ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ESTUDIO DEL HANDOVER EN REDES LTE MEDIANTE MEDICIONES DE CAMPO

ANÁLISIS DEL COMPORTAMIENTO DEL HANDOVER EN REDES CELULARES LTE MEDIANTE MODELOS DE CLASIFICACIÓN POR ÁRBOLES DE DECISIÓN Y KNN

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

JONATHAN RAMIRO VILLARREAL TAYÁN
(jonathan.villarreal@epn.edu.ec)

DIRECTOR: PhD. PABLO ANIBAL LUPERA MORILLO
(pablo.lupera@epn.edu.ec)

DMQ, Febrero 2023

CERTIFICACIONES

Yo, JONATHAN RAMIRO VILLARREAL TAYÁN declaro que el trabajo de integración curricular aquí descrito 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.

_____

JONATHAN RAMIRO VILLARREAL TAYÁN

Certifico que el presente trabajo de integración curricular fue desarrollado por JONATHAN RAMIRO VILLARREAL TAYÁN, bajo mi supervisión.

_____

PABLO ANÍBAL LUPERA MORILLO

DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

JONATHAN RAMIRO VILLARREAL TAYÁN

PABLO ANÍBAL LUPERA MORILLO

DEDICATORIA

A Dios, a mis padres Teresa y Ramiro, y a mis hermanos Alex y Andrés, quienes me han acompañado y apoyado incondicionalmente en mi diario caminar, y me han dado fortaleza para seguir adelante.

Su amor, paciencia, consejo, ejemplos y sacrificios seguirán siendo el aliento para superarme como persona, por lo que deseo honrarlos con mis éxitos y deseando que Dios siga bendiciendo nuestras vidas como lo ha hecho hasta el día de hoy.

AGRADECIMIENTO

Mi principal agradecimiento es a Dios por bendecirme con mi familia, por darme la salud y vida cada día.

Agradezco a mi madre Teresa por brindarme su amor y fortaleza en los momentos difíciles y darme sentido de responsabilidad en mi etapa académica y personal. A mi padre Ramiro por enseñarme el valor del trabajo y superación. A mi hermano Alex por ser compañero de niñez y juventud. A mi hermanito menor Andrés gracias por hacerme reír y animarme.

Agradezco a los amigos que conocí en mi vida universitaria, porque fueron más que compañeros en los momentos que necesitaba, a mis amigos Alex Ramos, Jonathan Álvarez, Antonio Peña y Alex Páez, quienes son muy buenos amigos y espero no perder su valiosa amistad y deseándoles grandes éxitos en su vida.

Agradezco de igual manera mis compañeros y amigos de tesis Jhon Cando y Patricio Aguagallo, quienes nos apoyamos en la realización y culminación del trabajo de titulación con éxito.

Finalmente, pero no menos importante al Dr. Pablo Lupera, mis sinceros agradecimientos por la acogida para ser parte del trabajo de titulación con su persona y ser un buen tutor siempre apoyándonos y enseñándonos, tanto en este trabajo como en sus asignaturas impartidas.

ÍNDICE DE CONTENIDO

RESUMEN

El presente trabajo realiza la detección y estudio del proceso de handover, identificando los parámetros que afectan la calidad del servicio al realizar una llamada de VoIP en una red LTE.

En el primer capítulo se aborda la elección de herramientas móviles con capacidad de recolectar parámetros de RF como potencia de transmisión y recepción, identificando las estaciones baque ofrecen el servicio de telefonía móvil en una zona determinada en tiempo real y localización GPS. Además, de ser compatibles con cualquier dispositivo móvil conocido como UE (User Equipment) de interfaz amigable con el usuario en la visualización de datos y mapas de cobertura de la celda servidora.

En el segundo capítulo comprende el procesamiento de los datos, realizando una depuración, eliminando los errores en la captura de datos por parte de las aplicaciones. Además, de adicionar parámetros dependientes de los obtenidos como el cálculo de velocidad, ángulo de azimut y distancia a la que se encuentra el usuario con respecto a la estación base, además de integrar la variable a predecir que es el handover mediante el cambio de estación base. Estos cálculos adicionales proveen confiabilidad y variedad en la aplicación de un modelo en general. Por otra parte, se menciona el software a utilizar en el procesamiento de los datos con lenguaje de programación R. De igual forma, se da a conocer en que consiste el machine learning y del tipo de casos que puede existir también de las técnicas que estos conllevan, finalmente se describen las líneas de comando a utilizar para el análisis de los datos y sus respectivas salidas.

Para el tercer capítulo se presentan los resultados obtenidos en el estudio a partir de los modelos de clasificación por árbol de decisión y vecino más cercano KNN.

PALABRAS CLAVE: VoIP, LTE, handover, UE, estación base, lenguaje R, Árbol de decisión, KNN.

ABSTRACT

This work detects and studies the handover process, identifying the parameters that affect the quality of service when making a VoIP call in an LTE network.

The first chapter addresses the choice of mobile tools with the ability to collect RF parameters such as transmission and reception power, identifying the stations that offer mobile telephone service in a given area in real time and GPS location. In addition, they are compatible with any mobile device known as UE (User Equipment) with a user-friendly interface for viewing data and coverage maps of the serving cell.

The second chapter it includes the data processing, performing a debugging, eliminating errors in data capture. In addition, to add dependent parameters such as the calculation of speed, azimuth angle, distance at which the user is with respect to the base station and the variable to be predicted, which is the handover, was integrated. On the other hand, the software to be used in the analysis is mentioned, the R programming language. In the same way, it describes what machine learning consists of and the type of techniques that this entail, finally the command lines to be used are described. for data analysis and their respective outputs.

For the third chapter, the results obtained in the study from the classification models by decision tree and nearest neighbor KNN are presented.

KEYWORDS: VoIP, LTE, handover, UE, base station, R language, Decision Tree, KNN.

1 INTRODUCCIÓN

Las redes móviles van evolucionando conforme las nuevas necesidades de comunicación de los usuarios, por lo que su capacidad y forma de otorgar el servicio va evolucionando constantemente con la tecnología.

En el Ecuador la red de telefonía celular 4G LTE es la más empleada por parte de los operadores de servicio en el país, los cuales han concentrado esta tecnología mayormente en lugares de concentración masiva de personas como en las zonas urbanas y lugares con disponibilidad de recursos para su conexión. Esta tecnología permite una conexión tanto de telefonía como de datos móviles para servicio de internet gracias a la gran acogida y necesidad del ser humano por utilizar un dispositivo móvil.

La conexión del terminal móvil con la estación de servicios se ha visto afecta por factores físicos que degradan el traspaso de la señal de una estación base a otra. Es por esta razón, el análisis de la relación de estos parámetros con el establecimiento de una llamada por VoIP (Voz sobre IP) se ve afectada por parámetros como la velocidad del móvil, la distancia a la que el usuario se encuentra de la celda servidora y del cómo la potencia transmitida por el proveedor de telefonía muchas de las veces podrían determinar o no la calidad del servicio. Con esta premisa el análisis se centra en el traspaso de la señal o también denominado *handover,* que es el cambio de radiocanal, el cual tiene como finalidad la de solucionar el problema del mantenimiento de la comunicación de una estación móvil con otra. Esto acontece cuando un dispositivo se encuentra en movimiento y pierde la señal dentro de su celda o pasa de una celda a otra.

Las razones que pueden darse a la pérdida de la señal, pueden ser porque se produzca un desvanecimiento de la señal, donde la potencia baje de sus parámetros estipulados; por otra parte, se produce cuando la capacidad de tráfico de la celda se encuentre próxima a su límite.

Para la toma de decisión al realizar un handover siempre es producida por la BSC (Base Station Control) que ocupa el dispositivo en ese momento, excepto si el traspaso se efectúa por motivos de tráfico, donde, en este caso la que decide el traspaso es la MSC (Mobile Switching Center).

Para estudiar el traspaso de una celda a otra, se realizarán mediciones de campo en una red de telefonía celular durante la ejecución de una llamada de VoIP. Los datos recolectados se analizarán mediante técnicas de Machine Learning (Aprendizaje Automático).

Como preámbulo, se puede mencionar que el Machine Learning (ML) es la práctica de programación de computadoras para aprender a partir de los datos, mediante la recolección de información y la aplicación de algoritmos que procesan los datos y son capaces de clasificar o predecir comportamientos futuros.

Para este estudio se recolectarán datos de forma manual mediante una herramienta de mediciones de parámetros técnicos de las redes de telefonía celular en un determinado sector de la ciudad de Quito, durante la recolección de datos se ejecutará una conexión activa y permanente con la red y se detectarán las zonas de cobertura en las cuales existen cortes o alteraciones en la conexión.

Los datos obtenidos serán analizados mediante las técnicas de Machine Learning para observar el comportamiento de los datos. Una de las técnicas consideradas será la de los árboles de decisión y al menos se considerará una técnica adicional para compararlas.

1.1 OBJETIVO GENERAL

Aplicar al menos una técnica de machine learning para clasificar las zonas que son propensas o no a la ejecución de handover deficientes en base a mediciones de campo ejecutadas en la ciudad de Quito.

1.2 OBJETIVOS ESPECÍFICOS

1. Realizar mediciones de los parámetros de radiofrecuencia en el modo conectado en una red conocida de comunicación móvil LTE de la ciudad de Quito.
2. Seleccionar los datos que se considerarán en los análisis.
3. Aplicar al menos una técnica de machine learning para clasificar los datos de las zonas propensas o no a handover deficientes.
4. Evaluar el rendimiento del algoritmo aplicado.

1.3 ALCANCE

Este estudio es una ampliación de lo ejecutado en [1] y [2], en donde las mediciones se realizarán cuando se encuentra activa una conexión permanente con la red. En este nuevo trabajo se aplicará al menos una técnica adicional de Machine Learning con la finalidad de generar un modelo predictivo que permita establecer cuáles son las condiciones y características de las zonas de cobertura que presentarían mayor probabilidad de que se produzcan handover fallidos. Para simplificar el estudio, se

supondrá que cualquier tipo de falla o interrupción de la conexión se deberá a algún factor presente en el interfaz de radio. Los resultados de este estudio permitirán contar con un modelo predictivo, en el cual al ingresar los datos de mediciones realizadas en un sector determinado de la ciudad de Quito se podrá predecir si en esa zona existe la posibilidad de que se produzcan handovers fallidos, lo cual ayudará a aplicar medidas correctivas en caso de ser necesario. Los resultados obtenidos de este estudio se compararán con los obtenidos en los trabajos precedentes para establecer las diferencias alcanzadas.

1.4 MARCO TEÓRICO

1.4.1 Telefonía móvil 4G.LTE

El término *4G* es empleado al referirse a la familia IMT-Advanced (*International Mobile Telecommunications-Advanced),* la cual se compone de una serie de tecnologías móviles inalámbricas; definidas y ratificadas por la UIT (Unión Internacional de Telecomunicaciones)[3].

La primera especificación de LTE se observa en la Release 8 (Rel-8) del 3GPP, en el año de 2008. De modo posterior, se introdujeron las normas para LTE adscritas en el Release 9 (Rel-9), centrándose en las mejoras de HSPA+ y LTE, terminada en diciembre de 2009. En la Release 10, se abordan especificaciones para LTE-Advanced, centrándose en la siguiente generación de LTE (Beyond LTE). La Release 11 (Rel-11) fue la última en desarrollarse y siendo concluida a finales del 2012, e iniciándose la Release 12 (Rel-12), congelándose a mediados del 2014[3].

Los aspectos tecnológicos a mejorar por parte de las versiones anteriores van siendo mejoradas en las versiones 12 y 13, abarcando aspectos de capacidad de red y calidad del servicio.

1.4.2 Definición y Características del Handover

En las telecomunicaciones, el proceso del handover o *"traspaso"*, es muy importante, dado que, implica mover la asignación de recursos para un teléfono móvil o una pieza del UE[1] de una estación base a otra. Este proceso es utilizado para proveer calidad de servicio (QoS) hacia los usuarios, permitiendo que estos continúen utilizando el servicio celular al moverse del rango original de la estación base servidora[4].

Es muy importante que el handover se realice rápidamente, y causen poca o ninguna interrupción en la experiencia del usuario y se completan con una tasa de éxito muy alta. Si un handover es fallido, la llamada en curso se cortará debido a que no hay suficientes

[1] UE (User Equipment): dispositivo final utilizado por el usuario para comunicarse.

recursos disponibles en una estación base o de la potencia RSS (Received Signal Strength) la cual, decae por debajo un cierto umbral necesario para mantener la llamada, este umbral en LTE se conoce como ruido o y tiene un valor de -97,5 dB, e indica que el traspaso tarda aproximadamente 0,25 segundos en completarse después de que se haya tomado la decisión de que se produzca un traspaso[4] [5], como se visualiza en la figura 1.1:

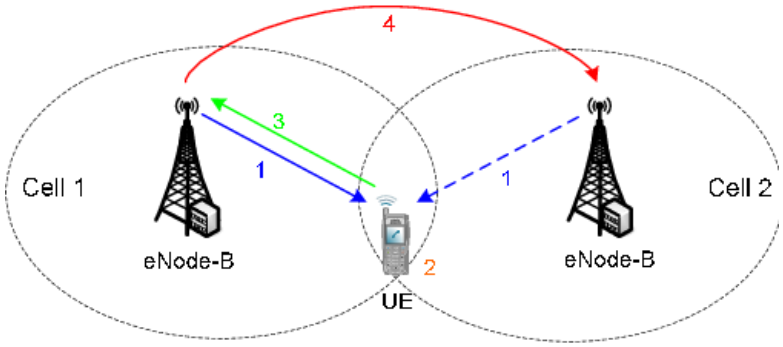

Figura 1.1 Proceso de Handover en redes LTE[5]

1.4.3 Herramientas utilizadas para la recolección de datos de redes LTE.

Las aplicaciones utilizadas para la recopilación de datos son las siguientes:

- CellMapper
- G-Net Track Lite
- Net Monitor Lite.

1.4.3.1 CellMapper

Es una aplicación que permite la localización de las estaciones base (BS) de las redes 2G/3G/4G/4G+, además, de su localización nos proporciona datos sobre la intensidad de la señal de la red otorgados a los usuarios finales. En la actualidad, es compatible para teléfonos celulares con sistema operativo Android 7.0 en adelante y con dispositivos con Windows 10 Mobile, con asistencia para su descarga desde Google Play.

Cellmapper crea una representación gráfica geográfica de los datos mediante un mapa entregado por Google Maps alcanzando una zona de cobertura de hasta cien kilómetros, por lo cual es aconsejable realizar la medición en un área lo más posible despejada, sin obstáculos, para conseguir datos mayormente confiables[6].

Esta aplicación además admite la subida de datos en tiempo real, ya que los datos recopilados se envían hacia los servidores de CellMapper conforme se realicen las mediciones.

Consideraciones de uso.

1. Instalar la aplicación desde Google Play para sistemas operativos Android 7.0 y dispositivos con Windows 10 Mobile como mínimo requeridos.
2. Habilitar el GPS.
3. Iniciar la aplicación y asegurarse de estar conectado a una estación base para telefonía celular sean 2G/3G/4G/4G+ que se desea medir, donde no se requiere de un plan de datos móviles, pero por la subida de datos en tiempo real se necesita acceso a la red o en algunos dispositivos permite que los datos se guarden en la tarjeta SIM o memoria interna del mismo.

Características:

Al abrir la aplicación en la pestaña de celda se detalla el número de sector de la celda, el MCC(Mobile Country Code), el cual es un código único de cada país, el MNC(Mobile Network Code), el cual es un código de red de cada operadora, identificador de estación base global LTE (CGI), la calidad de la señal recibida (RSRQ), la relación señal/ruido de la red LTE (SINR), donde, la escala se encuentra entre -20 y +30 dB, cabe recalcar que las estimaciones de velocidad se realizan con un ancho de banda de 20 MHz, la potencia recibida de la señal de referencia LTE (RSRP), además del identificador de intensidad de la señal recibida LTE (RSSI), también el código de área de seguimiento que consta de la variedad de estaciones base (TAC), y por último el código de área local (LAC) que da constancia de varias estaciones base y del número absoluto de canal RF (EA/UA/A RFCN)[6], como se observa a continuación en la figura 1.2:

Figura 1.2 Detalles de medición en la ventana de "Celda" en CellMapper

Por otra parte, se tiene a la pestaña de Mapa se visualizan las estaciones base cercanas al usuario final, así como al seleccionar una estación base LTE, se puede observar las características de dicha antena, como se muestra en la figura 1.3:

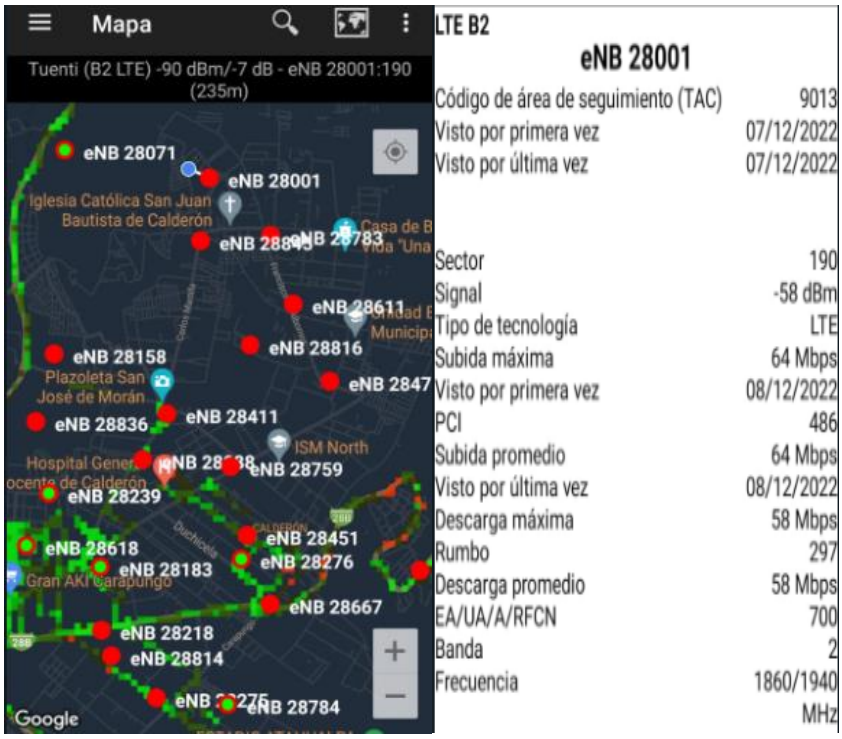

Figura 1.3 Ilustración geográfica de las estaciones base LTE y caracteristicas en CellMapper

De igual forma, la aplicación permite utilizar filtros por bandas, como la subida de datos y estadísticas a una cuenta creada previamente o que se guarda por defecto en la memoria interna del dispositivo en un archivo identificado por la fecha de toma de datos en formato .csv.

1.4.3.2 Net Monitor Lite

Esta aplicación permite obtener recopilar información sobre la red celular de 2G/3G/4G/4G+ y hasta 5G en las últimas actualizaciones es de uso gratuito, de código abierto y compatible para dispositivos con sistema operativo Android [7], mostrando el estado de la red. En la mayoría de los casos la estimación de los datos para una torre de telefonía celular se realiza con mejor precisión para sitios con tres celdas detectadas. Permite la compatibilidad con múltiples SIM, además de la localización geográfica para conocer la ubicación de las celdas como del usuario final.

