estimar el error esperado Err en lugar del error condicional Err_τ. Para el error al cuadrado, $0-1$, y otras funciones de pérdida, se puede demostrar de forma bastante general que

$$\omega = \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y_i), \tag{2.54}$$

donde Cov indica la covarianza. Por lo tanto, la cantidad en la que $\overline{err}$ subestima el verdadero error depende de la fuerza con la que y_i afecta a su propia predicción. Cuanto más difícil sea el ajuste de los datos, mayor será $Cov(\hat{y}_i, y_i)$, aumentando así el optimismo. Para la pérdida $0-1$, $\hat{y}_i \in 0,1$ es la clasificación en x_i, y para la pérdida de entropía, $\hat{y}_i \in [0,1]$ es la probabilidad ajustada de la clase 1 en xi. En resumen, tenemos la importante relación

$$E_y(Err_{in}) = E_y(\overline{err}) + \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y_i). \tag{2.55}$$

Esta expresión se simplifica si $\hat{y}_i$ se obtiene mediante un ajuste lineal con d entradas o funciones base. Por ejemplo,

$$\sum_{i=1}^{N} Cov(\hat{y}_i, y_i) = d\sigma_\varepsilon^2 \tag{2.56}$$

para el modelo de error aditivo $Y = f(X) + \varepsilon$, y así

$$E_y(Err_{in}) = E_y(\overline{err}) + 2 \cdot \frac{d}{N}\sigma_\varepsilon^2. \tag{2.57}$$

La expresión 2.56 es la base para la definición del número efectivo de parámetros. El optimismo aumenta linealmente con el número d de entradas o funciones base que utilicemos, pero disminuye a medida que aumenta el tamaño de la muestra de entrenamiento. Las versiones de 2.57 se mantienen aproximadamente para otros modelos de error, como los datos binarios y la pérdida de entropía. Una forma obvia de estimar el error de predicción es estimar el optimismo y luego añadirlo al error de entrenamiento $\overline{err}$. Los métodos descritos en la siguiente sección —C_p, AIC, BIC y otros— funcionan de esta manera, para una clase especial de estimaciones que son lineales en sus parámetros. Por el contrario, los métodos de validación cruzada y bootstrap, son estimaciones directas del error extra-muestra Err. Estas herramientas generales pueden utilizarse con cualquier función de pérdida y con técnicas de ajuste no lineales y adaptativas. El error dentro de la muestra no suele ser de interés directo, ya que no es probable que los valores futuros de las características coincidan con los del conjunto de entrenamiento. Pero para la comparación entre modelos, el error en la muestra es conveniente y a menudo conduce a una selección eficaz del modelo. La razón es que lo que importa es el tamaño relativo (y no absoluto) del error.

2.5.3. Estimación del error de predicción en la muestra

La forma general de las estimaciones dentro de la muestra es

$$\widehat{Err}_{in} = \overline{err} + \hat{\omega}, \tag{2.58}$$

donde $\hat{\omega}$ es una estimación del optimismo medio. El uso de la expresión 2.57, aplicable cuando los parámetros d se ajustan bajo pérdida de error cuadrado, conduce a una versión del llamado estadístico C_p,

$$C_p = \overline{err} + 2 \cdot \frac{d}{N}\widehat{\sigma_\varepsilon}^2. \tag{2.59}$$

Aquí $\widehat{\sigma_\varepsilon}^2$ es una estimación de la varianza del ruido, obtenida a partir del error cuadrático medio de un modelo de bajo sesgo. Con este criterio ajustamos el error

de entrenamiento en un factor proporcional al número de funciones base utilizadas. El criterio de información de Akaike es una estimación de Err_{in} similar pero más aplicable cuando se utiliza una función de pérdida de log-verosimilitud. Se basa en una relación similar a 2.57 que se mantiene asintóticamente como $N \to \infty$:

$$-2 \cdot E[logPr_{\hat{\theta}}(Y)] \approx -\frac{2}{N} \cdot E[loglik] + 2 \cdot \frac{d}{N}. \tag{2.60}$$

Aquí $Pr_{\hat{\theta}}(Y)$ es una familia de densidades para Y (que contiene la densidad "verdadera"), $\hat{\theta}$ es la estimación de máxima verosimilitud de θ, y "$loglik$" es la log-verosimilitud maximizada:

$$loglik = \sum_{i=1}^{N} logPr_{\hat{\theta}}(y_i). \tag{2.61}$$

Por ejemplo, para el modelo de regresión logística, utilizando la log-verosimilitud binomial, tenemos

$$AIC = -\frac{2}{N} \cdot loglik + 2 \cdot \frac{d}{N}. \tag{2.62}$$

Para el modelo Gaussiano (con varianza $\sigma_\epsilon^2 = \hat{\sigma}_\epsilon^2$ supuestamente conocida), el estadístico AIC es equivalente a C_p, por lo que nos referimos a ellos colectivamente como AIC.

Para utilizar el AIC en la selección de modelos, simplemente elegimos el modelo que da el menor AIC sobre el conjunto de modelos considerados. En el caso de los modelos no lineales y otros modelos complejos, tenemos que sustituir d por alguna medida de la complejidad del modelo.

Dado un conjunto de modelos $f_\alpha(x)$ indexados por un parámetro de ajuste α, denotamos por $\overline{err}(\alpha)$ y $d(\alpha)$ el error de entrenamiento y el número de parámetros para cada modelo. Entonces, para este conjunto de modelos definimos

$$AIC(\alpha) = \overline{err}(\alpha) + 2 \cdot \frac{d(\alpha)}{N} \hat{\sigma}_\varepsilon{}^2. \tag{2.63}$$

La función $AIC(\alpha)$ proporciona una estimación de la curva de error de la prueba, y encontramos el parámetro de ajuste $\hat{\alpha}$ que la minimiza. Nuestro modelo final elegido es $f_{\hat{\alpha}}(x)$. Nótese que si las funciones de base se eligen de forma adaptativa, 2.56 ya no se cumple. Por ejemplo, si tenemos un total de p entradas, y elegimos el modelo lineal de mejor ajuste con $d < p$ entradas, el optimismo superará $(\frac{2d}{N})\sigma_\epsilon^2$.

Dicho de otro modo, al elegir el modelo de mejor ajuste con d entradas, el número efectivo de parámetros ajustados es superior a d.

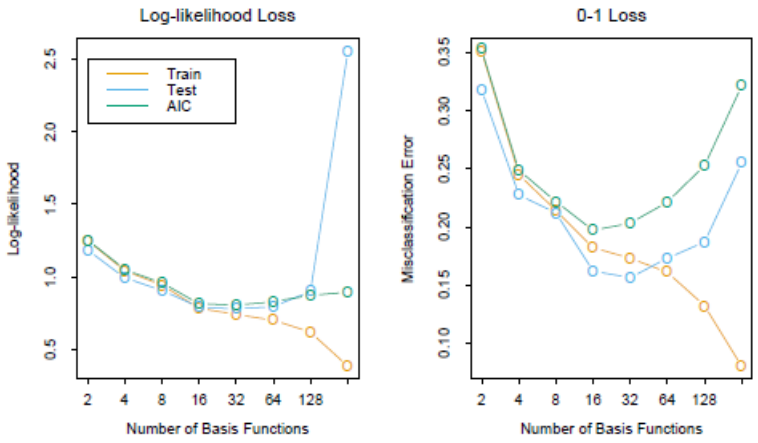
La figura 2.4 muestra el AIC en acción para el ejemplo de reconocimiento de fonemas. El vector de entrada es el log-periodograma de la vocal hablada, cuantificado en 256 frecuencias uniformemente espaciadas. Se utiliza un modelo de regresión logística lineal para predecir la clase de fonema, con función de coeficiente $\beta(f) = \sum_{m=1}^{M} h_m(f)\theta_m$, una expansión en M funciones de base spline. Para cualquier M, se utiliza una base de splines cúbicos naturales para el h_m, con nudos elegidos uniformemente en el rango de frecuencias (así $d(\alpha) = d(M) = M$). El uso del AIC para seleccionar el número de funciones de base minimizará aproximadamente $Err(M)$ tanto para la entropía como para la pérdida $0-1$.

La simple fórmula

$$(2/N)\sum_{i=1}^{N} Cov(\hat{y}_i, y_i) = (2d/N)\sigma_\epsilon^2$$

se mantiene exactamente para los modelos lineales con errores aditivos y pérdida de error al cuadrado, y aproximadamente para los modelos lineales y log-likelihoods. En particular, la fórmula no es válida en general para la pérdida 0-1 (Efron, 1986), aunque muchos autores la utilizan en ese contexto (panel derecho de la figura 2.4).

Figura 2.4: AIC utilizado para la selección de modelos para el reconocimiento de fonemas.



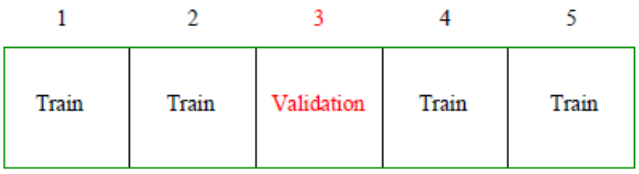
Fuente: (Hastie, Tibshirani y Friedman, 2009)

2.5.4. Validación cruzada

Probablemente, el método más sencillo y más utilizado para estimar el error de predicción es la validación cruzada. Este método estima directamente el error esperado error extra-muestra $Err = E[L(Y, \hat{f}(X))]$, el error medio de generalización cuando el método $\hat{f}(X)$ se aplica a una muestra de prueba independiente de la distribución conjunta de X e Y. Como se mencionó anteriormente, podríamos esperar que la validación cruzada estimará el error condicional, con el conjunto de entrenamiento τ mantenido fijo.

Validación cruzada K-Fold

Lo ideal sería que, si tuviéramos suficientes datos, reserváramos un conjunto de validación y lo utilizáramos para evaluar el rendimiento de nuestro modelo de predicción. Como los datos suelen ser escasos, esto no suele ser posible. Para suavizar el problema, la validación cruzada de $K - fold$ utiliza una parte de los datos disponibles para ajustar el modelo y otra parte para probarlo. Dividimos los datos en K partes de tamaño aproximadamente igual; por ejemplo, cuando $K = 5$, el escenario es el siguiente:



Para la k-ésima parte (la tercera), ajustamos el modelo a las otras $K - 1$ partes de los datos, y calculamos el error de predicción del modelo ajustado al predecir la k-ésima parte de los datos. Hacemos esto para $k = 1, 2, \ldots, K$ y combinamos las K estimaciones del error de predicción.

Aquí hay más detalles. Sea $\kappa : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ una función de indexación que indica la partición a la que se asigna la observación i por la aleatorización. Denotemos por $\hat{f}^{-k}(x)$ la función ajustada, calculada con la k-ésima parte de los datos eliminada. Entonces la estimación de validación cruzada del error de predicción es

$$CV(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i)). \tag{2.64}$$

Las opciones típicas de K son 5 o 10. El caso $K = N$ se conoce como validación cruzada con exclusión de uno de los lados. En este caso $\kappa(i) = i$, y para la i-ésima observación el ajuste se calcula utilizando todos los datos excepto el i-ésimo.

Dado un conjunto de modelos $f(x, \alpha)$ indexados por un parámetro de ajuste α, denote por $\hat{f} - k(x, \alpha)$ el α-ésimo modelo ajustado con la k-ésima parte de los datos eliminada. Entonces, para este conjunto de modelos definimos

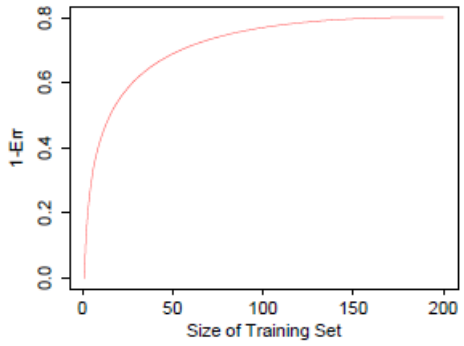
$$CV(\hat{f}, \alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha)). \tag{2.65}$$

La función $CV(\hat{f}, \alpha)$ proporciona una estimación de la curva de error de la prueba, y encontramos el parámetro de ajuste $\hat{\alpha}$ que la minimiza. Nuestro modelo final elegido es $f(x, \hat{\alpha})$, que luego ajustamos a todos los datos.

Es interesante preguntarse sobre qué cantidad estima la validación cruzada de K-fold. Con $K = 5$ o 10, podríamos suponer que estima el error esperado Err, ya que los conjuntos de entrenamiento en cada pliegue son bastante diferentes del conjunto de entrenamiento original. Por otro lado, si $K = N$ podríamos suponer que la validación cruzada estima el error condicional Err_τ . Resulta que la validación cruzada sólo estima de forma efectiva el error medio Err.

¿Qué valor debemos elegir para K? Con $K = N$, el estimador de validación cruzada es aproximadamente insesgado para el error de predicción verdadero (esperado), pero puede tener una alta varianza porque los N çonjuntos de entrenamiento"son muy similares entre sí. La carga computacional también es considerable, ya que requiere N aplicaciones del método de aprendizaje.

Figura 2.5: Curva de aprendizaje hipotética para un clasificador en una tarea determinada: un gráfico de $1 - Err$ frente al tamaño del conjunto de entrenamiento N.



Fuente: (Hastie, Tibshirani y Friedman, 2009)

Por otro lado, con $K = 5$ digamos, la validación cruzada tiene menor varianza. Pero el sesgo podría ser un problema, dependiendo de cómo varíe el rendimiento del método de aprendizaje con el tamaño del conjunto de entrenamiento. La figura 2.5 muestra una hipotética çurva de aprendizaje"para un clasificador en una tarea determinada, un gráfico de $1 - Err$ frente al tamaño del conjunto de entrenamiento N. El rendimiento del clasificador mejora a medida que el tamaño del conjunto de entrenamiento aumenta hasta 100 observaciones; aumentar el número hasta 200 sólo aporta un pequeño beneficio. Si nuestro conjunto de entrenamiento tuviera 200 observaciones, la validación cruzada quíntuple estimaría el rendimiento de nuestro clasificador sobre conjuntos de entrenamiento de tamaño 160, que según la Figura 2.5 es prácticamente el mismo que el rendimiento para un conjunto de entrenamiento de tamaño 200. Por lo tanto, la validación cruzada no sufriría mucho sesgo. Sin embargo, si el conjunto de entrenamiento tuviera 50 observaciones, la validación cruzada quíntuple estimaría el rendimiento de nuestro clasificador sobre conjuntos de entrenamiento de tamaño 40, y a partir de la cifra que sería una subestimación de $1 - Err$. Por lo tanto, como estimación de Err, la validación cruzada estaría sesgada al alza.

En resumen, si la curva de aprendizaje tiene una pendiente considerable con un tamaño de conjunto de entrenamiento determinado, la validación cruzada quíntuple o décupla sobrestimará el verdadero error de predicción. Que este sesgo sea un inconveniente en la práctica depende del objetivo. Por otro lado, la validación cruzada leave-one-out tiene un sesgo bajo pero puede tener una varianza alta. En general, se recomienda la validación cruzada de cinco o diez veces como un buen compromiso.

La Figura 2.7 muestra el error de predicción y la curva de validación cruzada de diez veces estimada a partir de un único conjunto de entrenamiento, del escenario del panel inferior derecho de la Figura 2.6. Se trata de un problema de clasificación de dos clases, utilizando un modelo lineal con regresión de los mejores subconjuntos de tamaño p. Se muestran las barras de error estándar, que son los errores estándar de las tasas de error de clasificación individuales de error de clasificación individual para cada una de las diez partes. Ambas curvas tienen mínimos en $p = 10$, aunque la curva del CV es bastante plana más allá de 10. A menudo se utiliza la regla de ün error estándarçon la validación cruzada, en el que elegimos el modelo más parsimonioso cuyo error no está más de un error estándar por encima del error del mejor modelo. Aquí parece que se elegiría un modelo con unos $p = 9$ predictores, mientras que el modelo verdadero utiliza $p = 10$.

La validación cruzada generalizada proporciona una aproximación conveniente a la validación cruzada de exclusión, para el ajuste lineal bajo pérdida de error cuadrado. Un método de ajuste lineal es aquel para el que podemos escribir

$$\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}. \tag{2.66}$$

Ahora, para muchos métodos de ajuste lineal,

$$\frac{1}{N}\sum_{i=1}^{N}[y_i - \hat{f}^{-i}(x_i)]^2 = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}}\right]^2, \tag{2.67}$$

donde S_{ii} es el i-ésimo elemento diagonal de $\mathbf{S}$. La aproximación GCV es

$$GCV(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - trace(\mathbf{S})/N}\right] \tag{2.68}$$

La cantidad $\mathbf{S}$ es el número efectivo de parámetros. El GCV puede tener una ventaja computacional en algunos entornos, donde la traza de $\mathbf{S}$ puede calcularse más fácilmente que los elementos individuales S_{ii}. En los problemas de suavización, el GCV también puede aliviar la tendencia de la validación cruzada a la falta de suavización. La similitud entre GCV y AIC puede verse en la aproximación $1/(1-x)^2 \approx 1 + 2x$.

La forma correcta e incorrecta de realizar validación cruzada

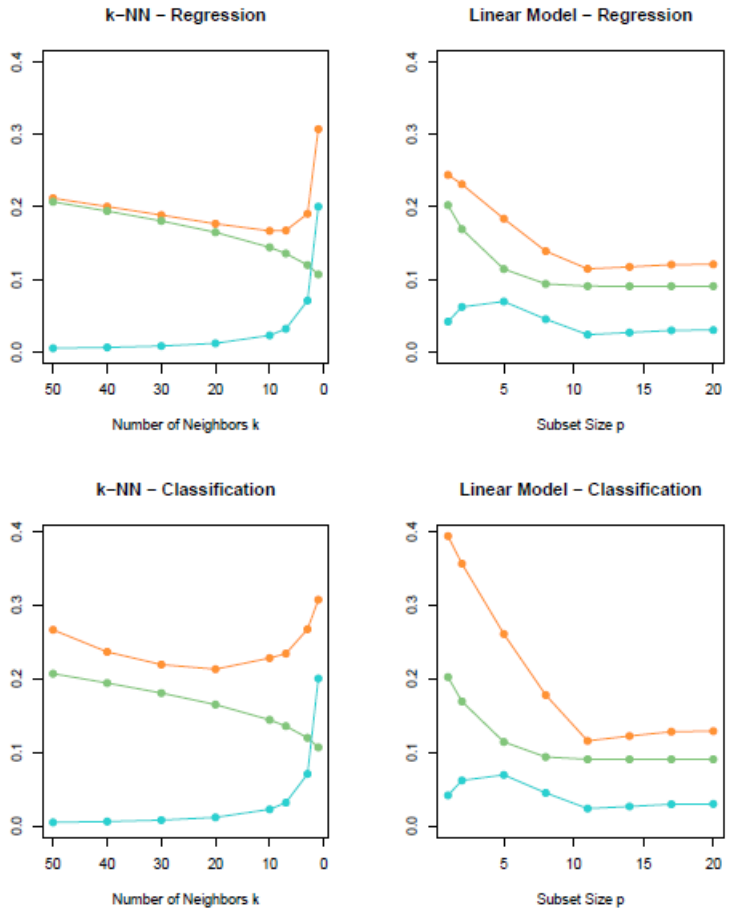
Consideremos un problema de clasificación con un gran número de predictores, como puede ocurrir, por ejemplo, en aplicaciones genómicas o proteómicas. Una estrategia típica de análisis podría ser la siguiente:

1. Seleccionar los predictores: encontrar un subconjunto de "buenos"predictores que muestren correlación bastante fuerte (univariante) con las etiquetas de clase.

2. Utilizando sólo este subconjunto de predictores, construir un clasificador multivariante.

3. Utilizar la validación cruzada para estimar los parámetros de ajuste desconocidos y estimar el error de predicción del modelo final.

Figura 2.6: Error de predicción esperado (naranja), sesgo al cuadrado (verde) y varianza (azul) para un ejemplo simulado. La fila superior es una regresión con pérdida de error al cuadrado; la fila inferior es una clasificación con pérdida $0-1$. Los modelos son los vecinos más cercanos (izquierda) y el mejor subconjunto de regresión de tamaño p (derecha). Las curvas de varianza y sesgo son las mismas en la regresión y la clasificación, pero la curva de error de predicción es diferente.

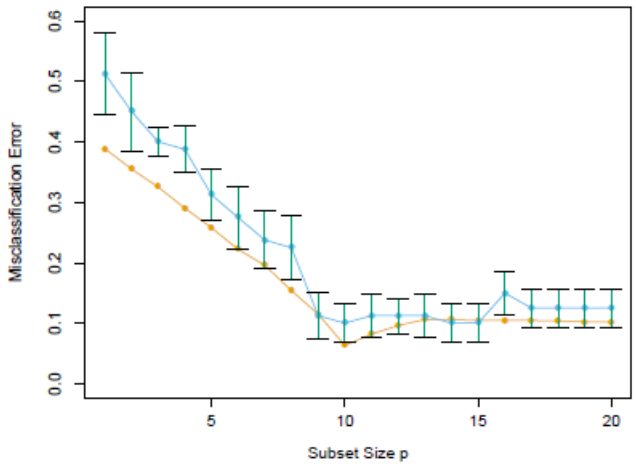


Fuente: (Hastie, Tibshirani y Friedman, 2009)

¿Es ésta una aplicación correcta de la validación cruzada? Considere un escenario con $N = 50$ muestras en dos clases de igual tamaño, y $p = 5000$ predictores cuantitativos (gaussianos estándar) que son independientes de las etiquetas de las clases. La tasa de error real (de prueba) de cualquier clasificador es del 50 %. Llevamos a cabo la anterior receta, eligiendo en el paso (1) los 100 predictores que tienen mayor correlación con las etiquetas de la clase y, a continuación, utilizando un clasificador de 1 vecino más cercano, basado en estos 100 predictores, en el paso (2). En más de 50 simulaciones de esta configuración, la tasa media de error de CV fue del 3 %. Esto es muy inferior a la tasa de error real del 50 %.

¿Qué ha ocurrido? El problema es que los predictores tienen una ventaja injusta, ya que fueron elegidos en el paso (1) sobre la base de todas las muestras. Dejar fuera

Figura 2.7: Error de predicción (naranja) y curva de validación cruzada de diez veces (azul) estimada a partir de un único conjunto de entrenamiento, del escenario del panel inferior derecho de la Figura 2.6



Fuente: (Hastie, Tibshirani y Friedman, 2009)

las muestras después de haber seleccionado las variables no imita correctamente la aplicación del clasificador a un conjunto de pruebas completamente independiente, ya que estos predictores za han visto"las muestras dejadas fuera.

La figura 2.8 (panel superior) ilustra el problema. Seleccionamos los 100 predictores que tienen la mayor correlación con las etiquetas de clase en las 50 muestras. A continuación, elegimos un conjunto aleatorio de 10 muestras, como haríamos en la validación cruzada de cinco pliegues, y calculamos las correlaciones de los 100 predictores preseleccionados con las etiquetas de clase sólo en estas 10 muestras (panel superior). Vemos que la media de las correlaciones es de $0,28$, en lugar de 0, como cabría esperar. Esta es la forma correcta de llevar a cabo la validación cruzada en este ejemplo:

1. Dividir las muestras en K pliegues de validación cruzada (grupos) al azar.
2. Para cada pliegue $k = 1, 2, \ldots, K$
   - *a*) Encuentre un subconjunto de "buenos"predictores que muestren una correlación bastante fuerte (univariante) con las etiquetas de clase, utilizando todas las muestras excepto las del pliegue k.
   - *b*) Utilizando sólo este subconjunto de predictores, construya un clasificador multivariante, utilizando todas las muestras excepto las del pliegue k.
   - *c*) Utilice el clasificador para predecir las etiquetas de clase para las muestras en el pliegue k.

Figura 2.8: Validación cruzada a la manera incorrecta y correcta: los histogramas muestran la correlación de las etiquetas de clase, en 10 muestras elegidas al azar, con los 100 predictores elegidos utilizando las versiones incorrecta (rojo superior) y correcta (verde inferior) de la validación cruzada.

Fuente: (Hastie, Tibshirani y Friedman, 2009)

Las estimaciones de error del paso 2(c) se acumulan entonces en todos los K pliegues, para producir la estimación de validación cruzada del error de predicción. El panel inferior de la Figura 2.8 muestra las correlaciones de las etiquetas de clase con los 100 predictores elegidos en el paso 2(a) del procedimiento correcto, sobre las muestras de un pliegue típico k. Vemos que la media es de cero, como debería ser.

En general, con un procedimiento de modelización de varios pasos, la validación cruzada debe aplicarse a toda la secuencia de pasos de modelización. En particular, hay que "dejar fuera"las muestras antes de aplicar cualquier paso de selección o filtrado. Hay un requisito: los pasos iniciales de cribado no supervisado pueden realizarse antes de dejar las muestras fuera. Por ejemplo, podríamos seleccionar los 1000 predictores con mayor varianza en las 50 muestras, antes de comenzar la validación cruzada. Como este filtrado no implica las etiquetas de clase, no da a los predictores una ventaja injusta.

¿Funciona realmente la validación cruzada?

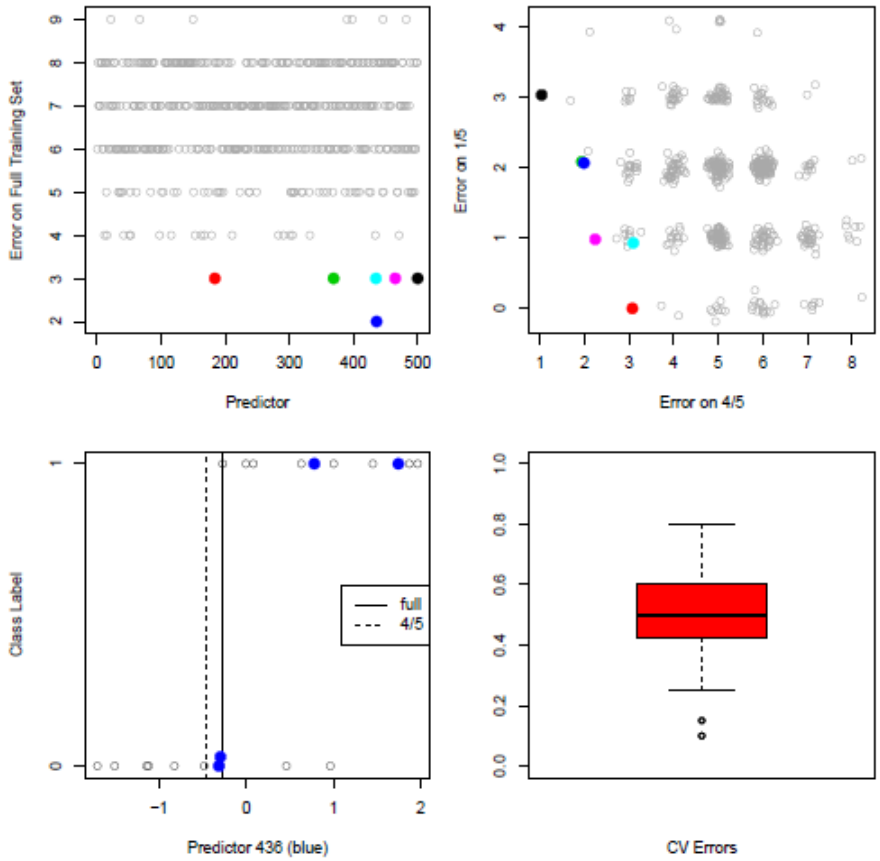
Volvemos a examinar el comportamiento de la validación cruzada en un problema de clasificación de alta dimensión. Consideremos un escenario con $N = 20$ muestras en dos clases de igual tamaño, y $p = 500$ predictores cuantitativos que son independientes de las etiquetas de clase. Una vez más, la tasa de error real de cualquier clasificador es del 50 %. Consideremos un clasificador univariante simple: una única división que minimice el error de clasificación (un "tocón"). Los tocones son árboles con una única división y se utilizan en los métodos de refuerzo. Un simple argumento sugiere que la validación cruzada no funcionará correctamente en este escenario:

> *Ajustando a todo el conjunto de entrenamiento, encontraremos un predictor que divida muy bien los datos. Si realizamos una validación cruzada de* 5 *veces, este mismo predictor debería dividir bien las* $4/5$ *partes y* $1/5$ *partes de los datos y, por tanto, su error de validación cruzada será pequeño (mucho menos del* 50 %*).*

Para investigar si este argumento es correcto, la figura 2.9 muestra el resultado de una simulación de esta configuración. Hay 500 predictores y 20 muestras, en cada una de las dos clases de igual tamaño, y todos los predictores tienen una distribución gaussiana estándar. El panel de la parte superior izquierda muestra el número de errores de entrenamiento para cada uno de los 500 tocones ajustados a los datos de entrenamiento. Hemos marcado en color los seis predictores que producen menos errores. En el panel superior derecho, se muestran los errores de entrenamiento para los tocones ajustados a una partición aleatoria de $4/5$ de los datos (16 muestras), y probados en el $1/5$ restante (cuatro muestras). Los puntos coloreados indican los mismos predictores marcados en el panel superior izquierdo. Vemos que el muñón del predictor azul (cuyo muñón era el mejor en el panel superior izquierdo), comete dos de cada cuatro errores de prueba (50 %) y no es mejor que el azar.