Características.

Al iniciar la aplicación se observa la supervisión de la red en tiempo real de 2G/3G/4G/4G+ y hasta 5G, además añade información de la celda actual y de la celda vecina con sus respectivos parámetros (MCC,MNC, LAC/TAC, CID/CI, RNC, PSC/PCI, canales, ancho de banda, frecuencias y bandas). Para visualizar los cambios de la señal se utilizan unidades en dBm, como se muestra en la siguiente figura:

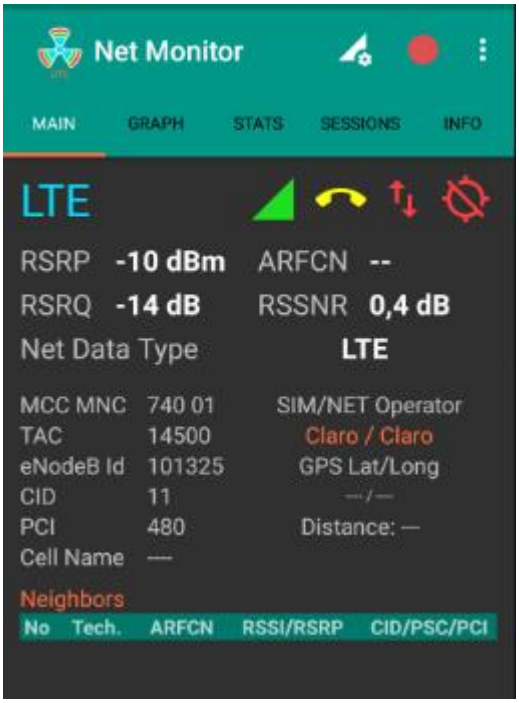

Figura 1.4 Información de cobertura de la estación base NetMonitor

Por otra parte, Netmonitor en su pestaña de graph, nos permite visualizar la ubicación de las celdas actuales como de las celdas vecinas con dirección basada en los servicios de geolocalización[7], indicando información como el RSRP, el RSRQ, el RSSNR, el TAC, el eNode y la distancia del dispositivo hacia la antena conectada en metros con soporte en Google Maps/OSM además nos muestra mediante gráficos estadísticos el servicio de la celda, la señal de vecino cercano, el throughput[2] de datos, y agrupación de los sectores de torres celulares en un mapa, observadas en la figura 1.5:

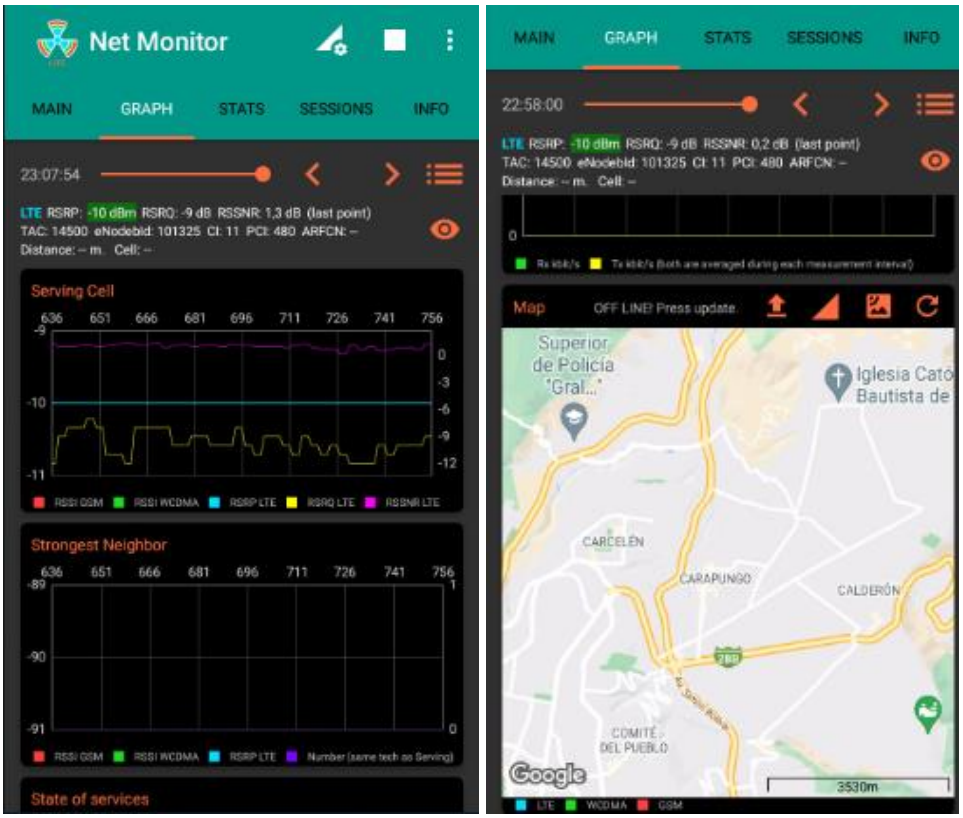

Figura 1.5 Ventana gráficos estadísticos y de ubicación del dispositivo en Netmonitor

Por último, el registro de los datos se realiza mediante un archivo csv y KML, exportado a la memoria interna del dispositivo usuario. Los archivos en formato KML[3] pueden ser abiertos en la aplicación Google Earth, visualizándose la velocidad del dispositivo y la ubicación exacta del mismo mediante una geolocalización. Net Monitor Lite para guardar sus datos recurre a la utilización de sesiones las cuales se mantienen almacenadas en la aplicación por defecto, mostradas en la figura 1.6:

[2] Throughput: velocidad de transporte de los datos es una red.
[3] Archivos KML(Keyhole Markup Language): es un formato de archivo que permite una visualización de información de forma geográfica.

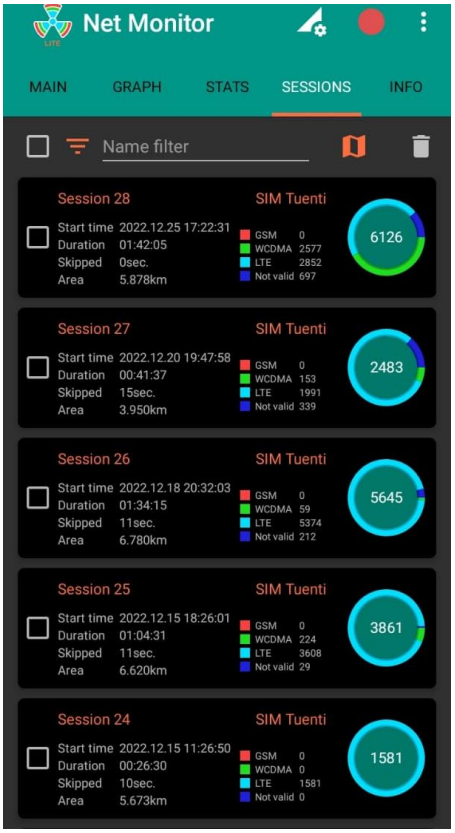

Figura 1.6 Ventana sesiones en Netmonitor

1.4.3.3 G-NetTrack Lite

Es una aplicación de prueba de manejo y monitoreo de red tanto para 2G como para 3G, 4G, 4G+ y hasta 5G en las últimas versiones. Permite el escaneo y monitoreo del servicio de telefonía móvil, además añade información de las celdas actuales y celdas vecinas, se encuentra en dos versiones una gratuita la cual es G-NetTrack Lite y otra versión con pago por su uso que es G-NetTrack Pro, con recomendación para su utilización en dispositivos con sistema operativo Android 7.0 y posteriores.

Características:

En la ventana de célula o celda, G-NetTrack Lite nos informa acerca de las propiedades de la estación base como nombre del operador móvil actual, el código de país o región móvil (MCC), el código de red móvil (MNC), el tipo de red actual (GPRS/EDGE/UMTS/HSPA/HSDPA/HSUPA/LTE), el identificador de celda (CID), el indicador de calidad del canal (CQI), el cual se mide solo en 4G, el indicación de intensidad de la señal recibida (RSSI), la potencia recibida de señales de referencia (RSRP) y la potencia recibida de la señal de referencia (RSSI) para redes LTE [8].

Además de la ubicación geográfica con sus parámetros de longitud, latitud, velocidad del dispositivo móvil, la dirección de movimiento del GPS, alto, altitud y nivel del suelo como también, la conexión Wifi del dispositivo con sus niveles de carga y descarga de la información, observada en la siguiente figura:

Figura 1.7 Información de la estación base analizada mediante G-NetTrack Lite

La ventana de vecinas, nos da a conocer la celda contigua a la celda actual del dispositivo que puede conectarse el dispositivo al realizar un traspaso de señal, visualizada en un gráfico estadístico, de la siguiente manera, en la figura 1.8:

Figura 1.8 Celdas vecinas en G-NetTrack Lite

A continuación, se tiene la ventana de mapa en la cual se observa la geolocalización del dispositivo, con los parámetros de longitud, latitud, velocidad del dispositivo expresada en km/h, la ARFCN (Absolute Radio Frequency Channel Number) o números de canales de radio frecuencia absolutos, la potencia recibida (RSSI), la potencia promedio sobre los símbolos (RSRP) la relación de potencia entre la RSRP y la RSSI o conocida como calidad de la señal de referencia recibida, así también del valor de SNR en unidades de dB y TA el cual se trata del avance de tiempo 4G, observada a continuación en la siguiente figura 1.9:

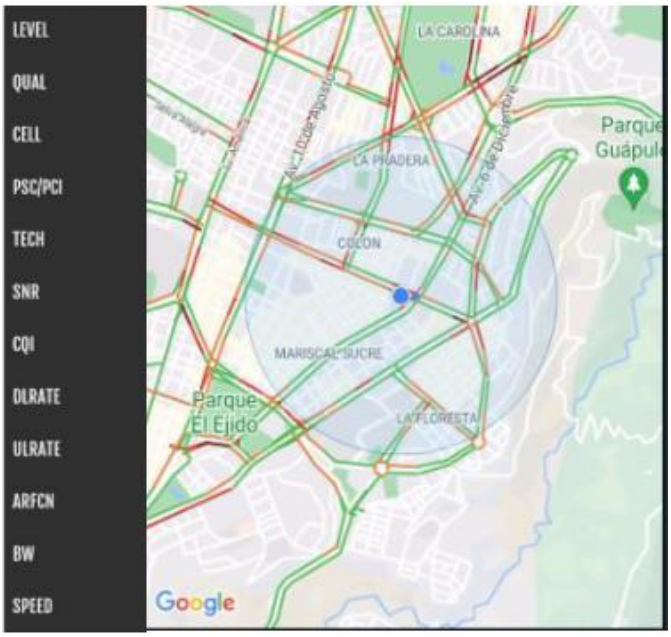

Figura 1.9 Geolocalización del dispositivo móvil en G-NetTrack Lite

Esta aplicación permite un registro del monitoreo en archivos en una carpeta llamada G-NetTracl_Logs folder ubicada en la memoria interna del dispositvo usuario con extensión XML/KML.

1.4.4 Descripción del Terminal Móvil

El terminal móvil utilizado para la recolección de datos es un dispositivo móvil Samsung Galaxy A01, de la versión SM-A015M/DS. Para lo cual se resumen sus especificaciones técnicas en la siguiente tabla:

Tabla 1.1 Resumen Especificaciones Técnicas del Terminal Móvil [9], [10].

ESPECIFICACIONES	
Características físicas	**Dimensiones:** 146.2 x 70.9 x 8.3 mm **Peso:** 149 g (5.26 oz) **Construcción:** Frente de vidrio, respaldo de plástico, marco de plástico **SIM**: Single SIM (Nano-SIM) o Dual SIM (Nano-SIM)
Modelo	**Nombre del modelo:** Galaxy A01 **Número de modelo:** SM-A015M/DS
Plataforma	**Sistema Operativo:** Android 10, actualizable a Android 11 **Memoria interna:** 16GB 2GB RAM **Card slot:** microSDXC

	Chipset: Qualcomm SDM439 Snapdragon 439 (12 nm) **CPU:** Octa-core (4x1.95 GHz Cortex-A53 & 4x1.45 GHz Cortex A53) **GPU:** Adreno 505	
Red	**Tecnología:** GSM / HSPA / LTE **Máximo carga/descarga:** 150/75 Mbps	
	GSM MHz band	Quad-Band 850/900/1800/1900
	3G-UMTS	UMTS 850/1700/1900
	4G-LTE	LTE Cat4 (2, 4, 5, 12, 14)
	5G	No soporta
	Red de datos primaria	GPRS, EDGE, UMTS, HSDPA, HSUPA, HSPA+, LTE
Comunicaciones	**Red de área local inalámbrica:** Wi-Fi 802.11 b/g/n, Wi-Fi Direct **Bluetooth:**4.2, A2DP, LE **Posicionamiento**: GPS, GLONASS, GALILEO, BDS **NFC:** No **Radio**: FM radio, RDS, recording **USB**: microUSB 2.0	

1.4.5 Herramientas de Análisis de Datos

1.4.5.1 R

Es un lenguaje computacional intuitivo y de código abierto que permite el análisis y exploración de datos, utilizado mayormente para un modelado estadístico, debido a la gran cantidad de paquetes de funciones con operaciones para estadística básica y avanzada. La ventana principal de R se muestra en la figura 1.10:

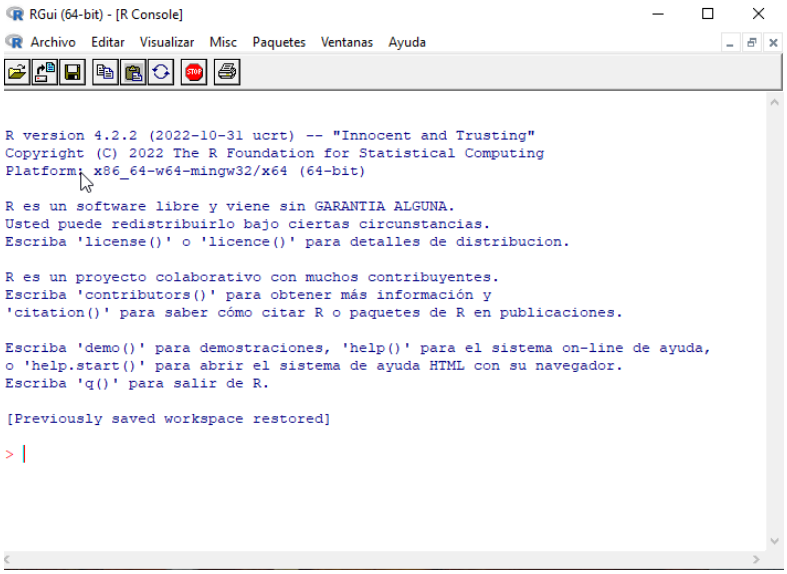

Figura 1.10 Ventana de inicio de R

La utilización del lenguaje R proporciona varias ventajas, entre ellas, que funciona en distintos sistemas operativos, como Windows, Linux y Mac. R cuenta con la capacidad de gestionar grandes cantidades de información y disminuye el tiempo de cálculo en la manipulación de datos y generación de gráficas. Por otra parte, permite la utilización de más de 15303 paquetes desarrollados y disponibles en el repositorio *CRAN (Comprehesive R Archive Network)*, que cubren varios campos como financieros, creación de gráficas, aplicación de técnicas de Machine Learning, entre otras [11].

1.4.5.2 RStudio

Es el entorno gráfico de trabajo para el lenguaje R. Permite la creación y ejecución del código en R para explorar y analizar datos. Amplía las operaciones básicas en R y agiliza muchas operaciones en el entorno de trabajo, puesto que, permite cargar secuencias de comandos de código[11], cargar conjuntos de datos o guardar el archivo de trabajo en un formato que se puede compartir. La ventana principal de RStudio se muestra en la siguiente figura.

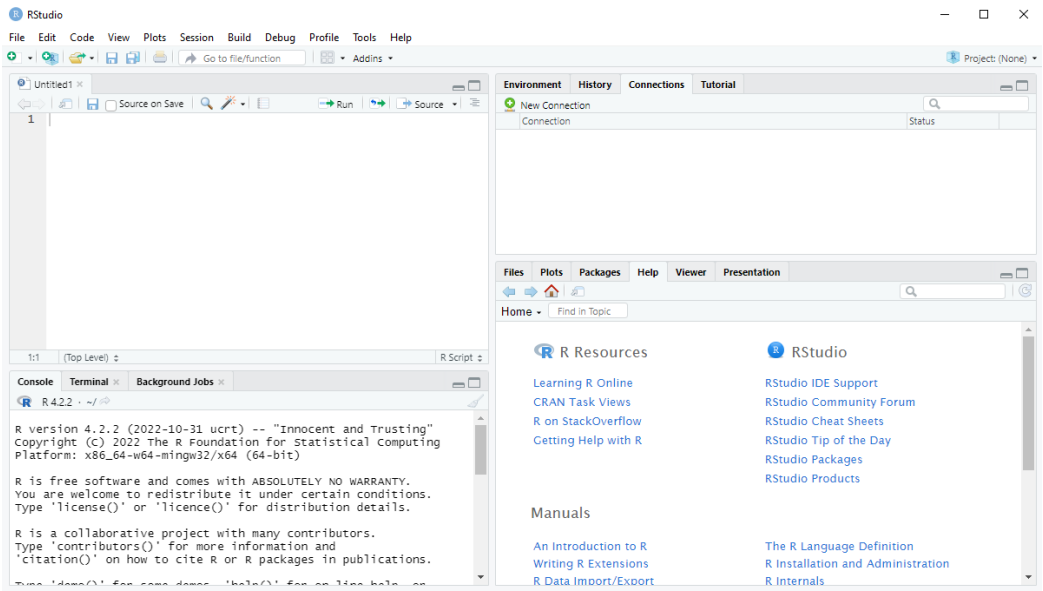

Figura 1.11 Ventana de inicio de RStudio

El IDE (Integrated Development Environment), o entorno de desarrollo integrado, cuenta de cuatro secciones de trabajo en las cuales se puede ejecutar lo siguiente [11]:

- Observar el área de trabajo
- Ejecución de partes de código
- Visualizar varios scripts simultáneamente
- Observar el historial y objetos del área de trabajo
- Gestionar librerías, gráficos, explorar archivos y acceder a su ayuda

1.4.6 Machine Learning

Es una disciplina de la inteligencia artificial (IA) que brinda a las máquinas la capacidad de aprender de modo automático a partir de información de entrada. Se utiliza principalmente para identificar patrones y realizar predicciones con la mínima intervención de una persona.

1.4.6.1 Técnicas de Machine Learning

Existen diferentes técnicas de Machine Learning, las cuales son [12]:

- **Supervisado**

  Los datos que son utilizados son referidos como etiquetas, y pueden predecir un comportamiento o identificar una determinada clase[8]. Entre las técnicas más importantes se encuentran[8]:

  - K-nearest neighbors (KNN, los o el vecino más cercano K)
  - Árboles de decisión

    - Red neuronal
    - Regresión logística
    - Máquinas de soporte de vectores
- **No supervisado**

  En este tipo de técnicas de Machine Learning los datos utilizados no tienen una etiqueta, por lo cual se usa un análisis de agrupamiento jerárquico para combinar varias características relacionadas a un determinado grupo [12].
- **Semi-supervisado**

  Utiliza datos con ambas técnicas con datos etiquetados y no etiquetados[4].
- **Machine Learning de Refuerzo**

  Es otra técnica, donde interactúa un agente con un sistema AI (Inteligencia Artificial), el cual observa el entorno y ejecuta varias acciones de las cuales las acciones bien ejecutadas son recompensadas y las erróneas son penalizadas [12].

2 METOLODOGÍA

2.1 ETAPA DE OBTENCIÓN DE DATOS

Esta etapa tiene como finalidad la identificación de las zonas geográficas, en las cuales se localizan las estaciones base servidoras de una operadora de telefonía celular en general, que ofrecen cobertura en un sector norte de la ciudad. Para este logro se plantearon los siguientes objetivos:

- Identificar las zonas geográficas y las estaciones base de la una operadora celular.
- Dividir las zonas geográficas por rutas para un análisis menos complejo.
- Realizar las mediciones mediante un servicio de VoIP por medio de una aplicación con datos móviles.
- Identificar los parámetros de red celular que actúan en el proceso de handover y los factores que influyen en la funcionalidad del servicio.

2.1.1 Descripción de los parámetros

Los parámetros que actúan en la comunicación de la estación base y el terminal móvil se mantienen inconstantes, dado que, intervienen diversos factores físicos que degradan el servicio de la telefonía celular. Estos parámetros son causa de análisis, los cuales pueden determinar un resultado exitoso o fallido en el proceso de handover. Los parámetros que contribuyen a dicho análisis se detallan a continuación:

- **RSRP (Reference Signal Received Power):** este parámetro es una métrica de intensidad de la señal específica de la celda. Es utilizada para clasificar diferentes celdas LTE-Advanced según su intensidad. También es usado para la decisión de la celdas en el proceso de handover. Se encuentra definido por la potencia promedio linear (en watts) de la celda especifica como portadora de la señales de referencia (RS), considerada dentro del ancho de banda de operación[4]. Se necesita un mínimo de -20 dB SINR para detectar la RSRP[13].
- **RSRQ(Reference Signal Received Quality):** de forma similar al RSRP, tambien ayuda a seleccionar celdas candidatas de LTE-Advanced acorde a la calidad de la señal en esenarios en los cuales el RSRP no proporciona suficiente información. Se encuentra definida como:

$$RSRQ = \frac{N*RSRP}{RSSI} \tag{2.1}$$

Donde, N es el bloque de recursos de la señal LTE-Advanced. mientras que RSRP es un indicador de la intensidad de la señal deseada, RSRQ tiene en cuenta el nivel de interferencias debido a la inclusión de RSSI, permitiendo combinar efectos de intensidad de señal con la interferencia que afecta la eficiencia de la señal[4].

- **SNR (Signal Noise Ratio):** es la relación que existe entre la potencia de señal deseado con la potencia de ruido, así:

$$SNR = \frac{P_{signal}}{P_{noise}} \quad (2.2)$$

- **RSSI(Received Signal Strength Indicator:** en LTE la señal RSSI se define como la potencia total promedio observada por el UE de todas las fuentes, incluidas las celdas cocanal que sirven y las que no están activas, la interferencia del canal adyacente y el ruido térmico. Sin embargo, la decisión de traspaso también se puede basar en los parámetros del enlace uplink[4]. Sin embargo, -70 dBm y valores más altos generalmente equivalen a un área de cobertura excelente. Cuanto más cerca de 0 dBm, más fuerte es la señal[14].

- **SNIR**: es la relación señal-interferencia a ruido, la cual, es una cantidad que se utiliza para conocer la capacidad del canal en los sistemas de comunicación inalámbrica[15].

- **PLMN:** que determina el conjunto de servicios específico para un país en este caso para Ecuador es el 740 y el identificador de la red del operador.

- **RRC_state**: que define los estados específicos del dispositivo, en los que puede estar presente el UE (User Equipment).

- **Cell_ID:** corresponde al identificador físico de la celda de un eNB o estación base[16].

- **Latitud:** indica la distancia con respecto al eje paralelo en grados, minutos y segundos de la posición de un objeto con respecto al Ecuador.

- **Longitud:** indica la distancia con respecto al eje paralelo con respecto al meridiano de Greenwitch

2.2 DETERMINACIÓN DE LAS ZONAS Y RUTAS

Para este estudio, se escogió una zona noreste que corresponde a la parroquia rural de Calderón, la cual contiene una alta densidad poblacional y siendo una de las parroquias más grandes de la ciudad con una extensión superficial de 79 km^2 y con una densidad poblacional de aproximadamente 300.000 habitantes[17].

Figura 2.1 Ubicación geográfica Parroquia Calderón[18]

2.2.1 Trayectoria de las rutas de estudio

Debido a la extensión de la zona el estudio se escogieron tres rutas con la mayoría de estaciones base que prestan servicio de telefonía celular, las cuales por su crecimiento poblacional y automotriz la calidad de la señal ha disminuido y logra ser un escenario de errores en la conexión. A continuación, se describen las rutas escogidas:

- **Ruta 1:**

  La ruta primera se localiza en el sector de Carapungo, cuyo recorrido se desarrolló en las calles principales de dicho sector que son: Av. Padre Luis Vaccari, calle Galo Plaza Lasso y calle Jaime Roldós Aguilera, con un recorrido de aproximadamente de 3 km. Cabe mencionar que es un sector muy concurrido debido a la cantidad de habitantes y congestión vehicular. Por tanto, es una buena opción para el análisis de las estaciones base que prestan servicio a esta región, visualizándose de manera geográfica en la siguiente figura:

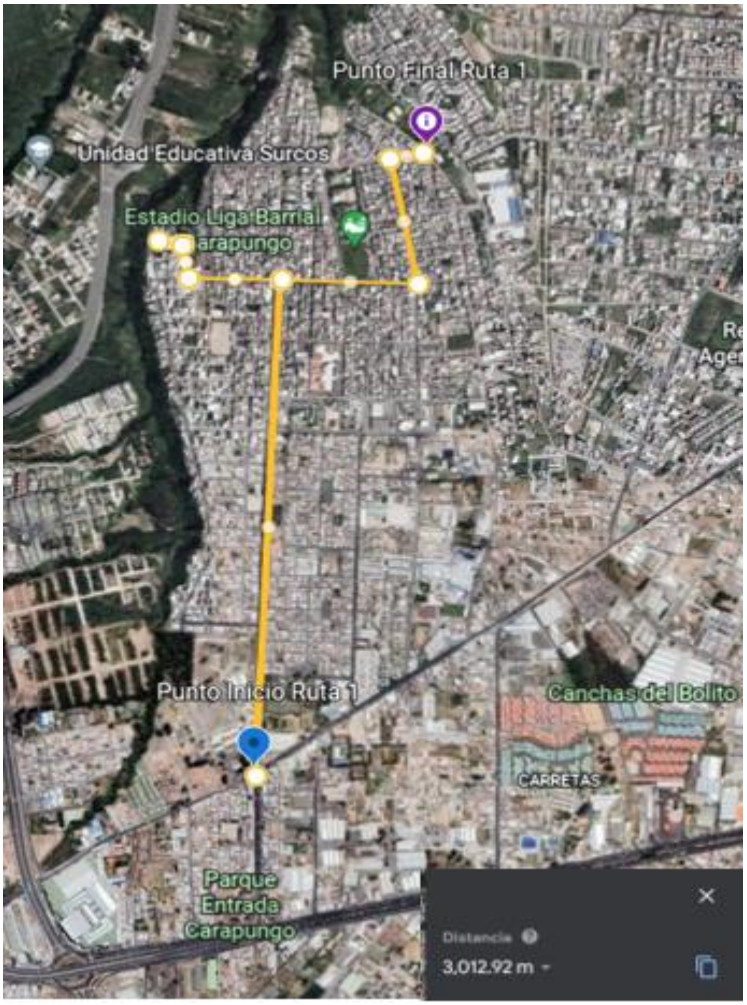

Figura 2.2 Ubicación geográfica recorrido Ruta 1-Google Earth

- **Ruta 2:**

  La ruta siguiente se localiza en los sectores de San José de Morán y San Juan de Calderón, cuyo recorrido se desarrolló en las calles principales que son: calle Carlos Mantilla (San José de Morán), calle Pio XII (San Juan de Calderón) y calle secundaria Calle Francisco Guañuna (San Juan de Calderón). Con un recorrido de aproximadamente de 3.9 km, representado en la siguiente figura:

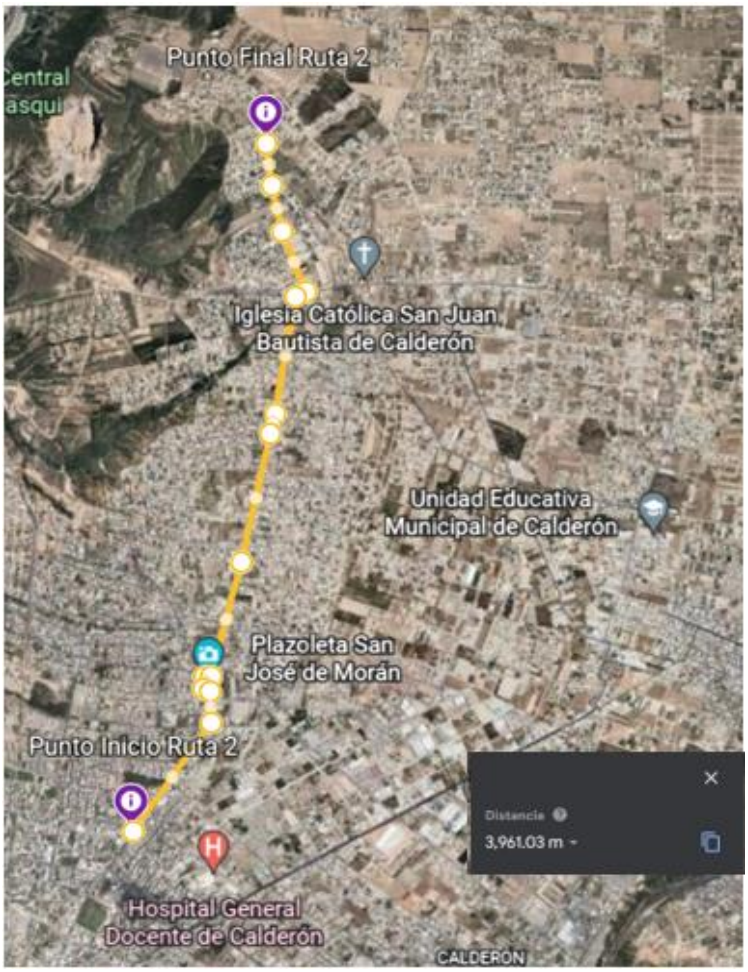

Figura 2.3 Ubicación geográfica recorrido Ruta 2-Google Earth

- **Ruta 3.**

  La última ruta se encuentra en el sector de San Juan de Calderón y Marianas, cuyo recorrido consistió en las calles principales y secundarias como: calle Francisco Guañuna (San Juan de Calderón), San Isidro (San Juan de Calderón), Francisco de Albornoz (San Juan de Calderón) y calle Capitán Giovanni Calles (Marianas), con un trayecto de 5.6 Km aproximadamente, como se representa en la siguiente figura:

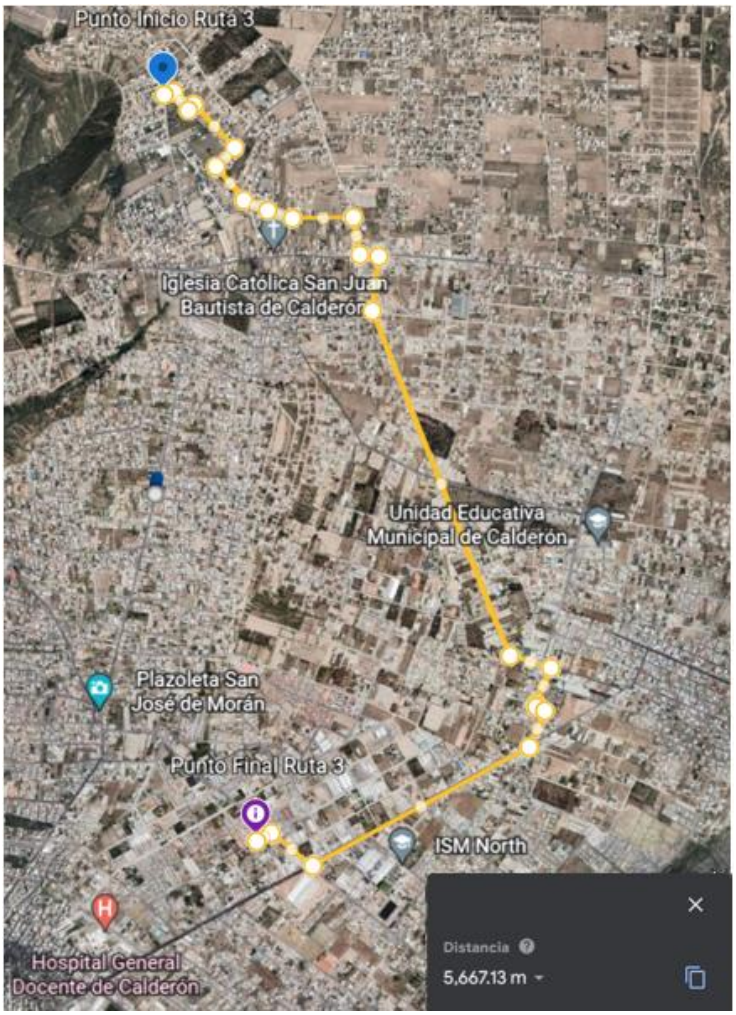

Figura 2.4 Ubicación geográfica recorrido Ruta 3-Google Earth

2.2.2 Recopilación de datos

En la etapa de recopilación de datos se ha considerado un lapso de tiempo de 30 días que comprende desde el 26 de noviembre hasta el 26 de diciembre, durante estos días se establecen los días y las horas adecuadas para las mediciones intentando cubrir las tres rutas a la semana. Además, para la recopilación se utiliza el servicio de VoIP mediante datos móviles, mientras que el terminal móvil realiza el servicio y se dirige a las rutas especificadas anteriormente sin un parámetro de velocidad, dado que, las mediciones se realizan en un ambiente ordinario y para el público en general. Se recomendó desarrollar las mediciones en distintos días, dado que, en su posterior análisis es mejor la comparación de los datos adquiridos.

El cronograma propuesto para la recolección de datos se presenta en el ANEXO 1.

2.2.3 Reconocimiento de las estaciones base

En las rutas de recolección de datos se reconocieron un total de 15 nodos y ubicados cercanamente en el trayecto, y los cuales pueden ser objeto de posibles estaciones base para el proceso de handover en la realización de la llamada de VoIP, detallados en la siguiente tabla 2.1:

Tabla 2.1 Ubicación de estaciones base mediante latitud y longitud

Ruta 1		
e_NodeB	*Latitud*	*Longitud*
28318	-0,105237319	-78,455651617944
28618	-0,097817167	-78,452219607476
28239	-0,092311595	-78,449768108404
28457	-0,090979166	-78,455440117697
28836	-0,089127744	-78,449512829290
28732	-0,082359669	-78,449191031277
Ruta 2		
e_NodeB	*Latitud*	*Longitud*
28845	-0,064443005	-78,4337285266362
28816	-0,074686233	-78,4348921844986
28411	-0,082826839	-78,4368868506241
28088	-0,088766887	-78,4397606836704
28183	-0,099966892	-78,4450680283825
Ruta 3		
e_NodeB	*Latitud*	*Longitud*
28783	-0,06377811	-78,424717812440
28611	-0,07268635	-78,422717518686
28474	-0,0764903	-78,416865215759
28759	-0,0882537	-78,430039834674

Con la ayuda de la aplicación CellMapper se ubicaron las estaciones base en las tres rutas propuestas.

- **Estaciones base Ruta 1**

Por medio de la aplicación CellMapper se ubicaron las celdas de manera geográfica, donde se obtuvieron las vías de acceso y lugares próximos al recorrido correspondiente a la Ruta1, observado en la figura 2.5:

Figura 2.5 Ubicación geográfica estaciones base Ruta 1-CellMapper

- **Estaciones base Ruta 2**

  Por medio de la aplicación CellMapper se ubicaron las celdas de manera geográfica, donde se obtuvieron las vías de acceso y lugares próximos al recorrido correspondiente a la Ruta2 (sector San José de Morán-San Juan de Calderón), observado en la figura 2.6:

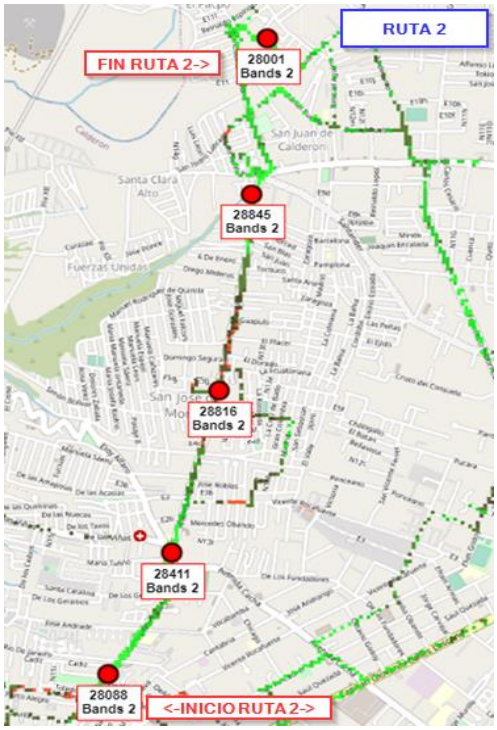

Figura 2.6 Ubicación geográfica estaciones base Ruta 2-CellMapper

- **Estaciones base Ruta 3**

  Por medio de la aplicación CellMapper se ubicaron las celdas de manera geográfica, donde se obtuvieron las vías de acceso y lugares próximos al recorrido correspondiente a la Ruta 3 (sector San Juan de Calderón-Marianas), observado en la figura 2.7:

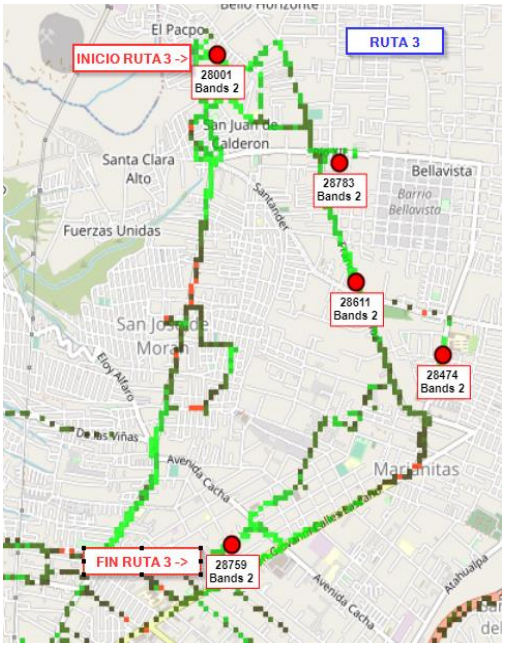

Figura 2.7 Ubicación geográfica estaciones base Ruta 3-CellMapper

2.3 PREPARACIÓN Y ANÁLISIS DE LOS DATOS RECOPILADOS POR LAS APLICACIONES EN EXCEL

Para analizar los datos capturados por las aplicaciones CellMapper, G-Net Track Lite y NetMonitor Lite, primero se procesaron los datos obtenidos utilizando sus archivos .csv, .txt y .kml. archivos de descarga por defecto en el terminal móvil, y de fácil lectura y escritura por medio de un computador apoyado en el cronograma de trabajo (ANEXO I).