¿Qué ha ocurrido? El argumento anterior ha ignorado el hecho de que en la validación cruzada, el modelo debe volver a entrenarse completamente para cada pliegue del proceso. En el presente ejemplo, esto significa que el mejor predictor y el punto de división correspondiente se encuentran a partir de $4/5$ de los datos. El efecto de la elección del predictor se ve en el panel superior derecho. Como las etiquetas de clase son independientes de los predictores, el rendimiento de un muñón en las $4/5$ partes de los datos de entrenamiento no contiene información sobre su rendimiento en las restantes $1/5$. El efecto de la elección del punto de división se

Figura 2.9



Fuente: (Hastie, Tibshirani y Friedman, 2009)

muestra en el panel inferior izquierdo. Aquí vemos los datos del predictor 436, correspondiente al punto azul en el gráfico superior izquierdo. Los puntos coloreados indican los datos de $1/5$, mientras que los puntos restantes pertenecen a las $4/5$ partes. Se indican los puntos de división óptimos para este predictor basados tanto en el conjunto de entrenamiento completo como en los datos de las $4/5$ partes. La división basada en los datos completos no comete errores en los datos de las $1/5$ partes. Pero la validación cruzada debe basar su división en los datos de $4/5$, y esto incurre en dos errores de cada cuatro muestras.

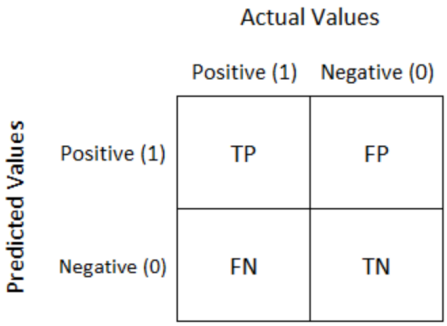
Los resultados de aplicar la validación cruzada de cinco veces a cada uno de los 50 conjuntos de datos simulados se muestran en el panel inferior derecho. Como es de esperar, el error medio de validación cruzada se sitúa en torno al 50 %, que es el verdadero error de predicción esperado para este clasificador. Por lo tanto, la validación cruzada se ha comportado como debería. Por otro lado, existe una variabilidad considerable en el error, lo que subraya la importancia de informar del error estándar estimado de la estimación del CV.

2.6. Evaluación y desempeño de modelos de clasificación

2.6.1. Matriz de confusión

La Matriz de confusión es una técnica utilizada para medir el rendimiento de problemas la clasificación donde la respuesta puede tener dos o más clases. Es una tabla con 4 combinaciones diferentes de valores predichos y reales (ver figura 2.10).

Figura 2.10: Matriz de confusión (caso binario)



Fuente: Elaboración Propia

Es extremadamente útil para medir la Sensibilidad (*Sensitivity*), la especificidad (*Specificity*), la precisión (*Accuracy*) y, lo que es más importante, la curva AUC-ROC [30], donde el significado de cada uno de los términos de la matriz es el siguiente:

- **TP**.- Es el número de predicciones correctas de clase positiva (positivos reales)
- **FP**.- Es el número de predicciones incorrectas de clase positiva (falsos positivos)
- **TN**.- Es el número de predicciones correctas de clase negativa (negativos reales)
- **FN**.- Es el número de predicciones incorrectas de clase negativa (falsos negativos).

Precisión

La precisión o "Accuracy" (AC) se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión. Se representa por la proporción entre el número de

[30] detallada en el siguiente apartado

predicciones correctas (tanto positivas como negativas) y el total de predicciones, y se calcula mediante la ecuación:

$$AC = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.69}$$

Sensibilidad

La Sensibilidad ("Recall"), también se conoce como Tasa de Verdaderos Positivos (*True Positive Rate* o TPR). Es la proporción de casos positivos que fueron correctamente identificadas por el algoritmo. Se calcula según la ecuación:

$$TPR = \frac{TP}{TP + FN} \tag{2.70}$$

Especificidad

La Especificidad, por otra parte, es la Tasa de Verdaderos Negativos, (*True Negative Rate* o TNR). Se trata de los casos negativos que el algoritmo ha clasificado correctamente. Su ecuación es:

$$TNR = \frac{TN}{TN + FP} \tag{2.71}$$

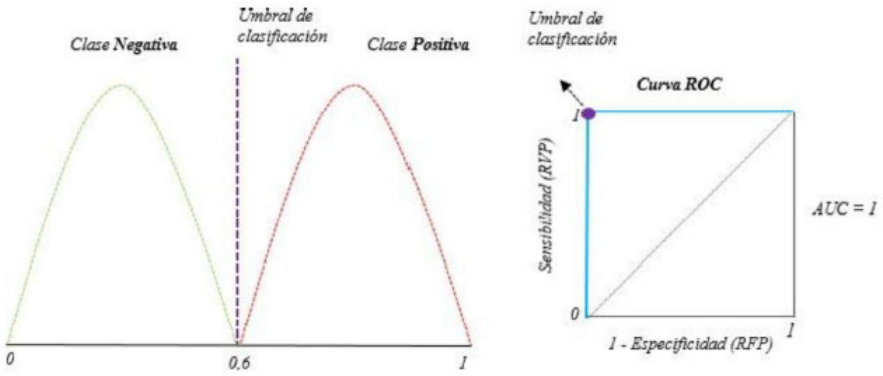
2.6.2. AUC y Curva ROC

El modelo construido divide al nuevo conjunto de datos en dos clases: clase positiva y clase negativa, siendo la clase positiva la clase objetivo. A partir de esto, se obtiene la distribución de cada clase y en base a ellas se genera la curva ROC haciendo recorrer el umbral de clasificación hacia la derecha y hacia la izquierda. Haciendo uso de la curva ROC (*Receiver Operating Characteristic*) se calcula el coeficiente AUC, que no es nada más que el área bajo la curva e indica que tan bueno es el modelo generado (Torres, 2010). A continuación, se muestran la curva ROC y su coeficiente AUC para diferentes casos:

Primer caso

Es el caso ideal de un problema de clasificación en el que las clases están totalmente separadas, es decir el modelo clasifica perfectamente y sin ningún error a positivos en positivos y a negativos en negativos (ver figura 2.11).

Figura 2.11: Caso1: Curva ROC

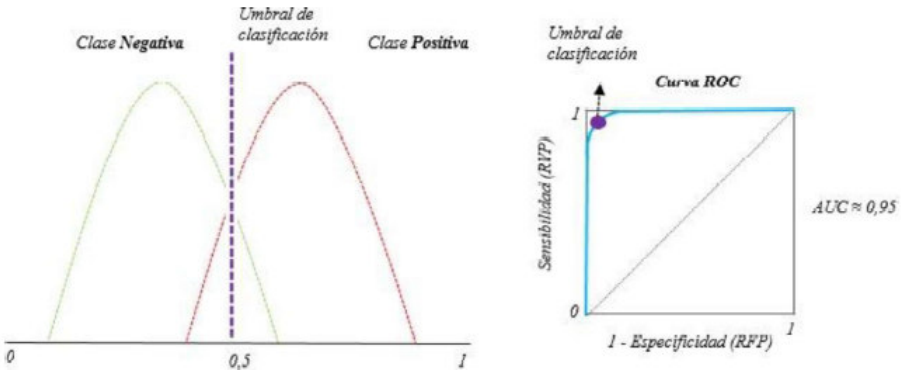


Fuente: (Torres, 2010)

Segundo caso

Se produce cuando las proporciones de positivos y negativos son iguales, pero la clasificación no es del todo correcta y por tanto se generan falsos positivos y falsos negativos (ver figura 2.12).

Figura 2.12: Caso2: Curva ROC



Fuente: (Torres, 2010)

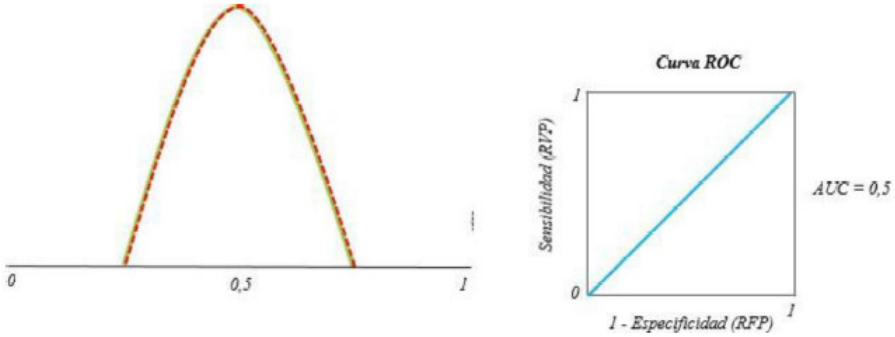
El área bajo la curva ROC o el AUC será menor a uno, pues la clasificación no es la adecuada, ya que existe una parte dónde las clases están solapadas.

Tercer caso

Ahora, al juntar las distribuciones vamos a obtener un nuevo caso, considerado como el peor de ellos, en el que el modelo no logra diferenciar entre positivos y negativos, y cuyo error sería el más grande.(ver figura 2.13).

Por tanto, de los casos mencionados anteriormente se concluye que el AUC es un valor entre $0.5 \leq AUC \leq 1$, y mientras más grande es su valor, más adecuado es

Figura 2.13: Caso3: Curva ROC



Fuente: (Torres, 2010)

el funcionamiento del modelo para la clasificación.

La Curva ROC se puede construir a partir de dos muestras que se encuentren disponibles de los puntajes de clientes malos (morosos) y buenos (no morosos). Estas dos muestras se obtendrían de la predicción del modelo sobre cada una de las poblaciones de clientes en análisis (buenos y malos). El área bajo la curva ROC, llamada comúnmente AUC (Área Bajo la Curva), es una transformación lineal del ratio de precisión, donde en el eje horizontal se encuentra el porcentaje de falsos positivos, es decir, clientes malos que el modelo clasificó como clientes buenos (no morosos), y en el eje vertical se encuentra el porcentaje de verdaderos positivos que son los clientes buenos (no morosos) que fueron clasificados como buenos en el modelo. El área bajo la curva AUC puede tomar valores entre 0.5 y 1. Un porcentaje del AUC del 50 % implica que el modelo no es mejor que hacer una suposición aleatoria; y un valor de 100 % indicaría la improbable aparición de predicciones perfectamente correctas.

Sin embargo, para el caso de sistemas de clasificación lo ideal es tomar valores del AUC que sean mayores a 0.7, y entonces se dirá que esos modelos tienen una buena capacidad de clasificación (Anderson, 2007).

2.6.3. Estadístico de Kolmogorov - Smirnov (KS)

Una de las estadísticas comúnmente utilizadas en la calificación crediticia, es la estadística KS (Anderson, 2007). La cual se basa en un análisis de la función de distribución empírica acumulativa ($ECDF$). Es una medida no paramétrica que se usa para evaluar el error o "bondad de ajuste" en el ajuste de curvas. Segun Anderson

(2007), la estadística KS es la estadística más utilizada para medir el poder predictivo de los sistemas de clasificación. Para calcular el estadístico de Kolmogorov-Smirnov, sea D la función de distribución de las muestras $F_{n1}(x)$ y $F_{n2}(x)$ hay que calcular su máxima diferencia absoluta (siendo la diferencia absoluta máxima entre las dos curvas: la de clientes malos acumulados y clientes buenos acumulados), sobre todos los valores x tal que:

$$D = \max_{x} |F_{n1}(x) - F_{n2}(x)| \tag{2.72}$$

Mays (2004) también menciona que, como medida de la capacidad de clasificación, los valores de KS deben oscilar entre el 20 % (por debajo del cual debe cuestionarse el valor del modelo), y el 70 % (por encima del cual "probablemente sea demasiado bueno para ser cierto").

2.6.4. Coeficiente Gini

Mide la eficacia del modelo. Hace esto comparando el porcentaje de los buenos contra el porcentaje de los malos clientes para los mismos puntajes. Si el porcentaje de malos se traza contra el porcentaje buenos para una serie de bandas de puntajes, el resultado es una curva ABC (ver figura 2.14). El coeficiente Gini es el área entre la curva ABC y la línea de la eficiencia nula AC establecida como un porcentaje del área del triángulo ACZ.

El coeficiente Gini = área ABC / área del triángulo AZC, y su cálculo se lo realiza con la siguiente fórmula:

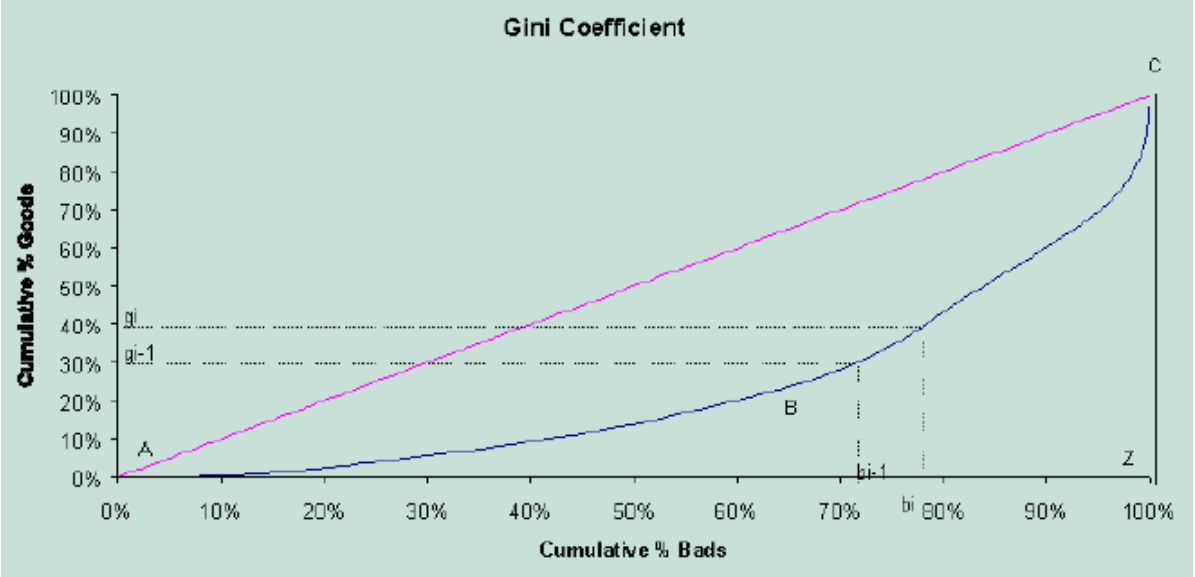
$$Gini = 1 - \sum_{i=1}^{n} \left((b_i - b_{i-1}) * (g_i + g_{i-1}) \right), \quad b_0 = g_0 = 0 \tag{2.73}$$

donde :

- g_i : % acumulado de buenos en un puntaje dado
- g_{i-1} : % acumulado de buenos en un puntaje anterior a g_i
- b_i : % acumulado de malos en un puntaje dado
- b_{i-1} : % acumulado de malos en un puntaje anterior a b_i

En la práctica i representa al percentil i, es decir se divide a toda la población en percentiles y para cada percentil se calcula el % acumulado de buenos y malos

Figura 2.14: Ejemplo Índice de Gini



Fuente: Elaboración Propia

clientes. Según Lisim [31], de su experiencia empírica, un coeficiente Gini debe estar alrededor del 30 % para un modelo de aprobación que incluye información de la solicitud, y estará generalmente más cercano al 60 % para un modelo de comportamiento, dependiendo de las limitaciones sobre los datos.

2.6.5. Tabla de ODDS

Las tablas de ODDS también conocidas como tablas de performance o rendimiento, son herramientas en las cuales podemos visualizar la calidad de discriminación del modelo por cada decil de la probabilidad de default estimada. Generalmente, para analizar el rendimiento de clasificación del modelo se particiona la probabilidad estimada en 10 intervalos y se analiza el número y porcentaje de sujetos totales, sujetos defraudadores y sujetos no defraudadores en cada uno de ellos.

Una tabla de performance tiene una estructura similar a la tabla de la Figura 2.15, en dónde podemos observar ciertas columnas que proporcionan información relevante, divididas en cuatro bloques:

- **Probabilidad**: Se presentan los límites inferior (*Min*) y superior (*Max*) de la probabilidad de default estimada, subdividida comúnmente en 10 rangos.

[31] Consultora colombiana experta en temas de Scoring.

Figura 2.15: Tabla de Odds, aplicado a un modelo de fraude

KS	ROC	Gini							
43,7	77,5	55,0							
Probabilidad		Total			Fraude			Razón de Fraude	
Min	Max	Int#	Int%	Cum%	Int#	Int%	Cum%	Int	Cum
0,765	0,999	12.151	10%	10%	1.529	48%	48%	12,58%	12,6%
0,599	0,765	12.288	10%	20%	478	15%	62%	3,89%	8,2%
0,525	0,599	11.739	10%	29%	275	9%	71%	2,34%	6,3%
0,480	0,525	12.487	10%	40%	212	7%	78%	1,70%	5,1%
0,455	0,480	10.331	8%	48%	151	5%	82%	1,46%	4,5%
0,430	0,455	13.588	11%	59%	183	6%	88%	1,35%	3,9%
0,416	0,430	10.149	8%	67%	115	4%	92%	1,13%	3,6%
0,401	0,416	11.866	10%	77%	105	3%	95%	0,88%	3,2%
0,387	0,401	13.517	11%	88%	93	3%	98%	0,69%	2,9%
0,001	0,387	14.643	12%	100%	72	2%	100%	0,49%	2,6%
Total		*122.759*			*3.213*				

Fuente: (Pérez, 2019)

- **Total**: Se presentan la cantidad total de sujetos en cada rango ($Int\#$); el porcentaje dentro del bloque total, 10 aproximadamente por la división en deciles; y el porcentaje acumulado del total($Cum\,\%$).
- **Default**: Se presentan la cantidad total de sujetos etiquetados como *default* en cada rango ($Int\#$), el porcentaje dentro del bloque $default$ y el porcentaje acumulado de default ($Cum\,\%$).
- **Razón de default**: Se presentan la tasa de default dentro de cada rango ($Int = \frac{Int\#default}{Int\#Total}$) y el porcentaje acumulado de default respecto al total de sujetos.

Finalmente, se describen ciertas propiedades importantes que debe disponer un modelo de clasificación para ser considerado válido y adecuado:

- Un modelo adecuado captará porcentajes significativos de default en los deciles más altos.
- El porcentaje o tasa de default debe aumentar conforme la probabilidad va creciendo.

2.7. Interpretable Machine Learning Models

En este apartado, se presentará las nociones de modelos interpretables, en especial ahondaremos en LIME. Los siguientes resultados fueron obtenidos a partir de los trabajos de Escalante et al. (2018) y Rothman (2020).

Hoy en día, se dedica cada vez más interés al concepto de "aprender de los datos" es decir, utilizar los datos recopilados sobre el proceso para predecir su resultado (Hastie, Tibshirani y Friedman, 2009). Los ingredientes principales de su éxito reciente son la gran disponibilidad de fuentes de datos y el aumento de la potencia computacional, que permite que algoritmos complejos proporcionen resultados en un tiempo relativamente corto.

En estadística, hacer predicciones sobre el futuro es un tema de especial relevancia. Para abordar el tema, se han desarrollado algoritmos y métodos simples a lo largo de los años, siendo los más famosos la regresión lineal y los modelos lineales generalizados. Sin embargo, con la llegada de potentes herramientas informáticas, se han desarrollado técnicas más sofisticadas. En particular, los modelos de aprendizaje automático pueden realizar tareas inteligentes que suelen realizar los humanos, lo que respalda la automatización de procesos impulsados por datos.

A pesar de la precisión mejorada, los modelos de aprendizaje automático muestran debilidades, especialmente en lo que respecta a la interpretabilidad, es decir, "la capacidad de explicar o presentar los resultados, en términos comprensibles, a un humano" (Hall y Gill, 2018). Por lo general, adoptan grandes estructuras de modelos y refinan la predicción utilizando una gran cantidad de iteraciones. La lógica subyacente al modelo termina escondida bajo potencialmente muchos estratos de cálculos matemáticos, así como dispersa en una arquitectura demasiado vasta, lo que impide que los humanos la capten.

Para lograr la interpretabilidad, se han propuesto bastantes técnicas en la literatura reciente. Estos enfoques se pueden agrupar en función de diferentes criterios Molnar (2020), Guidotti et al. (2018) tales como

- Local, global o basado en ejemplos.
- Modelo agnóstico [32] o modelo específico.
- Intrínseco o post-hoc.
- Basado en perturbaciones o prominencia.

[32] El enfoque de un agnóstico consiste en utilizar modelos de aprendizaje automático para estudiar la estructura subyacente sin asumir que el modelo puede describirla con precisión debido a su naturaleza.

Los métodos explicables se agrupan en técnicas de explicabilidad global y local (Guidotti et al., 2018). Los métodos globales tienen como objetivo brindar una comprensión del modelo en su conjunto: la explicación debe aplicarse a todos los registros del conjunto de datos. En cambio, los métodos locales intentan proporcionar una muy buena comprensión solo para una pequeña parte de los registros.

En este trabajo, nos enfocamos en LIME (Local Interpretable Model-agnostic Explanations), un framework [33] de interpretación local, desarrollado por Ribeiro et al. (2016).

2.7.1. LIME

LIME (Ribeiro et al., 2016) es un método para explicar modelos de caja negra, es decir, modelos cuya lógica interna está oculta y no es claramente comprensible. En este apartado se da a conocer brevemente dicho método, pero para su mejor comprensión se sugiere se lea detenidamente el artículo "Why Should I Trust You? Explaining the Predictions of Any Classifier" expuesto por (Ribeiro et al., 2016).

Antes de presentar el sistema de explicación, es importante distinguir entre características y representaciones de datos interpretables. Como se mencionó anteriormente, las explicaciones interpretables deben usar una representación que sea comprensible para los humanos, independientemente de las características reales utilizadas por el modelo. Por ejemplo, una posible *representación interpretable* para la clasificación de texto es un vector binario que indica la presencia o ausencia de una palabra, aunque el clasificador puede usar características más complejas (e incomprensibles) como las incrustaciones de palabras. Asimismo, para la clasificación de imágenes, una *representación interpretable* puede ser un vector binario que indica la "presencia" o "ausencia" de un parche contiguo de píxeles similares (un superpíxel), mientras que el clasificador puede representar la imagen como un tensor con tres canales de color por píxel.

Sea $x \in \mathbb{R}^d$ la representación original de una instancia que se está explicando, y usamos $x' \in \{0,1\}^{d'}$ para denotar un vector binario para su representación interpretable.

[33] Un framework es un conjunto particular de reglas, ideas o creencias que utiliza en la resolución de problemas.

Formalmente, se define una explicación como un modelo $g \in G$, donde G es una clase de modelos potencialmente interpretables, tal como la regresión lineal, árboles de decisión, etc., es decir, un modelo $g \in G$ puede ser presentado fácilmente al usuario con artefactos visuales o textuales.

El dominio de g es $\{0,1\}^{d'}$, es decir, g actúa sobre la ausencia/presencia de los componentes interpretables. Como no todos los $g \in G$ pueden ser lo suficientemente simples como para ser interpretables, definimos $\Omega(g)$ como una medida de complejidad (contraposición de la interpretabilidad) de la explicación $g \in G$. Por ejemplo, para los árboles de decisión $\Omega(g)$ puede ser la profundidad del árbol, mientras que para modelos lineales $\Omega(g)$ puede ser el número de coeficientes distinto de cero.

Denotemos el modelo que está siendo explicado por $f : \mathbb{R}^d \longrightarrow \mathbb{R}$. En problemas de clasificación, $f(x)$ es la probabilidad (o indicador binario) que x pertenezca a cierta clase. Además usamos $\pi_x(z)$ como la medida de proximidad entre una instancia z y x para definir la localidad alrededor de x.

Finalmente, sea $L(f, g, \pi_x)$ una medida de cuán distante es g al aproximar f en una vecindad definida por π_x. Para asegurarnos tanto de la interpretabilidad como la fidelidad local, debemos minimizar $L(f, g, \pi_x)$ mientras que $\Omega(g)$ sea lo suficientemente bajo para ser interpretado por humanos. La explicación producida por LIME se obtiene de la siguiente manera:

$$\xi(x) = \underset{g \in G}{\arg\min}\, L(f, g, \pi_x) + \Omega(g) \tag{2.74}$$

Esta formulación se puede utilizar con diferentes familias de explicaciones G, funciones de fidelidad L y medidas de complejidad Ω. Aquí nos enfocamos en modelos lineales dispersos como explicaciones y en realizar la búsqueda usando perturbaciones.

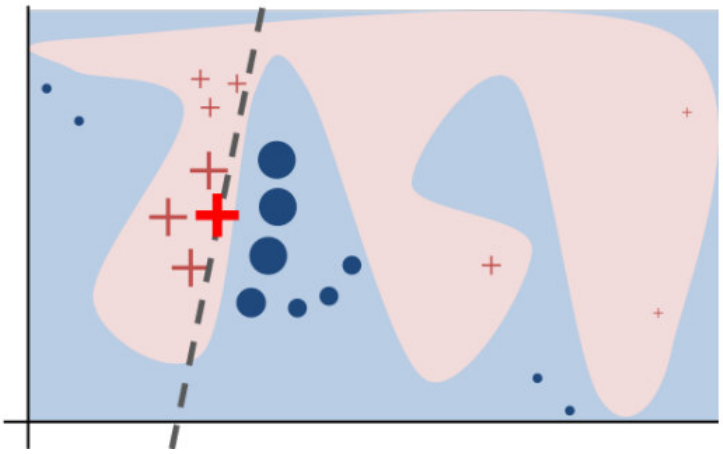
Queremos minimizar la pérdida consciente de la localidad $L(f, g, \pi_x)$ sin hacer suposiciones sobre f, ya que queremos que el explicador sea un modelo agnóstico. Por lo tanto, para aprender el comportamiento local de f a medida que varían las entradas interpretables, aproximamos $L(f, g, \pi_x)$ extrayendo muestras, ponderadas

por π_x.