2.3.1 Preparación de los datos

Los datos recopilados por las aplicaciones en el terminal móvil se recolectaron de manera simultánea y en tiempo real, de manera que, se pueden grabar errores en la captura de los parámetros. Para esta fase se exportaron los archivos .csv, .kml y .txt, observándose errores de captura como valores erróneos de potencia, valores duplicados, puntos geográficos inconsistentes como latitud de “-1.0” y longitud de “-1.0”, y problemas de medición en la aplicación mostrando un valor atípico de “2147483647”, ilustrado en la figura 2.8:

tech	lte_neig	rssi_strongest	node_id	cid	rssi	rsrq	rssnr	accurac	lat	long
LTE	1	-97	28618	192	-84	-12	17	-1	-0.09810777	-78,45334160
LTE	0	2147483647	28618	192	-81	-12	17	-1	-1.0	-1.0
LTE	0	2147483647	28618	192	-81	-12	17	-1	-1.0	-1.0
LTE	0	2147483647	28618	192	-81	-10	17	51	-0.09787241	-78,453410990

Figura 2.8 Localización de errores en las mediciones de campo-Netmonitor

2.3.2 Zona de Handover

Tomando en cuenta que en el servicio de telefonía móvil existe un margen de histéresis, el cual es un parámetro que controla el proceso del traspaso de señal entre dos celdas, las celdas manejan un algoritmo de decisión que evita que se inicie la nueva decisión de handover. Con este concepto, se comprende que la velocidad, potencia, distancia hacia la estación base y cantidad de usuarios afectan en la degradación de la señal, este problema es adquirido en el compendio de información de las aplicaciones, debido a esto se insertan columnas que muestren cuando el terminal móvil estando en una zona de intersección de dos estaciones base si el terminal pierde o recupera la señal de la llamada mientras se recorre el trayecto especificado. Se visualiza en la figura 2.9 la adición de la columna y filtrado del nuevo componente.

lte_vecino	rssi_stronge	node_id	cid	rssi	rsrq	rssnr	handover_1	handover_2
3	-98	eNB28088	191	-98	-19	-7	1	exitoso
3	-98	eNB28088	191	-98	-19	-7	1	exitoso
3	-98	eNB28088	191	-98	-19	-6	1	exitoso
4	-95	eNB28183	190	-99	-18	-6	1	exitoso
4	-95	eNB28183	190	-99	-18	-6	1	exitoso
3	-99	eNB28183	190	-97	-13	-6	1	exitoso
3	-99	eNB28183	190	-97	-13	-7	1	exitoso
4	-99	eNB28183	190	-95	-14	-7	1	exitoso
4	-99	eNB28183	190	-95	-14	-6	0	zona_interna
4	-100	eNB28183	190	-96	-16	-6	0	zona_interna

Figura 2.9 Detección de la zona de handover mediante el cambio de conexión de las estaciones base

2.3.3 Ángulo azimutal

El ángulo azimutal o de azimut se trata del ángulo medido entre el Norte, medido en el sentido de las agujas del reloj alrededor del horizonte del observador, y un punto de interés previamente escogido en el plano que la dirección de referencia, que en nuestro caso se toma como referencia el norte geográfico (sea verdadero, arbitrario o magnético), en otros de los casos se escoge como punto de un cuerpo celeste como el sol o la luna, pero a veces se utiliza el sur geográfico como referencia[25], [26].

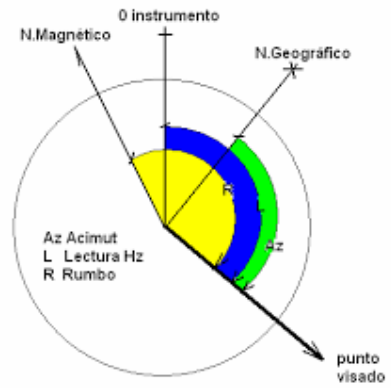

Figura 2.13 Representación del cálculo del ángulo de azimut[27]

Se lo calcula mediante la siguiente fórmula, y se presenta generalmente en grados.

$$Az = arctg\left(\frac{\Delta Longitud}{\Delta Latitud}\right) \quad (2.5)$$

2.3.4 Distancia entre 2 puntos geográficos

Para calcular la distancia geográfica en la tierra, si se tiene dos valores diferentes de latitud y longitud de dos puntos diferentes en la tierra, entonces, con la ayuda de la fórmula de Haversine, puede calcular fácilmente la distancia (la distancia más corta entre dos puntos en la superficie de una esfera). El término Haversine fue acuñado por el Prof. James Inman en 1835. Haversine es una fórmula muy popular y de uso frecuente al

desarrollar una aplicación GIS (Sistema de Información Geográfica) o al analizar caminos y campos. Se calcula mediante la fórmula siguiente[28]:

$$hav(\theta) = hav(\phi_2 - \phi_1) + \left(1 - hav(\phi_2 - \phi_1) - hav(\phi_2 + \phi_1)\right).hav(\lambda_2 - \lambda 1) \quad (2.6)$$

otra forma de calcular es con la fórmula del semiverseno para conocer la distancia del círculo máximo, debido a que la tierra puede verse como un círculo (teorema del coseno esférico)[28]:

$$\cos(c) = \cos(a)\cos(b) + \sin(a)\sin(b)\cos(C)$$

Donde este valor se lo multiplica por el radio de la tierra de 6.370 km aproximadamente.

2.4 TÉCNICAS DE MACHINE LEARNING POR CLASIFICACIÓN PARA EL MODELADO DE LOS DATOS

2.4.1 K-nearest neighbor (KNN, vecino más cercano K)

La técnica *K-nearest neighbor - KNN,* es una de las más utilizadas para el reconocimiento de patrones y correlaciones entre datos. Realiza la clasificación en base a las características similares que comparten los datos[19]; se puede proponer en la solución de búsquedas y detección de anomalías.

Para clasificar un nuevo objeto y asignarlo a un grupo determinado se analizan las características de sus *k* vecinos más cercanos y se realiza la correspondiente asignación[20], una descripción de esta técnica se presenta en la figura 2.10.

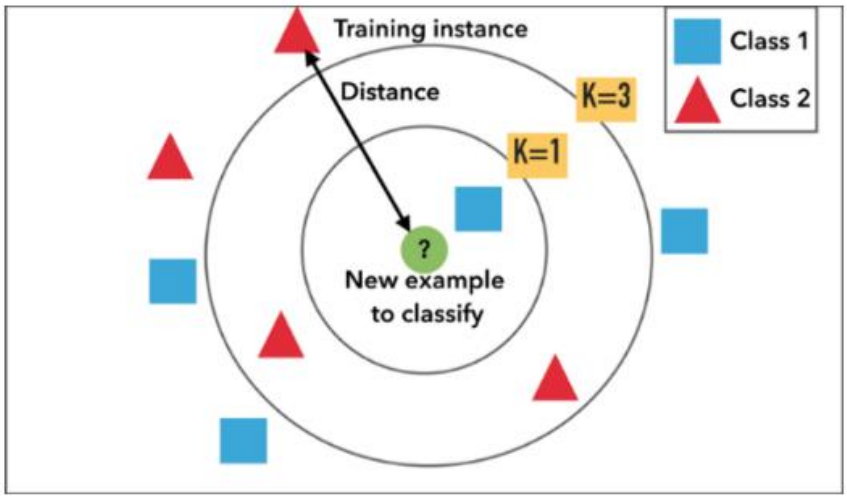

Figura 2.10. Diagrama de ejemplo técnica de ML de clasificación k-nearest neighbor [20]

Esta en una de las técnicas a aplicar en este estudio, tomando como base de datos la información recolectada durante cuatro semanas de diferentes estaciones base en el sector Norte de Quito. El modelo obtenido clasificará cuando el proceso de handover puede ser fallido o acertado, conociendo las características de dicha conexión y clasificándolos mediante los *k*-vecinos más cercanos.

2.4.2 Árboles de decisión

La técnica *DecisionTree* o árboles de decisión, al igual que *k-nearest neighbor (*k-vecino más cercano*)*, es uno de los algoritmos más utilizados para la extracción y clasificación de información. Permite una representación gráfica de las posibles soluciones a una decisión fundamentado en ciertas características.

La idea se fundamenta en la estructura de un árbol, en el cual contiene una raíz, ramas y hojas. En esta técnica, la raíz está identificada por los nodos desplegados hacia abajo, mientras que las ramas se despliegan a izquierda y derecha del mismo, y, por lo tanto, los nodos terminales corresponden a las hojas[19].

Los árboles de decisión realizan la clasificación y agrupamiento de la información mediante valores que las características contienen, siendo la raíz el atributo con mayor importancia, y cada hoja pertenecerá a un subconjunto de una misma clase o etiqueta. En la figura 2.11 se muestra un ejemplo sencillo, pero fácil de interpretar, de un árbol de decisión para la aceptación o rechazo de una oferta de trabajo (figura tomada de la referencia indicada).

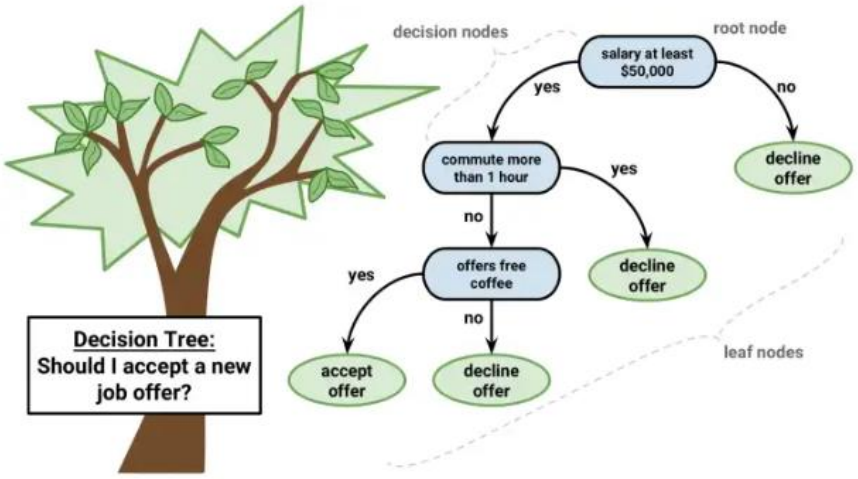

Figura 2.11 Ejemplo técnica ML Árboles de decisión, decisión de aceptar o no una oferta de trabajo[21]

La técnica de árboles de decisión será de gran ayuda en este estudio, ya que permitirá obtener una ilustración gráfica de los factores fundamentales que influyen al clasificar si la conexión celular en una red LTE puede fallar en el momento del traspaso de la conexión de una estación base a otra cuando el dispositivo se encuentra en movimiento. Mediante el modelo obtenido se establecerá el factor más relevante que influye en dicho proceso y se colocará en la raíz del árbol, ayudando a identificar el comportamiento de los datos y agrupando las características menos relevantes en las hojas de la ilustración.

2.4.3 Evaluación del modelo

2.4.3.1 Matriz de Confusión

La matriz de confusión se trata básicamente por una tabulación de información en forma de matriz, donde se muestra la comparación entre los valores pronosticados correctos e incorrectos. Se puede utilizar tanto en la clasificación de una sola etiqueta o de múltiples etiquetas, lo cual dependiendo del número de etiquetas se trazará una matriz más grande o reducida de dimensión N x N, donde N representa el número total de clases objetivo. De este modo, al clasificar una sola etiqueta la dimensión de la matriz será de 2 x 2 siendo el número de etiquetas de evaluación para el modelo[22].

Esta matriz muestra información del número de positivos verdaderos (TP), negativos verdaderos (TN), negativos falsos (FN) y positivos falsos (FP), los cuales son devueltos al aplicar un algoritmo de clasificación a un conjunto de datos de prueba. Brindando una comprensión integral de la efectividad del modelo de clasificación aplicado y de los tipos de errores que se producen en el mismo[23]. Representado en la figura 2.12[22]:

Figura 2.12 Representación matriz de confusión para un problema de clasificación binaria de dimensión 2x2, con 4 valores.

2.4.3.2 Métricas de evaluación del modelo en base a la matriz de confusión

Una vez generada la matriz de confusión con sus determinados el número de positivos verdaderos (TP), negativos verdaderos (TN), negativos falsos (FN) y positivos falsos (FP); se procede a medir el rendimiento general del modelo planteado, para lo cual, se realiza el cálculo de las métricas de evaluación estadística del modelo mediante la precisión, la exactitud, la sensibilidad y la especificidad[22].

- **Exactitud de clasificación.**

Es uno de los factores más críticos para la evaluación del modelo, dado que mide con qué frecuencia el modelo predice una correcta salida. Por tanto, cuanto mayor sea la exactitud, mejor será el modelo de clasificación[22]. Para calcular la exactitud se considera la siguiente fórmula:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{2.1}$$

- **Sensibilidad**

Permite medir la capacidad del modelo hacia la detección muestras positivas. Cuanto mayor sea el valor de precisión, mayor cantidad de muestras positivas el modelo ha logrado detectar. Se calcula mediante la relación entre el número de verdaderos positivos (TP) y la suma del número de positivos verdaderos (TP) con el número de negativos falsos (FN)[22].

$$Recall = \frac{TP}{TP+FN} \tag{2.2}$$

- **Precisión**

Compara el número de positivos verdaderos (TP) con el número total de muestras positivas (positivos verdaderos y positivos falsos). Se relaciona de una manera proporcional con el rendimiento para clasificar muestras positivas[22],[24].

$$Precision = \frac{TP}{TP+FP} \tag{2.3}$$

- **Especificidad**

También conocida como tasa negativa verdadera, se calcula como la relación entre el número total de predicciones negativas verdaderas (TN) y el número total de valores reales negativos (positivos falsos (FP) y negativos verdaderos (TN)). En donde, la mejor especificidad es de 1,0. Por el contrario, el valor de 0,0 corresponderá a una deficiencia en el modelo[24].

$$Specificity = \frac{TN}{TN+FP} \quad (2.4)$$

2.5 IMPLEMENTACIÓN DE LOS MODELOS DE CLASIFICACIÓN

Una vez obtenido el dataset depurado sin errores que afecten el análisis de las tres rutas establecidas y que contiene los parámetros claramente identificados que actúan directamente en el proceso de handover en el servicio de VoIP activo durante el recorrido de cada ruta, se procede a su análisis e implementación de las técnicas de machine learning de aprendizaje supervisado a través de K-nearest neighbor (KNN, K vecino más cercano) y árboles de decisión.

2.5.1 Importación de los archivos .xlsx

La importación de los archivos depurados se realiza en el lenguaje de programación R con el uso del IDE de Rstudio, la lectura de los datos .xlsx se hace de manera general, puesto que, se trabajará con la misma base de información por ruta en los distintos modelos, el código de este proceso se puede visualizar en la tabla 2.2:

Tabla 2.2 Codificación de importación de archivos .xlsx en el lenguaje de R

```
##ESCUELA POLITÉCNICA NACIONAL
#TRABAJO DE INTEGRACIÓN CURRICULAR
#MODELO PREDICTIVO COM ML DEL PROCESO DE HANDOVER
#JONATHAN RAMIRO VILLARREAL TAYÁN
#######################################################

#Liberias de manejo de archivos
library("openxlsx")#Permite abrir los archivos .xlsx (excel)
library("readxl")#Permite leer los archivos .xlsx
library("dplyr")#Permite el manejo de datos en tramas de datos

#Elección de la base de datos, nos muestra la ubicación del archivo
file.choose()

#Importación de los archivos .xlsx de varias mediciones en un solo archivo por ruta
#RUTA 1
# SECTOR DE CARAPUNGO
#carga y lectura de la base de datos para ruta
```

```
r1data1<-read_xlsx("C:\\Users\\Desktop-DTB\\Desktop\\mediciones\\Procesamiento\\Arboles_ruta1\\Ruta1_data.xlsx")

#RUTA 2
#SECTORES DE SAN JOSÉ DE MORÁN Y SAN JUAN DE CALDERÓN
r2data1<-read_xlsx("C:\\Users\\Desktop-DTB\\Desktop\\mediciones\\Procesamiento\\Arboles_ruta2\\Ruta2_data.xlsx")

#RUTA 3
#SECTOR DE SAN JUAN DE CALDERÓN Y MARIANAS
r3data1<-read_xlsx("C:\\Users\\Desktop-DTB\\Desktop\\mediciones\\Procesamiento\\Arboles_ruta3\\Ruta3_data.xlsx")
```

En este caso para la importación de los datos se utilizó la función **file.choose()**, la cual al ejecutarla devuelve una ventana de visualización de los archivos en el sistema, al seleccionar un archivo, la sentencia devuelve la ruta absoluta del archivo o directorio. La función **read_xlsx**, como su nombre lo indica lee datos de archivos de Excel y los importa a R para su posterior análisis.

2.5.2 Eliminación de variables faltantes o perdidas

En la fase de limpieza de datos se procede a eliminar los parámetros incongruentes con las mediciones tomadas. Esta sección se puede realizar mediante un filtro de búsqueda, reemplazo y luego borrado de las filas con errores en los archivos .xlsx de cada ruta, o mediante programación en R una vez cargados los archivos, como, por ejemplo, la eliminación del problema de captura de parámetros con el error **2147483647.** Este proceso se implementa mediante las siguientes instrucciones:

Tabla 2.3 Codificación de eliminación de errores y variables faltantes

```
#Eliminación de los datos erróneos en la medición->error  2147483647
r1data1[r1data1==2147483647]<-NA #Se asigna un valor no existente
#A continuación se omiten las líneas de datos con el valor inexistente
```

```
r1data2<-na.omit(r1data1)
```

La función **na.omit**, ayuda a borrar las filas con los parámetros que tienen datos inexistentes, y se crea una nueva base de datos, debido a que estos pueden tratarse de datos faltantes o perdidos, por tanto, al eliminar estos valores se evitan errores en el posterior análisis de los datos.