Tomamos muestras aleatorias de instancias alrededor de x' dibujando elementos de x' que sean distintos de cero y obtenidos de manera uniforme (donde el número de tales extracciones también se muestrea uniformemente). Dada una muestra perturbada $z' \in \{0,1\}^{d'}$ (que contiene una fracción de los elementos distintos de cero de x'), recuperamos la muestra en la representación original $z \in R^d$ y obtenemos $f(z)$, que se usa como etiqueta para el modelo de explicación.

Dado este conjunto de datos Z de muestras perturbadas con las etiquetas asociadas, optimizamos la ecuación 2.74 para obtener una explicación $\xi(x)$. La intuición principal detrás de LIME se presenta en la Figura 2.16, donde se obtienen muestras de instancias tanto en la vecindad de x (que tienen un alto peso debido a π_x) como lejos de x (bajo peso de π_x). Aunque el modelo original puede ser demasiado complejo para explicarlo globalmente, LIME presenta una explicación que es localmente fiel (lineal en este caso), donde la localidad es capturada por π_x. Vale la pena señalar que el método es bastante robusto para muestrear ruido, ya que las muestras se ponderan con π_x en la ecuación 2.74.

Figura 2.16: Ejemplo básico para presentar la intuición de LIME.



Fuente: (Ribeiro, Singh y Guestrin, 2016)

En la figura 2.16 La función de decisión compleja f del modelo de caja negra (desconocida para LIME) está representada por el fondo azul/rosa, que no puede aproximarse bien mediante un modelo lineal. La cruz roja en negrita es el caso que se explica. LIME toma muestras de instancias, obtiene predicciones usando f y las pesa por la proximidad a la instancia que se explica (representada aquí por tama-

ño). La línea discontinua es la explicación aprendida que es fiel localmente (pero no globalmente).

Características importantes

Lime es un framework que tiene como objetivo hacer que las predicciones de cualquier modelo de aprendizaje automático sean más interpretables, se podría decir que Lime recibe un modelo de aprendizaje automático (caja negra) e investiga la relación entre los datos de entrada y de salida, representada por el modelo.

Lime es independiente del modelo, lo que significa que se puede aplicar a cualquier modelo de aprendizaje automático. La técnica intenta comprender el modelo perturbando la entrada de muestras de datos y comprendiendo cómo cambian las predicciones.

Los enfoques específicos del modelo tienen como objetivo comprender el modelo de aprendizaje automático del modelo negro mediante el análisis de los componentes internos y cómo interactúan.

Capítulo 3

Metodología Analítica y Resultados

El presente capítulo tiene como objetivo describir los principales pasos de la metodología utilizada en la creación y validación del modelo de *credit scoring*. Se busca presentar una visión completa de todo el proceso de modelización, empezando por el procesamiento de la información en donde se revisarán temas como la descripción de la base de datos, la selección de la ventana de muestreo, etc.; llegando a tener como resultado de esta etapa las tablas de datos que se utilizarán tanto para el entrenamiento como la evaluación del modelo. En la segunda sección se detallan todos los puntos considerados en la estimación del modelo, y éste será el resultado de esta etapa; para en la tercera sección trabajar exclusivamente en la validación estadística del modelo. En el cuarto y último apartado se detallan todas las consideraciones que se tomaron en cuenta en la estimación del modelo LIME, útil para la interpretabilidad de las predicciones del modelo XGBoost.

Todos los cálculos serán realizados en el software estadístico R[1], y los códigos se adjuntan en el apéndice B

[1] R es un entorno de software libre para computación estadística y gráficos. Compila y se ejecuta en una amplia variedad de plataformas UNIX, Windows y MacOS

3.1. Procesamiento de la información

3.1.1. Descripción de la Base de Datos

En este estudio se utilizará una base de datos de corte transversal [2] de una cartera de créditos de consumo concedidos por una institución bancaria del Ecuador, recordemos que el segmento objetivo de este estudio son las personas sin antecedentes crediticios, y tal como se mencionó en la introducción se busca calificar a los clientes bajo esas circunstancias, es por ello que para la selección de los casos de estudio se tuvieron las siguientes consideraciones:

- Cada caso, deberá ser una operación de crédito analizada en su fecha de concesión, es decir, se observarán las características que presentaba el cliente al momento del otorgamiento del crédito para luego evaluar su comportamiento de pago en el futuro.
- Se eligen las operaciones de clientes que antes de la fecha de concesión no presentaban un historial crediticio (se les otorgó el crédito por el análisis de un asesor especializado).
- Debe considerarse como fecha máxima de estudio, aquella que permita observar a la operación de crédito 12 meses en el futuro (ventana de análisis, detallada en el siguiente apartado).

La información disponible cuenta con un total de 59076 registros y 35 variables explicativas divididas en tres bloques importantes de datos que se detallan a continuación:

- **Información demográfica** que contiene datos como la edad, estado civil, ciudad de nacimiento, profesión y nivel de educación.
- **Pasivos bancarios** Contiene información acerca de los saldos en las cuentas o inversiones que posee el cliente con la entidad financiera.
- **Buró de Crédito del núcleo familiar** Al tratarse de un modelo en el cuál no se posee información de buró del cliente, se intenta capturar el comportamiento de pago a través de los datos de buró del núcleo familiar del individuo, basados en la idea de Contagio y Riesgo Sistémico en Redes, presentada en el trabajo de Dolfin, et al (2019).

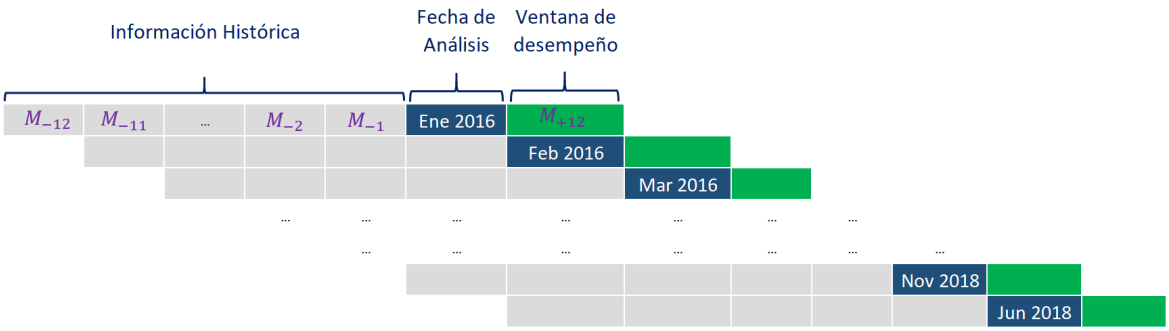
[2] Observaciones en un momento determinado y, en donde, el orden de las observaciones es irrelevante

3.1.2. Selección de la ventana de muestreo

En base a la recomendación de la Superintendencia de Bancos y Seguros del Ecuador sobre creación de modelos de *credit scoring*, la entidad bancaria construye todos sus modelos con al menos 3 años de información histórica (Superintendencia de Bancos Ecuador, 2019, p.5), es por ello que para el estudio se cuenta con los datos de operaciones de crédito desde enero 2016 hasta junio 2019.

Para comprender de mejor manera la generación de variables involucradas en el desarrollo del modelo, observemos la Figura 3.1

Figura 3.1: Esquema de generación de información



Fuente: Elaboración Propia

- **Fecha de Análisis (azul):** Corresponde a los meses en donde se selecciona la población de modelamiento (M_0), es decir, se seleccionarán como casos de estudio todas las operaciones de crédito que fueron concedidos en esas fechas en específico. Se consideran todos los meses desde enero 2016 hasta junio 2018, dejando los 12 meses posteriores para la ventana de desempeño que se describirá a continuación.

- **Ventana de comportamiento (gris):** Es la ventana construida por 12 meses anteriores a la fecha de análisis. ($M_{-1}, M_{-2}..., M_{-12}$). Sobre este periodo se construyen las variables independientes que son aquellas que permitirán estimar la probabilidad de que un cliente sea un mal pagador.

- **Ventana de desempeño (verde):** Es aquella ventana de tiempo posterior a la fecha de análisis, generalmente 12 meses (M_{+12}) por recomendación de Basilea, en esta ventana se cuenta con variables de comportamiento de pago como días mora, saldos vencidos, entre otros. Con el fin de evaluar las operaciones y

determinar la variable dependiente cuya construcción se detalla en el apartado 3.1.3.

La distribución de registros seleccionados de acuerdo a la fecha de concesión se presenta en la Tabla 5.1

Cuadro 3.1: Distribución del número de registros por fecha de análisis

Fecha_Análisis	Num_Registros	Porc_Clientes
Ene 2016	601	1.0 %
Feb 2016	1343	2.3 %
Mar 2016	1712	2.9 %
Abr 2016	1128	1.9 %
May 2016	1879	3.2 %
Jun 2016	1084	1.8 %
Jul 2016	967	1.6 %
Ago 2016	1347	2.3 %
Sep 2016	3282	5.6 %
Oct 2016	2412	4.1 %
Nov 2016	1941	3.3 %
Dic 2016	1063	1.8 %
Ene 2017	1153	2.0 %
Feb 2017	1880	3.2 %
Mar 2017	2903	4.9 %
Abr 2017	4566	7.7 %
May 2017	4943	8.4 %
Jun 2017	1945	3.3 %
Jul 2017	1848	3.1 %
Ago 2017	3026	5.1 %
Sep 2017	2439	4.1 %
Oct 2017	1342	2.3 %
Nov 2017	832	1.4 %
Dic 2017	1193	2.0 %
Ene 2018	1393	2.4 %
Feb 2018	1186	2.0 %
Mar 2018	3264	5.5 %
Abr 2018	2804	4.7 %
May 2018	1972	3.3 %
Jun 2018	1628	2.8 %

Fuente: Elaboración Propia

3.1.3. Definición de la variable dependiente

La definición de la variable dependiente (Y) tiene un impacto muy grande en el proceso de modelización, debido a que engloba el fenómeno que se desea analizar. La construcción de la variable dependiente se realiza utilizando la información de días mora de un cliente disponible en la ventana de desempeño, y esta variable será denominada *default*.

Para definir el *default* o incumplimiento se consideran 3 tipos de condiciones que el cliente debe cumplir y que se detallan a continuación:

- Que el cliente haya alcanzado o superado los 90 días de mora en cualquier mes de la ventana de desempeño (12 meses a partir de la concesión del crédito).
- Que la operación haya sido castigada en algún punto de la ventana de desempeño.
- Que la operación haya sido demandada en algún punto de la ventana de desempeño.

Estas condiciones nacen de las recomendaciones de los acuerdos de Basilea (Bassel Commite on Banking Supervision, 2005) y de la experiencia de la entidad bancaria en la creación de modelos similares para otros segmentos de clientes, a tal punto que constan en la políticas de la entidad bancaria para la creación de modelos de *credit scoring*. Esta situación se da debido a que en la entidad financiera existen muchos otros modelos para los diferentes segmentos de clientes y tipos de créditos, razón por la cual, la organización necesita que los resultados de los modelos sean comparables, es decir, un cliente malo signifique lo mismo en todos los modelos. Los clientes que cumplan con cualquiera de estas tres condiciones, serán categorizado como clientes malos y como se trata de una clasificación binaria todo aquel cliente que no sea malo será catalogado como bueno.

A pesar que la definición de la variable dependiente estaba preestablecida, fue importante también realizar el análisis *Roll Rate* [3] para determinar si la definición antes mencionada asegura las condiciones para tener una variable dependiente robusta para este modelo. Los resultados del análisis se muestran en la Figura 3.2, en donde las filas representan el rango de morosidad máxima que alcanzó un cliente,

[3] Es un modelo de Markov simple en el que las cuentas se agrupan según su estado de morosidad durante X meses y, posteriormente, si la cuenta entró en incumplimiento en los próximos Y meses

las columnas son los meses de maduración tomados en cuenta en el análisis y los valores nos indican el porcentaje de los clientes que no terminaron pagando el crédito al final del plazo establecido.

Figura 3.2: Análisis Roll Rate

		Meses de Maduración		
		6	9	12
Rango de Morosidad Máxima	15	14%	17%	25%
	30	43%	50%	59%
	60	72%	75%	79%
	90	90%	91%	93%

Fuente: Elaboración Propia

Se puede ver que de todos los clientes con un rango de mora igual o superior a los 90 días en un periodo de madurez de 12 meses, el 93 % de ellos no terminan cancelando el crédito, es decir, solamente el 7 % de ellos se recuperan y vuelven a paga, esto nos asegura que la definición dada por la organización se ajusta perfectamente a las características de este problema. A pesar que una combinación de 90 días y 9 meses de madurez parecería ser una opción muy tentadora como definición de la variable objetivo ya que permitiría detectar a los clientes malos con más antelación (3 meses), la condición de que los diferentes modelos sean comparables prima sobre este resultado.

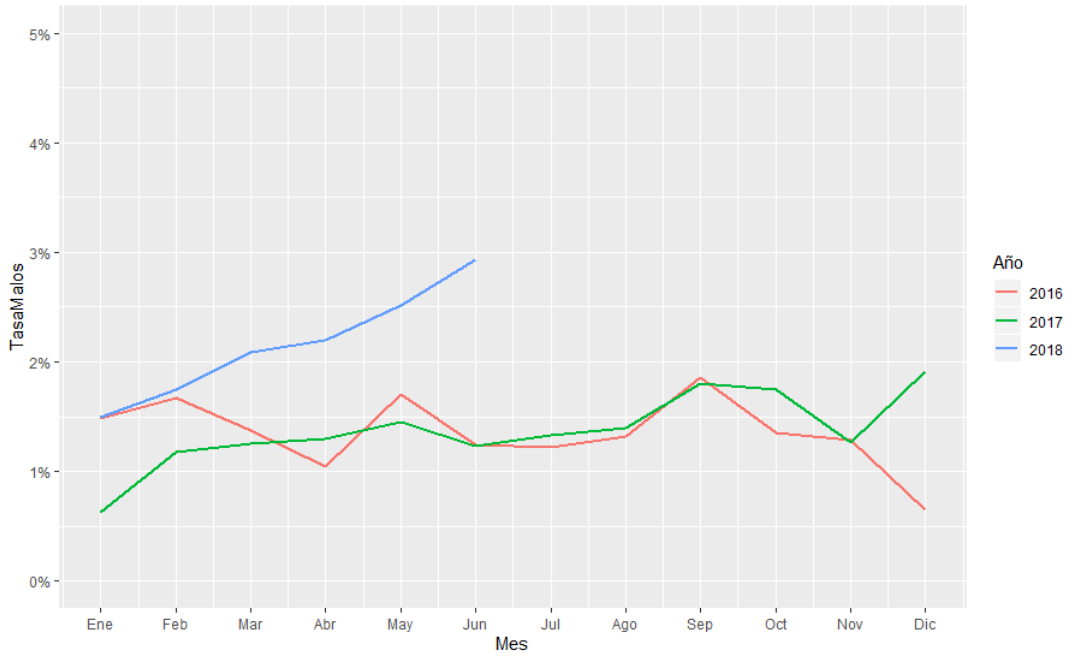
La variable Y definida bajo las consideraciones antes mencionadas, al interactuar en el modelo analítico con las variables mencionadas en la sección 3.1.1 entregarán como resultado una probabilidad de incumplimiento o *default* para cada cliente, cuantificando la situación de que si otorgase el crédito a un cliente nuevo, este incumpla con sus obligaciones en los 12 siguientes meses.

Comportamiento de la variable dependiente

Una vez se ha asignado el valor de la variable Y a cada cliente, procedemos a analizar su comportamiento a través del tiempo. A continuación se muestra gráficamente la tasa de malos [4] durante el periodo de estudio.

[4]Cantidad de clientes malos (definidos como *default*) dividido para el total de clientes en la fecha específica.

Figura 3.3: Tasa de malos vs FechaAnálisis



Fuente: Elaboración Propia

En la figura 3.3 se observa que la tasa de malos no presenta variaciones significativas para las operaciones concedidas en los años 2016 y 2017, pero vemos que a partir del segundo mes de 2018 presenta un crecimiento significativo en relación a los años anteriores, esto puede deberse a que en ese año se aplicaron políticas de crecimiento de cartera.

3.1.4. Análisis Descriptivo de la Base de Datos

Con el fin de escoger las variables que aportan mayor información al modelo predictivo, en una primera aproximación se realizará el análisis exploratorio de los datos.

Las bases de datos con las que se trabajará son completamente internas, como concierne en un modelo de originación, la variedad de información proporcionada para el estudio del modelo estadístico, incluye información de calidad pero también mucha información irrelevante. En la tabla 3.2 se muestra una descripción de las variables con las que se cuenta inicialmente.

Cuadro 3.2: Descripción de las variables en la base de datos

Variable	Tipo de Dato	Descripción
FechaAnalisis	Fecha	Fecha en la que se desembolsó el crédito
Y	Entera	Variable dependiente
Genero	Categórica	Género del cliente
EstadoCivil	Categórica	Estado civil del cliente
NivelEstudio	Categórica	Nivel de estudio del cliente
CodigoProfesion	Categórica	Código de la profesión del cliente
Hijos	Numérica	Número de hijos que tiene el cliente
Latitud	Numérica	Latitud geográfica del domicilio del cliente
Longitud	Numérica	Longitud geográfica del domicilio del cliente
AntiguedadLaboral	Numérica	Número de meses desde inicio de sus actividades laborales a la fecha de análisis
Edad	Numérica	Edad del cliente
LongitudAgencia	Numérica	Latitud geográfica de la agencia de anclaje del cliente
LatitudAgencia	Numérica	Longitud geográfica de la agencia de anclaje del cliente
AntiguedadPasivo	Numérica	Tiempo en meses desde que el cliente adquirió su primero pasivo bancario
NumeroCuentasActivas	Numérica	Número de cuentas activas que tiene el cliente en la fecha de análisis
NumeroCuentasCerradas	Numérica	Número de cuentas que el cliente ha cancelado hasta la fecha de análisis
CantidadCuentasAF	Numérica	Número de cuentas de tipo Ahorro Futuro que tiene el cliente en la FA
CantidadCuentasCO	Numérica	Número de cuentas de tipo Corriente que tiene el cliente en la FA
CantidadCuentasAH	Numérica	Número de cuentas de tipo Ahorro que tiene el cliente en la FA
SaldoUltimos30Dias	Numérica	Saldo del cliente en sus cuentas 30 días antes de la fecha de análisis
SaldoUltimos90Dias	Numérica	Saldo del cliente en sus cuentas 90 días antes de la fecha de análisis
SaldoUltimos180Dias	Numérica	Saldo del cliente en sus cuentas 180 días antes de la fecha de análisis
SaldoUltimos360Dias	Numérica	Saldo del cliente en sus cuentas 360 días antes de la fecha de análisis
TotalSaldoUltimos90Dias	Numérica	Suma del saldo de las cuentas en los últimos 90 días antes de la FA
TotalSaldoUltimos180Dias	Numérica	Suma del saldo de las cuentas en los últimos 180 días antes de la FA
TotalSaldoUltimos360Dias	Numérica	Suma del saldo de las cuentas en los últimos 360 días antes de la FA
PromedioSaldoUltimos90Dias	Numérica	Promedio mensual del saldo de las cuentas en los últimos 90 días antes de la FA
PromedioSaldoUltimos180Dias	Numérica	Promedio mensual del saldo de las cuentas en los últimos 180 días antes de la FA
PromedioSaldoUltimos360Dias	Numérica	Promedio mensual del saldo de las cuentas en los últimos 360 días antes de la FA
ScoreBuroPadre	Numérica	Calificación del buró de crédito del padre del cliente en la fecha de análisis
ClasificacionPadre	Categórica	Perfil del cliente en buró de crédito del padre del cliente en la fecha de análisis
ScoreBuromadre	Numérica	Calificación del buró de crédito de la madre del cliente en la fecha de análisis
Clasificacionmadre	Categórica	Perfil del cliente en buró de crédito de la madre del cliente en la fecha de análisis
ScoreBuroConyuge	Numérica	Calificación del buró de crédito del cónyuge del cliente en la fecha de análisis
ClasificacionConyuge	Categórica	Perfil del cliente en buró de crédito del cónyuge del cliente en la fecha de análisis

Fuente: Elaboración Propia

Con el fin de obtener una visión general de dicha información en los siguientes apartados se procede a calcular algunos estadísticos descriptivos para los predictores cuantitativos y presentar algunos valores relevantes para los predictores cualitativos, además el código de R a partir del cuál se obtuvieron los resultados se adjunta en el apéndice B.1.

Variables Cuantitativas

En este apartado se procederá a calcular los siguientes valores para cada una de las variables:

- Porcentaje de valores perdidos
- Porcentaje de ceros
- Media
- Desviación estándar
- Mínimo
- Cuartil 1
- Mediana
- Cuartil 3
- Máximo

Cuadro 3.3: Análisis exploratorio de variables numéricas

Variable	Por_Nulos	Por_Ceros	Media	Desv_Est	Minimo	Cuartil1	Mediana	Cuartil3	Maximo
Hijos	66.35	0	2.17	1.2	1	1	2	3	14
Latitud	0.05	0.15	-0.91	1.14	-25.26	-1.78	-0.33	-0.17	37.39
Longitud	0.05	0.15	-78.92	3.29	-100.32	-79.8	-78.65	-78.5	0
AntiguedadLaboral	12.23	0.04	46.23	30.68	0	25	38	61	284
Edad	0	0	35.23	11.22	18	26	32	43	73
LongitudAgencia	1.49	0	-0.86	1.04	-4.07	-1.67	-0.29	-0.18	1.29
LatitudAgencia	1.49	0	-79	1.01	-90.32	-79.72	-78.61	-78.48	-75.74
AntiguedadPasivo	0	0.1	72.31	63.29	0	26	52	101	705
NumeroCuentasActivas	0	0	1.48	0.62	1	1	1	2	7
NumeroCuentasCerradas	0	0	1.48	0.62	1	1	1	2	7
CantidadCuentasAF	0	67.46	0.33	0.48	0	0	0	1	5
CantidadCuentasCO	0	93.77	0.06	0.26	0	0	0	0	6
CantidadCuentasAH	0	6.2	1.02	0.4	0	1	1	1	10
SaldoUltimos30Dias	0	0.25	1468.57	5779.07	0	117.04	351.42	924.27	372958.09
SaldoUltimos90Dias	0	1.46	1379.34	5713.82	-1770.71	52.65	290.04	872.2	450410.06
SaldoUltimos180Dias	0	2.44	1262.7	4831.3	-302.63	36.58	253.74	795.06	232402.93
SaldoUltimos360Dias	0	20.29	939.32	4007.24	-3823.81	1.06	92.86	544.44	247760.27
TotalSaldoUltimos90Dias	0	0.16	4278.06	16715.29	0	388.2	1036.89	2805.95	1275997.82
TotalSaldoUltimos180Dias	0	0.14	8196.48	29534.16	0	777.57	2044.76	5533.47	1775535.43
TotalSaldoUltimos360Dias	0	0.13	14703.25	50204.12	-7806.89	1417.16	3759.71	10326.67	2595260.57
PromedioSaldoUltimos90Dias	0	0.16	191.47	807.87	0	14.55	42.04	123.32	63856.64
PromedioSaldoUltimos180Dias	0	0.14	364.2	1434.9	0	29.49	83.94	242.57	122894.21
PromedioSaldoUltimos360Dias	0	0.13	626.39	2170.38	-229.61	55.12	154.06	434.86	122894.23
ScoreBuroPadre	78.38	2.57	723.64	350.41	0	527	924	964	999
ScoreBuromadre	79.23	2.36	729.16	342.45	0	566	924	960	999
ScoreBuroConyuge	81.15	1.23	795.99	286,48	0	801	931	962	999

Fuente: Elaboración Propia

Basados en la tabla 3.3, podemos sacar algunas conclusiones acerca de la calidad de los datos. Por ejemplo, la variable Hijos posee un 66.35 % de valores perdidos y no presenta ningún caso de valores iguales a cero. por tanto, nos lleva a pensar

que los valores Nulos se generan por el algoritmo que calcula este campo [5] y no significa la ausencia de un valor sino simplemente que no se encuentra una relación padre-hijo en las bases de datos, razón por la cual se procederá a reemplazar todos estos valores Nulos por el valor cero. Por otro lado también podemos observar la presencia de valores negativos en los saldos de las cuentas (observar el valor mínimo en las variables relacionadas con los 'Saldos') y a pesar que esto pueda parecer un error en el cálculo de los datos, estos casos se presentan por el tipo de cuenta que posee el cliente y las características del producto, ya que los sobregiros [6] se registran en valores negativos en las tablas de datos de Saldos de Cuentas. De esta manera junto con la validación de colaboradores expertos se pudo llegar a determinar que los datos mostrados en la tabla 3.3 presentan concordancia con lo que pasa en la realidad.

Variables Cualitativas

En este apartado se procederá a calcular los siguientes valores para cada una de las variables:

- Porcentaje de Nulos
- Número de categorías
- Nombre de la clase mayoritaria
- Frecuencia categoría mayoritaria
- Nombre de la clase minoritaria
- Frecuencia categoría minoritaria

Cuadro 3.4: Análisis exploratorio de variables categóricas

Variable	Por_Nulos	Num_Categorias	Clase_mayoritaria	Freq_Clas_mayoritaria	Clase_minoritaria	Freq_Clas_minoritaria
Genero	0	2	MASCULINO	56.94	FEMENINO	43.06
EstadoCivil	0	5	SOLTERO	50.38	UNION LIBRE	0.66
NivelEstudio	0	9	BACHILLER	41.54	NINGUNA	0.05
CodigoProfesion	0	955	E30	30.38	A05	0.001
ClasificacionPadre	78.39	6	1	26.56	6	10.33
Clasificacionmadre	79.23	6	1	24.47	6	11.54
ClasificacionConyuge	81.15	6	1	25.38	4	8.68

Fuente: Elaboración Propia

Dada la naturaleza del problema, las pocas variables categóricas presentes en las fuentes de datos y sus respectivas características se presentan a detalle en la

[5] Después se validó el código y se corroboró la hipótesis.

[6] Un sobregiro ocurre cuando se retira dinero de una cuenta bancaria y el saldo disponible desciende por debajo de cero. En esta situación, se dice que la cuenta está "sobregirada".

tabla 3.4, en donde lo más importante a resaltar es que las variables provenientes de fuentes de buró (para el respectivo núcleo familiar) presentan alrededor de un 80 % de valores perdidos, en este caso la falta de datos se da porque la entidad no realizó la consulta de los individuos a la entidad de buró correspondiente en la fecha mencionada. En el apartado 3.1.6 se detallará como se procederá a trabajar con estas variables antes del entrenamiento del modelo.

3.1.5. Análisis Bivariado

A continuación se presenta un análisis estadístico sobre la relación que existe entre cada una las variables independientes y la variable dependiente, además el código de R a partir del cuál se obtuvieron los resultados se adjunta en el apéndice B.1.95

Variable dependiente vs variables cuantitativas

Para cuantificar la relación que existe entre las variables cuantitativas y la variable objetivo haremos uso del estadístico KS (Kolmogorov-Smirnov), con el cual se determina la eficacia que tienen los modelos. Mientras la estadística KS sea más grande, por ende, la variable analizada tendrá mayor poder predictivo y entonces el modelo será más eficaz (Yap, Ong y Mohamed, 2011).

Para calcular el estadístico de Kolmogorov-Smirnov, sea D la función de distribución de las muestras $F_{n1}(x)$ y $F_{n2}(x)$ hay que calcular su máxima diferencia absoluta (siendo la diferencia absoluta máxima entre las dos curvas: la de clientes malos acumulados y clientes buenos acumulados), sobre todos los valores x tal que:

$$D = \max_{x} |F_{n1}(x) - F_{n2}(x)| \tag{3.1}$$

Se realiza un análisis con las variables cuantitativas presentes en la base de datos de este estudio. En la tabla 3.5 se muestra el top 10 de las variables cuantitativas ordenadas por su respectivo estadístico KS, mientras que en el ANEXO C.1 se mostrará el cálculo del estadístico KS para el total de las variables cuantitativas.