2.5.3 Creación de nuevas variables en el dataset en R

Para la creación de nuevas variables en el lenguaje R, se asignan cadenas de caracteres con valores no existentes “NA”, los cuales son actualizados posteriormente. Para este caso se crea la columna de calidad de recepción “calidad_rx”, que comprende en dar una descripción corta del atributo creado tomando en cuenta el nivel de señal recibida, como se aprecia en la tabla 2.4:

Tabla 2.4 Codificación de creación de nuevas categorías

```
#Creacion de nuevas variables en la base de datos

###Añade columna de calidad de RX
r1data2$calidad_rx<-"NA"
r1data2$calidad_rx[r1data2$rsrq>=(-5)]<-"Excelente"
r1data2$calidad_rx[r1data2$rsrq<(-5)&r1data2$rsrq>=(-9)]<-"Media"
r1data2$calidad_rx[r1data2$rsrq<(-9)]<-"Mala"
```

2.5.4 Implementación del algoritmo de clasificación por Árboles de Decisión

a. Compilación de librerías

Para el uso de librería en el lenguaje R, se escribe el comando **library**, y entre paréntesis la librería que se desea utilizar, como se presenta a continuación en la tabla 2.5:

Tabla 2.5 Inserción de librerías

```
#Librerías de procesamiento de datos
library(recipes)#permite realizar una receta de los datos, creación y edición
```

```
library(rsample)#Permite una validación cruzada y separacion de datos en conj. de entreamiento y prueba
library(themis)#Permite proporcionar y manejar datos faltantes
library(dplyr) #Permite el manejo de datos en tramas de datos

#Libreria de evaluación de los datos
library(workflows)#Proporciona una forma estructurada de realizar un análisis
library(parsnip)#Permite una interfaz unificada para crear un modelo
library(tune)#Permite la optimización en modelos de aprendizaje automático
library(yardstick)#Permite evaluar el rendimiento del modelo
library(dials)#Permite un ajuste de los parámetros del modelo
library(rpart)#Permite construir árboles de decisión
library(kknn)#Permite el aprendizaje del vecino más cercano
library(data.table)#Permite manipular datos eficientemente

#Librerias de presentación del modelo de clasificacion
library(rpart.plot) #Proporciona una función para visualizar árboles de decisión ajustados.
library(rattle)#Proporciona una interfaz gráfica de usuario (GUI) para ajustar modelos de aprendizaje automático en R.
library(RColorBrewer)#Proporciona paletas de colores personalizadas para visualizaciones en R, árboles de desición
```

b. Creación de la receta para preprocesamiento de los datos de entrenamiento

Se crea una receta para preprocesar los datos de entrenamiento, para luego aplicarlos a los datos de prueba mediante la utilización de la función **bake()**, después se equilibran las frecuencias de las categorías de la variable de respuesta, que en este caso es el atributo *handover_2*. Este último atributo fue constituido por las características si el handover fue exitoso o fallido, luego se crean variables ficticias excepto para la variable de respuesta, eliminándose todas las variables con cero varianza y normalizando las variables sobrantes para que contengan media cero y desviación estándar de uno. Finalmente, este

proceso se aplica a los datos de prueba y se almacenan estos datos preprocesados en la variable de prueba **test_proc**, como se indica en la tabla 2.6:

Tabla 2.6 Creación de la receta

```
#procesamiento de los datos para el modelo
r1data2_rec<-recipe(handover_2~.,data=r1data2_train)%>%
  themis::step_downsample(handover_2)%>%
  recipes::step_dummy(all_nominal(),-all_outcomes())%>%
  recipes::step_zv(all_numeric())%>%
  recipes::step_normalize(all_numeric())%>%
  recipes::prep()
test_proc<-bake(r1data2_rec,new_data=r1data2_test)
test_proc
```

A continuación, como se observa en la tabla siguiente, se utiliza la función **juice()** para extraer los datos de un objeto y se cuenta con la función **count()** el número de observaciones de cada nivel del atributo en cuestión que en este caso es *handover_2*.

Tabla 2.7 Extraemos la data de la receta y conteo de observaciones

```
#Extraemos la data de la receta y se cuentan las observaciones
juice(r1data2_rec)%>%
  count(handover_2)
```

c. Especificación del modelo de árbol de decisión

Gracias a la librería **tidymodels**, la función **decision_tree()** crea una plantilla en blanco para definir un modelo de árbol de decisión. Como se observa en la tabla 2.8, también se utiliza la función **set_engine()**, que especifica el motor del modelo que en este caso se trata de la librería **rpart** para ajustar el modelo a un árbol de decisión, y se elige el tipo de modelo, es decir, si se trata de un modelo de clasificación o de regresión.

Tabla 2.8 Especificación del modelo

```
###ESPEFICACION DEL ARBOL DE DECISION
```

```
tree_spec<-decision_tree()%>%
  set_engine("rpart")%>%# Ajustar el modelo a un árbol de decisión
  set_mode("classification")
```

d. Ajuste del modelo

Para esta sección, se ajusta el modelo creado previamente con la función **tidymodels**, ajustando la variable de salida de *handover_2*, y que todas las demás se tomen como datos de entrada de la receta y guardándolo en una nueva variable llamada **tree_fit**, dicho código se muestra en la tabla 2.9:

Tabla 2.9 Ajuste del modelo

```
#Ajuste del modelo
tree_fit<-tree_spec%>%
  fit(handover_2~.,data=juice(r1data2_rec))
tree_fit
```

e. Selección de atributos de ingreso

Para seleccionar atributos a considerarse en el modelo se hace uso de la función **select**, la cual permite escoger uno o varios parámetros por columnas del dataset que son de utilidad para el modelo de clasificación. En este estudio se probó la selección de las variables de mayor consideración, que llegaron a ser los valores de rsrq, rssi, rssi_strongest, el ángulo de azimut y la velocidad, en cambio, en la salida la variable seleccionada fue la de handover, siendo el objetivo de estudio. Se obtuvo que al escoger todas las variables de entrada el árbol para este estudio es extenso en su comprensión. A continuación, se muestra en la tabla 2.10 la etapa de selección de atributos:

Tabla 2.10 Selección de atributos

```
#Seleccion de las columnas a en el dataframe
r1data2_train<-
select(r1data2_train,c(rsrq,rssnr,rssi,rssi_strongest,handover_2,azimut,v
elocidad,dist_bs_mov))
str(r1data2_train)
```

La función str(), muestra la estructura del objeto o atributo.

f. División en conjuntos de entrenamiento y prueba

Se procede a dividir el conjunto de datos en los porcentajes de entrenamiento y prueba correspondiente al 80% y 20%, mediante una función implementada manualmente, donde la especificación denominada **size** establece el tamaño del conjunto de entrenamiento, si la condición se cumple; o el conjunto de prueba, si la sentencia devuelve un valor de falso, el código asociado a lo mencionado se puede visualizar en la tabla 2.11:

Tabla 2.11 Creación de conjuntos de entrenamiento y prueba

```
#Elaboración de conjuntos de prueba y entrenamiento
create_train_test<-function(data,size=0.8,train=TRUE){
  n_row=nrow(data)
  r1data2_train_row=size*n_row
  train_sample<-1:r1data2_train_row
  if(train==TRUE){
    return(data[train_sample,])
  }else{
    return(data[-train_sample,])
  }
}
```

g. Muestreo de la matriz

Para garantizar una mezcla aleatoria de los datos desde la primera fila hasta la fila última, se aplica la función **sample()**, como se observa en la tabla 2.12 :

Tabla 2.12 Muestreo de la matriz

```
#Muestreo de la matriz principal
r1data2_train<-r1data2_train[sample(1:nrow(r1data2_train)),]
r1data2_train#visualización datos de entrenamiento
```

h. Generación del conjunto de datos de entrenamiento y prueba

Se crean los conjuntos de entrenamiento y de prueba utilizando una división 80/20, es decir, 80% para entrenamiento y 20% para prueba, como se muestra

en la tabla 2.13. El conjunto de entrenamiento se utiliza para ajustar y validar el modelo, mientras que el conjunto de prueba se usa para evaluación del modelo.

Tabla 2.13 Conjunto de datos de entrenamiento y prueba

```
#Especificacion de los datos de 80% de entrenamiento y 20% de prueba
train<-create_train_test(r1data2_train,0.8,train = TRUE)
test<-create_train_test(r1data2_train,0.8,train=FALSE)
```

i. Generación del árbol de decisión

Se realiza un ajuste del árbol de decisión con la librería **rpart** y la función **fit**, donde se especifica la variable de respuesta que en este caso es *handover_2*. Se ingresa el conjunto de datos de entrenamiento, además se especifica el método a usar, que en este caso se trata de un problema de clasificación. Es importante mencionar que la sentencia **minsplit**, especifica un número mínimo de observaciones que debe tener cada nodo para que se produzca una división adicional en el árbol de decisión, en este caso se determinó un valor adecuado de 8 observaciones para que el árbol de decisiones tenga mejor apreciación. En cambio, la sentencia **minbucket**, se trata del número de observaciones mínimo que debe tener cada hoja; el código completo se muestra en la tabla 2.14 :

Tabla 2.14 Generación del árbol de decisión

```
#GENERACION DEL ARBOL DE DECISION
fit<-rpart(handover_2~.,data = train,method = "class",minsplit=8,minbucket=1)
```

j. Predicción de los datos de prueba

Se utiliza el árbol de decisión ajustado anteriormente con el conjunto de datos de entrenamiento para predecir las clases del conjunto de prueba mediante la función **predict()**, además, se especifica el tipo de salida que en este caso es de clasificación, como se muestra en la siguiente tabla:

Tabla 2.15 Predicción de los datos de prueba

```
#Prediccion de los datos de prueba
prueba<-predict(fit,test,type="class")
```

k. Representación gráfica del árbol de decisiones

Se utiliza la librería **rpart** y mediante la función **fancyRpartPlot()** se devuelve el árbol de decisiones con los ajustes realizados y se especificaron los siguientes argumentos para mejorar su visualización:

- Cex: cambia el tamaño de letra del árbol
- Main: coloca un título a la gráfica
- split.box.col: establece una paleta de colores para los nodos (0,1…)
- yspace: establece espaciado en el eje y del árbol
- space: espaciado en el eje x del árbol
- split.cex= cambia el tamaño de letra del nodo
- shadow: modifica el tamaño de la sombra de las hojas
- yshift: modifica el espacio entre hojas y nodos
- gap: modifica el tamaño del árbol en la hoja de impresión

A continuación, se presenta la configuración en la tabla 2.16:

Tabla 2.16 Representación gráfica del árbol de decisiones

```
#GRAFICO DEL ARBOL DE DECISION
fancyRpartPlot(fit,cex=0.3,main = "RUTA 1",split.box.col=5, yspace = 0.5, space=1,split.cex = 1,shadow = 0.3,yshift=1.4,gap=1)
```

l. Matriz de confusión

La matriz de confusión muestra un resumen de los valores pronosticados contra los valores reales. En la herramienta de R la matriz de confusión se la obtiene como se muestra en la tabla 2.17, se especifican los datos de entrada y la variable de salida.

Tabla 2.17 Obtención de la matriz de confusión en R

```
#CREACION DE LA MATRIZ DE CONFUSION
matriz<-table(test$handover_2,prueba)
```

```
matriz
```

m. Exactitud del modelo

Para obtener una métrica importante, la cual se trata de la exactitud del modelo, se usa la matriz antes obtenida, como se indicó en la ecuación (2.1), se realiza el algoritmo indicado en la tabla 2.18 para extraer los datos de la matriz y se imprime el valor obtenido de la operación de acuerdo a lo presentado en la tabla 2.18:

Tabla 2.18 Obtención de la exactitud del modelo

```
#EXACTITUD DEL MODELO
accuracy_test<-sum(diag(matriz))/sum(matriz)
print(paste("Exactitud del modelo de clasificación",accuracy_test))
```

La implementación del algoritmo se observa en ANEXOII.

2.5.5 Implementación del algoritmo de clasificación por vecino más cercano (KNN)

La implementación del algoritmo para la obtención del modelo del vecino más cercano (KNN) es de manera similar al del árbol de decisión; pero con un grado menor de dificultad, dado que, se utilizan algunos de los pasos antes realizados, como son: la creación del data set, la creación del conjunto de datos de entrenamiento y prueba que se establece por medio de una semilla, la aplicación de la receta es similar y la diferencia radica en que se realiza la especificación del modelo y su correspondiente evaluación descritas a continuación:

a. Creación de los conjuntos de prueba y entrenamiento para el vecino más cercano

Se establece una semilla para la generación de números aleatorios para garantizar que los resultados de la división sean reproducibles, luego de creada la semilla se dividen del conjunto de datos los porcentajes de entrenamiento y pruebas correspondiente al 75% y 25%, y al final son almacenados de forma tabulada con su correspondiente nombre.

Tabla 2.19 Creación de los conjuntos de prueba y entrenamiento-KNN

```
#CREAMOS UNA SEMILLA
set.seed(1234)
#creamos una división aleatoria por defecto es del 75% para entrenamiento
y 25% para prueba
r1data2_split<-initial_split(r1data2)
#Creamos la base de entrenamiento y de prueba
r1data2_train<-training(r1data2_split)
r1data2_test<-testing(r1data2_split)
```

b. Especificación del modelo del vecino más cercano

El siguiente código de la tabla 2.20, establece mediante la función **nearest_neighbor()** el modelo a implementar (vecino más cercano), por otra parte, con la función **set_engine()** se establece la librería a utilizar, que en este caso es **kknn**, y el tipo de modelo a elegirse, el cual se escoge entre clasificación y regresión.

Tabla 2.20 Creación de los conjuntos de prueba y entrenamiento-KNN

```
###ESPEFICACION DEL VECINO MAS CERCANO
kknn_spec<-nearest_neighbor()%>%
  set_engine("kknn")%>%
  set_mode("classification")  #Especificacion del modelo
```

El ajuste del modelo y creación de la receta se realiza de igual manera al modelo anterior y se procede a la evaluación del modelo actual.

c. Evaluación del modelo del vecino más cercano

Para la evaluación del modelo se realiza una validación cruzada de MonteCarlo, mediante la función **mc_cv**¸a continuación, se establece la porción a tomar de la muestra, como se observa en la tabla 2.21, se configura el 90% de los datos para la muestra de entrenamiento y 10% de los datos para la muestra de prueba; se ajusta el modelo KNN de clasificación. Para finalizar,se recopilan las métricas de evaluación, donde se incluyen a la precisión, sensibilidad, especificidad y exactitud, utilizando la función **collect_metrics**, de la librería **yardstick**.

Tabla 2.21 Evaluación del modelo del vecino más cercano

```
#EVALUACION DEL RENDIMIENTO
set.seed(1234)
validation_split<-mc_cv(juice(r1data2_rec),prop = 0.9,strata = handover_2)
validation_split
kknn_res<-
tune::fit_resamples(kknn_spec,handover_2~.,validation_split,control=control_
resamples(save_pred = TRUE))
kknn_res%>%
  collect_metrics()   #para extraer métricas
```

Para la extracción de las métricas del modelo del árbol de decisión se procede de manera similar (ANEXO II y III).

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

En este capítulo se realizará un análisis de los resultados obtenidos al desarrollar los modelos del árbol de decisión y del vecino más cercano (KNN) en base a los algoritmos implementados considerando los atributos de entrada y salida y las métricas de evaluación de cada modelo por cada ruta.

3.1 ANÁLISIS DE LOS MODELOS DE CLASIFICACIÓN POR RUTAS

3.1.1 Árbol de decisión general

Para esta sección se generó un árbol de decisión general siguiendo el proceso de implementación, para la creación del árbol general se juntaron los datos de las tres rutas propuestas con los atributos que se consideran de importancia en su totalidad, una vez realizado el proceso de depuración de los datos y adicionando la columna de la zona de handover se obtuvo el siguiente árbol de decisiones:

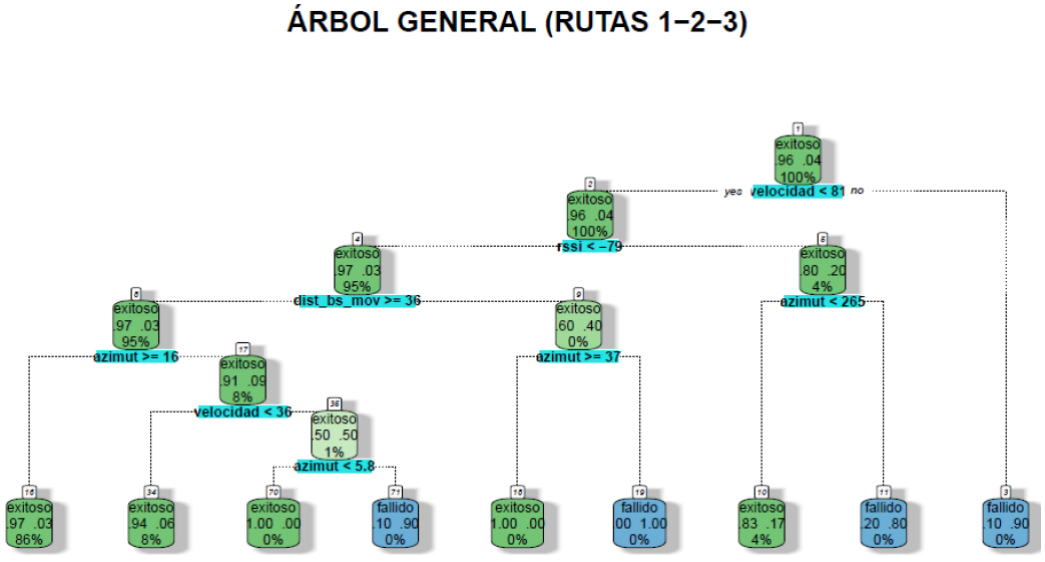

Figura 3.1 Árbol de decisión Ruta1-2-3

El árbol de decisión general de las tres rutas y se comprueba que los parámetros de mayor importancia son la velocidad del móvil, el azimut, la potencia RSSI y la distancia de la estación base al móvil, donde el factor predominante es la velocidad, si el móvil alcanza una velocidad mayor a 81 km/h el handover obtendrá una categoría de fallido, caso contrario será exitoso, como segundo parámetro de elección es la potencia RSSI,

que es la intensidad de la señal, donde si obtiene valores <-79 dBm influye la distancia de la estación base y tendrá que se menor de 4300m con un ángulo de azimut mayor o igual que 16 grados y menor que 265 grados para su proceso exitoso.

3.1.2 Árbol de decisión Ruta 1

Para generar el árbol de decisión de la Ruta 1 (Sector Carapungo) se realizó la implementación indicada en el capítulo anterior, donde se escogieron los atributos de entrada considerando los parámetros que mayormente afectan al modelo de clasificación, los cuales son el rssi, rssi_strongest, rsrq, la velocidad con la cual el móvil se traslada por el trayecto, el ángulo de azimut relacionado con el norte geográfico, y como último parámetro la distancia del terminal móvil con respecto a la estación base, y como salida se estableció al estado del handover. Con estos antecedentes se obtuvo el árbol de la siguiente figura 3.2, donde se observa un árbol de decisión codificado mediante la herramienta Rstudio y sus librerías respectivas.