Cuadro 3.5: Estadístico KS para las variables numéricas

Variable	KS
SaldoUltimos90Dias	13.127
SaldoUltimos360Dias	12.460
SaldoUltimos180Dias	11.656
NumeroCuentasActivas	7.138
NumeroCuentasCerradas	7.138
CantidadCuentasAF	7.127
Hijos	6.983
TotalSaldoUltimos360Dias	4.856
ScoreBuromadre	4.824
TotalSaldoUltimos180Dias	4.490

Fuente: Elaboración Propia

Se puede observar que las variables relacionadas con el Saldo de cuentas son las que presentan una relación más fuerte con la variable objetivo. Aparte existen variables como el número de hijos y Score de Buró de la madre que presentan un KS mucho más grande que otras variables financieras que se encuentran fuera del top 10.

Variable dependiente vs variables cualitativas

Para cuantificar la relación que existe entre las variables cualitativas y la variable objetivo haremos uso del estadístico Information Value que fue introducido por Fair Isaac [7] y es usada para medir el poder predictivo de una característica, se la conoce técnicamente como la medida de divergencia de Kullback, y se utiliza para medir la diferencia entre dos distribuciones. Yap, Ong y Mohamed, (2011) hacen uso de la medida del Information Value (IV) para clasificar a las variables predictoras según su poder clasificador, y poder separar los altos riesgos de los bajos riesgos.

El Information Value (IV) se calcula de la siguiente forma:

$$IV = \sum_{i=1}^{k} \left[\left(\frac{N_i}{\sum N} - \frac{P_i}{\sum P} \right) * \log \left(\frac{\frac{N_i}{\sum N}}{\frac{P_i}{\sum P}} \right) \right] \tag{3.2}$$

[7] Los pioneros más conocidos de la calificación crediticia son el ingeniero Bill Fair y el matemático Earl Isaac, quienes fundaron su consultora, Fair Isaac (FI), en San Francisco en 1956.

Donde:

P : es la ocurrencia (positiva), es decir, el número de malos clientes.

N: es la no ocurrencia (negativa), es decir, el número de buenos clientes.

i: es el índice del atributo que se está evaluando.

k: es el número total de atributos.

Para poder interpretar los resultados del Information Value, se tiene la siguiente regla:

Cuadro 3.6: Reglas para el Information Value.

Information Value	**Poder de predicción**
<0.02	Predictor inútil
0.02 to 0.1	Predictor débil
0.1 to 0.3	Predictor medio
0.3 to 0.5	Predictor fuerte
>0.5	Predictor sospechoso

Fuente: (Yap, Ong y Mohamed, 2011)

Se realiza un análisis con las variables categóricas presentes en la base de datos de este estudio.

Cuadro 3.7: Information value para las variables categóricas

Variable	**Information Value**
Genero	0.4 %
EstadoCivil	6.4 %
NivelEstudio	3.1 %
CodigoProfesion	13.2 %
ClasificacionPadre	0.1 %
Clasificacionmadre	0.5 %
ClasificacionConyuge	1.9 %

Fuente: Elaboración Propia

En la tabla 3.7 se puede observar que la variable con el mayor poder predictivo es Código de profesión teniendo un poder de predicción medio según la tabla 3.6. Las demás variables tienen un poder predictivo débil o inútil, es necesario poder determinar si este resultado sigue presentándose al momento de incluir estas variables al modelo.

3.1.6. Preprocesado de datos

Tomando como base la revisión descriptiva realizada en apartado 3.1.4, tanto de las variables numéricas como de las variables categóricas, procedemos a llevar a cabo tres procesos importantes: el primero, relacionado con la generación de nuevas variables explicativas a partir de las variables cuantitativas, el segundo en donde se realiza la recategorización de variables cualitativas, y el tercero, en el que se abordan los análisis de completitud y depuración de los datos en general.

Generación de Variables Cuantitativas

Se conoce de trabajos anteriores que realizar operaciones como la suma, promedio o desviaciones a las variables de montos, saldos y días mora en los diferentes periodos de tiempo producen variables que ayudan a mejorar la efectividad de los modelos. También realizar transformaciones como logaritmos o ratios de variables numéricas presentan el mismo efecto en el modelo (Pérez, 2019). En nuestro caso al momento de obtener la data ya se solicitó la creación de algunas de esta variables, y se utiliza este criterio para calcular las variables que complementan esta lista de transformaciones que se detallan a continuación:

- r_Saldo30s90dias
- r_Saldo30s180dias
- r_Saldo30s360dias
- r_Saldo90s180dias
- r_Saldo90s360dias
- r_Saldo180s360dias
- log_SaldoUltimos30dias
- log_TotalSaldoUltimos90dias
- log_TotalSaldoUltimos180dias
- log_TotalSaldoUltimos360dias
- log_PromSaldoUltimos90dias
- log_PromSaldoUltimos180dias
- log_PromSaldoUltimos360dias

En donde todas las variables cuyo nombre presenta la estructura 'r_Saldo**N1**s**N2**dias' son los ratios que resultan de dividir la variable Saldo con corte **N1** días sobre la variable Saldo a corte **N2** días; además se obtiene el logaritmo natural de todas aquellas variables numéricas de TotalSaldo y PromSaldo, a los cuales se les asignó nombres con el prefijo 'log_'.

Recategorización de variables categóricas

Según Pérez (2019) cada variable categóricas debería ser recategorizada en base a la tasa de fraude (en nuestro caso tasa de default) y representatividad de cada una de las categorías originales, agrupando las categorías que presenten similar tasa de default, de tal manera que el nuevo grupo sea representativo. En la práctica, este procedimiento es utilizado basándose en la experticia de las personas de negocios, sin embargo, existen técnicas estadísticas que agrupan las categorías de manera diferente, por ejemplo, K-medias, Mezclas Gaussianas, Clusterización Univariada, entre otras. En este caso, no se profundizó en este tipo de técnicas, ya que no son parte de nuestro enfoque, pero podrían ser utilizadas en estudios posteriores.

Algo que se evidenció en el análisis descriptivo de variables categóricas es la presencia de una variable con 955 categorías, esto podría representar un problema potencial a la hora de crear el modelo, ya que, para incluir esta variable se deberán crear $n - 1$ (n es el número de categorías) variables dicotómicas llamadas *dummy* o ficticias, lo cual incurrirá en este ejemplo con la inclusión de 954 variables *dummies* en el modelo. Por otro lado, se conoce que las variables dicotómicas que presentan pocos niveles positivos son menos representativas ya que pueden generar estimaciones no confiables (Iñiguez y Morales, 2009), es por tal razón que se recomienda combinar las categorías de menor frecuencia en otro nivel para así garantizar la representatividad de las variables dicotómicas.

En la mayoría de los casos del mundo real, los datos siguen la regla del "80/20" [8], lo que significa que habrá pocos niveles con mucha frecuencia y que la mayoría de los niveles no tendrán suficientes datos. Por tal razón, basados en este principio se procede a reagrupar al 20 % de los datos de menor frecuencia en un nivel con el nombre "Otros" para la variable 'CodigoProfesion' que en nuestro caso es la única que presenta el problema de alta dimensión de categorías, de tal manera se obtienen los siguientes resultados:

[8]Principio de Pareto.- establece que aproximadamente el 80 % de las consecuencias provienen del 20 % de las causas.

Cuadro 3.8: Comparativo variable 'CodigoProfesion' antes y después de la recategorización

Variable	Por_Nulos	Num_Categorias	Clase_mayoritaria	Freq_Clas_mayoritaria	Clase minoritaria	Freq_Clas_minoritaria
CodigoProfesion	0	955	E30	30.38	A05	0.001
New_CodigoProfesion	0	23	E30	30.38	L24	0.63

Fuente: Elaboración Propia

Se observa que al realizar el proceso de recategorización la variable 'CodigoProfesion' pasa de tener 955 niveles a tener solamente 23, que es un número razonable de categorías para poder incluirse en el modelo.

Análisis de completitud y depuración de datos

La presencia de datos perdidos ha sido siempre considerada un problema en el campo del análisis de datos, debido a que disminuye el poder estadístico. Asimismo, el enfoque clásico de manejo de datos perdidos plantea una serie de medidas poco efectivas como la eliminación de casos cuyos ítems tiene valores perdidos o la sustitución por la media en variables cuantitativas (Enders, 2010; Baraldi y Enders, 2010).

En la actualidad se puede contar con enfoques para el manejo de datos perdidos más eficientes y fáciles de implementar gracias a los avances en computación (Enders, 2010; Graham, 2012), permitiendo así recuperar los valores perdidos y restablecer el poder estadístico. Debido a esto, el principal problema ya no es la presencia de valores perdidos, el verdadero problema es como lidiamos con los datos perdidos.

Vale la pena recalcar que gracias a las características que incorporan los modelos a utilizar en el tratamiento de valores perdidos, tal como se lo mencionó en el apartado 2.3.7, el proceso de depuración de la información fue menos riguroso ya que se le permite al modelo capturar de mejor manera la realidad de los datos, y de esta forma se busca mejorar el comportamiento del modelo con datos de producción. De esta forma no se profundizó en el uso de las técnicas antes mencionadas, limitándonos a validar que el proceso de extracción de los datos fuese al adecuado.

3.1.7. Muestra de entrenamiento y prueba

En este apartado se describe el procedimiento utilizado, para obtener los conjuntos de datos de entrenamiento (muestra modelamiento) y prueba (muestra prueba) del modelo estadístico. En la mayoría de los artículos de *Machine Learning*, se recomienda como particiones adecuadas aquellas que dividen al conjunto de datos aleatoriamente en las siguientes proporciones: 50 %/50 %, 60 %/40 %, 70 %/30 % y 80 %/20 %. En nuestro caso, basados en el juicio experto de los administradores de riesgo de la institución, se toman las siguientes consideraciones para realizar la partición del conjunto de datos:

- Considerando la limitada cantidad de datos que presenta este modelo con relación a otros en donde existe una mayor cantidad de información, se considera utilizar el 70 %/30 %, para brindarle mayor información al modelo y tener una cantidad considerable de datos para la evaluación.
- La partición no se la realizará de forma aleatoria, dado que en un futuro el modelo deberá ser presentado ante el ente regulador para su aprobación, se considera utilizar el conjunto de prueba como insumo para una evaluación de tipo *forward testing* [9]

Con las consideraciones mencionadas se obtiene que la base de entrenamiento contará con los registros de las fechas Enero 2016 y Agosto 2018 mientras que la base de prueba irá desde Septiembre 2018 hasta Junio 2018, con lo cual se cuenta con 10 meses para la evaluación de *forward testing*. A continuación se indica la distribución de los conjuntos de datos

Cuadro 3.9: Partición del conjunto de datos en entrenamiento y prueba

Partición	Registros	Porcentaje Registros	TasaDefault
Entrenamiento	41023	69.44 %	1.21 %
Prueba	18053	30.55 %	2.08 %

Fuente: Elaboración Propia

[9] El método de forward testing permite evaluar al modelo en condiciones reales o de producción observando cual sería su comportamiento en el futuro.

3.2. Construcción del modelo predictivo XGBoost

La construcción del modelo se la realizará utilizando el algoritmo XGBoost detallado en el apartado 2.3.7, ya que es un algoritmo que ha demostrado ser exitoso en muchos campos y es uno de los métodos líderes para ganar competencias de Kaggle [10]. La idea general del algoritmo es que va construyendo árboles de decisión débiles sucesivamente, donde cada árbol va aprendiendo del anterior y por ende mejorando su precisión. Al final la sucesión de árboles producen un clasificador robusto cuyo rendimiento es muy difícil de superar.

3.2.1. Selección de variables

Al crear un modelo de aprendizaje automático, es casi raro que todas las variables del conjunto de datos sean útiles para crear un modelo. Agregar variables redundantes reduce la capacidad de generalización del modelo y también puede reducir la precisión general de un clasificador. Además, agregar más y más variables a un modelo aumenta la complejidad general del mismo.

De acuerdo con la Ley de la Parsimonia (*Occam's razor*) [11], la mejor explicación para un problema es la que implica la menor cantidad de suposiciones posibles. Por lo tanto, la selección de características se convierte en una parte indispensable de la construcción de modelos de aprendizaje automático (Ferreira et al., 2012).

El objetivo de la selección de características en el aprendizaje automático es encontrar el mejor conjunto de variables que permitan construir modelos útiles para la aplicación de los fenómenos estudiados, las técnicas para la selección de variables en el aprendizaje automático desde un punto de vista taxonómico se pueden clasificar en:

- **Métodos de filtrado**.- Los métodos de filtro recogen las propiedades intrínsecas de las características medidas a través de estadísticas univariadas en lugar del rendimiento de validación cruzada. Estos métodos son más rápidos y menos costosos computacionalmente que los métodos de envoltura. Cuando se

[10] Kaggle es una plataforma con recursos para aprender sobre Aprendizaje Automático y Ciencia de Datos

[11] Es el principio de resolución de problemas de que "las entidades no deben multiplicarse sin necesidad", o más simplemente, la explicación más simple suele ser la correcta.

trata de datos de alta dimensión, es computacionalmente más económico utilizar métodos de filtrado.

- **Métodos de envoltura**.- Los envoltorios requieren algún método para buscar en el espacio todos los posibles subconjuntos de características, evaluando su calidad, aprendiendo y evaluando un clasificador con ese subconjunto de características. El proceso de selección de características se basa en un algoritmo de aprendizaje automático específico que intentamos encajar en un conjunto de datos determinado. Sigue un enfoque de búsqueda codiciosa al evaluar todas las posibles combinaciones de características contra el criterio de evaluación. Los métodos de envoltura generalmente dan como resultado una mejor precisión predictiva que los métodos de filtro.
- **Métodos integrados**.-Estos métodos abarcan los beneficios de los métodos de envoltura y de filtro, al incluir interacciones de características pero también mantener un costo computacional razonable. Los métodos integrados son iterativos en el sentido de que se encargan de cada iteración del proceso de entrenamiento del modelo y extraen cuidadosamente las características que más contribuyen al entrenamiento para una iteración en particular.

Dento de los métodos de envoltura se encuentran el *Forward Feature Selection* [12] y *Backward Feature Elimination* [13]. En trabajos como el de Pérez (2019), se realiza como primera instancia un proceso de filtrado usando medidas de divergencia como el estadístico de *KS* y medidas de asociación como el *Information Value* para realizar una selección preliminar de variables. En nuestro caso, dado que el conjunto de datos no tiene una alta dimensionalidad en número de variables (por la naturaleza del problema) optamos por utilizar el método *Backward Feature Elimination*, ya que de esta forma permitimos que el algoritmo capture las relaciones no lineales que existe entre las variables y decida si la relación es fuerte o débil.

El procedimiento considera crear en el primer paso un modelo con todas las variables disponibles e ir eliminando en cada iteración una variable, para lo cual haremos uso del valor de *feature importance* del modelo como criterio seleccionador

[12]Método iterativo en el que comenzamos con la variable de mejor rendimiento contra el objetivo. A continuación, seleccionamos otra variable que ofrezca el mejor rendimiento en combinación con la primera variable seleccionada. Este proceso continúa hasta que se alcanza el criterio preestablecido.

[13]Este método funciona exactamente de manera opuesta al método Forward Feature Selection. Aquí, comenzamos con todas las variables disponibles y construimos un modelo. A continuación, tomamos la variable del modelo que da el mejor valor de medida de evaluación. Este proceso continúa hasta que se alcanza el criterio preestablecido.

de qué variable debería retirarse del modelo en dicho paso, para al final quedarnos con el modelo más simple y que mejor rendimiento posea.

3.2.2. El problema de muestras desproporcionadas

Al enfrentarse a la situación de crear un modelo de clasificación es habitual que las clases no se encuentran balanceadas. Esto es, el número de registros para una de las clases es inferior al resto. Cuando el desequilibrio es pequeño, esto no supone un inconveniente, pero cuando es grande, éste representa un problema para la mayoría de los modelos de clasificación. Esta situación se conoce como el Problema del Desequilibrio de Clases (*Class Imbalance Problem*) y está presente en nuestro caso, ya que el número de clientes morosos es muy pequeño en relación al número de clientes buenos como se muestra en la Tabla 3.9.

Existen métodos de remuestreo como *oversamplig*, *undersampling* y métodos más sofisticados como *SMOTE* [14] que sirven para lidiar con el problema de muestras desproporcionadas y que han demostrado tener buenos resultados, pero que en nuestro caso no es aplicable ya que al utilizar estos métodos en modelos de credit scoring el resultado del modelo sobrestima la verdadera probabilidad de incumplimiento.

Iñiguez y Morales (2009) hablan del problema de las muestras desproporcionadas y el efecto que tiene realizar un remuestreo de los datos cuando se estima el modelo con una regresión logística, en el trabajo se demuestra que se debe realizar una transformación sobre la constante del modelo para retornar a la distribución real. En nuestro caso al hacer uso de un modelo no paramétrico como lo es el XGBoost no podemos realizar dicha transformación, por lo tanto, si se realiza un remuestreo de los datos vamos a encontrarnos con el problema que la distribución resultante estará sobrestimando a la distribución real de la probabilidad de default y dado que el modelo va a ser utilizado en el cálculo de provisiones eso representaría un gran problema.

Existen técnicas recomendadas por grandes consultoras de Riesgo de Crédito, como por ejemplo realizar transformaciones de tipo exponencial a la distribución de probabilidad resultante del modelo. A pesar que la idea parece muy codiciosa, ya que solucionaría el problema de la sobrestimación de la PD y podríamos hacer

[14] Synthetic Minority Oversampling Technique

uso de las técnicas de remuestreo brindando mejoras en el rendimiento del modelo, se deja planteada la idea para próximas investigaciones. En nuestro caso se prefirió no hacer uso de las técnicas de remuestreo ya que basados en los resultados de Petropoulos, et al (2018) sabemos que el modelo XGBoost maneja de forma eficiente el problema de muestras desproporcionadas teniendo un excelente desempeño.

3.2.3. Implementación del Algoritmo

Una vez se cuenta con los datos procesados y el conjunto de entrenamiento definido, procedemos a estimar el modelo analítico en el Lenguaje Estadístico *R* (versión 3.6.2). Se detallan a continuación las librerías más importantes que se utilizaron en el proceso de entrenamiento.

- xgboost
- data.table
- lubridate
- stringr
- caret
- Matrix
- dplyr
- MLmetrics

Búsqueda de Hiperparámetros

La forma tradicional de realizar la optimización de hiperparámetros ha sido la búsqueda en cuadrícula (*grid search*, traducido al inglés), que es simplemente una búsqueda exhaustiva a través de un subconjunto especificado manualmente del espacio de hiperparámetros de un algoritmo de aprendizaje automático. Un algoritmo de búsqueda de cuadrícula debe estar guiado por alguna métrica de rendimiento, que en nuestro caso será el AUC [15]. Esta métrica es medida generalmente utilizando validación cruzada en un conjunto de evaluación que generalmente es el conjunto de datos de validación.

Dado que el espacio de parámetros de un algoritmo de aprendizaje automático puede incluir espacios de valores acotados para ciertos parámetros o ilimitados para otros parámetros, es posible que sea necesario establecer límites y discretización manualmente antes de aplicar la búsqueda de cuadrícula. Por ejemplo, un clasifi-

[15]Se encuentra implementada en el algoritmo XGBoost para problemas de clasificación binaria

cador SVM [16] de margen suave típico equipado con un kernel RBF tiene al menos dos hiperparámetros que deben ajustarse para un buen rendimiento en datos no vistos: una constante de regularización C y un hiperparámetro del kernel γ. Ambos parámetros son continuos, por lo que para realizar una búsqueda en la cuadrícula, se selecciona un conjunto finito de valores "razonables" para cada uno, digamos $C \in \{10, 100, 1000\}$ y $\gamma \in \{0.1, 0.2, 0.5, 1.0\}$

Además existen otros métodos como *Random search*, *Bayesian optimization*, *Gradient-based optimization*, *Evolutionary optimization* , entre otros, pero en nuestro trabajo vamos a utilizar el método *grid search* por su fácil implementación y buenos resultados. El código utilizado para realizar la búsqueda de los hiperparámetros de nuestro modelo se adjunta en el anexo B.4, además la tabla con los valores sobre los cuales se hizo la búsqueda se adjunta en el anexo B.3; y a través de los cuales se obtuvieron los siguientes valores para los hiperparámetros finales del modelo:

Cuadro 3.10: Hiperparámetros óptimos

Parámetro	Descripción	Valor óptimo
eta	Tasa de aprendizaje	0.1
gamma	Regularización del árbol	0.01
min_child_weight	Peso mínimo para que se cree un nuevo nodo.	50
subsample	Porcentaje de submuestras de los casos.	0.6
colsample_bytree	Porcentaje de submuestras de variables.	0.5
max_depth	profundidad de los árboles	10
max_delta_step	Ajuste de desbalanceo	10
alpha	Regularización	1
lambda	Regularización	1

Fuente: Elaboración Propia

Entrenamiento del modelo

Basados en la configuración de hiperparámetros recomendados en el apartado 2.3.7 se realiza la primera iteración del modelo, con lo cual se obtiene un resultado

[16] Las máquinas de soporte vectorial (SVM, por sus siglas en inglés) son modelos de aprendizaje supervisado con algoritmos de aprendizaje asociados que analizan datos para el análisis de clasificación y regresión.

base que servirá para evaluar si las siguientes iteraciones del modelo dan mejores o peores resultados, se construye el modelo bajo el siguiente esquema:

1. Crear un modelo 'base' con todas las variables resultantes del proceso mencionado en el apartado 3.1.6 y con la configuración predeterminada de hiperparámetros. De esta manera tendremos un valor de la métrica AUC medida sobre el conjunto de validación, esta medida nos permitirá comparar si los siguientes modelos presentan un mejor o un peor ajuste.

2. Continuar el entrenamiento basados en el método *Backward Feature Elimination* mencionado en el apartado 3.2.1 hasta encontrar la combinación adecuada de variables a incluir en el modelo.

3. Una vez se haya seleccionado las variables que estarán en el modelo, se realiza una última calibración de los hiperparámetros utilizando el método de *grid serach* mencionado en el apartado anterior y cuyos resultados se muestran en la tabla 3.10, esto con el fin de que el modelo 'final' se ajuste de la mejor manera a los datos específicos de nuestro proyecto.

4. Una vez definidas tanto las variables como los hiperparámetros, se procede con el entrenamiento del modelo final.

3.2.4. Resultados del Algoritmo

Tras haberse ejecutado el entrenamiento del modelo utilizando el esquema del apartado anterior, el resultado de éste es un objeto de **R** de tipo *.model* dentro del cual se almacena toda la información referente al proceso de entrenamiento, como por ejemplo:

- *handle*.- un identificador (puntero) al modelo xgboost en la memoria.

- *raw*.- un vertedero de memoria en caché del modelo xgboost guardado como tipo crudo de R.

- *niter*.- número de iteraciones.

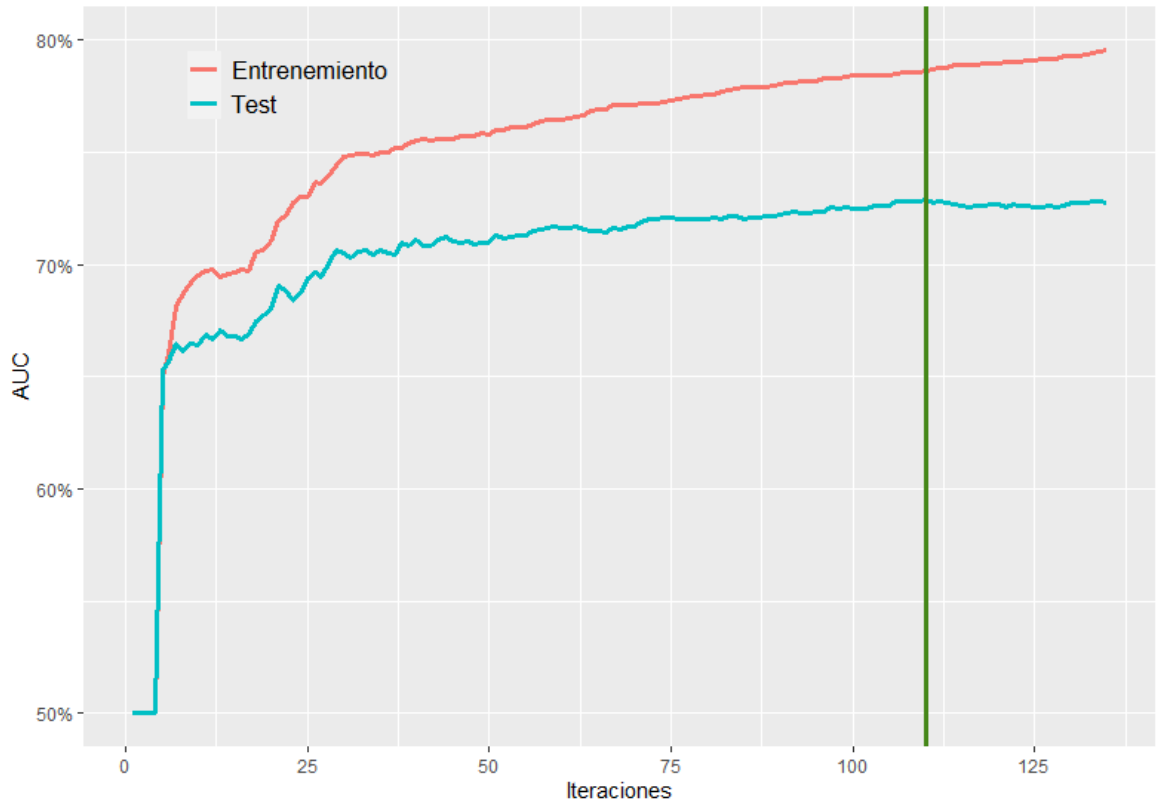
- *evaluation_log*.- historial de evaluación almacenado como una tabla de datos con la primera columna correspondiente al número de iteración y el resto correspondiente a los valores de las métricas de evaluación.

- *params*.- parámetros que se pasaron a la librería para entrenar el modelo xgboost.
- *feature_names*.- nombres de las variables del conjunto de datos de entrenamiento.
- *best_iteration*.- número de la iteración con el mejor valor de métrica de evaluación.

Iteraciones del modelo final

Presentamos a continuación la evolución del coeficiente *AUC* a medida que se incluían más árboles al modelo final.