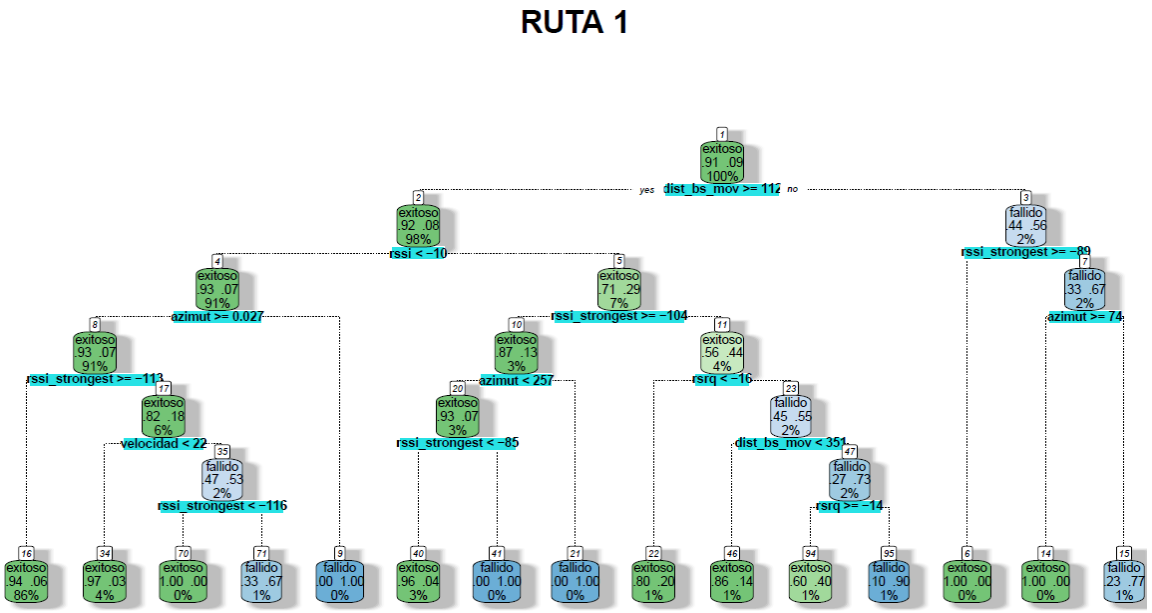

Figura 3.2 Árbol de decisión -Ruta1

En la figura 3.2, el árbol de decisión muestra el estado del proceso de handover tomando como parámetro de mayor prioridad la distancia del terminal móvil con respecto a la estación base, en este caso de la RUTA 1. En el primer parámetro de comparación la distancia del terminal hacia la estación base debe tener un valor mayor o igualde 112 metros, donde el 91% de los datos el handover se da de una manera “exitosa” y del restante que es un 9% del total de los datos como un estado de “fallido” si no sobrepasa este valor de distancia. Dado que dependiendo de la distancia a la que se encuentre conectado el dispositivo puede alcanzar menor o mayor nivel de señal, lo que se puede

visualizar en el nodo siguiente del árbol de decisión. El parámetro de RSSI (Intensidad de señal recibida) debe ser menor a -10 dBm para que el handover sea exitoso, tomando en cuenta que mientras mayor sea la potencia de RSSI la señal se recibe con mayor intensidad, tomando en cuenta que en el 92% de los datos corresponden al caso de éxito.

En el nodo 4 del árbol se observa que el parámetro del ángulo de azimut también llega a ser un factor determinante en el estado de éxito o falla del handover, si llega a obtenerse un ángulo mayor a los 0.027 grados el handover se produce de manera exitosa, esto se podría deber a lugares con baja potencia de recepción, ya que si el terminal móvil continúa moviéndose por la misma dirección puede desconectarse o reconectarse a la red.

Además, se observa que si el rssi_strongest es mayor o igual a -113 dBm existe una gran probabilidad de que el handover sea exitoso.

La velocidad de movimiento se ve reflejada en el nodo 17, donde si la velocidad del móvil es menor a 22 km/h el handover será exitoso, esto se puede deber a que si se aumenta este valor la celda no tiene recursos para un traspaso rápido de señal a otra estación base.

Se resumen los datos más relevantes en la siguiente tabla 3.1, tomando en cuenta el número total de nodos y el árbol de decisión en forma de texto generado por R (ANEXO IV):

Tabla 2.3 Resumen modelo árbol de decisión Ruta 1

Categoría (Handover)	**Distancia eNB-UE**	**RSSI(dBm)**	**RSSI_Strongest (dBm)**	**RSRQ(dB)**	**Azimut(grados)**	**Velocidad (km/h)**
Exitoso	<112 m	>-10	<-89	>=-14	<267	<19
Fallido	>616 m	<-10	>-86	<16	>267	>19

3.1.2.1 Matriz de confusión Ruta 1

Del análisis de la matriz de confusión de la Ruta 1 se observa que de un total de 219 datos utilizados para prueba en la categoría de “exitoso”, 204 datos son acertados, por el contrario de 9 datos que fueron utilizados en la categoría de “fallido”, 2 han sido acertados. Por lo tanto, se comprende que más del 50% de los datos tomados son acertados. Obteniendo los valores de las métricas calculado con las ecuaciones del segundo capítulo exactitud del 90.3%, sensibilidad del 93,15%, precisión del 96,68 y especificidad del 22,22%.

```
#Matriz de confusión Ruta 1
matriz
          prueba
           exitoso fallido
exitoso      204        7
fallido       15        2
```

Figura 3.3 Matriz de confusión -Ruta1

3.1.2.2 Métricas de evaluación del modelo Ruta 1

Otros valores de importancia son las métricas de evaluación, donde el lenguaje de R permite obtener la exactitud del modelo, como se observa en la siguiente figura, donde la exactitud en porcentaje corresponde 98,4%.

```
#EXACTITUD DEL MODELO
print(paste("Exactitud del modelo de clasificación",accuracy_test))
.] "Exactitud del modelo de clasificación 0.98443579766537"
```

Figura 3.4 Exactitud del modelo-Ruta1

Una técnica para evaluar los modelos de clasificación es la curva ROC (Receiver Operating Characteristic) que es también conocida como representación de la sensibilidad, donde mientras más área se abarque de la curva mayor sensibilidad tendrá el modelo, estos valores varían entre 0 y 1. En la siguiente figura, se observa un valor de 0,638, cuyo valor se relacionará más adelante para una comparativa entre modelos.

```
  #Extracción métricas-Ruta1
  collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.64     25  0.0225 Preprocessor1_Model1
roc_auc  binary     0.662    25  0.0265 Preprocessor1_Model1
```

Figura 3.5 ROC y Exactitud-Ruta1

3.1.2.3 Modelo de clasificación del vecino más cercano (KNN)-Ruta 1

El algoritmo para el modelo del vecino más cercano se describió en el capítulo anterior y se codifica similarmente a la ruta general (ANEXO III), este modelo al no tener de salida una representación gráfica, obteniendo una tasa de error del 0,28, lo que indica que la tasa de clasificación incorrecta es del 28%. En otras palabras, el modelo clasifica correctamente el 72% de las observaciones. Para conocer el rendimiento del modelo su evaluación se obtiene mediante las métricas de la exactitud y curva ROC que se muestran en la figura 3.6:

```
#metricas KKN
  collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.653    25  0.0159 Preprocessor1_Model1
roc_auc  binary     0.666    25  0.0192 Preprocessor1_Model1
```

Figura 3.6 ROC y Exactitud- KNN- Ruta1

Al comparar ambos valores se aprecia que el modelo del vecino más cercano tiene una mayor exactitud que el modelo del árbol de decisiones para la Ruta1, donde el vecino más cercano obtuvo una exactitud promedio del 65,5%, con una estimación ROC de 66,6% frente al árbol de decisión con exactitud promedio del 64%, y estimación ROC del 66,2% por lo cual, el modelo que obtiene mejores predicciones es el del vecino más cercano (KNN). Se debe mencionar que dicho modelo puede trabajar con dos estados de salida posibles, como se dio en nuestro caso (“exitoso”,”fallido”), si el modelo contiene más variables se producen errores en su aplicación.

3.1.3 Árbol de decisión Ruta 2

Para generar el árbol de decisión de la Ruta 2 (Sector San José de Morán – San Juan de Calderón) se realizó la implementación presentada en el capítulo anterior y considerando los mismos atributos que para la ruta 1. El resultado se observa en el árbol de decisión de la ruta 2 que se presenta en la siguiente figura.

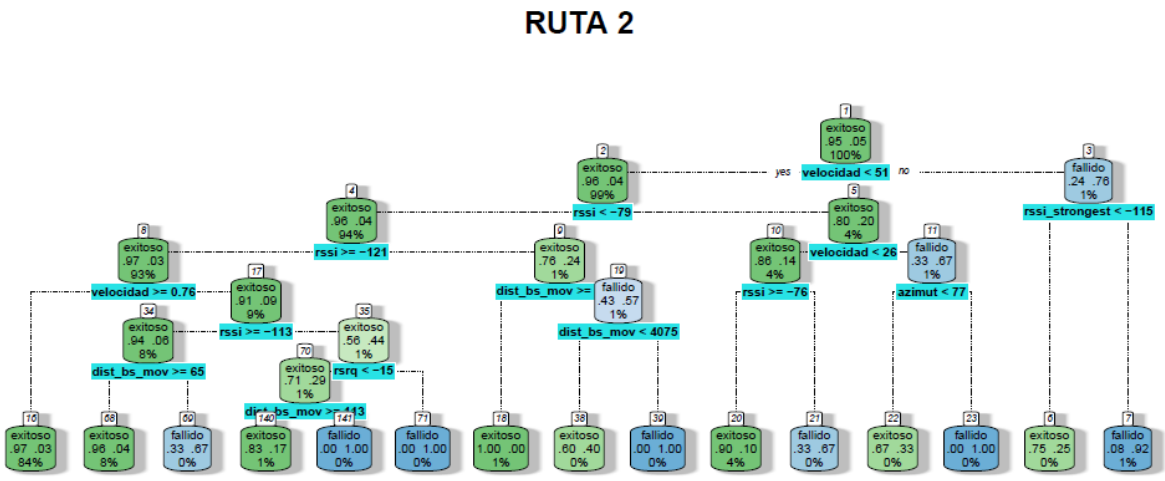

Figura 3.7 Árbol de decisión -Ruta 2

En la figura 3.7, el árbol de decisión muestra el estado del proceso de handover; en este caso el primer parámetro de comparación es la velocidad del UE, dado que la velocidad promedio del recorrido es de 51 km/h, se observa que si se sobrepasa esa velocidad

existe gran probabilidad que el handover falle., además, si simultáneamente el rssi_strongest es mayor a – 115 dBm aumenta la probabilidad de falla del handover.

De la totalidad de los datos el 95% de ellos resulta como “exitoso”.

La potencia reflejada en el parámetro RSSI también determina si el handover es exitoso y se tiene que con valores menores -79dBm y mayores a -121 dBm y si se mantiene una velocidad mayor de 0,76 km/h, el cual correspondería al movimiento normal de una persona, se tiene un handover “exitoso”. Además, se obtuvo que, si el móvil alcanza una velocidad menor 26 km/h, alcanzando a 4% de los datos, existe alta probabilidad de que el handover sea exitoso.

A continuación, se resumen los datos más relevantes en la siguiente tabla 3.2, tomando en cuenta el número total de nodos y el árbol de decisión en forma de texto generado por R (ANEXO V):

Tabla 3.2 Resumen del modelo del árbol de decisión de la Ruta 2

Categoría (Handover)	**Distancia eNB-UE**	**RSSI(dBm)**	**RSSI_Strongest (dBm)**	**RSRQ(dB)**	**Azimut (grados)**	**Velocidad (km/h)**
Exitoso	<113 m	>-79	>-121	<-15	<76.83	<0.26
Fallido	>4075 m	<-113	<-121	>-15	>=76.83	>51

3.1.3.1 Matriz de confusión Ruta 2

Al analizar la matriz de confusión de la Ruta 2, de la figura 3.8, se observa que de un total de 284 datos utilizados para prueba en la categoría de “exitoso”, 273 datos son acertados, por el contrario de 6 datos que fueron utilizados en la categoría de “fallido”, 5 han sido acertados. Por lo tanto, se deduce que más del 50% de los datos recolectados son acertados.

```
#Matriz Ruta 2
matriz
        prueba
         exitoso fallido
exitoso     273       1
fallido      11       5
```

Figura 3.8 Matriz de confusión -Ruta2

3.1.3.2 Métricas de evaluación del modelo Ruta 2

Otros valores de importancia son las métricas de evaluación, donde el lenguaje en R permite obtener la exactitud del modelo, como se observa en la figura 3.9, en donde se

puede ver que la exactitud alcanza un porcentaje que corresponde al 95,8%. Obteniendo los valores de las demás métricas calculado con las ecuaciones del segundo capítulo sensibilidad del 96,12%, precisión del 99,63% y especificidad del 83,33%.

```
print(paste("Exactitud del modelo",accuracy_test))
] "Exactitud del modelo 0.958620689655172"
```

Figura 3.9 Exactitud del modelo-Ruta2

3.1.3.3 Estimación de exactitud y Curva ROC-Ruta2

En la figura 3.10, se observa un valor en promedio de 73,2% de exactitud y un valor de 81,3% que se utilizará más adelante para una comparativa entre modelos.

```
  #resultado de métricas
  collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.732    25  0.0189 Preprocessor1_Model1
roc_auc  binary     0.813    25  0.0196 Preprocessor1_Model1
```

Figura 3.10 ROC y Exactitud- Árbol de decisión- Ruta2

3.1.3.4 Modelo de clasificación del vecino más cercano (KNN)-Ruta 2

Se obtuvo una minimización de la tasa de error, es de 0.40, lo que indica que la tasa de clasificación incorrecta es del 40%. En otras palabras, el modelo clasifica correctamente el 60% de las observaciones (ANEXO III).

Para conocer el rendimiento del modelo su evaluación se obtiene con las métricas de la exactitud y curva ROC que se muestran en la figura 3.11:

```
  #Metricas ruta 2
collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.602    25  0.0255 Preprocessor1_Model1
roc_auc  binary     0.684    25  0.0269 Preprocessor1_Model1
```

Figura 3.11 ROC y Exactitud- KNN- Ruta2

Al comparar ambos valores se aprecia que el modelo del árbol de decisiones tiene una mayor exactitud promedio de 73,2%, respecto que el modelo del vecino más cercano con un 60,2% de exactitud promedio, y de forma similar el valor de estimación ROC de 81,4% y de 68,4% para el KNN parala Ruta2, por lo cual, el modelo del árbol de decisiones que obtiene mejores resultados al clasificar los datos es el del vecino más cercano (KNN), tomando en cuenta que dicho modelo puede trabajar con dos estados de salida posibles como se dio en nuestro caso (“exitoso”,”fallido”).

3.1.4 Árbol de decisión Ruta 3

Se genera el árbol de decisión y se obtiene la siguiente figura.

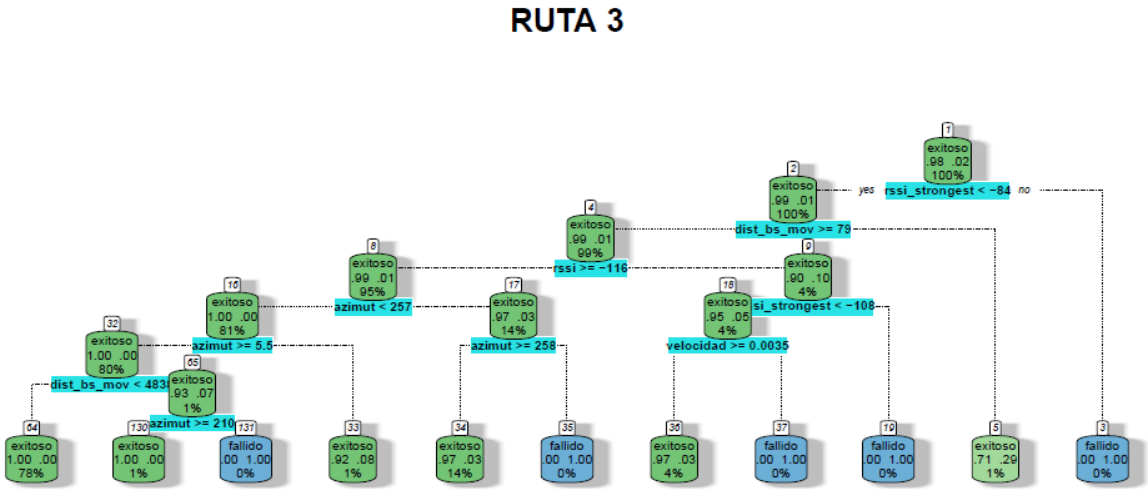

Figura 3.12 Árbol de decisión -Ruta 3

En la figura 3.12, el árbol de decisión toma en cuenta como parámetro de mayor prioridad el rssi_strongest, se observa que si la potencia de la celda vecina alcanza un valor mayor de -84 dBm el handover es fallido en todos los casos de acuerdo a los datos recolectados. El siguiente parámetro relevante esla potencia RSSI, en donde se observa que con valores mayores e iguales a -116 dBm y distancias entre estación móvil y la estación base menores a 4830 m se tiene un handover “exitoso”.

Se resumen los resultados más relevantes en la tabla 3.3, tomando en cuenta el número total de nodos y el árbol de decisión en forma de texto generado por R (ANEXO VI):

Tabla 3.3 Resumen del modelo del árbol de decisión de la Ruta 3

Categoría (Handover)	**Distancia eNB-UE**	**RSSI(dBm)**	**RSSI_Strongest (dBm)**	**RSRQ(dB)**	**Azimut (grados)**	**Velocidad (km/h)**
Exitoso	<4838 m	>-116	<-84	>=-18	<218.78	<0.0036
Fallido	>4838 m	<-116	>-108	<=-17	>218.78	>16

3.1.4.1 Matriz de confusión Ruta 3

Del análisis de la matriz de confusión de la Ruta 3, presentado en la figura 3.13, se observa que de un total de 252 datos utilizados para prueba en la categoría de “exitoso”, 252 datos son acertados, por el contrario de 5 datos que fueron utilizados en la categoría

de “fallido”, 4 han sido acertados. Por lo tanto, se puede concluir que más del 80% de los datos tomados son acertados.

```
#Matriz de confusión Ruta3
matriz
          prueba
           exitoso fallido
exitoso     252       4
fallido       0       1
```

Figura 3.13 Matriz de confusión -Ruta3

3.1.4.2 Métricas de evaluación del modelo Ruta 3

Otros valores de importancia son las métricas de evaluación que el lenguaje en R permite obtener como es la exactitud del modelo que se observa en la figura 3.14, donde la exactitud en porcentaje corresponde 98,4%. Obteniendo los valores de las demás métricas calculado con las ecuaciones del segundo capítulo de sensibilidad del 100%, precisión del 98,43% y especificidad del 20%, debido al poco número de verdaderos negativos.

```
print(paste("Exactitud del modelo",accuracy_test))
] "Exactitud del modelo 0.98443579766537"
```

Figura 3.14 Exactitud del modelo del árbol de decisión de la Ruta3

3.1.4.3 Estimación de exactitud y Curva ROC-Ruta3

En la figura 3.15, se observa un valor en promedio de 70% de exactitud y un valor de 64,5% de estimación.

```
  #Extraccion métricas Ruta 3
  collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.7      25  0.0595 Preprocessor1_Model1
roc_auc  binary     0.645    25  0.0702 Preprocessor1_Model1
```

Figura 3.15 ROC y Exactitud- Árbol de decisión- Ruta3

3.1.4.4 Modelo de clasificación del vecino más cercano (KNN)-Ruta 3

Se obtuvo una minimización de la tasa de error, es de 0.18, lo que indica que la tasa de clasificación incorrecta es del 18%. En otras palabras, el modelo clasifica correctamente el 82% de las observaciones (ANEXOIII).

El resultado de la evaluación del modelo se muestra en la siguiente figura 3.16:

```
  collect_metrics()
A tibble: 2 x 6
.metric  .estimator  mean     n std_err .config
<chr>    <chr>      <dbl> <int>   <dbl> <chr>
accuracy binary     0.775    20  0.0358 Preprocessor1_Model1
roc_auc  binary     0.844    20  0.0461 Preprocessor1_Model1
```

Figura 3.16 ROC y Exactitud- KNN- Ruta3

Al comparar ambos valores se aprecia que el modelo del vecino más cercano tiene una mayor estimación de exactitud que el modelo del árbol de decisiones para la Ruta3, donde se obtuvo un valor de estimación de 84,4%, y estimación ROC del 77,5% por cual, el modelo que obtiene mejores resultados en el proceso de clasificación de datos es el del vecino más cercano (KNN), frente al 64,5% y estimación ROC del 70% obtenido en el árbol de decisión para la misma ruta.

3.2 CONCLUSIONES

En este trabajo se presenta una propuesta sobre la recopilación, depuración y posterior análisis de información acerca de las condiciones del traspaso de señal entre estaciones base de una zona geográfica delimitada.

El proceso de captura de datos se ejecutó mediante aplicaciones de libre acceso y funcionando al mismo tiempo mientras se realizaba una llamada de VoIP durante el recorrido de las rutas especificadas. Se configuraron las aplicaciones para el registro de la ubicación y se realizó la recolección de datos en tiempo real.

El árbol de decisión general de las 3 rutas seleccionadas en la zona de Calderón permitió escoger los atributos de entrada, los cuales son: la potencia rssi, rsrq, rssi_strongest, velocidad del terminal móvil, azimut y la distancia de la estación base hacia el UE, y se comprobó que los parámetros de mayor importancia son la velocidad del móvil, el azimut, la potencia RSSI y la distancia de la estación base al móvil.

Al analizar la ruta 1, correspondiente a la zona de Carapungo, se concluye que como factores predominantes en el traspaso de la señal se encuentran la distancia del UE hacia la estación base y la intensidad de la señal recibida; se debe mencionar que esta zona es muy concurrida. Además, esta zona cuenta con estaciones base muy cercanas una de otra y por tanto existe alta probabilidad de la ejecución de handover donde el vecino más cercano obtuvo una mejor exactitud promedio del 65,5%; con una estimación ROC de 66,6% frente al árbol de decisión con exactitud promedio del 64%, y estimación ROC del 66,2%, donde el ROC es un indicador de sensibilidad, por lo cual, el modelo que obtiene mejores predicciones es el del vecino más cercano (KNN)

En la ruta 2, que corresponde a las zonas de San José de Morán y San Juan de Calderón, los factores de decisión es el RSSI_Strongest, el RSSI y la velocidad, debido a que es la ruta intermedia de la zona seleccionada, por lo tanto, las estaciones base vecinas (se encuentran en rutas diferentes) tienen efecto en la señal portadora de las celdas servidora, causando interferencias y disminuyendo el margen de efectividad del handover en esta ruta. Al comparar los resultados, el modelo del árbol de decisiones tiene una mayor exactitud promedio de 73,2%, respecto al modelo del vecino más cercano con un 60,2% de exactitud promedio, y de forma similar el valor de estimación ROC(sensibilidad) de 81,4% y de 68,4% para el KNN parala Ruta2, por lo cual, el modelo del árbol de decisiones que obtiene mejores resultados al clasificar los datos.

Asimismo, se analizó la ruta 3 correspondiente al sector de San Juan de Calderón y Marianas, donde los parámetros predominantes son la potencia de la estación vecina RSSI_Strongest y la distancia hacia la estación base, debido a la creciente población y demanda del servicio, la degradación del handover puede darse en el caso donde los nodos que brindan servicio están conectados a una gran cantidad de usuarios. Cabe destacar que este lugar contiene una menor cantidad de estaciones base de la operadora y existen lugares con gran cantidad de obstáculos donde la potencia es mínima y la señal disminuye. Donde, el modelo del vecino más cercano tiene una mayor estimación de exactitud que el modelo del árbol de decisiones para la Ruta3, obteniendo un valor de 84.4%, y estimación ROC del 77,5% frente al 64,5% y estimación ROC del 70% obtenido en el árbol de decisión; por cual, el modelo que obtiene mejores resultados en el proceso de clasificación de datos es el del vecino más cercano (KNN) gracias un mayor valor de estimación de sensibilidad y de exactitud.

3.3 RECOMENDACIONES

Se recomienda que en un estudio futuro se prueben los resultados de este trabajo para la implementación de técnicas que permitan mejorar la efectividad de la ejecución del proceso de handover.

Se recomienda la captura de la mayor cantidad de datos en las mismas rutas y a diferentes horas del día su fuere el caso, movilizándose en automóvil en algunos días y en otros en forma estática o caminando hacia las estaciones base, debido que la velocidad es importante para determinar de manera precisa la zona geográfica del handover.

Se recomienda utilizar un dispositivo móvil de 4 a 8 Gb de memoria ram, debido a que las mediciones se realizan en tiempo real con las tres aplicaciones recolectando de forma simultánea.

4 REFERENCIAS BIBLIOGRÁFICAS

[1] C. Rocha, «Propuesta de un Modelo Predictivo para Identificar Factores que se Manifiestan en las Fallas de Handover en Redes LTE Utilizando Mediciones de Campo y un Árbol de Decisión en R Studio (Caso de Estudio Voip)», Escuela Politécnica Nacional, 2022.

[2] H. Navarrete, «Estudio de las Regiones Limítrofes de Handover en la Parroquia Rural de Calderón con la Ayuda de la Herramienta R Aplicando Técnicas de Machine Learning», Escuela Politécnica Nacional, 2022.

[3] J. M. Huidobro y J. Luque Ordóñez., *Comunicaciones por radio : tecnologías, redes y servicios de telecomunicaciones. El espectro electromagnético*, Primera edición. México : Alfaomega, 2014.

[4] Mohamed Raafat OMRI محمد رأفت عمري, «Handover Parameters Self-optimization by Q-Learning in 4G Networks», 12:39:00 UTC. Accedido: 23 de febrero de 2023. [En línea]. Disponible en: https://www.slideshare.net/raafatomri/handover-parameters-selfoptimization-by-qlearning-in-4g-networks-64309021?from_action=save

[5] W. Khalid y K. S. Kwak, «Handover optimization in femtocell networks», *2013 International Conference on ICT Convergence (ICTC)*, pp. 122-127, oct. 2013, doi: 10.1109/ICTC.2013.6675321.

[6] «Cellular tower maps», *CellMapper*. https://www.cellmapper.net/First_Time_Startup/en, https://www.cellmapper.net/it (accedido 8 de diciembre de 2022).

[7] «Descripción de Netmonitor. Netmonitor cómo utilizar la aplicación. Software de monitoreo de red para teléfonos inteligentes basado en el sistema operativo Android. Quién puede beneficiarse del menú de servicio». https://jeraff.ru/es/netmonitor-opisanie-netmonitor-kak-polzovatsya-prilozheniem-programmy/ (accedido 29 de julio de 2022).

[8] «Manual g-nettrack – gyokov solutions». https://gyokovsolutions.com/manual-g-nettrack/#appfolder (accedido 8 de diciembre de 2022).

[9] «Samsung Galaxy A01 - Full phone specifications». https://www.gsmarena.com/samsung_galaxy_a01-9999.php (accedido 16 de febrero de 2023).

[10] «Samsung Galaxy A01 - Specs», 8 de abril de 2020. https://www.phonemore.com/specs/samsung/galaxy-a01/ (accedido 16 de febrero de 2023).

[11] R. Dominic y S. N. Roberto, *Introducción a los SIG con R*. Prensas de la Universidad de Zaragoza, 2019.

[12] R. Russell, *Machine Learning guia paso a paso para implementar algoritmos de Machine Learning con python*. Copyright 2018 Rudolph Russell. Accedido: 22 de febrero de 2023. [En línea]. Disponible en: https://ricardollarves.com.ve/wp-content/uploads/2020/07/Machine-Learning-Gu%C3%ADa-Paso-a-Paso-Para-Implementar-Algoritmos-De-Machine-Learning-Con-Python-by-Rudolph-Russell.pdf

[13] «RSRP and RSRQ - Teltonika Networks Wiki». https://wiki.teltonika-networks.com/view/RSRP_and_RSRQ (accedido 11 de marzo de 2022).

[14] «RSSI - Teltonika Networks Wiki». https://wiki.teltonika-networks.com/view/RSSI (accedido 11 de marzo de 2022).

[15] «SINR - Teltonika Networks Wiki». https://wiki.teltonika-networks.com/view/SINR (accedido 11 de marzo de 2022).

[16] R. Valor, Y. Torres, y A. Osman, «Identificador físico de celda para redes LTE: recomendaciones para la planificación y asignación», *Revista INGENIERÍA UC*, vol. 25, n.º 2, 2018, Accedido: 11 de marzo de 2022. [En línea]. Disponible en: https://www.redalyc.org/journal/707/70757669012/html/

[17] «Las “siete maravillas” de la parroquia Calderón y su lado oscuro», *Primicias*. https://www.primicias.ec/noticias/sociedad/siete-maravillas-parroquia-calderon-lado-oscuro/ (accedido 19 de febrero de 2023).

[18] «Título: UNIVERSIDAD CENTRAL DEL ECUADOR FACULTAD DE ARQUITECTURA Y URBANISMO CARRERA DE ARQUITECTURA». https://www.google.com/imgres (accedido 24 de febrero de 2023).

[19] J. ZAMORANO RUIZ, «COMPARATIVA Y ANÁLISIS DE ALGORITMOS DE APRENDIZAJE AUTOMÁTICO PARA LA PREDICCIÓN DEL TIPO PREDOMINANTE DE CUBIERTA ARBÓREA», UNIVERSIDAD COMPLUTENSE DE MADRID, 2018, pp. 52-56. Accedido: 22 de febrero de 2023. [En línea]. Disponible en: https://eprints.ucm.es/id/eprint/48800/1/Memoria%20TFM%20Machine%20Learning_Juan_Zamorano_para_difundir%20(2).pdf

[20] A. Bronshtein, «A quick introduction to k-nearest neighbors algorithm», *Medium*, 6 de mayo de 2019. https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7 (accedido 13 de enero de 2023).

[21] C. Gray, «Decision tree hugging», *Medium*, 6 de junio de 2017. https://towardsdatascience.com/decision-tree-hugging-b8851f853486 (accedido 13 de enero de 2023).

[22] «What Is a Confusion Matrix in Machine Learning?», *Plat.AI*, 18 de agosto de 2022. https://plat.ai/blog/confusion-matrix-in-machine-learning/ (accedido 17 de febrero de 2023).

[23] A. Tripathi, «What is Confusion Matrix in ML - Elements, Examples & More», *Blogs & Updates on Data Science, Business Analytics, AI Machine Learning*, 14 de noviembre de 2022. https://www.analytixlabs.co.in/blog/confusion-matrix/ (accedido 16 de febrero de 2023).

[24] A. Mitrani, «Evaluating Categorical Models II: Sensitivity and Specificity», *Medium*, 6 de diciembre de 2019. https://towardsdatascience.com/evaluating-categorical-models-ii-sensitivity-and-specificity-e181e573cff8 (accedido 24 de febrero de 2023).

[25] DOBLEVIA, «Rumbo y Azimut», *Doble Vía*, 19 de marzo de 2007. https://doblevia.wordpress.com/2007/03/19/rumbo-y-azimut/ (accedido 27 de febrero de 2023).

[26] «Understanding Azimuth and Elevation», *PhotoPills*. https://www.photopills.com/articles/understanding-azimuth-and-elevation (accedido 27 de febrero de 2023).

[27] Unknown, «Topografía IUPSM Ing. Civil: RUMBO Y AZIMUT», *Topografía IUPSM Ing. Civil*, 12 de noviembre de 2012. http://topografiadeobrasciviles.blogspot.com/2012/11/rumbo-y-azimut.html (accedido 27 de febrero de 2023).

[28] «CALCULO 2 - LEY DE HARVESIAN - La fórmula del semiverseno es una importante ecuación para la - Studocu». https://www.studocu.com/co/document/politecnico-grancolombiano/calculo-ii/calculo-2-ley-de-harvesian/25158062 (accedido 27 de febrero de 2023).

5 ANEXOS

ANEXO I. CRONOGRAMA DE ACTIVIDADES

Semana referencial / Etapas	**Fecha inicio-fin (si aplica)**	**ACTIVIDADES**	**Resultado esperado (si aplica)**
1	2022-11-26 2022-12-26	Realización de mediciones	Herramientas de mediciones
2	2022-10-11 2022-11-26	Recopilación de la información	Consulta de bibliografía

3	2022-12-26 2023-01-10	Análisis de datos y Selección de los datos que serán de utilidad para el análisis del componente seleccionado	Presentación de avances
4	2022-12-26 2023-01-15	Escritura del Capítulo 1	
5	2022-12-26 2023-01-08	Implementación de algoritmo básico	Software a realizar la implementación
6	2023-01-09 2023-01-19	Aplicación de algoritmo	
7	2023-01-20 2023-01-28	Escritura del Capítulo 2	
8	2023-01-29 2023-02-10	Análisis del algoritmo implementado	
9	2023-02-11 2023-02-15	Buscar fuentes y referencias relacionadas al componente y relacionarlas a los resultados esperados	
10	2023-02-15 2023-02-01	Comparación de resultados	Presentación de avances
11	2023-02-01 2023-02-18	Escritura del Capítulo 4	Evaluación de resultados
12	2023-02-19 2023-02-19	Colocación de conclusiones, recomendaciones y bibliografía	
13	2023-02-20 2023-02-25	Revisión del documento	

ANEXO II. CÓDIGO IMPLEMENTADO ÁRBOL DE DECISIONES

```
##ESCUELA POLITÉCNICA NACIONAL
#TRABAJO DE INTEGRACIÓN CURRICULAR
#MODELO PREDICTIVO COM ML DEL PROCESO DE HANDOVER
#JONATHAN RAMIRO VILLARREAL TAYÁN
#########################################################

#Liberias de manejo de archivos
library("openxlsx")#Permite abrir los archivos .xlsx (excel)
library("readxl")#Permite leer los archivos .xlsx
library("dplyr")#Permite el manejo de datos en tramas de datos

#Librerías de procesamiento de datos
library(recipes)#permite realizar una receta de los datos, creación y edición
```

```
library(rsample)#Permite una validación cruzada y separacion de datos en conj. de entreamiento y prueba
library(themis)#Permite proporcionar y manejar datos faltantes
library(dplyr)

#Libraria de evaluacion de los datos
library(workflows)#Proporciona una forma estructurada de realizar un análisis
library(parsnip)#Permite una interfaz unificada para crear un modelo
library(tune)#Permite la optimización en modelos de aprendizaje automático
library(yardstick)#Permite evaluar el rendimiento del modelo
library(dials)#Permite un ajuste de los parámetros del modelo
library(rpart)#Permite construir arboles de decision
library(kknn)#Permite el aprendizaje del vecino más cercano
library(data.table)#Permite manipular datos eficientemente
#Libreria de modelado
library(dplyr)#hace funcionar %>% para disminuir el tiempo de desarrollo
library(tidyverse)
library(tidymodels)#Ayuda a seleccionar el modelo
library(rpart.plot)#grafica el árbol de decisión
library(rattle)
library(RColorBrewer)



#Eleccion de la base de datos
file.choose()

#carga de la base de datos para ruta
r1data1<-read_xlsx("C:\\Users\\Desktop-DTB\\Desktop\\mediciones\\Procesamiento\\Arboles_ruta_1_2_3\\Ruta1_2_3.xlsx")
r1data2<-mutate(r1data1,margen_potencia=rssi-rssi_strongest,dia=day)
#Creacion de nuevas variables en la base de datos

###Añade columna de calidad de RX
r1data2$calidad_rx<-"NA"
r1data2$calidad_rx[r1data2$rsrq>=(-5)]<-"Excelente"
r1data2$calidad_rx[r1data2$rsrq<(-5)&r1data2$rsrq>=(-9)]<-"Media"
```

```
r1data2$calidad_rx[r1data2$rsrq<(-9)]<-"Mala"
############
#Cambio del formato de hora a entero
r1data2$sys_hour <- strptime(r1data2$sys_hour, format = "%H:%M:%S")
r1data2$sys_hour <- format(as.POSIXct(r1data2$sys_hour), format = "%H")
r1data2$sys_hour <- as.integer(r1data2$sys_hour)
r1data2
#APLICACION MODELO ML

#Técnica del Árbol de decision-RUTA1
##############################################
#CREAMOS UNA SEMILLA
set.seed(1234)
#creamos una division aleatoria por defecto es del 75% para entrenamiento y 25% para
prueba
r1data2_split<-initial_split(r1data2)

#CREAMOS LA BASE DE ENTRENAMIENTO Y DE PRUEBA
r1data2_train<-training(r1data2_split)
r1data2_test<-testing(r1data2_split)

#procesamiento de los datos para el modelo
r1data2_rec<-recipe(handover_2~.,data=r1data2_train)%>%
  themis::step_downsample(handover_2)%>%
  recipes::step_dummy(all_nominal(),-all_outcomes())%>%
  recipes::step_zv(all_numeric())%>%
  recipes::step_normalize(all_numeric())%>%
  recipes::prep()
test_proc1<-bake(r1data2_rec,new_data=r1data2_test)
test_proc1
#EXTRAEMOS LA DATA PARA LA RECETA
juice(r1data2_rec)%>%
  count(handover_2)

###ESPEFICACION DEL
#ARBOL DE DECISION
```

```
tree_spec<-decision_tree()%>%
  set_engine("rpart")%>%#ESPECIFICACION QUE ES UN ARBOL DE CLASIFICACION
  set_mode("classification")

#Ajuste del modelo
tree_fit<-tree_spec%>%
  fit(handover_2~.,data=juice(r1data2_rec))
tree_fit
r1data2_rec

#REVISION DE LA BASE DE DATOS
head(r1data2_train)
count(r1data2_train)
#Seleccion de las columnas a en el dataframe
#r2data2_train<-
select(r2data2_train,c(node_id,rsrq,rssnr,rssi_strongest,handover_2,azimut,velocidad,dist_bs_mov,margen_potencia,dia,calidad_rx,eNB28001,eNB28088,eNB28411,eNB28816,eNB28845,eNB28183))
#r1data2_train<-
select(r1data2_train,c(hora,node_id,rssnr,rssi,rssi_strongest,handover_2,Azimut,velocidad,dist_bs_mov,dia,calidad_rx,eNB28318,eNB28618,eNB28239,eNB28457,eNB28836,eNB28732))
r1data2_train<-
select(r1data2_train,c(rsrq,rssi,rssi_strongest,azimut,handover_2,velocidad,dist_bs_mov))
str(r1data2_train)

#r2data2_train$sys_hour<-strptime(r2data2_train$sys_hour,format = "%H:%M:%S")
#r2data2_train$sys_hour<-format(as.POSIXct(r2data2_train$sys_hour),format = "%H")
#r2data2_train$sys_hour<-as.integer(r2data2_train$sys_hour)
r1data2_train
#ENTRENAMIENTO
create_train_test<-function(data,size=0.8,train=TRUE){
  n_row=nrow(data)
  r1data2_train_row=size*n_row
```

```
  train_sample<-1:r1data2_train_row
  if(train==TRUE){
    return(data[train_sample,])
  }else{
    return(data[-train_sample,])
  }
}
############################################################333
#MUESTREO DE LA MATRIZ PRINCIPAL
r1data2_train<-r1data2_train[sample(1:nrow(r1data2_train)),]
r1data2_train
#####################################################################33
#ESPECIFICACION DE LOS DATOS DE 80% DE ENTRENAMIENTO Y 20% DE PRUEBA
train<-create_train_test(r1data2_train,0.8,train = TRUE)
test<-create_train_test(r1data2_train,0.8,train=FALSE)
test
train
#############################################################3333
#GENERACION DEL ARBOL DE DECISION
fit<-rpart(handover_2~.,data = train,method = "class",minsplit=1,minbucket=1)
fit
prueba<-predict(fit,test,type="class")
#####################################################33
#GRAFICO DEL ARBOL DE DECISION
fancyRpartPlot(fit,cex=0.4,caption = NULL,yshift=1.2,main = "ÁRBOL GENERAL (RUTAS 1-2-3)",split.box.col=5,gap=12, space=0.1,split.cex = 1)
#####################################################
#EXACTITUD DEL MODELO
accuracy_test<-sum(diag(matriz))/sum(matriz)
#EXACTITUD DEL MODELO
print(paste("Exactitud del modelo de clasificación",accuracy_test))
#CREACION DE LA MATRIZ DE CONFUSION
matriz<-table(test$handover_2,prueba)
#Matriz de confusión Ruta 1
matriz
```

```
#EVALUACION DEL RENDIMIENTO
set.seed(1234)
validation_split<-mc_cv(juice(r1data2_rec),prop = 0.9,strata = handover_2)
validation_split
tree_res<-
tune::fit_resamples(tree_spec,handover_2~.,validation_split,control=control_resample
s(save_pred = TRUE))
tree_res%>%
  #Extracción métricas-Ruta1
  collect_metrics()
```