Figura 3.4: Entrenamiento Modelo XGBoost



Fuente: Elaboración Propia

Se evalúa la métrica *AUC* tanto en el conjunto de entrenamiento como en el de validación a medida que el algoritmo va añadiendo árboles al modelo en cada iteración (ver figura 3.4). En el eje x se representan el número de iteraciones, mientras

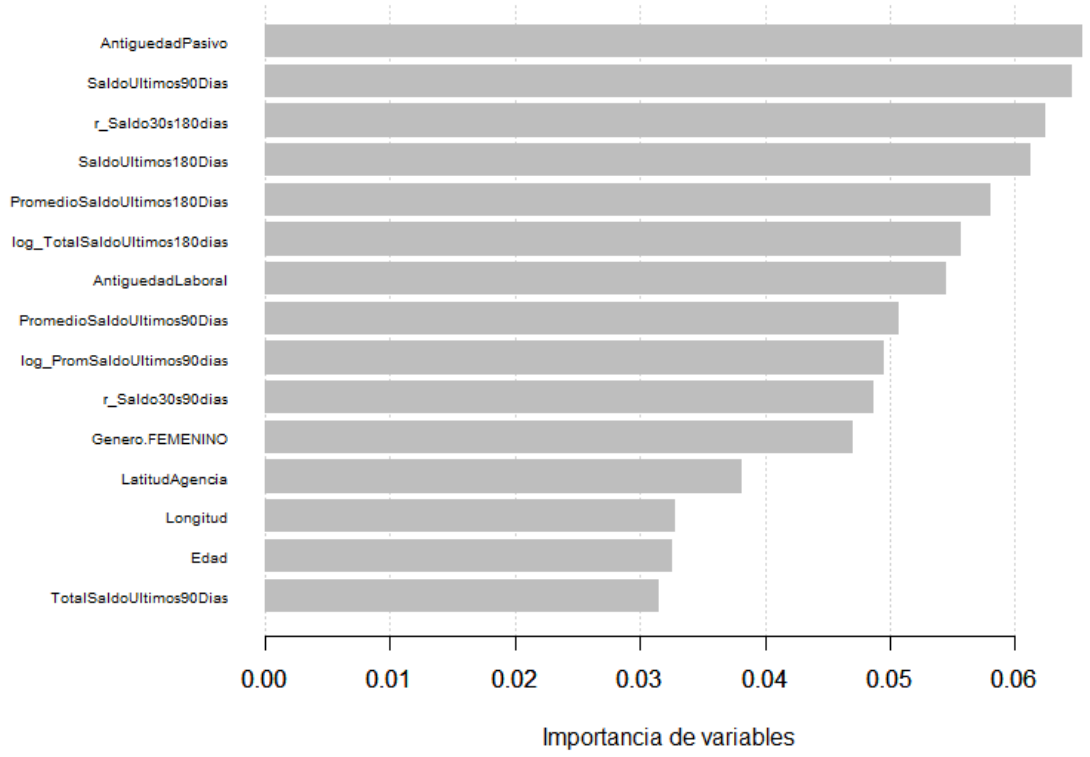
que la línea roja nos indica la evolución de la métrica AUC en el conjunto de entrenamiento y la línea turquesa en el conjunto de validación. Además se marca una línea verde en la iteración 110 que es el número de árboles óptimo que se eligió para el modelo, se puede ver que a partir de esa iteración el rendimiento en el conjunto de validación se estabiliza, mientras que en el conjunto de entrenamiento sigue aumentando con lo cuál si incrementaríamos el número de árboles caeríamos en un problema de sobreajuste (*Overfitting*) [17].

Feature Importance (modelo final)

Una vez elaborado el modelo estadístico, podemos hacer uso de sus cualidades, una de ellas es la generación del ranking de las variables explicativas según su nivel de influencia dentro del problema de crédito (Figura 3.5), a este hecho se conoce como importancia de variables, visto con mayor detenimiento en el apartado 2.3.6.

[17] En estadística, el sobreajuste es "la producción de un análisis que se corresponde demasiado cerca o exactamente con un conjunto particular de datos y, por lo tanto, puede fallar en ajustar datos adicionales o predecir observaciones futuras de manera confiable"

Figura 3.5: Importancia de variables



Fuente: Elaboración Propia

De acuerdo a la Figura 3.5, las variables que influyen fuertemente para determinar si un cliente va a pagar o no un crédito son: Antigüedad en la cuenta de ahorro, los saldos de las cuentas en las distintas ventanas de tiempo, la antigüedad laboral, el género, la ubicación geográfica tanto del domicilio como de la agencia de negocios. En este gráfico se incluyeron únicamente las 15 principales variables, y en el apéndice C se presenta los valores para todo el conjunto de variables.

Se puede determinar claramente que ante la falta de variables de tipo: Historial Crediticio, pasan a tener un rol más importante dentro del problema de *credit scoring* de originación aquellas variables relacionadas al manejo de ahorros y antigüedad en el sistema financiero, además de otras de tipo socio demográfico y de ubicación. Estas variables serán consideradas las mejores para resolver este tipo de problemas.

3.3. Evaluación estadística del modelo Xgboost

Una vez concluida la etapa de modelamiento y ya con un modelo final seleccionado, procedemos con la evaluación de su rendimiento, la misma que será realizada sobre un nuevo conjunto de datos (muestra de prueba) utilizando técnicas estadística que permitan verificar si el modelo es el adecuado para el problema que se desea solucionar, las técnicas propuestas para esta validación se describen a continuación.

3.3.1. Estadísticos de rendimiento

Los estadísticos que resumen la efectividad del modelo son generados a partir del test de Kolmogorov - Smirnov (test KS), Curva ROC y test de Gini, detallados en los apartados 2.6.3, 2.6.2, 2.6.4 respectivamente.

De acuerdo a la descripción de los diferentes estadísticos que resumen el rendimiento del modelo, se puede concluir que el desempeño del modelo será sumamente alto, cuanto más alto sean los valores de KS, AUC y GINI. Una vez adquirido este conocimiento, se procede a calcular dichos estadísticos para nuestro modelo de credit scoring, cuyos valores se presentan en la Tabla 3.11

Cuadro 3.11: Medidas de discriminación. XGBoost

Partición	KS	AUC	Gini
Entrenamiento	34.03	72.86	45.73
Prueba	30.48	71.22	42.45

Fuente: Elaboración Propia

Según la consultora McKinsey & Company [18] tener un coeficiente de Gini superior a 35 en modelos de *Credit Scoring* de tipo originación muestra tener un buen resultado y un coeficiente superior a 40 puntos de Gini representa tener un modelo excelente que se compara con los resultados de los modelos de grandes bancos latinoamericanos. Dada la posibilidad de no solamente medir el rendimiento del modelo, sino de poder tener un *benchmark* [19] a nivel regional la entidad bancaria utiliza el coeficiente de Gini como métrica oficial para medir sus modelos de Score

[18] Asesora de algunos de los bancos más grandes de la región `https://www.mckinsey.com/`

[19] El benchmarking es un punto de referencia sobre el cual las empresas comparan algunas de sus resultados con los del mercado.

Crediticio.

En nuestro caso, al tener un coeficiente de Gini de 42.45 puntos, consideramos que es un excelente modelo y procedemos a realizar las demás pruebas de validación.

Matriz de confusión

Se procede a mostrar la matriz de confusión que relaciona la variable objetivo Y y la predicción del modelo transformada a categórica a partir del punto de corte óptimo [20], cuya explicación se encuentra detallada en el apartado 2.6.1 y cuyo objetivo es minimizar el valor de los casos que son Falsos Positivos y paralelamente minimizar los Falsos negativos.

Figura 3.6: Matriz de confusión. XGBoost

		Y Real		
	Frecuencia	1	0	
	Porcentaje			
	Porcentaje Fila			Total
Y predecido	1	**125**	**2103**	**2228**
		0.7%	11.6%	12.3%
		5.6%	94.4%	
	0	**251**	**15574**	**15825**
		1.4%	86.3%	87.7%
		1.6%	98.4%	
	Total	**376**	**17677**	**18053**
		2.1%	97.9%	100%

Fuente: Elaboración Propia

Es importante mencionar que el objetivo del modelo es la construcción de perfiles de clientes (se detalla en el capítulo 5) y no el de buscar un punto de rechazo. La tabla nos indica que si se hubiese contado con el modelo al momento del otorgamiento de las operaciones estudiadas, se hubiera bajado la tasa de default a un 1.6 %, pero todo esto a un costo de perder 2103 operaciones a las que el modelo hubiese negado su aprobación a pesar que terminarían siendo buenas. Todas estas afirmaciones se basan en el supuesto que se elija el punto de corte como discriminador de

[20] Este método define el valor del punto de corte óptimo como el valor cuya sensibilidad y especificidad son las más cercanas al valor del área bajo la curva ROC y el valor absoluto de la diferencia entre los valores de sensibilidad y especificidad es mínimo.

ser sujeto a crédito, aunque por lo general los bancos no utilizan esta estrategia y la tabla de confusión simplemente nos brinda una mirada de la relación que existe entre la variable Y y la categorización de la predicción.

También podemos ver que si utilizáramos el punto de corte como discriminador de a quién otorgar o no el crédito, se tendría una tasa de aprobación del 87.7 % que nos indica que existe un margen muy grande, en relación a la tasa real de *default*, de clientes que a pesar que no son malos el modelos los clasifica como potenciales incumplimientos. Esto no representa un problema ya que como se verá en apartado 5.2, la creación de perfiles de riesgo permite a la entidad administrar de manera adecuada su portafolio de clientes.

3.3.2. Tabla de ODDS

Se procede a realizar el cálculo de la tabla de odds tanto para el conjunto de entrenamiento como para el conjunto de prueba, basados en la información detallada en el apartado 2.6.5.

Entrenamiento

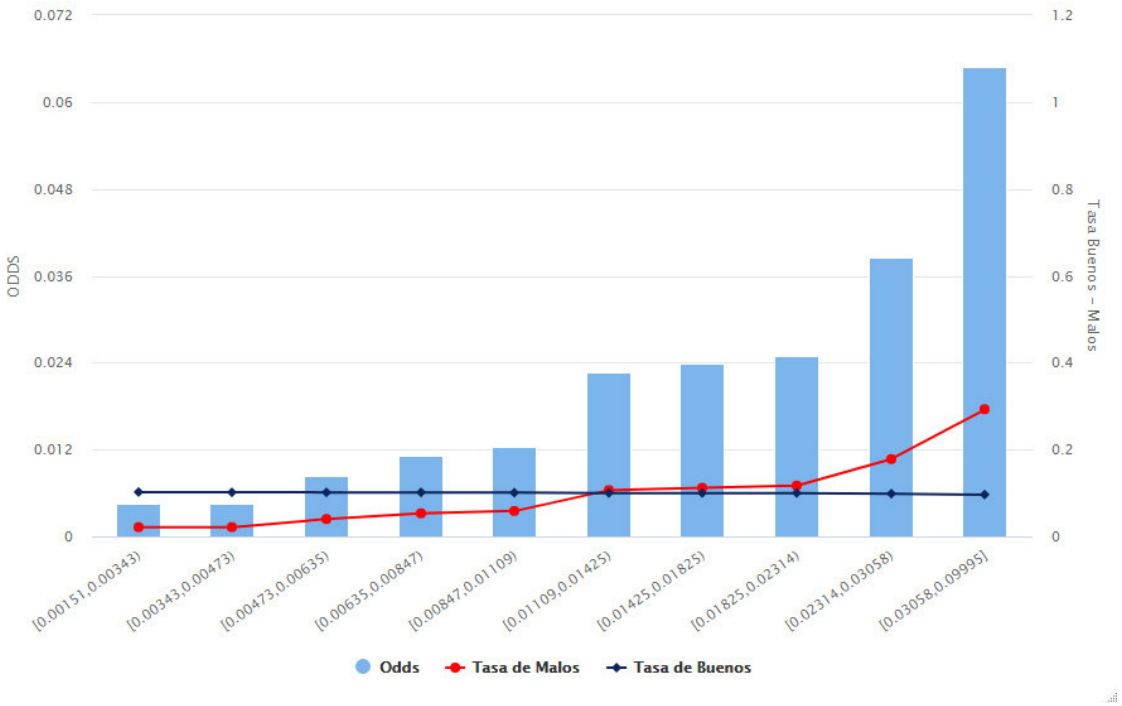
Cuadro 3.12: Tabla de Odds (Entrenamiento)

Decil	Rango	FreqMalos	FreqBuenos	TasaMalos	TasaBuenos	Odds
1	[0.00152 , 0.00301)	2	3289	0.0049	0.1012	0.0006
2	[0.00301 , 0.00379)	4	3287	0.0099	0.1011	0.0012
3	[0.00379 , 0.00478)	7	3284	0.0173	0.1010	0.0021
4	[0.00478 , 0.00611)	13	3278	0.0321	0.1009	0.0040
5	[0.00611 , 0.00797)	22	3268	0.0543	0.1005	0.0067
6	[0.00797 , 0.01074)	33	3258	0.0815	0.1002	0.0101
7	[0.01074 , 0.01453)	34	3257	0.0840	0.1002	0.0104
8	[0.01453 , 0.01961)	56	3235	0.1383	0.0995	0.0173
9	[0.01961 , 0.02724)	65	3226	0.1605	0.0993	0.0201
10	[0.02724 , 0.09106]	169	3121	0.4173	0.0960	0.0541

Fuente: Elaboración Propia

La tabla 3.12 presenta la información resumida de todos los clientes que pertenecen al conjunto de entrenamiento, de acuerdo a la predicción que le asigna el modelo fueron asignados a cada uno de los deciles, dentro de los cuales se obtienen variables como la frecuencia de malos, frecuencia de buenos, tasa de malos y tasas de buenos y el valor más importante es el Ratio Odds (Frecuencia de Malos/ Frecuencia de buenos). Por simplicidad, resulta mas intuitivo ver estos indicadores de forma gráfica, por tal motivo se los presentan en la Figura 3.7

Figura 3.7: Odds del Modelo (Entrenamiento)



Fuente: Elaboración Propia

Prueba

De igual manera se presentan los datos para el conjunto de test y para este se brinda un análisis de estos indicadores.

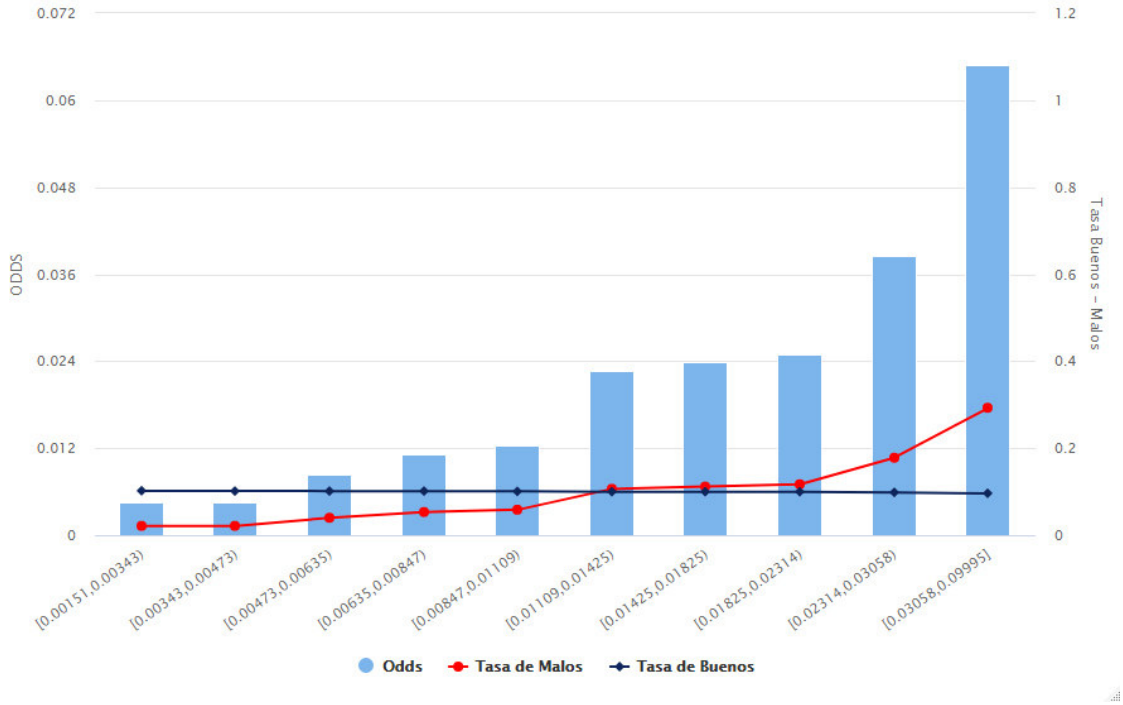
Cuadro 3.13: Tabla de Odds (Prueba)

Decil	Rango	FreqMalos	FreqBuenos	TasaMalos	TasaBuenos	Odds
1	[0.00151 , 0.00343)	8	1798	0.0213	0.1017	0.0044
2	[0.00343 , 0.00473)	8	1797	0.0213	0.1017	0.0045
3	[0.00473 , 0.00635)	15	1790	0.0399	0.1013	0.0084
4	[0.00635 , 0.00847)	20	1786	0.0532	0.1010	0.0112
5	[0.00847 , 0.01109)	22	1783	0.0585	0.1009	0.0123
6	[0.01109 , 0.01425)	40	1765	0.1064	0.0998	0.0227
7	[0.01425 , 0.01825)	42	1764	0.1117	0.0998	0.0238
8	[0.01825 , 0.02314)	44	1761	0.1170	0.0996	0.0250
9	[0.02314 , 0.03058)	67	1738	0.1782	0.0983	0.0386
10	[0.03058 , 0.09995]	110	1695	0.2926	0.0959	0.0649

Fuente: Elaboración Propia

Es importante notar que el decil 1 es el grupo con los clientes con una PD más baja y a medida que van aumentando el decil el grupo se va a haciendo más malo. Vamos a analizar la métrica más importante, ésta es el valor de Ratio ODDS (Frecuencia de malos/Frecuencia de buenos), en el caso del decil 1, se puede decir que en este grupo se esperaría encontrar 4 cliente malo por cada 1000 clientes buenos, y se espera que este indicador presente un comportamiento creciente a manera que aumenta el decil.

Figura 3.8: Odds del Modelo (Prueba)



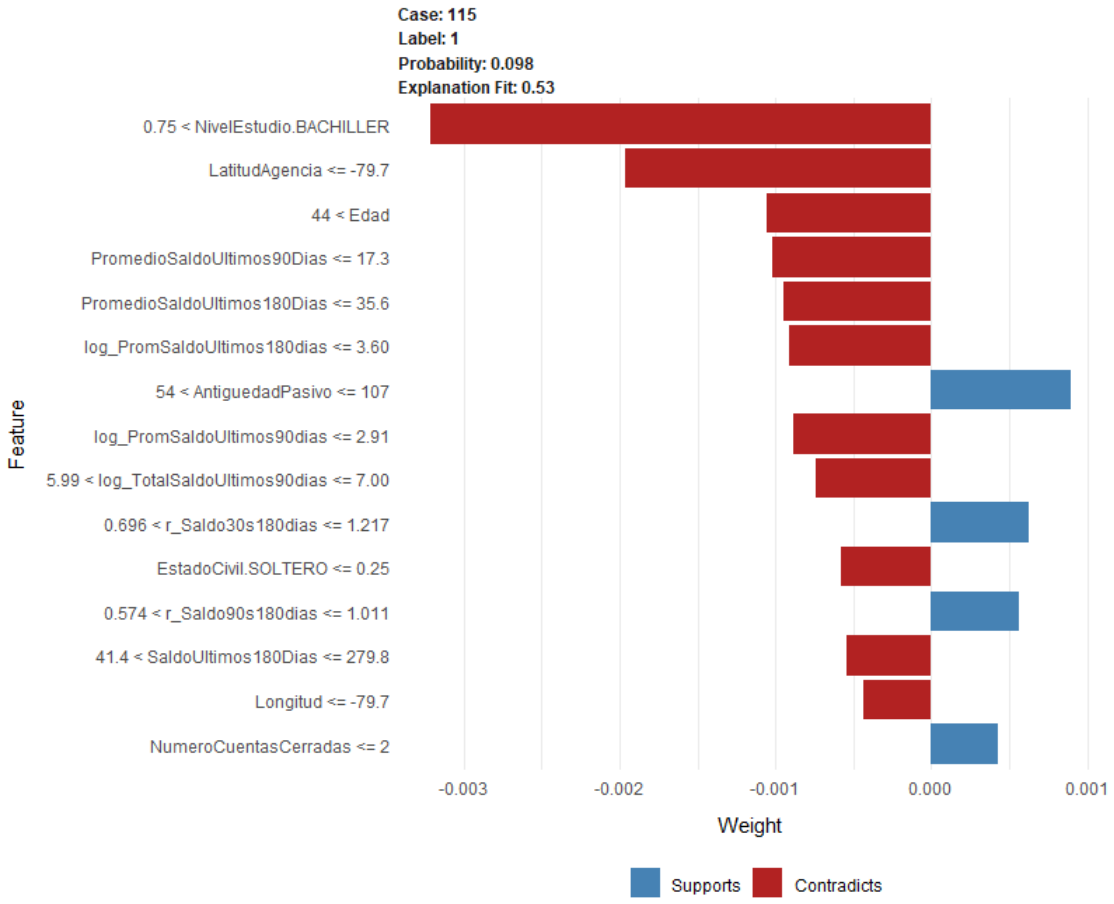
Fuente: Elaboración Propia

3.4. Construcción del Modelo LIME

El modelo LIME al ser un modelo de tipo local genera una explicación para cada uno de los casos de estudio, a modo de ejemplificación se eligió 1 caso aleatorio para detallar los resultados de este modelo que se muestra en la Figura 3.9 y el código con el que se fue generado se adjunta en el apéndice B.6.

Nos encontramos que el caso seleccionado es el indexado por el número 11, el valor de la variable Y es 1, La predicción del modelo XGBoost es 0.098 que según el punto de corte óptimo es considerado como mal cliente y el modelo LIME tiene una capacidad de explicación del 53 % considerando las variables estudiadas. Además se presentan en color rojo aquellas variables que inciden en forma negativa al resultado del modelo, y en azul las que inciden en forma positiva. Para este caso se tiene que tener un nivel de estudios de BACHILLER, estar situado en el sur del país, ser mayor a 44 años y tener un Promedio de Saldo en las cuentas menor a $17.3 en los últimos 90 días perjudica la calificación de crédito de este cliente, mientras que tener una cuenta con una antigüedad entre 54 y 107 años lo beneficia.

Figura 3.9: Resultado modelo LIME para 1 caso aleatorio1



Fuente: Elaboración Propia

Es importante recordar que este modelo brinda una aproximación a una explicación adecuada del porque se generan las distintas calificaciones, más no, es una característica propia del modelo XGBoost que contenga la información de relación entre las variables, es por eso que debe considerarse como guía y no se le debe cargar la responsabilidad de las decisiones del modelo. En este punto es bueno considerar que dada la complejidad de cálculo de este modelo, se debe considerar un uso adecuado del mismo, ya que hacer las estimaciones para todos los casos de prueba tomó como 6 horas de cálculo.

Capítulo 4

Comparación del modelo XGBoost con técnicas tradicionales(precisión e interpretabilidad)

Pérez (2019) en su trabajo resuelve un problema de características similares en el cual se hace un comparativo entre un algoritmo Random Forest y una Regresión Logística con el método de Firth, dado que el algoritmo Random Forest al igual que el XGBoost forman parte de la familia de *ensemble methods* [1] basados en árboles de decisión y los casos de estudio también presentan muestras desproporcionadas, se toma la decisión de utilizar la misma metodología de comparación de modelos para nuestro proyecto.

En el apartado 2.4, se hizo mención a la regresión logística con estimación a través del método de Firth, técnica elegida para ser comparada con nuestra técnica estrella XGBoost, pues según Lay (2015) también maneja adecuadamente el problema de desbalanceo significativo de clases aparte de ser una de las técnicas que más ha sido usada históricamente en este tipo de problemas. Dicha comparación se lleva a cabo, con la finalidad de identificar la técnica que mejor se ajusta a los datos de nuestro problema de *credit scoring* de originación.

Para que la comparación sea válida, el modelo desarrollado mediante la regresión logística con estimación a través del método de Firth utilizará las mismas mues-

[1] En estadísticas y aprendizaje automático, los métodos de conjunto utilizan múltiples algoritmos de aprendizaje para obtener un mejor rendimiento predictivo que el que se podría obtener solo con cualquiera de los algoritmos de aprendizaje constituyentes.

tras de modelamiento y validación (Tabla 3.9), que se usaron para construir y validar el modelo XGBoost. Puesto que, la regresión logística con estimación a través del método de Firth (sección 2.4) se encuentra diseñada para trabajar directamente con clases desbalanceadas, no se efectúa ningún tipo de remuestreo previo en la muestra de modelamiento, limitándonos únicamente a realizar la imputación de valores perdidos de acuerdo a las técnicas mencionadas en la sección 3.1.6, ya que este procedimiento no se lo realizó en el modelo XGBoost y es importante aplicarlo antes de entrenar el modelo de regresión logística.

Una vez dicho esto, se procede a explicar brevemente el desarrollo del modelo de regresión logística con estimación a través del método de Firth, para el cual vale la pena aclarar que no se hará una búsqueda exhaustiva hasta encontrar el mejor modelo mejor modelo ya que no es el objetivo de este trabajo y únicamente nos limitaremos a encontrar un modelo adecuado que cumpla con las características de un modelo robusto.

4.1. Modelo de regresión logística con estimación a través del método de Firth

4.1.1. Estimación del modelo

En la tabla C.2 (ver apéndice C) se presentan las variables explicativas usadas en el estimación del modelo XGBoost junto con su respectivo valor de importancia dentro del modelo, estás fueron seleccionadas a través del método *Backward Feature Elimination* mencionado en el apartado 3.2, y son las más influyentes dentro del problema de *credit scoring*. Por esta razón, del conjunto de las 31 variables explicativas se procede a seleccionar las que mejor ajustan el modelo de regresión logística con estimación a través del método de Firth, mediante el algoritmo *backward* (King y Zeng, 2001). El algoritmo *backward* parte de un modelo que incorpora todas las variables explicativas (modelo completo) y en cada iteración elimina una de ellas considerada como la menos influyente sobre la variable dependiente, la eliminación de variables usualmente se basa en el criterio de información de Akaike [2]. El algoritmo termina cuando ya no existan variables que suprimir. Como resultado se obtiene el modelo

[2] AIC por sus siglas en inglés, estima la cantidad relativa de información perdida por un modelo determinado: cuanto menos información pierde un modelo, mayor es la calidad de ese modelo

presentado en la tabla 4.1, donde se muestra que todas las variables incorporadas en el modelo de regresión logística con estimación a través del método de Firth son significativas. El código de construcción del modelo se adjunta en el apéndice B.2.

Cuadro 4.1: Modelo Regresión Logística Firth

Variable	Coeficiente	Std. Error	z value	Pr(>|z|)	
Intercepto	-1.92611	0.64051	-3.00700	0.00264	**
r_Saldo30s90dias	0.00001	0.00000	2.58800	0.00966	**
AntiguedadLaboral	-0.00693	0.00046	-15.06000	<2e-16	***
AntiguedadPasivo	-0.00439	0.00025	-17.36600	<2e-16	***
PromedioSaldoUltimos90Dias	0.00038	0.00008	4.90700	0.00000	***
Edad	0.01516	0.00122	12.43100	<2e-16	***
r_Saldo90s180dias	0.00002	0.00000	5.85500	0.00000	***
Longitud	-0.04322	0.00809	-5.34200	0.00000	***
Genero.FEMENINO	-0.49301	0.02336	-21.10400	<2e-16	***
SaldoUltimos180Dias	0.00006	0.00003	2.17900	0.02934	*
SaldoUltimos30Dias	0.00006	0.00001	8.46400	<2e-16	***
NivelEstudio.SUPERIOR	-0.44874	0.03233	-13.88100	<2e-16	***
TotalSaldoUltimos90Dias	0.00012	0.00001	8.02400	0.00000	***
TotalSaldoUltimos180Dias	-0.00014	0.00001	-10.83100	<2e-16	***
Hijos	0.03665	0.01156	3.17000	0.00153	**
log_PromSaldoUltimos180dias	1.86510	0.20294	9.19000	<2e-16	***
log_PromSaldoUltimos90dias	-1.90613	0.20427	-9.33100	<2e-16	***
CodigoProfesion.E30	0.26069	0.02507	10.39700	<2e-16	***
log_TotalSaldoUltimos180dias	-1.94988	0.19371	-10.06600	<2e-16	***
EstadoCivil.CASADO	-0.39421	0.03051	-12.92100	<2e-16	***
NumeroCuentasCerradas	0.38074	0.03085	12.34000	<2e-16	***
log_TotalSaldoUltimos90dias	1.63160	0.18837	8.66100	<2e-16	***
CantidadCuentasAF	-0.16205	0.03777	-4.29100	0.00002	***
NivelEstudio.BACHILLER	-0.07400	0.02569	-2.88100	0.00397	**

Fuente: Elaboración Propia

En este modelo todas las variables son significativas, sin embargo es necesario realizar un análisis de multicolineadidad para solventar posibles problemas en la estimación de los parámetros. Para calcular el grado de multicolinealidad entre las variables regresoras se usa frecuentemente el Factor de Inflación de Varianza (VIF

por sus siglas en ingles) [3] el cual fue calculado para las variables de la tabla 4.1 y cuyos resultados se muestran en la tabla 4.2

Cuadro 4.2: VIF Primer modelo Regresión Logíistica

Variable	VIF
log_PromSaldoUltimos180dias	517.32
log_TotalSaldoUltimos180dias	514.97
log_PromSaldoUltimos90dias	512.45
log_TotalSaldoUltimos90dias	491.53
TotalSaldoUltimos180Dias	55.30
TotalSaldoUltimos90Dias	51.76
SaldoUltimos30Dias	5.47
PromedioSaldoUltimos90Dias	3.99
SaldoUltimos180Dias	3.57
NumeroCuentasActivas	3.00
CantidadCuentasAF	2.94
Hijos	1.93
EstadoCivil.CASADO	1.78
Edad	1.59
AntiguedadPasivo	1.43
NivelEstudio.SUPERIOR	1.37
NivelEstudio.BACHILLER	1.32
CodigoProfesion.E30	1.18
AntiguedadLaboral	1.14
r_Saldo30s90dias	1.02
Genero.FEMENINO	1.02
Longitud	1.00
r_Saldo90s180dias	1.00

Fuente: Elaboración Propia

Se puede observar claramente que estamos ante un problema de multicolinealidad, razón por la cual se procede a realizar el análisis de las variables dentro del

[3]El VIF, cuantifica la intensidad de la multicolinealidad en un análisis de regresión de mínimos cuadrados y proporciona un índice de medición, el cual mide hasta qué punto la varianza de un coeficiente de regresión estimado se incrementa a causa de la colinealidad. Los valores de VIF mayores que 10 se consideran indicativos de multicolinealidad

modelo. Existen muchos métodos, por ejemplo aplicar un algoritmo ACP [4] y utilizar las proyecciones en lugar de las variables originales o en su defecto utilizar algún otro algoritmo de reducción de dimensionalidad. En este caso nos limitamos a eliminar las variables que presenten un mayor valor del VIF basándonos también en la matriz de correlación hasta tener un modelos en el cual ya no se presente el problema de la multicolinealidad. Como resultado tenemos un modelo con las siguientes características:

Cuadro 4.3: Modelo Final Regresión Logística Firth

Variable	Coeficiente	Std. Error	z value	Pr(>|z|)	
Intercepto	-2.20739	0.71818	-3.07	0.0021	**
AntiguedadLaboral	-0.00695	0.00077	-8.98	0.0000	***
AntiguedadPasivo	-0.00475	0.00042	-11.20	0.0000	***
Edad	0.01756	0.00190	9.24	0.0000	***
r_Saldo90s180dias	0.00015	0.00000	3.55	0.0004	***
Longitud	-0.02024	0.00900	-2.25	0.0245	*
Genero.FEMENINO	-0.53004	0.03954	-13.41	0.0000	***
SaldoUltimos180Dias	-0.00037	0.00004	-10.38	0.0000	***
SaldoUltimos30Dias	0.00020	0.00001	3.82	0.0001	***
NivelEstudio.SUPERIOR	-0.48121	0.04906	-9.81	0.0000	***
CodigoProfesion.E30	0.23863	0.04066	5.87	0.0000	***
EstadoCivil.CASADO	-0.34662	0.03967	-8.74	0.0000	***
log_TotalSaldoUltimos90dias	-0.23900	0.01389	-17.20	0.0000	***
NumeroCuentasActivas	0.37473	0.04615	8.12	0.0000	***
CantidadCuentasAF	-0.22246	0.06094	-3.65	0.0003	***

Fuente: Elaboración Propia

El modelo final elegido que se presenta en la tabla 4.3 posee 14 variables en comparación al modelo inicial donde se tenía 23 variables, además se puede observar en la tabla 4.4 que no existen problemas de multicolinealidad. Por otro lado, los signos de los coeficientes de las variables finales hacen sentido con el giro del negocio, por ejemplo a mayor antigüedad laboral disminuye (signo negativo) la probabilidad que un cliente caiga en default (Y=1 es un cliente malo); con esto cual podemos asegurar que tenemos un modelo robusto con un ajuste adecuado de sus coeficientes.

[4] Análisis de Componentes Principales

Cuadro 4.4: VIF Modelo final Regresión Logística

Variable	VIF
AntiguedadLaboral	1.133
AntiguedadPasivo	1.372
Edad	1.445
r_Saldo90s180dias	1.006
Longitud	1.006
Genero.FEMENINO	1.024
SaldoUltimos180Dias	1.258
SaldoUltimos30Dias	1.085
NivelEstudio.SUPERIOR	1.074
CodigoProfesion.E30	1.166
EstadoCivil.CASADO	1.08
log_TotalSaldoUltimos90dias	1.32
NumeroCuentasActivas	2.604
CantidadCuentasAF	2.835

Fuente: Elaboración Propia

4.1.2. Evaluación del modelo

Tanto el modelo de XGBoost como el modelo de regresión logística con estimación a través del método de Firth emplean las mismas medidas de rendimiento detalladas en los apartados 2.6.3, 2.6.2, 2.6.4, para evaluar su desempeño o rendimiento de discriminación. A continuación, se presentan los valores obtenidos para cada estadístico y las respectivas tablas de rendimiento o de performance.

Cuadro 4.5: Medidas de discriminación. Regresión Firth

Partición	KS	AUC	Gini
Entrenamiento	34.68	70.41	40.82
Prueba	29.07	69.09	38.18

Fuente: Elaboración Propia

Basados en las observaciones realizadas en el apartado 3.3, podemos afirmar que el modelo estimado con la regresión logística presenta un buen desempeño al tener

un coeficiente de Gini de 38.18 puntos.

En cuanto a las tablas de rendimiento, se presentan los resultados para el conjunto de test, que es el que nos interesa comparar. Basados en la tabla 4.6 y en el gráfico 4.1, podemos concluir que el modelo estimado a través de la regresión logística con estimación a través del método de Firth cumple con los siguientes requisitos:

- De ordenamiento, pues la tasa de malos en cada rango se incrementa conforme la probabilidad de default estimada aumenta.
- Adicionalmente, se verifica el requisito de acumulación de personas no pagadoras (malas) en los deciles más altos, pues capta un porcentaje importante del 57.18 % de los malos pagadores en los tres deciles más altos.

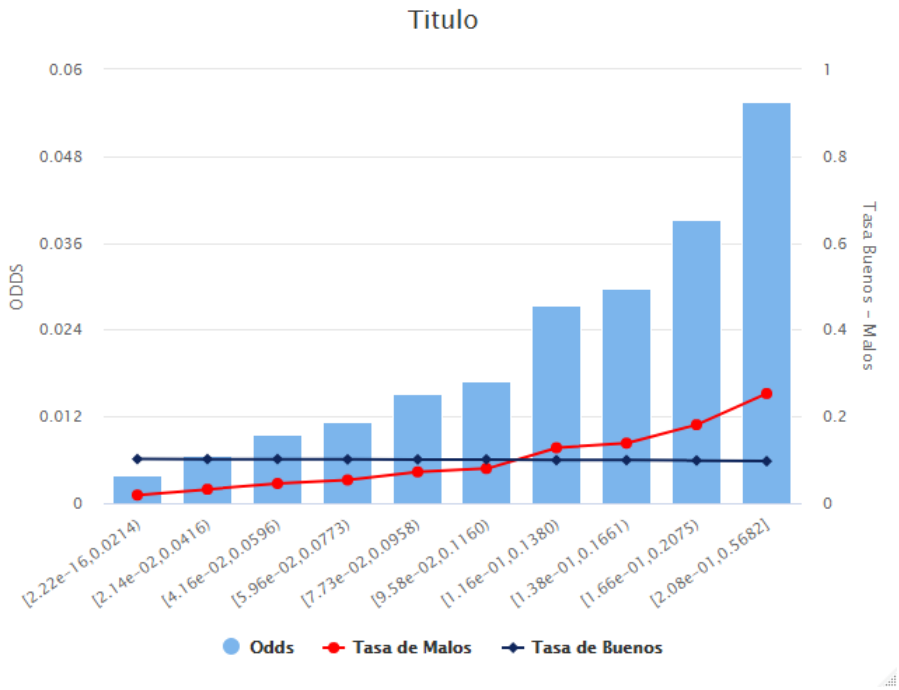
Con lo cual tenemos que el modelo presenta un buen rendimiento y podría ser utilizado para realizar nuevas predicciones.

Cuadro 4.6: Tabla de Odds Modelo Regresión Logística (Prueba)

Decil	Rango	FreqMalos	FreqBuenos	TasaMalos	TasaBuenos	Odds
1	[2.22e-16,0.0214)	7	1799	0.01860	0.10180	0.00389
2	[2.14e-02,0.0416)	12	1793	0.03190	0.10140	0.00669
3	[4.16e-02,0.0596)	17	1788	0.04520	0.10110	0.00951
4	[5.96e-02,0.0773)	20	1786	0.05320	0.10100	0.01120
5	[7.73e-02,0.0958)	27	1778	0.07180	0.10060	0.01519
6	[9.58e-02,0.1160)	30	1775	0.07980	0.10040	0.01690
7	[1.16e-01,0.1380)	48	1758	0.12770	0.09950	0.02730
8	[1.38e-01,0.1661)	52	1753	0.13830	0.09920	0.02966
9	[1.66e-01,0.2075)	68	1737	0.18090	0.09830	0.03915
10	[2.08e-01,0.5682]	95	1710	0.25270	0.09670	0.05556

Fuente: Elaboración Propia

Figura 4.1: Odds del Regresión Logística (Prueba)



Fuente: Elaboración Propia

4.2. XGBoost vs regresión logística con estimación a través del método de Firth

4.3. Comparación en precisión

Considerando la Tabla 4.5 y 3.11, se observa que los valores de los distintos estadísticos empleados para medir el rendimiento de discriminación del modelo, favorecen significativamente al modelo XGBoost. Analizando el valor del estadístico Gini, que es el considerado para la comparación (utilizado por la entidad bancaria), se tiene que el modelo XGBoost supera en 4 puntos de Gini al modelo de regresión Logística. Diferencias similares se observan si analizamos el valor de los otros estadísticos. De esta manera, se evidencia estadísticamente que el poder predictivo del XGBoost es superior al obtenido mediante la regresión logística con estimación a través del método de Firth.

Comparando ahora el rendimiento de discriminación a través de una tabla de performance 3.12 y 4.6 es posible observar que el modelo XGBoost capta a los malos pagadores en los tres primeros rangos frente al porcentaje captado por la regresión

logística con estimación a través del método de Firth. Adicionalmente, los Odds ratios en el modelo de XGBoost crece de una manera más pronunciada que en la regresión logística con estimación a través del método de Firth.
Por lo tanto, en base al análisis previo, podemos mencionar que el modelo de XGBoost resulta tener un mejor ajuste y rendimiento de discriminación que el estimado a través de la regresión logística con estimación a través del método de Firth, por ende concluimos que el método XGBoost permite generar un modelo más eficiente para abordar el problema de *credit scoring*.

4.4. Comparación de variables usadas en el modelo

Tanto en el modelo XGBoost como en la regresión logística con el método de Firth se eligieron las variables de tal manera que solamente se mantengan las variables representativas para cada modelo. Al hacer el comparativo del número de variables usadas vemos que la regresión logística utilizan 14 variables mientras que en el modelo XGBoost se usan 31 variables.

Por otro lado, para la regresión logística conocemos que la interpretabilidad puede obtenerse a través de los coeficientes del modelo o a partir de la tabla de efectos marginales (Tabla 4.7), la cual nos brinda una medida adecuada para saber el aporte de cada una de las variables en el modelo, es decir, podemos conocer cuánto cambiaría la probabilidad de default al realizar una variación en alguna variable que esté presente en el modelo. Como se había mencionado en el apartado 1.1.2 los algoritmos de aprendizaje automático no presentan la cualidad mencionada anteriormente pero se intenta realizar un aproximación local (para individuos cercanos) a través de un proceso de simulación que permite conocer cuál es el efecto de las variables en cada uno de los resultados de predicción para cada cliente (modelo LIME), tal como se mostró en el apartado 3.4. También se puede utilizar la tabla de Importancia de variables (Tabla C.2) para realizar el comparativo.

Al no ser los dos métodos 100 % comparables vamos a realizar un comparativo aproximado utilizando las características antes mencionadas.

- Las 3 variables más importante para el modelo Regresión Logística son:

    1. $Genero.FEMENINO$ (dF/dx = -0.03067)

Cuadro 4.7: Tabla de efectos marginales (Regresión Logística)

Variable	dF/dx	Std. Err.	z	P>|z|
AntiguedadLaboral	-0.00041	0.0000	-8.817	0.000
AntiguedadPasivo	-0.00028	0.0000	-10.933	0.000
Edad	0.00104	0.0001	9.09	0.000
r_Saldo90s180dias	0.00001	0.0000	3.527	0.000
Longitud	-0.00120	0.0005	-2.248	0.025
Genero.FEMENINO	-0.03067	0.0024	-13.047	0.000
SaldoUltimos180Dias	-0.00002	0.0000	-13.314	0.000
SaldoUltimos30Dias	0.00001	0.0000	3.792	0.000
NivelEstudio.SUPERIOR	-0.02653	0.0026	-10.187	0.000
CodigoProfesion.E30	0.01485	0.0027	5.558	0.000
EstadoCivil.CASADO	-0.02020	0.0023	-8.712	0.000
log_TotalSaldoUltimos90dias	-0.01421	0.0010	-14.507	0.000
NumeroCuentasActivas	0.02227	0.0028	8.008	0.000
CantidadCuentasAF	-0.01322	0.0036	-3.637	0.000

Fuente: Elaboración Propia

2. $NivelEstudio.SUPERIOR$ (dF/dx = -0.026525)
3. $NumeroCuentasActivas$ (dF/dx = 0.022273)

Mientras que para el modelo XGBoost son:

1. $AntiguedadPasivo$ (Importancia = 6.5 %)
2. $SaldoUltimos90Dias$ (Importancia = 6.5 %)
3. $r\_Saldo30s180dias$ (Importancia = 6.2 %)

A pesar que los modelos finales consideran variables similares, el modelo XGBoost da más importancia a las variables que están relacionadas con datos financieros mientras que el modelo de regresión logística a variables sociodemográficas.

- En la regresión logística se puede determinar si una variable tiene un efecto positivo o negativo en el resultado de la PD, en el caso del modelo de aprendizaje automático se debe recurrir al modelo LIME para poder hacer una comparación de este tipo. Aún así no es un método 100 % efectivo, ya que por ejemplo en la regresión logística el efecto negativo del Nivel Educativo Superior recae sobre todos los individuos, mientras que en el caso del modelo LIME, al ser un método local, cambia dependiendo del sujeto de estudio.

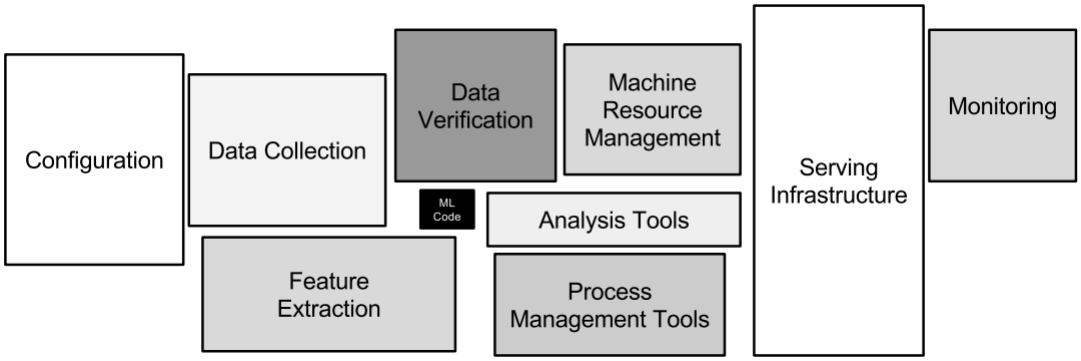
Ya que el trabajo propone el uso del modelo XGBoost se ve la necesidad de usar el modelo LIME para obtener un modelo semejante al de la regresión logística. Pero al ser técnicas pertenecientes al estado del arte todavía no presentan una manera adecuada de garantizar que la interpretabilidad del modelo sea la adecuada y es uno de los puntos que se podría realizar en estudios futuros. En el caso de la entidad bancaria, es de vital importancia contar con un modelo que tenga un excelente rendimiento pero también es necesario que se pueda determinar las razones de rechazo de manera individual para que los asesores de crédito puedan brindar una adecuada recomendación a los clientes. Es por tal razón que el resultado del modelo LIME en principio es una buena aproximación y cumple con las necesidades de la entidad bancaria.

Capítulo 5

Implementación del modelo en la entidad bancaria

Una de las verdades conocidas del mundo del aprendizaje automático (ML, por sus siglas en inglés) es que se necesita mucho más tiempo y esfuerzo para implementar modelos de ML en producción que para desarrollarlos. El famoso artículo: "***Hidden Technical Debt in Machine Learning Systems***" expuesto por Sculley et al. (2018) nos dice que : "Solo una pequeña fracción de los sistemas de ML del mundo real está compuesta por el código ML.". La infraestructura circundante requerida es vasta y compleja, como se puede observar en la Figura 5.1; solamente la zona de color negro corresponde al código del modelo y es muy pequeña en relación al resto de componente del sistema.

Figura 5.1: Componente de una solución de ML



Fuente: Sculley et al. (2018)

Y al igual que sucede en empresas a nivel mundial, la entidad bancaria para la cual se creó el modelo no fue la excepción, encontrándose con una dificultad muy grande al momento de poner en producción el modelo creado y explicado en el ca-

pítulo 3. Es por tal razón que en los siguientes apartados se presenta una serie de consejos para poder desplegar un modelo en producción bajo las mejores prácticas que beneficien tanto el trabajo del investigador como los requerimientos de la organización.

El objetivo es no dejar que el trabajo realizado en la creación del modelo sea desechado, sino que sirva para la toma de decisiones y la mejor manera de hacerlo es que la entidad pueda contar con una recomendación basada en el resultado del modelo en todos los casos que se requieran, como por ejemplo, en los procesos de calificación masiva, en los sistemas que usan los asesores de crédito o inclusive en empresas comerciales con las cuales se tenga convenios, mediante las disponibilización de API's [1] de calificación externa.

La forma de disponibilizar el modelo al resto de la organización no será detallará ya que va fuera de los límites de este trabajo, sin embargo, se va a mencionar un paso importante que se debe tener en cuenta antes del despliegue del modelo. Este paso es el comportamiento funcional del modelo en situaciones reales, para ello en los siguientes apartados se presentan las principales validaciones que se deberían realizar para que un modelo de *credit scoring* sea considerado listo para el despliegue.

5.1. Forward Testing

El Forward Testing de un modelo tiene el propósito de probar el rendimiento del modelo fuera de la muestra de construcción **simulando** un esquema de producción, generalmente se utilizan los últimos periodos de datos disponibles para realizar las pruebas correspondientes.

Si bien las validaciones estadísticas realizadas en el capítulo 3 muestran que el modelo es lo suficientemente bueno y presenta un performance adecuado, es necesario mostrar a las autoridades del banco que el modelo puede tener un buen rendimiento cuando este sea desplegado para la colocación de créditos de la institución, adaptándose a las necesidades del negocio; para ello se realizan pruebas como

[1] Es el acrónimo de *Application Programming Interface* que es un conjunto de funciones que permite a las aplicaciones acceder a datos e interactuar con componentes de software externos, sistemas operativos o microservicios.

la estabilidad temporal de las predicciones.

5.1.1. Evaluación estadística

En nuestro caso, dado que la partición del conjunto de datos se la hizo de manera temporal y no aleatoria (ver apartado 3.1.7), el conjunto de datos de prueba se acopla al esquema de *forward testing*, por tal razón todos los resultados presentados en el apartado 3.3 son considerados como válidos para la evaluación estadística del *forward testing*.

5.1.2. Estabilidad de la población

Es importante para una institución financiera poder contar con un determinado número de clientes preaprobados mensualmente para poder cumplir con las metas de negocio, por tal razón, si bajo un escenario económico 'normal', la calificación mensual [2] del modelo llegará a presentar cambios significativos en la *probabilidad de default* con respecto al comportamiento 'normal', teniendo como resultado un número menor de clientes preaprobados, esto representaría un gran problema a la institución porque se estarían perdiendo clientes potenciales, Por tal razón es importante evaluar la estabilidad de la distribución de la *probabilidad de default* a través de los meses de análisis del periodo de *forward testing*.

Índice de estabilidad de la población

El índice de estabilidad de la población (PSI por sus siglas en inglés) es una estadística muy utilizada que mide cuánto ha cambiado una variable a lo largo del tiempo. Un PSI alto puede alertar a la entidad bancaria sobre un cambio en las características de una población. Este cambio puede requerir investigación y posiblemente una actualización del modelo. El PSI se usa comúnmente entre los bancos para medir el cambio entre los datos de desarrollo del modelo y los datos actuales. Los bancos pueden enfrentar riesgos adicionales si los modelos se utilizan sin la validación adecuada. El uso incorrecto de PSI puede traer riesgos inesperados para estas instituciones. Sin embargo, no hay muchos estudios sobre las propiedades estadísticas del PSI. En la práctica, se utiliza la siguiente "regla empírica":

[2] Se realiza la calificación mensual, ya que la actualización de las bases de datos de la entidad se la realiza con esa temporalidad

- PSI <10 % significa un “pequeño cambio”.
- 10 % <PSI <25 % significa un “cambio moderado”.
- PSI >25 % significa un “cambio significativo, se requiere acción”.

Estos puntos de referencia se utilizan sin referencia a las tasas de error estadístico de tipo I o tipo II, en nuestro caso se presentan los siguientes valores para este indicador a través de los meses de análisis del forward testing:

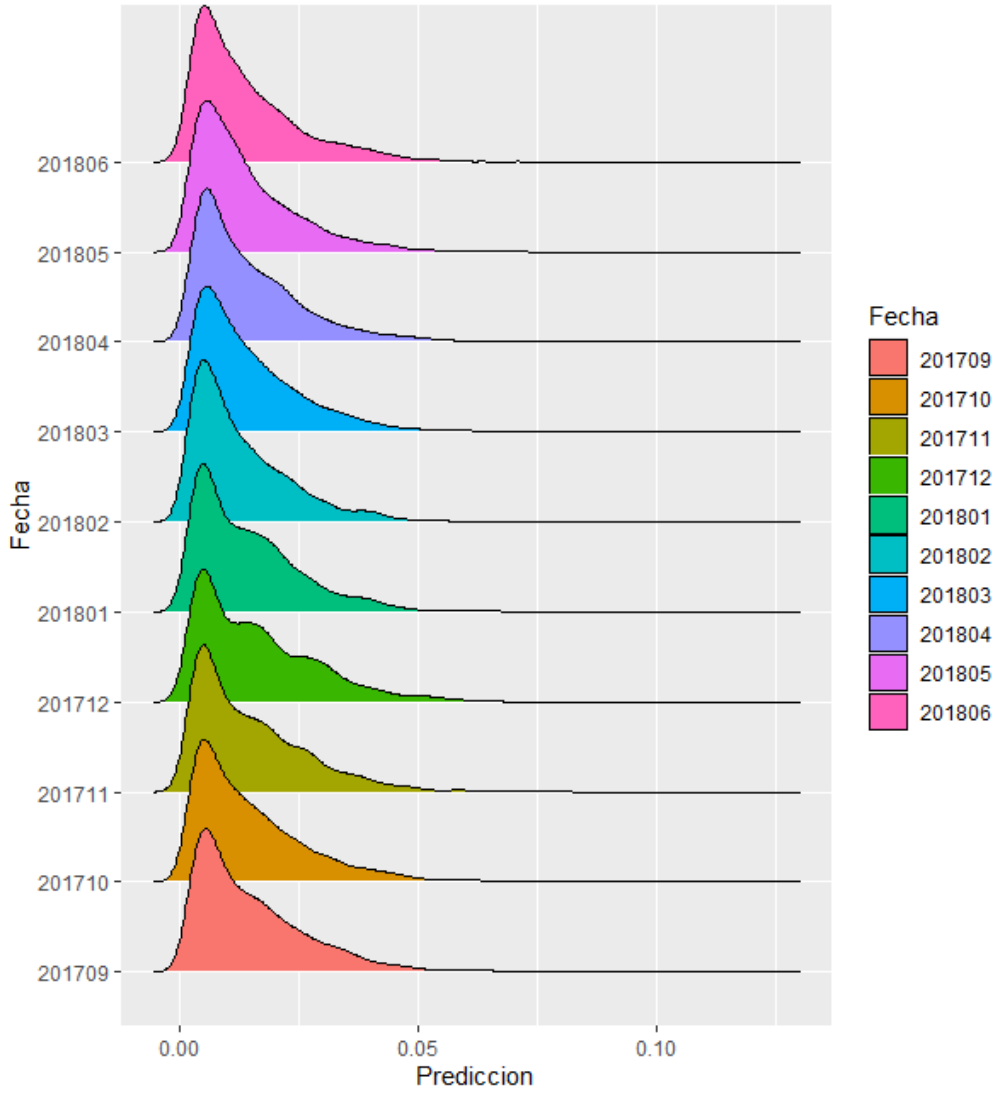
Cuadro 5.1: Distribución del número de registros por fecha de análisis

Fecha_Análisis	**PSI**
Sep 2017	1.9 %
Oct 2017	2.3 %
Nov 2017	3.4 %
Dic 2017	2.8 %
Ene 2018	2.9 %
Feb 2018	3.0 %
Mar 2018	2.5 %
Abr 2018	2.3 %
May 2018	2.9 %
Jun 2018	2.8 %

Fuente: Elaboración Propia

Se puede evidenciar que los indicadores se mantienen bajo el 10 % , por lo tanto, podemos asegurar que existe una estabilidad de la población. Además del cálculo del Índice de estabilidad de la población, en la figura 5.2 se observa gráficamente a partir de la distribución de las predicciones que la predicción del modelo es similar en todos los meses del periodo de análisis del *forward testing*

Figura 5.2: Estabilidad de las predicciones



Fuente: Elaboración Propia

5.2. Construcción de Perfiles de Riesgo

Recordemos que el resultado del modelo es la probabilidad de que un cliente caiga en default, si bien existen muchas metodologías para encontrar un punto de corte óptimo (CUTOFF) [3] que permita clasificar de mejor manera a un cliente bueno y a uno malo, basados en el concepto que a mayor probabilidad de default un cliente es más riesgoso, las entidades bancarias aprovechan esta variable respuesta no solamente como discriminador de otorgamiento de crédito sino que la usan para crear perfiles de riesgo, los cuales para adecuarse a las definiciones del banco tendrán 4

[3] Es el puntaje de crédito más bajo posible que se puede tener y aun así calificar para un préstamo.

categorías que se detallan continuación:

- Perfil de Riesgo Bajo
- Perfil de Riesgo Medio
- Perfil de Riesgo Alto
- Perfil de Riesgo Rechazado

Es importante el manejo de estas categorías dentro de la entidad bancaria ya que al conocer las características financiera de estos grupos se pueden ejecutar estrategias de colocación que van desde las más conservadoras hasta unas muy ambiciosas; dicho de otra manera, la institución podría empezar aprobando créditos a los clientes con Perfil de Riesgo Bajo y dependiendo de apetito de colocación se podrá ir aumentando el número de registros con clientes del siguiente perfil de riesgo, siempre teniendo claro cuales son las posibles pérdidas que podrían generarse. Para poder determinar dichas categorías a partir del resultado de nuestro modelo, vamos a empezar realizando el cálculo del punto de corte del modelo.

5.2.1. Determinación del Punto de Corte

Iñiguez y Morales (2019) presenta 2 metodologías para el cálculo del punto de corte, el primer método a partir de la curva ROC y el segundo utilizando matrices de transición. Dado que en los marcos metodológicos de la institución financiera se indica que se debe utilizar el método de la curva ROC, éste es el procedimiento que se consideró en este proyecto.

CUTOFF utilizando la curva ROC

A cada cliente que pertenece a la muestra de construcción, el modelo le asigna una probabilidad, que va de 0 a 1, el problema se centra ahora en determinar el puntaje sobre el cual se rechazan a los clientes, es decir en términos de probabilidad, sería la máxima probabilidad aceptada de que un cliente sea malo, por ejemplo si determinamos que la máxima probabilidad aceptada es 70 % y tenemos un cliente con probabilidad de ser malo mayor a 70 % automáticamente se le negaría cualquier operación.

Para identificar un punto de corte adecuado, se puede utilizar la curva ROC[4], su origen se dio en la estimación de errores en la transmisión de mensajes y actualmente es muy usada en el campo del aprendizaje automático. Dos definiciones importantes relacionadas con la curva ROC, son la sensibilidad y la especificidad:

- Sensibilidad (Se): Tasa de verdaderos positivos, es la proporción de casos positivos que fueron correctamente identificadas por el algoritmo.
- Especificidad (Sp): Tasa de verdaderos negativos, se trata de los casos negativos que el algoritmo ha clasificado correctamente. Expresa cuan bien puede el modelo detectar esa clase.

La curva ROC es un gráfico $1 - Sp$ vs Se, describiendo la propiedad de clasificación de la PD, cuando el punto de corte varía. La PD que maximiza el KS, por ejemplo, corresponde al punto en la curva ROC cuya distancia horizontal desde el eje es máxima. Este es el punto C en el gráfico 5.3. De esto se sigue que el punto es $(P_M(s), P_B(s))$, donde:

$$\begin{aligned} p_M(s) &= \frac{n_M(s)}{N_M} = \frac{\text{Clientes malos con PD igual a } s \text{ en una muestra de tamaño } n}{\text{Clientes malos en la muestra de tamaño } n} \\ p_B(s) &= \frac{n_B(s)}{N_B} = \frac{\text{Clientes buenos con PD igual a } s \text{ en una muestra de tamaño } n}{\text{Clientes buenos en la muestra de tamaño } n} \end{aligned} \tag{5.1}$$

y las distribuciones acumuladas se denotan por:

$$P_M(s) = \sum_{S \leq s} p_M(s) \quad \text{y} \quad P_B(s) = \sum_{S \leq s} p_B(s) \tag{5.2}$$

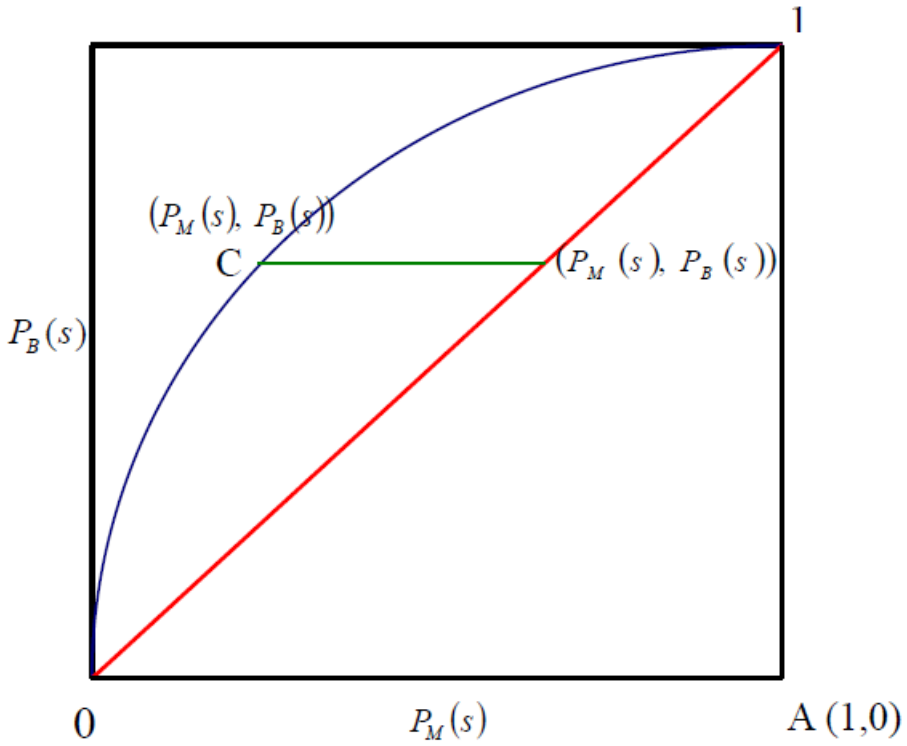
Entonces esta distancia horizontal es $(P_M(s) - P_B(s))$, la explicación de lo anterior se detalla en el gráfico 5.3.

En la práctica la construcción de la curva requiere la realización de los siguientes pasos:

1. Ordenar la PD en forma ascendente.
2. Determinar el porcentaje de clientes buenos y malos que comparten la misma PD ($p_M(s)$ y $p_B(s)$).

[4]Receiver Operating characteristic curve – Curva de características operativas del receptor, también llamada diagrama de Lorentz

Figura 5.3: Definición del Cutoff con la curva ROC



Fuente: (Iñiguez,2019)

3. Calcular el porcentaje acumulado ($P_M(s)$ y $P_B(s)$).

4. Calcular las diferencias ($P_M(s) - P_B(s)$) entre porcentajes acumulados por PD entre buenos y malos.

5. Identificar el score que produce la máxima diferencia.

Siguiendo los pasos descritos anteriormente, la máxima diferencia se produce en la PD igual a 0.02837, y esta será considerada el punto de discriminación para determinar un cliente bueno y un cliente malo.

5.2.2. Cálculo del Score

Antes de empezar con el perfilamiento de los clientes es conveniente realizar el cálculo del Score de crédito [5]. Este cálculo se da ya que la probabilidad que resulta del modelo no es muy amigable en el argot bancario, y para solucionar este inconveniente se ha manejado históricamente puntajes de 0 a 1000, que se obtienen usando

[5]Se puede determinar el Score a partir de la Probabilidad mediante el uso de una función de cambio de escala.

la fórmula $Score = 1000 * (1 - PD)$.

Al hacer el análisis estadístico sobre el punto de corte para determinar la probabilidad de que un cliente caiga en default, se determinó para nuestro modelo un valor de probabilidad de corte de aproximadamente 0.02837, este valor transformándolo a score equivale a un puntaje de 971. Con el fin de que nuestra escala se ajuste al estándar de algunas importantes empresas de calificación de riesgo como: Moody's, FICO, Equifax, entre otras; las cuales presentan sus scores de riesgo con un punto de corte (rechazo) que oscila entre los valores de 600 y 700, se procede a suavizar la distribución del *score* a través de una transformación exponencial [6] de la siguiente forma:

$$Score = 1000 * (1 - PD^{x})$$

tal que el punto de corte se traslade a 615 [7]. Para lograr dicho objetivo, se tendrá que encontrar el valor de x tal que se cumpla la igualdad para los valores $PD = 0.02837$ y $score = 615$.

Como resultado se obtiene que $x = 0.2679$, con lo cual la fórmula del cálculo del Score sería:

$$Score = 1000 * (1 - PD^{0.2679}) \tag{5.3}$$

5.2.3. Construcción de Perfiles utilizando la cartera vencida como indicador de corte

Se llama perfiles a las agrupaciones de score, cada perfil se identificará por su ratio de cartera vencida, lo que daría un mejor criterio para generar las políticas de control.

Se tiene como objetivo crear grupos o perfiles homogéneos con el fin de tener distintos niveles de riesgo asociado a cada grupo para poder manejar de mejor manera las estrategias de negocio basados en el apetito de riesgo del banco. Se construyen los puntos de corte con la base de prueba, con el fin de que estos sean lo más cercanos a la realidad, teniendo las siguientes consideraciones:

- Los umbrales son definidos basados en los requerimientos de presupuesto del Banco. El procedimiento consiste en hallar puntos de corte en los cuales los

[6] Recomendación de firma de consultoría McKinsey & Company
[7] Valor definido a partir del punto de corte de otros modelos de la entidad bancaria.

valores de cartera vencida de los clientes aprobados estén dentro del umbral requerido.

- El primer paso es dividir a los clientes rechazados de los aprobados considerando el punto de corte óptimo calculado en el apartado 5.2.1.
- Como siguiente paso, se escogió un umbral para el grupo "Riesgo Bajo" en el cual la cartera vencida corresponda al 0.5 %.
- Finalmente, se distribuyó de manera uniforme la cantidad de clientes en los rangos intermedios para determinar esos puntos de cortes restantes. Se obtiene el siguiente resultado

Cuadro 5.2: Perfiles por Ratio de Cartera Vencida

Perfiles	**Score Mínimo**	**Score Máximo**	**Proporción de Clientes**	**Pérdida Generada**
Riesgo Bajo	745	1000	26.92 %	0.55 %
Riesgo Medio	685	744	30.91 %	1.30 %
Riesgo Alto	616	684	29.80 %	2.74 %
Rechazado	0	615	12.34 %	5.24 %

Dependiendo del porcentaje de pérdida que la institución establezca se podría excluir uno o más perfiles de aquellos que generan más pérdida; por ejemplo, inicialmente se excluye el perfil Rechazado que genera en promedio un 5.24 % de pérdida, esto significaría que el modelo generaría una tasa de rechazo de aproximadamente 12.34 % de las solicitudes.

Las políticas de selección y control incrementarán su rigurosidad a medida que se avanza en los perfiles. Hasta el perfil Medio, se abarca un 57.83 % de la población y una pérdida promedio aproximada del 1.30 %, por tanto en estos perfiles habrá más colocación, mientras que en el perfil Alto existirá más limitación y control.

Cabe mencionar que la definición de los perfiles por Cartera Vencida, depende del enfoque que cada institución financiera tiene, por ejemplo existirán entidades para las cuales los perfiles definidos en este trabajo sean considerados de alto riesgo y otras que consideren lo contrario. Por tanto la construcción de los perfiles, en el

modo indicado, dependerá del riesgo que cada institución esté dispuesta a asumir.

Las políticas de gestión del riesgo de crédito son muy variadas, su tratamiento puede ser muy amplio, pero no es el enfoque del presente estudio, en el siguiente apartado se realizará un acercamiento a los temas que competen al monitoreo del sistema scoring construido.

5.3. Esquema de Pilotos

Siempre existirá un desconcierto al momento de implementar un nuevo modelo en una entidad financiera, ya que esto puede representar algunos problemas que van desde una mala ejecución de modelos en los sistemas , hasta que el número de clientes preaprobados disminuya drásticamente. Es por esta razón que basados en la forma de trabajo de la metodología *AGILE* [8] se dispone a realizar el despliegue del modelo de manera paulatina y controlada. Esto le permite a la organización estar alineada con la puesta en producción del modelo desde el principio hasta el fin de la implementación y sobretodo tener la visibilidad de los resultados que se van generando.

El proceso consiste en el lanzamiento de pequeñas campañas focalizadas en grupos de clientes con el fin de poder medir el performance del modelo tanto en rendimiento (efectividad) como en riesgo (mora). Para lograr este objetivo de detallan los pasos a seguir:

1. Obtener la predicción del modelo de *credit score* de originación para todos los clientes sin antecedentes crediticios, y utilizando la metodología detallada en el apartado 5.2 crear el Perfil de Riesgo de cada cliente; el cual será insumo para realizar todo el proceso de calificación y filtros de clientes, obteniendo como resultado una lista con todos los clientes aptos a un crédito según las definiciones de la institución.

2. Con esta lista de clientes se debe obtener una muestra aleatoria que será el grupo de clientes sobre el cual se evaluará al modelo.

[8] La entrega continua es un subconjunto de AGILE, en el que el equipo mantiene su producto listo para su lanzamiento o actualización en todo momento,asegurando una entrega paulatina y segura.

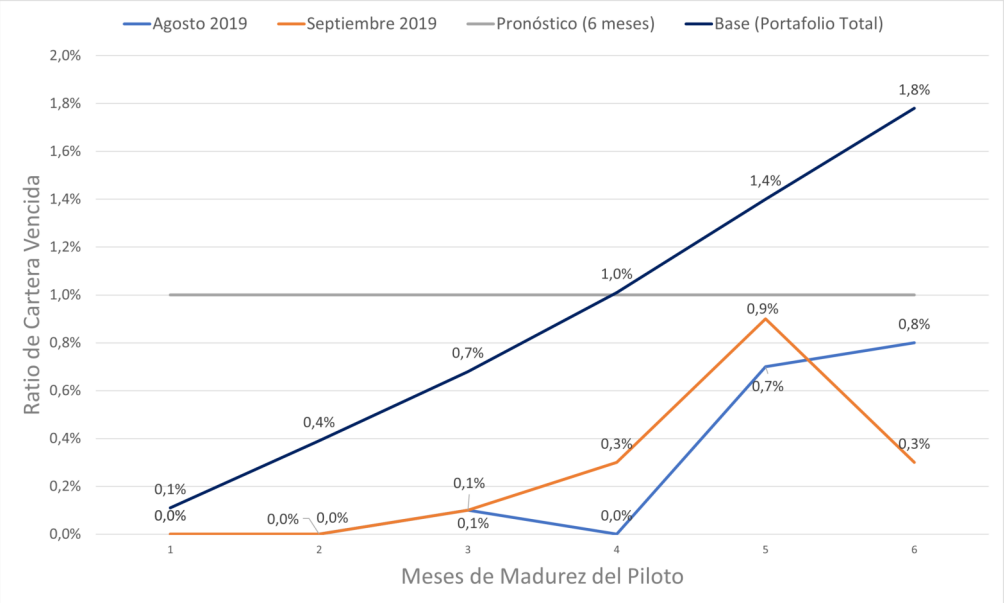
3. Se entrega esta lista de clientes al área de negocio para que puedan ser distribuidos a las distintas fuerzas de venta y comercializados.

4. Se empieza a medir los indicadores de efectividad en el corto plazo y los indicadores de salud de portafolio en el (mediano plazo).

5. Si el piloto genera buenos indicadores, se incrementará incrementará el tamaño de la base en las siguientes campañas de forma progresiva hasta que el 100 % de los clientes sean calificados con este modelo, pero en el caso que el piloto presente indicadores no muy buenos, se tiene que actuar de forma inmediata, realizando los debidos ajustes en el modelo y retomar el proceso iterativo de campañas hasta que el modelo sea implementado en la entidad bancaria.

Esta forma de trabajo permite realizar un despliegue seguro y controlado de los modelos, permitiendo reaccionar de manera inmediata en el caso que el modelo no se encuentre bien calibrado. En el caso de nuestro proyecto se realizó dos iteraciones de las cuales se presentan los resultados de cosechas [9] en la figura 5.4.

Figura 5.4: Resultado Pilotos



Fuente: Elaboración Propia

Se puede observar que todas las operaciones concedidas en los dos pilotos presentan buenos indicadores hasta el sexto mes de madurez de la operación a partir de la fecha de concesión, obteniendo resultados mucho menores a los esperados.

[9] Técnica utilizada para medir la salud de un portafolio de crédito a través del indicador de Ratio de Vencida

Capítulo 6

Conclusiones y Recomendaciones

La finalidad del presente estudio fue incorporar a las metodologías tradicionales de modelos de credit score, técnicas de aprendizaje automático, las cuales son herramientas estadísticas, modernas y objetivas que permitan determinar la probabilidad de incumplimiento de un cliente sin antecedentes crediticios dentro de una institución bancaria del Ecuador, con el fin de determinar el perfil de riesgo del cliente para mejorar la administración del portafolio de crédito de una entidad bancaria del Ecuador y adicional complementar la metodología con la implementación de un algoritmo que facilite la interpretabilidad del modelo.

6.1. Conclusiones

Una vez desarrollado el modelo estadístico, se puede elaborar una serie de conclusiones en base a sus resultados. A continuación, se enumeran los principales hallazgos encontrados a lo largo del desarrollo del presente trabajo:

1. Uno de los resultados proporcionados por el modelo estadístico es la importancia de las variables (tabla C.2) del modelo, a partir de la cual se puede identificar que para el segmento de clientes sin antecedentes crediticios las variables que ayudan a predecir de mejor manera el comportamiento de pago futuro [1] son principalmente aquellas relacionadas con los datos de cuentas de ahorro (suma, promedio y ratios de saldos en cuentas). En relación con modelos enfocados en otros segmentos de clientes, y para los cuales se ha determinado a través de varios trabajos de titulación que las variables más importantes son

[1] En caso de concederles su primer crédito.

las relacionadas a su historial de crédito [2], se puede concluir que este conjunto de variables de tipo *PASIVOS BANCARIOS* se convierten en una base para problemas de este estilo.

2. Comparando los diferentes estadísticos (KS, GINI, AUC-ROC) del modelo obtenido mediante la técnica XGBoost (Tabla 3.11), con los valores obtenidos para el modelo desarrollado mediante el método de la regresión logística con estimación a través del método de Firth (Tabla 4.5), tenemos que el modelo XGBoost presenta un mayor poder de predicción que el modelo de Firth, es decir, existe una mayor discriminación entre buenos pagadores y malos pagadores; y por ende una cantidad más reducida de verdaderos negativos, obteniendo así una clasificación más precisa. En cuanto a las tablas de rendimiento del modelo XGBoost 3.13 y modelo de Firth 4.6, podemos concluir que ambos modelos son adecuados, pero nuevamente el modelo XGBoost es el que presenta un mejor desempeño.

3. Desde un enfoque de negocios, el método de XGBoost no es muy apetecido por su difícil interpretabilidad, ya que lo ideal sería obtener una serie de reglas explícitas que determinan la probabilidad de incumplimiento de cada cliente. Sin embargo, el uso del modelo LIME (apartado 3.4) ha permitido evidenciar las ventajas y virtudes del uso de técnicas de aprendizaje automático, aprovechando el poder predictivo de las mismas. Por lo tanto se puede concluir que la implementación del modelo LIME (*Interpretable Machine Learning*) facilitó la aprobación por parte de las autoridades de la entidad bancaria, la implementación del modelo XGBoost como clasificador de credit score.

4. La estimación de un modelo en el que la variable de estudio (Y) presente un gran desequilibrio de clases, se refuerza empleando un método de remuestreo previamente a la aplicación del clasificador binario, y no aplicando directamente el clasificador. Dado que la naturaleza de nuestro problema nos exige obtener la probabilidad real [3] y al no contar con una metodología estándar para anular el efecto del remuestreo en la distribución resultante del modelo, como en la regresión logística; se procedió a estimar nuestro modelo ayudándonos de las característicos del algoritmo XGBoost las cuales permitieron crear un estimador robusto sin la necesidad de utilizar las técnicas de remuestreo.

[2] Variables creadas a partir de los días de mora, saldos vencidos, saldos por vencer, etc.

[3] Utilizar un método de remuestreo hace que la probabilidad de incumplimiento generada por el modelo sobreestime a la probabilidad de default real

5. Dentro del ciclo de crédito la etapa de preventa permite ofrecer al cliente un producto financiero de forma proactiva, generalmente esta se da en forma de campañas de crédito. La aplicación de este modelo scoring en particular permite: mejorar la rentabilidad incrementando la tasa de aceptación y/o la reducción de la morosidad, ahorrar recursos, mejorar el servicio al cliente y apoyar a la toma de decisiones sin eliminar el criterio del analista de riesgo y la experiencia de los asesores de crédito.

6.2. Recomendaciones

A partir de la elaboración del modelo estadístico se pueden extraer diversas recomendaciones que servirán de ayuda para trabajos futuros y que permitirán mejorar el desempeño del modelo de detección de fraude crediticio:

1. Se determinó en el trabajo que las variables de tipo Pasivos Bancarios son las más representativas a la hora de enfrentarse a un modelo para clientes sin antecedentes crediticios, aunque en el proceso se encontró que existía una fuerte relación entre la variable dependiente y variables provenientes de uso de tarjetas de débito. Esta relación prometía mejorar la predictibilidad del modelo pero al final no se pudieron incluir estas variables porque no se contaba con la información histórica requerida [4] Se recomienda en trabajos futuros considerar la inclusión de estas variables ya que pueden aportar a que mejore el poder predictivo del modelo, esto sería de gran ayuda ya que por la naturaleza misma del problema se cuenta con una cantidad limitada de variables y la adición de nuevas fuentes de datos beneficia a la clasificación de clientes.

2. La nuevas tecnologías cada vez van brindando mayores facilidades al ser humano, por tal razón, se recomienda a la institución bancaria aprovechar sus fuentes generadoras de datos para poder contar con más variables de tipo ubicación geográfica, que se vio eran de gran importancia para el modelo. Estas variables pueden obtenerse a través de la localización de las aplicaciones móviles, a partir de las cuales se podría conocer información acerca de viajes, visita de lugares suntuarios, obtener información actualizada del domicilio y lugar de trabajo; para en un futuro añadirlas al modelo de credit score.

[4] Exigía un periodo largo de tiempo para poder reprocesar la información para la construcción de las variables.

3. En relación con la recomendación anterior, es sumamente importante aprovechar al máximo las nuevas herramientas tecnológicas que se van creando día a día. En nuestro caso se contaba con la dirección de los clientes como una variable de texto (ejemplo: "Sangolquí, calle xxx."), y al hacer uso de la plataforma *Google Cloud* con su API *Google Maps* [5] a través de un algoritmo adecuado se pudo obtener las coordenadas geográficas de esas direcciones, permitiendo capturar la correlación espacial de las ubicaciones que como se puede ver en las variables importantes del modelo son de gran impacto. Por tal motivo se recomienda a la organización, el uso de técnicas modernas de ingeniería de datos [6] más actuales para poder aprovechar otro tipo de información como la semi-estructurada y la no estructurada.

4. A partir de las distintas iteraciones que se hizo en la construcción del modelo, se pudo notar que las variables mientras más información del pasado poseían, mejor era el rendimiento del modelo. Sin embargo por la naturaleza del problema, los clientes a los que se califica son nuevos, por lo tanto es necesaria una pronta estrategia de bancarización porque la entidad bancaria no desea esperar que el cliente tenga, es por eso que a pesar de contar con información histórica de 12 meses, se decidió utilizar solamente una ventana de tiempo hacia atrás de 6 meses. En este punto si bien es importante buscar el mejor rendimiento del modelo también se busca maximizar la cantidad de clientes calificados, para nuestro caso el modelo con información de 6 meses de historia perdió solamente 1 punto porcentual de $GINI$ frente al modelo con 12 meses de historia, pero se pudo calificar a un 35 % más de clientes que de no haber tenido esa consideración no hubieran podido tener una recomendación del modelo. Por lo tanto, como recomendación se diría a la entidad bancaria que se considere este aspecto a la hora de la construcción de nuevos modelos ya que una adecuada elección de la ventana de tiempo puede mejorar la administración de modelos trayendo consigo mejores rendimientos de los modelos y una mayor calificación de clientes.

5. De la tabla C.2 se puede notar que la variable *ScoreBuroConyuge* fue incluida en el modelo sobre algunas otras como información de cuentas corrientes, esto nos lleva a pensar que al obtener la información no solamente del cliente sino de su entorno familiar ayudaría a mejorar la predicción de incumplimiento

[5] https://cloud.google.com/maps-platform/maps

[6] Los ingenieros de datos son responsables de encontrar tendencias en conjuntos de datos y desarrollar algoritmos para ayudar a que los datos sin procesar sean más útiles para la empresa.

de clientes del segmento sin información crediticia, por eso se recomienda a la institución diseñar un esquema de datos donde se pueda comprar y almacenar la información del comportamiento de pago de las personas que conforman el núcleo familiar de los potenciales clientes, sobretodo si la estrategia del banco le apunta a bancarizar a este nuevo segmento de clientes.

6. Es importante realizar el monitoreo y versionamiento de este tipo de modelos de manera periódica ya que como se discutió en el capitulo 5, el paso a producción de los modelos presenta una alta complejidad y llevar una administración adecuada de los mismos facilitará el seguimiento del rendimiento de los modelos. Es por tal razón que se recomienda a la entidad bancaria contar con una metodología robusta de monitoreo que además recoja la información detalla en el capitulo 5 para poder facilitar el proceso de creación, implementación, monitoreo y calibración de modelos con el fin de siempre contar con las mejores calificaciones adecuadas para el otorgamiento de crédito.

7. Si bien el algoritmo de interpretabildad LIME presenta resultados de forma local (para cada caso de estudio), y necesita de un tratamiento adicional para hacer generalizable los resultados a toda la población, es de utilidad poder contar con esta herramienta en las máquinas de los asesores ya que muchos de los casos en donde se le niegue el crédito a un cliente van a tener que pasar por el procesos manual (revisión del asesor de crédito) y para ellos es de gran importancia poder saber cuales son as características de las personas por las cuales se le negó el crédito.

8. Si bien la forma de despliegue progresiva de modelos mencionada en el apartado 5.3 se adecuó de manera perfecta a los intereses del área de riesgo, la misma puede servir para manejar un esquema de evaluación de nuevas estrategias de negocio y descubrir nuevos segmentos de clientes, permitiendo probar hipótesis y siempre asegurando el control sobre el presupuesto destinado y el riesgo asumido.

9. Se recomienda a la entidad bancaria diseñar productos financieros enfocados en la bancarización de clientes, de esta manera con el uso adecuado del modelo XGBoost creado para este segmento de clientes, se puede potenciar la oferta de tarjetas de crédito y créditos emergente (montos bajos) para así ir ampliando el portafolio de clientes y a la vez que se mantener controlado el riesgo asumido.

Apéndice A

ANEXO 1: Definiciones y algoritmos

La siguientes definiciones fueron tomadas de Breiman et al. (1984) y Hastie et al. (2009) .

A.1. Función de pérdida de Huber

En la literatura sobre robustez estadística, se ha propuesto una variedad de criterios para la función de pérdida en problemas regresión que brindan una fuerte resistencia (si no inmunidad absoluta) a valores atípicos brutos, mientras que son casi tan eficientes como los mínimos cuadrados para los errores gaussianos. A menudo son mejores que cualquiera de los dos para distribuciones de error con colas moderadamente pesadas. Uno de estos criterios es el criterio de pérdida de Huber. cuya fórmula se detalla a continuación.

$$L\left(y, f(x)\right) = \begin{cases} [y - f(x)]^2 & \text{para } |y - f(x)| \leq \delta \\ \delta\left(|y - f(x)| - \frac{\delta}{2}\right) & \text{otros casos.} \end{cases} \tag{A.1}$$

A.2. Función de pérdida de desviación

El objetivo del algoritmo de clasificación es producir márgenes positivos con la mayor frecuencia posible. Cualquier criterio de pérdida utilizado para la clasificación debería penalizar más los márgenes negativos que los positivos, ya que las observaciones de márgenes positivos ya están correctamente clasificadas. Para ello se utiliza la función de pérdida de la desviación que aprovecha las características de la función de pérdida de Huber para el caso de clasificación y cuya fórmula se

detalla a continuación.

$$\begin{aligned} L\left(y, p(x)\right) &= -\sum_{k=1}^{K} I(y = G_k) \log p_k(x) \\ &= -\sum_{k=1}^{K} I(y = G_k) f_k(x) + \log\left(\sum_{l=1}^{K} e^{f_l(x)}\right) \end{aligned} \tag{A.2}$$

A.3. Función Sofmax

La función softmax, también conocida como softargmax o función exponencial normalizada, es una generalización de la función logística a múltiples dimensiones. Se utiliza en la regresión logística multinomial y a menudo se utiliza como la última función de activación de una red neuronal para normalizar la salida de una red a una distribución de probabilidad sobre las clases de salida predichas.

$$p_k(x) = \frac{e^{f_k(x)}}{\sum_{l=1}^{K} e^{f_l(x)}} \tag{A.3}$$

A.4. Clasificador de Bayes

Cuando buscamos un clasificador $G(x)$ que tome valores en G con varias categorías es suficiente conocer las probabilidades condicionales de la clase $p_k(x) = Pr(Y = G_{k|j})$; con $k = 1; 2; ...; K$, con lo cual entonces el clasificador de Bayes es:

$$G(x) = G_k \ \textit{donde}\ k = \arg\max_{l} p_l(x) \tag{A.4}$$

A.5. Algoritmo Impulso progresivo por etapas (Forward stagewise boosting)

Algoritmo 3 Forward Stagewise Additive Modeling

1: Inicializar $f_0(x) = 0$
2: **para** $m = 1$ hasta M **hacer**
3: Calcular

$$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma))$$

4: Fijar

$$f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma_m)$$

5: **fin para**
6: **devolver** $f_m(x)$

Apéndice B

ANEXO 2: Código para la creación del modelo

B.1. Funciones para EDA

B.1.1. Variables Numéricas

```
descrip_num <- function(DT){
result <- data.table(Variable=character(),
                     Por_Nulos=numeric(),
                     Por_Ceros=numeric(),
                     Media=numeric(),
                     Desv_Est=numeric(),
                     Minimo=numeric(),
                     Cuartil1=numeric(),
                     Mediana=numeric(),
                     Cuartil3=numeric(),
                     Maximo=numeric()
                     )
n <- nrow(DT)
for (x in names(DT)){
    Variable <- x
    Por_Nulos <- 100*round(sum(is.na(DT[[x]]))/n,4)
    Por_Ceros <- 100*round(sum(DT[[x]]==0, na.rm = TRUE)/n,4)
```

```
    Media <- round(mean(DT[[x]], na.rm = TRUE), 2)
    Desv_Est <- round(sd(DT[[x]], na.rm = TRUE), 2)
    Minimo <- round(min(DT[[x]], na.rm = TRUE), 2)
    Cuartil1 <- round(quantile(DT[[x]],0.25,na.rm = TRUE), 2)
    Mediana <- round(median(DT[[x]], na.rm = TRUE), 2)
    Cuartil3 <- round(quantile(DT[[x]], 0.75,na.rm = TRUE), 2)
    Maximo <- round(max(DT[[x]], na.rm = TRUE), 2)
    result <- rbind(result, data.table(Variable, Por_Nulos,
                                       Por_Ceros, Media, Desv_Est,
                                       Minimo, Cuartil1, Mediana,
                                       Cuartil3, Maximo))
}
return(result)
}
}
```

B.1.2. Variables Categóricas

```
descrip_cat <- function(DT){
result <- data.table(Variable=character(),
                     Por_Nulos=numeric(),
                     Num_Categorias = integer(),
                     Clase_mayoritaria=character(),
                     Freq_Cat_mayoritaria = numeric(),
                     Clase_minoritaria=character(),
                     Freq_Cat_minoritaria = numeric()
                     )
#DT <- dttrain[, VarCat, with=F]
n <- nrow(DT)
#x <- names(DT)[4]
for (x in names(DT)){
    Variable <- x
    tab <- sort(prop.table(table(as.character(DT[[x]]))))
    tab1 <- names(tab)
    Por_Nulos <- 100*round(sum(is.na(DT[[x]]))/n,4)
```

```
    Num_Categorias <- length(levels(DT[[x]]))
    Clase_mayoritaria <- tab1[length(tab1)]
    Freq_Cat_mayoritaria <- 100*round(tab[length(tab)],4)
    Clase_minoritaria <- tab1[1]
    Freq_Cat_minoritaria <- 100*round(tab[1],4)
    result <- rbind(result, data.table(Variable, Por_Nulos,
                    Num_Categorias, Clase_mayoritaria,
                    Freq_Cat_mayoritaria, Clase_minoritaria,
                    Freq_Cat_minoritaria))
}
return(result)
}
```

B.2. Funciones para Análisis Bivariado

B.2.1. Variables Numéricas vs Dependiente

```
library(MLmetrics)

# dttrain.- Conjunto de datos con el que se esta trabajando
# Varnum.- Lista de variables numericas en el analisis

Variables <- character(0)
KS <- character(0)

for (i in VarNum) {
    Variables <- c(Variables, i)
    zz <- na.omit(dttrain[, c(i, 'Y'),with=F])
    xx <- zz[[i]]
    yy <- zz[['Y']]
    KS <- c(KS, KS_Stat(y_pred = xx, y_true = yy))
}
data.frame(Variables, KS)
```

B.2.2. Variables Categóricas vs Dependiente

```
library(InformationValue)

# dttrain. Conjunto de datos con el que se esta trabajando
# VarCat.- Lista de variables categoricas en el analisis

Variables <- character(0)
IVal <- character(0)

for (i in VarCat) {
    Variables <- c(Variables, i)
    IVal <- c(IVal, IV(dttrain[[i]], dttrain[['Y']],
                valueOfGood = 1))
}

data.frame(Variables, IVal)
```

B.3. Búsqueda de hiperparámetros

Cuadro B.1: Hiperparámetros iniciales de búsqueda

Parámetro	Valor1	Valor2	Valor3	Valor4
eta	0.01	0.05	0.08	-
gamma	0.01	0.05	-	-
min_child_weigh	50	75	-	-
subsample	0.75	0.65	-	-
colsample_bytree	0.65	0.55	-	-
max_depth	8	10	12	-
alpha	0	0.1	1	10
lambda	0	0.1	1	10
max_delta_step	1	10	100	-

Fuente: Elaboración Propia

B.4. Búsqueda de hiperparámetros

```
ntrees <- 1000

searchGridSubCol <- expand.grid(
    eta = c(0.01, 0.05, 0.08)
    ,gamma = c(0.01, 0.05)
    ,min_child_weight = c(50,75)
    ,subsample = c(0.75, 065)
    ,colsample_bytree = c(0.65 0.55)
    ,max_depth = c(8, 10, 12)
    ,alpha = c(0, 0.1, 1, 10)
    ,lambda = c(0, 0.1, 1, 10)
    ,max_delta_step = c(1, 10, 100)
)
gc()

AUC_Errors_Hyperparameters <-
    apply(searchGridSubCol, 1, function(parameterList){
    currentEta <- parameterList[["eta"]]
    currentGamma <- parameterList[["gamma"]]
    currentMinChildW <- parameterList[["min_child_weight"]]
    currentSubsampleRate <- parameterList[["subsample"]]
    currentColsampleRate <- parameterList[["colsample_bytree"]]
    currentDepth <- parameterList[["max_depth"]]
    currentAlpha <- parameterList[["alpha"]]
    currentLambda <- parameterList[["lambda"]]
    currentmax_delta_step <- parameterList[["max_delta_step"]]
    xgboostModelCV <-
        xgb.cv(data =  dtrain,
                nrounds = ntrees,
                watchlist = watchlista,
                early_stopping_rounds = 10,
                nfold = 5,
                booster = 'gbtree',
                showsd = TRUE,
```

```
                    verbose = FALSE,
                    nthread = 4,
                    eval_metric = "auc",
                    objective = "binary:logitraw",
                    max_delta_step = 10,
                    eta = currentEta,
                    gamma = currentGamma,
                    min_child_weight = currentMinChildW,
                    subsample = currentSubsampleRate,
                    colsample_bytree = currentColsampleRate,
                    max.depth = currentDepth,
                    alpha = currentAlpha,
                    lambda = currentLambda )

    xvalidationScores <-
        as.data.frame(xgboostModelCV$evaluation_log)
    auc_test <- tail(xvalidationScores$test_auc_mean,1)
    auc_train <- tail(xvalidationScores$train_auc_mean,1)
    output <- return(
            c(auc_train=auc_train,
                auc_val=auc_test,
                Eta = currentEta,
                Gamma = currentGamma,
                MinChidWeight = currentMinChildW,
                SubsampleRate = currentSubsampleRate,
                ColsampleRate = currentColsampleRate,
                MaxDepth = currentDepth,
                Alpha = currentAlpha,
                Lambda = currentLambda,
                max_delta_step = currentmax_delta_step
            )
        )
})

hp <- data.table(t(AUC_Errors_Hyperparameters))
hp[order(-auc_val)]
```

B.5. Estimación del modelo XGBoost

B.5.1. Estimación del modelo

```
set.seed(123)
xgb <-
    xgb.train(
        data = xgb.DMatrix(data = as.matrix(train_variables),
        label = as.matrix(train_label)),
        watchlist = watchlista,
        print_every_n = 10,
        booster = "gbtree",
        eval_metric = "auc",#error@0.2
        objective = "binary:logistic",
        nthread = 4,
        showsd = TRUE,
        #normalize_type= "forest",
        #tree_method = "auto",
        #scale_pos_weight = scala,
        nround = 1000, #Numero maximo de iteraciones
        early_stopping_rounds = 25,
        eta = 0.1,
        gamma = 0.01,
        min_child_weight = 50,
        subsample = 0.6,
        colsample_bytree = 0.5,
        max_depth = 10,
        max_delta_step = 10,
        alpha = 1, # parametro de regularizacion
        lambda = 1 # parametro de regularizacion L2
        #,nfold = 10
    )
gc()
```

B.5.2. Iteraciones del modelo

```
xgb$best_iteration
ggplot(data = df,
    mapping = aes(x=iter,
        y = Entrenamiento_auc,
            color = 'Entrenemiento')) +
    geom_line(size =1.1)+
    geom_line(data = df,
        mapping = aes(x=iter,
            y = Validacion_auc,
            color = 'Test'),
        size =1.1)+
    scale_x_continuous(name = 'Iteraciones',
        breaks = seq(from=0, to=200, by=25))+
    geom_vline(xintercept = xgb$best_iteration,
        size =1.1,
        color = '#458718')+
    scale_y_continuous(
        name = 'AUC',
        labels = function(x) paste0(x*100, "%"),
        limits = c(0.5, 0.8)) +
    theme(legend.title=element_blank(),
          legend.text=element_text(size=13),
          legend.position = c(.19,.9),
          legend.box = 'horizontal',
          legend.margin = margin(r = 125, l = 125),
          legend.background = element_rect(fill = NA))
```

B.5.3. Métricas de evaluación del modelo

```
prediccion_train_xgb <- predict(xgb, dtrain)
prediccion_val_xgb <- predict(xgb, dval)
prediccion_test_xgb <- predict(xgb, dtest)

ks_train <-
    MLmetrics::KS_Stat(y_pred = prediccion_train_xgb,
```

```
        y_true = dttrain1$Y)
ks_val <-
    MLmetrics::KS_Stat(y_pred = prediccion_val_xgb,
        y_true = dtval1$Y)
ks_test <-
    MLmetrics::KS_Stat(y_pred = prediccion_test_xgb,
        y_true = dttest1$Y)

AUC_train <-
    MLmetrics::AUC(y_pred = prediccion_train_xgb,
        y_true = dttrain1$Y)
AUC_val <-
    MLmetrics::AUC(y_pred = prediccion_val_xgb,
        y_true = dtval1$Y)
AUC_test <-
    MLmetrics::AUC(y_pred = prediccion_test_xgb,
        y_true = dttest1$Y)

gini_train <-
    2*MLmetrics::AUC(y_pred = prediccion_train_xgb,
        y_true = dttrain1$Y)-1
gini_val <-
    2*MLmetrics::AUC(y_pred = prediccion_val_xgb,
        y_true = dtval1$Y)-1
gini_test <-
    2*MLmetrics::AUC(y_pred = prediccion_test_xgb,
        y_true = dttest1$Y)-1

data.frame(Particion=c('Entrenamiento','Validacion','Test')
           ,KS = c(ks_train, ks_val, ks_test)
           ,AUC = c(AUC_train, AUC_val, AUC_test)
           ,Gini = c(gini_train, gini_val, gini_test))
```

B.5.4. Importancia de variables

```
feature_importance <-
```

```
    xgb.importance(model = xgb) %>%
    as.data.table()

feature_importance %>% head(20)


xgb.plot.importance(feature_importance,
    rel_to_first = TRUE,
    xlab = "Relative importance",
    top_n = 15)

xgb.plot.importance(feature_importance,
    rel_to_first = FALSE,
    xlab = "Importancia de variables",
    top_n = 15)
```

B.6. Código LIME

```
library(lime)
load('Datos/ModelData.RData')

test_set <-
    sample(seq_len(nrow(test_variables)), 20)

explainer <-
    lime(train_variables, xgb, bin_continuous = T)

explanation <-
    data.table(explain(test_variables[test_set, ],
                       explainer = explainer,
                       n_labels = 1,
                       n_features = 15,
                       n_permutations = 1000))

explanation1 <- explanation[case==11,]
```

```
plot_features(explanation1)
```

B.7. Implementación algoritmo Regresión logística con el método de Firh

```
# Variables a inluir en el modelo
incluir <- c("r_Saldo30s90dias", "SaldoUltimos90Dias",
        "AntiguedadLaboral", "AntiguedadPasivo",
        "PromedioSaldoUltimos90Dias", "Edad",
        "r_Saldo90s180dias", "Longitud",
        "LatitudAgencia", "r_Saldo30s180dias",
        "Genero.FEMENINO", "SaldoUltimos180Dias",
        "LongitudAgencia", "PromedioSaldoUltimos180Dias",
        "SaldoUltimos30Dias", "NivelEstudio.SUPERIOR",
        "Latitud", "TotalSaldoUltimos90Dias",
        "TotalSaldoUltimos180Dias", "Hijos",
        "log_PromSaldoUltimos180dias",
        "log_PromSaldoUltimos90dias", "EstadoCivil.SOLTERO",
        "CodigoProfesion.E30", "log_TotalSaldoUltimos180dias",
        "EstadoCivil.CASADO", "NumeroCuentasCerradas",
        "ScoreBuroConyuge", "log_TotalSaldoUltimos90dias",
        "NumeroCuentasActivas", "Clasificacionmadre.6",
        "CantidadCuentasAF", "NivelEstudio.BACHILLER",
        "ClasificacionConyuge.4", "Clasificacionmadre.2",
        "ScoreBuromadre", "ScoreBuroPadre")

# Formula a partir de la cual se estima el modelo
ff <- formula(paste0('Y~', paste(incluir, collapse = '+')))

# Fijar semilla para hacer los calculos reproducibles
set.seed(1)

#Estimacion del modelo
m1 <- logistf(formula = ff, data = dttrain1, pl = F,
    control = logistf.control(
```

```
        maxit = 50000,
        maxhs = 5,
        maxstep = 5,
        lconv = 1e-05,
        gconv = 1e-05,
        xconv = 1e-05,
        collapse = TRUE
))
```

Apéndice C

ANEXO 3: Resultados del modelo

C.1. Estadístico KS para variables numéricas

Cuadro C.1: Estadístico KS para las variables numéricas

Variable	KS	Variable	KS
r_Saldo30s90dias	13.128	SaldoUltimos30Dias	0.504
r_Saldo30s360dias	12.460	TotalSaldoUltimos90Dias	0.208
r_Saldo30s180dias	11.657	log_TotalSaldoUltimos90dias	0.208
NumeroCuentasActivas	7.138	PromedioSaldoUltimos90Dias	0.188
NumeroCuentasCerradas	7.138	log_PromSaldoUltimos90dias	0.188
CantidadCuentasAF	7.128	TotalSaldoUltimos180Dias	0.169
Hijos	6.983	log_TotalSaldoUltimos180dias	0.169
r_Saldo90s360dias	4.856	PromedioSaldoUltimos180Dias	0.166
ScoreBuromadre	4.824	log_PromSaldoUltimos180dias	0.166
r_Saldo180s360dias	4.491	TotalSaldoUltimos360Dias	0.160
LongitudAgencia	4.122	PromedioSaldoUltimos360Dias	0.157
Latitud	3.020	log_TotalSaldoUltimos360dias	0.151
Longitud	2.889	log_PromSaldoUltimos360dias	0.148
LatitudAgencia	2.806	AntiguedadLaboral	0.071
r_Saldo90s180dias	2.649	SaldoUltimos360Dias	0.034
ScoreBuroPadre	2.513	SaldoUltimos90Dias	0.015
ScoreBuroConyuge	2.311	SaldoUltimos180Dias	0.009
CantidadCuentasAH	1.583	AntiguedadPasivo	0.000
Edad	0.734	CantidadCuentasCO	0.000

Fuente: Elaboración Propia

C.2. Importancia de Variables

Cuadro C.2: Importancia de variables

Variable	Cover	Frequency	Importance
AntiguedadPasivo	5.4 %	7.2 %	6.5 %
SaldoUltimos90Dias	10.4 %	5.8 %	6.5 %
r_Saldo30s180dias	4.6 %	6.8 %	6.2 %
SaldoUltimos180Dias	4.5 %	5.6 %	6.1 %
PromedioSaldoUltimos180Dias	12.8 %	4.8 %	5.8 %
log_TotalSaldoUltimos180dias	6.1 %	5.1 %	5.6 %
AntiguedadLaboral	3.1 %	5.8 %	5.5 %
PromedioSaldoUltimos90Dias	4.1 %	4.8 %	5.1 %
log_PromSaldoUltimos90dias	4.5 %	3.4 %	4.9 %
r_Saldo30s90dias	7.7 %	5.3 %	4.9 %
Genero.FEMENINO	3.4 %	4.1 %	4.7 %
LatitudAgencia	2.8 %	5.1 %	3.8 %
Longitud	2.1 %	3.9 %	3.3 %
Edad	1.8 %	3.4 %	3.3 %
TotalSaldoUltimos90Dias	2.7 %	3.1 %	3.2 %
TotalSaldoUltimos180Dias	5.6 %	2.4 %	3.0 %
log_PromSaldoUltimos180dias	5.0 %	2.4 %	3.0 %
SaldoUltimos30Dias	1.4 %	2.9 %	2.6 %
LongitudAgencia	1.1 %	2.4 %	2.2 %
Latitud	1.4 %	2.4 %	2.2 %
NumeroCuentasCerradas	1.0 %	1.9 %	1.8 %
NivelEstudio.SUPERIOR	1.3 %	1.4 %	1.8 %
EstadoCivil.CASADO	0.9 %	1.4 %	1.4 %
r_Saldo90s180dias	2.2 %	2.2 %	1.4 %
log_TotalSaldoUltimos90dias	0.7 %	1.2 %	1.3 %
EstadoCivil.SOLTERO	0.8 %	1.4 %	1.2 %
Hijos	1.2 %	1.4 %	1.0 %
NumeroCuentasActivas	0.2 %	0.5 %	0.5 %
CodigoProfesion.E30	0.4 %	0.7 %	0.5 %
NivelEstudio.BACHILLER	0.3 %	0.5 %	0.4 %
ScoreBuroConyuge	0.6 %	0.5 %	0,3 %

Fuente: Elaboración Propia

Bibliografía

[1] H. ABDOU, M. TSAFACK, C. NTIM, AND R. BAKER, *Predicting creditworthiness in retail banking with limited scoring data*, SSRN Electronic Journal, (2016).

[2] P. ADDO, D. GUEGAN, AND B. HASSANI, *Credit risk analysis using machine and deep learning models*, Risks, 6 (2018), p. 38.

[3] E. ALPAYDIN, *Introduction to Machine Learning*, Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, 3 ed., 2014.

[4] R. ANDERSON, *The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation*, in AAAI, 2007.

[5] A. BARALDI AND C. ENDERS, *An introduction to modern missing data analyses*, Journal of School Psychology, 4 (2010), pp. 5–37.

[6] T. N. BECKMAN, *Credits and collections in theory and practice.*, 1924.

[7] M. BONILLA, I. OLMEDA, AND R. PUERTAS, *An Application of Hybrid Models in Credit Scoring*, 01 2000, pp. 69–78.

[8] L. BREIMAN, *Bagging predictors*, Machine Learning, 24 (1996a), pp. 123–140.

[9] L. BREIMAN, *Heuristics of instability and stabilization in model selection*, The Annals of Statistics, 24 (1996b), pp. 2350 – 2383.

[10] L. BREIMAN, *Random forests*, Machine Learning, 45 (2001), pp. 5 – 32.

[11] L. BREIMAN, J. H. FRIEDMAN, R. A. OLSHEN, AND C. J. STONE, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA, 1984.

[12] S. DE BANCOS ECUADOR, *Glosario de términos*.

[13] A. DE LARA HARO, *Medición y control de riesgos financieros*, LIMUSA, 2004.

[14] M. DOLFIN, D. KNOPOFF, M. LIMOSANI, AND M. XIBILIA, *Credit risk contagion and systemic risk on networks*, Mathematics, 7 (2019), p. 713.

[15] C. ENDERS, *Applied missing data analysismethodology in social sciences*, New York: Guilford Press., 1 (2010). An optional note.

[16] H. J. ESCALANTE, S. ESCALERA, I. GUYON, X. BARO, Y. GUCLUTURK, U. GUCLU, AND M. VAN GERVEN, *Explainable and Interpretable Models in Computer Vision and Machine Learning*, Springer Publishing Company, Incorporated, 1st ed., 2018.

[17] A. J. FERREIRA AND M. A. T. FIGUEIREDO, *Efficient feature selection filters for high-dimensional data*, Pattern Recogn. Lett., 33 (2012), p. 1794–1804.

[18] D. FIRTH, *Bias reduction of maximum likelihood estimates*, Biometrika, 80 (1993), pp. 27–38.

[19] Y. FREUND AND R. E. SCHAPIRE, *Experiments with a new boosting algorithm*, in Proceedings of the Thirteenth International Conference on International Conference on Machine Learning, ICML'96, San Francisco, CA, USA, 1996, Morgan Kaufmann Publishers Inc., p. 148–156.

[20] J. FRIEDMAN, T. HASTIE, AND R. TIBSHIRANI, *Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors)*, The Annals of Statistics, 28 (2000), pp. 337 – 407.

[21] J. GOBAT, *¿qué es un banco?*, Finanzas & Desarrollo, (2012), pp. 38–39.

[22] J. GRAHAM, *Missing data: Analysis and design*, SpringerThe name of the journal, 4 (2012), pp. 201–213.

[23] R. GUIDOTTI, A. MONREALE, F. TURINI, D. PEDRESCHI, AND F. GIANNOTTI, *A survey of methods for explaining black box models*, CoRR, abs/1802.01933 (2018).

[24] M. GUTIERREZ, *Credit scoring models: what, how, when and for what purposes*, Banco Central de la República de Argentina, (2007).

[25] S. HA AND R. KRISHNAN, *Predicting repayment of the credit card debt*, Computers & OR, 39 (2012), pp. 765–773.

[26] D. HAND AND W. HENLEY, *Statistical classification methods in consumer credit scoring: a review*, Journal of The Royal Statistical Society Series A-statistics in Society, 160 (1997), pp. 523–541.

[27] T. HASTIE, R. TIBSHIRANI, AND J. FRIEDMAN, *The elements of statistical learning: data mining, inference and prediction*, Springer, 2 ed., 2009.

[28] S. HIDALGO, *Random forests para detección de fraude en medios de pago*, Master's thesis, Departamento de Ingeniería Informática. Universidad Autónoma de Madrid, Madrid, Sep 2014.

[29] C. A. IÑIGUEZ SALAS AND M. G. MORALES ARIAS, *Selección de perfiles de clientes mediante regresión logística para muestras desproporcionadas, validación, monitoreo y aplicación en la proyección de provisiones*, Dic 2009.

[30] A. KHANDANI, A. KIM, AND A. LO, *Consumer credit risk models via machine-learning algorithms*, SSRN Electronic Journal, (2010).

[31] G. KING AND L. ZENG, *Logistic regression in rare events data*, Political Analysis, 9 (2001), p. 137–163.

[32] R. C. MERTON, *A functional perspective of financial intermediation*, Financial Management, 24 (1995), pp. 23–41.

[33] C. MOLNAR, *Interpretable Machine Learning*, 2019. `https://christophm.github.io/interpretable-ml-book/`.

[34] C. F. NACIONAL, *Glosario de términos financieros*, 2016.

[35] H. PATRICK AND G. NAVDEEP, *Introduction to Machine Learning Interpretability*, O'Reilly Media, Incorporated, 2018.

[36] A. PETROPOULOS, V. SIAKOULIS, E. STAVROULAKIS, AND A. KLAMARGIAS, *A robust machine learning approach for credit risk analysis of large loan level datasets using deep learning and extreme gradient boosting*, aug 2018.

[37] M. RESTREPO AND D. RESTREPO, *Inestabilidad financiera y regulación: una reseña a partir de la crisis financiera de 2008*, Perfil de Coyuntura Económica, (2009), pp. 33–51. Universidad de Antioquia.

[38] M. T. RIBEIRO, S. SINGH, AND C. GUESTRIN, *"why should I trust you?": Explaining the predictions of any classifier*, CoRR, abs/1602.04938 (2016).

[39] M. T. RIBEIRO, S. SINGH, AND C. GUESTRIN, *Anchors: High-precision model-agnostic explanations*, in AAAI, 2018.

[40] B. D. RIPLEY, *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.

[41] D. RODRIGUEZ, J. BECERRA AREVALO, AND D. CARDONA, *Modelos y metodologías de credit score para personas naturales: una revisión literaria*, Revista CEA, 3 (2017), pp. 13–28.

[42] T. RONCALLI, *Handbook of Financial Risk Management*, 04 2020.

[43] D. ROTHMAN, *Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps*, Packt Publishing, 2020.

[44] D. SCULLEY, G. HOLT, D. GOLOVIN, E. DAVYDOV, T. PHILLIPS, D. EBNER, V. CHAUDHARY, M. YOUNG, J.-F. CRESPO, AND D. DENNISON, *Hidden technical debt in machine learning systems*, in Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS'15, Cambridge, MA, USA, 2015, MIT Press, p. 2503–2511.

[45] V. SRINIVASAN AND Y. H. KIM, *Credit granting: A comparative analysis of classification procedures*, The Journal of Finance, 42 (1987), pp. 665–681.

[46] SUPERINTENDENCIA DE BANCOS, *Normas de control para las entidades de los sectores financieros público y privado.*, Quito-Ecuador. Libro I, Título IX, Capítulo II.

[47] C. S. TAPIERO, *Risk and financial management: mathematical and computational methods*, Wiley, 1 ed., 2004.

[48] L. THOMAS, J. CROOK, AND D. EDELMAN, *Credit Scoring and Its Applications, Second Edition*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.

[49] A. TORRES, *Curvas roc para datos de supervivencia*, in AAAI, 2010.

[50] B. W. YAP, S. H. ONG, AND N. H. M. HUSAIN, *Using data mining to improve assessment of credit worthiness via credit scoring models*, Expert Syst. Appl., 38 (2011), p. 13274–13283.

[51] C. YÉPEZ, *EvaluaciÓn del riesgo crediticio en una cartera de consumo bajo dos sistemas de clasificaciÓn: ProcÍclico y acÍclico*, Dic 2019.

[52] C. ZOTT AND R. AMIT, *Business model design and the performance of entrepreneurial firms*, Organization Science, 18 (2021).