ANEXO III. CÓDIGO IMPLEMENTADO Y RESULTADOS VECINO MÁS CERCANO KNN

```
##Técnica del vecino más cercano-RUTA2

library("openxlsx")#Permite abrir los archivos .xlsx (excel)
library("readxl")#Permite leer los archivos .xlsx
library("dplyr")#Permite el manejo de datos en tramas de datos

#Librerías de procesamiento de datos
library(recipes)#permite realizar una receta de los datos, creación y edición
library(rsample)#Permite una validación cruzada y separacion de datos en conj. de
entreamiento y prueba
library(themis)#Permite proporcionar y manejar datos faltantes
library(dplyr)



#Libraria de evaluacion de los datos
library(workflows)#Proporciona una forma estructurada de realizar un análisis
library(parsnip)#Permite una interfaz unificada para crear un modelo
library(tune)#Permite la optimización en modelos de aprendizaje automático
library(yardstick)#Permite evaluar el rendimiento del modelo
library(dials)#Permite un ajuste de los parámetros del modelo
library(rpart)#Permite construir arboles de decision
library(kknn)#Permite el aprendizaje del vecino más cercano
library(data.table)#Permite manipular datos eficientemente
```

```
file.choose()#Encontramos la ruta del archivo

#Carga de los datos de la ruta
r2data1<-read_xlsx("C:\\Users\\Desktop-
DTB\\Desktop\\mediciones\\Procesamiento\\Arboles_ruta2\\Ruta2_data.xlsx")
r2data2<-mutate(r2data1,margen_potencia=rssi-rssi_strongest,dia=dia)
###Añade columna de calidad de RX
r2data2$calidad_rx<-"NA"
r2data2$calidad_rx[r2data2$rsrq>=(-5)]<-"Excelente"
r2data2$calidad_rx[r2data2$rsrq<(-5)&r2data2$rsrq>=(-9)]<-"Media"
r2data2$calidad_rx[r2data2$rsrq<(-9)]<-"Mala"
#APLICACION MODELO ML
##############################################

#CREAMOS UNA SEMILLA
set.seed(1234)
#creamos una division aleatoria por defecto es del 75% para entrenamiento y 25% para
prueba
r2data2_split<-initial_split(r2data2)

#CREAMOS LA BASE DE ENTRENAMIENTO Y DE PRUEBA
r2data2_train<-training(r2data2_split)
r2data2_test<-testing(r2data2_split)

#r2data2_rec<-
recipe(sys_time~.,data=r2data2_train,sys_hour~.,data=r2data2_train,day~.,data=r2da
ta2_train)
r2data2_rec<-recipe(handover_2~.,data=r2data2_train)%>%
  themis::step_downsample(handover_2)%>%
  recipes::step_dummy(all_nominal(),-all_outcomes())%>%
  recipes::step_zv(all_numeric())%>%
  recipes::step_normalize(all_numeric())%>%
  recipes::prep()
```

```
test_proc1<-bake(r2data2_rec,new_data=r2data2_test)
test_proc1
#EXTRAEMOS LA DATA PARA LA RECETA
juice(r2data2_rec)%>%
  count(handover_2)

###ESPEFICACION DEL VECINO MAS CERCANO
kknn_spec<-nearest_neighbor()%>%
set_engine("kknn")%>%
set_mode("classification")  #Especificacion del modelo
##Se ajusta el modelo
library(dplyr)
kknn_fit<-kknn_spec%>%
fit(handover_2~.,data=juice(r2data2_rec))
kknn_fit

############################################################
##EVALUACION DEL MODELO
#set.seed(1234)
validation_split<-mc_cv(juice(r2data2_rec),prop = 0.9,strata = handover_2)
validation_split
#EVALUACION DEL RENDIMIENTO
kknn_res<-
tune::fit_resamples(kknn_spec,handover_2~.,validation_split,control=control_resampl
es(save_pred = TRUE))
kknn_res%>%
  #Metricas ruta 2
collect_metrics()
################################################
```

RESULTADOS DEL KNN

```
> ###KNN-RUTA1
> kknn_fit
parsnip model object


Call:
kknn::train.kknn(formula = handover_2 ~ ., data = data, ks = min_rows(5,     data, 5))

Type of response variable: nominal
Minimal misclassification: 0.28
Best kernel: optimal
Best k: 5
```

```
> ###KNN-RUTA2
> kknn_fit
parsnip model object


Call:
kknn::train.kknn(formula = handover_2 ~ ., data = data, ks = min_rows(5,     data, 5))

Type of response variable: nominal
Minimal misclassification: 0.4038462
Best kernel: optimal
Best k: 5
```

```
> ###KNN-RUTA3
> kknn_fit
parsnip model object


Call:
kknn::train.kknn(formula = handover_2 ~ ., data = data, ks = min_rows(5,     data, 5))

Type of response variable: nominal
Minimal misclassification: 0.1818182
Best kernel: optimal
Best k: 5
```

ANEXO IV. ÁRBOL DE DECISIONES EN FORMA TEXTUAL -RUTA 1

```
##árbol de decisiones en forma textual-ruta1
> fit
n= 909

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 909 81 exitoso (0.910891089 0.089108911)
    2) dist_bs_mov>=111.7083 893 73 exitoso (0.918253080 0.081746920)
      4) rssi< -10.5 834 57 exitoso (0.931654676 0.068345324)
        8) rsrq>=-14.5 316 10 exitoso (0.968354430 0.031645570)
         16) azimut< 185.5434 185  1 exitoso (0.994594595 0.005405405) *
         17) azimut>=185.5434 131  9 exitoso (0.931297710 0.068702290)
           34) dist_bs_mov>=115.0996 130  8 exitoso (0.938461538 0.061538462)
             68) azimut>=185.6775 129  7 exitoso (0.945736434 0.054263566)
              136) rssi_strongest>=-110.5 106  3 exitoso (0.971698113 0.028301887) *
              137) rssi_strongest< -110.5 23  4 exitoso (0.826086957 0.173913043)
                274) dist_bs_mov>=257.9533 15  0 exitoso (1.000000000 0.000000000) *
                275) dist_bs_mov< 257.9533 8  4 exitoso (0.500000000 0.500000000)
                  550) rsrq< -13.5 3  0 exitoso (1.000000000 0.000000000) *
                  551) rsrq>=-13.5 5  1 fallido (0.200000000 0.800000000)
```

```
            1102) azimut< 191.389 1  0 exitoso (1.000000000 0.000000000) *
            1103) azimut>=191.389 4  0 fallido (0.000000000 1.000000000) *
         69) azimut< 185.6775 1  0 fallido (0.000000000 1.000000000) *
      35) dist_bs_mov< 115.0996 1  0 fallido (0.000000000 1.000000000) *
   9) rsrq< -14.5 518 47 exitoso (0.909266409 0.090733591)
   18) velocidad< 28.17339 443 33 exitoso (0.925507901 0.074492099)
      36) rssi_strongest< -103.5 355 20 exitoso (0.943661972 0.056338028) *
      37) rssi_strongest>=-103.5 88 13 exitoso (0.852272727 0.147727273)
         74) rssi_strongest>=-102.5 71  7 exitoso (0.901408451 0.098591549)
         148) dist_bs_mov< 1424.392 67  5 exitoso (0.925373134 0.074626866)
            296) rssi_strongest< -98.5 36  0 exitoso (1.000000000 0.000000000) *
            297) rssi_strongest>=-98.5 31  5 exitoso (0.838709677 0.161290323)
               594) rssi_strongest>=-97.5 26  1 exitoso (0.961538462 0.038461538) *
               595) rssi_strongest< -97.5 5  1 fallido (0.200000000 0.800000000)
                1190) rsrq< -17 1  0 exitoso (1.000000000 0.000000000) *
                1191) rsrq>=-17 4  0 fallido (0.000000000 1.000000000) *
         149) dist_bs_mov>=1424.392 4  2 exitoso (0.500000000 0.500000000)
            298) rsrq< -18 2  0 exitoso (1.000000000 0.000000000) *
            299) rsrq>=-18 2  0 fallido (0.000000000 1.000000000) *
         75) rssi_strongest< -102.5 17  6 exitoso (0.647058824 0.352941176)
         150) dist_bs_mov>=401.3724 9  1 exitoso (0.888888889 0.111111111)
            300) dist_bs_mov< 2228.406 8  0 exitoso (1.000000000 0.000000000) *
            301) dist_bs_mov>=2228.406 1  0 fallido (0.000000000 1.000000000) *
         151) dist_bs_mov< 401.3724 8  3 fallido (0.375000000 0.625000000)
            302) rsrq< -17.5 1  0 exitoso (1.000000000 0.000000000) *
            303) rsrq>=-17.5 7  2 fallido (0.285714286 0.714285714)
               606) velocidad< 0.2419351 3  1 exitoso (0.666666667 0.333333333)
                1212) azimut< 70.97366 2  0 exitoso (1.000000000 0.000000000) *
                1213) azimut>=70.97366 1  0 fallido (0.000000000 1.000000000) *
               607) velocidad>=0.2419351 4  0 fallido (0.000000000 1.000000000) *
   19) velocidad>=28.17339 75 14 exitoso (0.813333333 0.186666667)
      38) velocidad>=28.49084 73 12 exitoso (0.835616438 0.164383562)
         76) azimut< 268.1456 72 11 exitoso (0.847222222 0.152777778)
         152) rssi_strongest>=-109.5 54  5 exitoso (0.907407407 0.092592593) *
         153) rssi_strongest< -109.5 18  6 exitoso (0.666666667 0.333333333)
            306) rsrq>=-16.5 11  2 exitoso (0.818181818 0.181818182)
```

```
            612) dist_bs_mov>=290.5287 10  1 exitoso (0.900000000 0.100000000) *
            613) dist_bs_mov< 290.5287 1  0 fallido (0.000000000 1.000000000) *
          307) rsrq< -16.5 7  3 fallido (0.428571429 0.571428571)
            614) rssi< -114.5 4  1 exitoso (0.750000000 0.250000000)
             1228) azimut< 156.8741 3  0 exitoso (1.000000000 0.000000000) *
             1229) azimut>=156.8741 1  0 fallido (0.000000000 1.000000000) *
            615) rssi>=-114.5 3  0 fallido (0.000000000 1.000000000) *
        77) azimut>=268.1456 1  0 fallido (0.000000000 1.000000000) *
      39) velocidad< 28.49084 2  0 fallido (0.000000000 1.000000000) *
  5) rssi>=-10.5 59 16 exitoso (0.728813559 0.271186441)
   10) rssi_strongest>=-104 30  4 exitoso (0.866666667 0.133333333)
     20) rssi_strongest< -85.5 28  2 exitoso (0.928571429 0.071428571) *
     21) rssi_strongest>=-85.5 2  0 fallido (0.000000000 1.000000000) *
   11) rssi_strongest< -104 29 12 exitoso (0.586206897 0.413793103)
     22) rsrq< -16.5 11  1 exitoso (0.909090909 0.090909091)
       44) velocidad< 24.27228 10  0 exitoso (1.000000000 0.000000000) *
       45) velocidad>=24.27228 1  0 fallido (0.000000000 1.000000000) *
     23) rsrq>=-16.5 18  7 fallido (0.388888889 0.611111111)
       46) dist_bs_mov< 350.5871 5  1 exitoso (0.800000000 0.200000000)
         92) dist_bs_mov>=246.2789 4  0 exitoso (1.000000000 0.000000000) *
         93) dist_bs_mov< 246.2789 1  0 fallido (0.000000000 1.000000000) *
       47) dist_bs_mov>=350.5871 13  3 fallido (0.230769231 0.769230769)
         94) rssi_strongest>=-106.5 5  2 exitoso (0.600000000 0.400000000)
          188) azimut>=188.331 2  0 exitoso (1.000000000 0.000000000) *
          189) azimut< 188.331 3  1 fallido (0.333333333 0.666666667) *
         95) rssi_strongest< -106.5 8  0 fallido (0.000000000 1.000000000) *
3) dist_bs_mov< 111.7083 16  8 exitoso (0.500000000 0.500000000)
  6) rssi_strongest>=-89 4  0 exitoso (1.000000000 0.000000000) *
  7) rssi_strongest< -89 12  4 fallido (0.333333333 0.666666667)
   14) dist_bs_mov>=44.5632 8  4 exitoso (0.500000000 0.500000000)
     28) rssi< -94.5 2  0 exitoso (1.000000000 0.000000000) *
     29) rssi>=-94.5 6  2 fallido (0.333333333 0.666666667)
       58) azimut< 7.918646 3  1 exitoso (0.666666667 0.333333333)
        116) velocidad< 36.44296 2  0 exitoso (1.000000000 0.000000000) *
        117) velocidad>=36.44296 1  0 fallido (0.000000000 1.000000000) *
       59) azimut>=7.918646 3  0 fallido (0.000000000 1.000000000) *
```

```
      15) dist_bs_mov< 44.5632 4  0 fallido (0.000000000 1.000000000) *
```

ANEXO V. ÁRBOL DE DECISIONES EN FORMA TEXTUAL -RUTA 2

```
##árbol de decisiones en forma textual-ruta2
> fit
n= 1159

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 1159 62 exitoso (0.94650561 0.05349439)
    2) velocidad< 50.62217 1142 49 exitoso (0.95709282 0.04290718)
      4) rssi< -79.5 1092 39 exitoso (0.96428571 0.03571429)
        8) rssi>=-121.5 1075 35 exitoso (0.96744186 0.03255814)
         16) velocidad>=0.7551707 969 25 exitoso (0.97420021 0.02579979) *
         17) velocidad< 0.7551707 106 10 exitoso (0.90566038 0.09433962)
           34) rssi>=-113.5 97  6 exitoso (0.93814433 0.06185567)
             68) dist_bs_mov>=65.09438 94  4 exitoso (0.95744681 0.04255319) *
             69) dist_bs_mov< 65.09438 3  1 fallido (0.33333333 0.66666667) *
           35) rssi< -113.5 9  4 exitoso (0.55555556 0.44444444)
             70) rsrq< -15.5 7  2 exitoso (0.71428571 0.28571429)
              140) dist_bs_mov>=113.3401 6  1 exitoso (0.83333333 0.16666667) *
              141) dist_bs_mov< 113.3401 1  0 fallido (0.00000000 1.00000000) *
             71) rsrq>=-15.5 2  0 fallido (0.00000000 1.00000000) *
        9) rssi< -121.5 17  4 exitoso (0.76470588 0.23529412)
         18) dist_bs_mov>=4079.755 10  0 exitoso (1.00000000 0.00000000) *
         19) dist_bs_mov< 4079.755 7  3 fallido (0.42857143 0.57142857)
           38) dist_bs_mov< 4075.269 5  2 exitoso (0.60000000 0.40000000) *
           39) dist_bs_mov>=4075.269 2  0 fallido (0.00000000 1.00000000) *
      5) rssi>=-79.5 50 10 exitoso (0.80000000 0.20000000)
       10) velocidad< 25.588 44  6 exitoso (0.86363636 0.13636364)
         20) rssi>=-76 41  4 exitoso (0.90243902 0.09756098) *
         21) rssi< -76 3  1 fallido (0.33333333 0.66666667) *
       11) velocidad>=25.588 6  2 fallido (0.33333333 0.66666667)
         22) azimut< 76.83528 3  1 exitoso (0.66666667 0.33333333) *
```

```
        23) azimut>=76.83528 3  0 fallido (0.00000000 1.00000000) *
    3) velocidad>=50.62217 17  4 fallido (0.23529412 0.76470588)
      6) rssi_strongest< -115.5 4  1 exitoso (0.75000000 0.25000000) *
      7) rssi_strongest>=-115.5 13  1 fallido (0.07692308 0.92307692) *
```

ANEXO VI. ÁRBOL DE DECISIONES EN FORMA TEXTUAL -RUTA 3

```
##árbol de decisiones en forma textual-ruta3

fit
n= 1025

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 1025 16 exitoso (0.984390244 0.015609756)
    2) rssi< -80 1016 13 exitoso (0.987204724 0.012795276)
      4) velocidad>=0.003473583 997 10 exitoso (0.989969910 0.010030090)
        8) dist_bs_mov< 4733.706 976  8 exitoso (0.991803279 0.008196721)
         16) azimut>=5.461491 967  7 exitoso (0.992761117 0.007238883)
           32) dist_bs_mov>=298.7134 852  3 exitoso (0.996478873 0.003521127)
             64) rsrq>=-18.5 785  1 exitoso (0.998726115 0.001273885) *
             65) rsrq< -18.5 67  2 exitoso (0.970149254 0.029850746)
              130) velocidad>=16.58666 48  0 exitoso (1.000000000 0.000000000) *
              131) velocidad< 16.58666 19  2 exitoso (0.894736842 0.105263158)
                262) velocidad< 15.63562 18  1 exitoso (0.944444444 0.055555556) *
                263) velocidad>=15.63562 1  0 fallido (0.000000000 1.000000000) *
           33) dist_bs_mov< 298.7134 115  4 exitoso (0.965217391 0.034782609)
             66) dist_bs_mov< 295.7844 114  3 exitoso (0.973684211 0.026315789)
              132) rssi>=-118 112  2 exitoso (0.982142857 0.017857143)
                264) azimut< 258.4487 93  0 exitoso (1.000000000 0.000000000) *
                265) azimut>=258.4487 19  2 exitoso (0.894736842 0.105263158)
                  530) azimut>=260.8203 18  1 exitoso (0.944444444 0.055555556) *
                  531) azimut< 260.8203 1  0 fallido (0.000000000 1.000000000) *
              133) rssi< -118 2  1 exitoso (0.500000000 0.500000000) *
             67) dist_bs_mov>=295.7844 1  0 fallido (0.000000000 1.000000000) *
```

```
        17) azimut< 5.461491 9  1 exitoso (0.888888889 0.111111111) *
      9) dist_bs_mov>=4733.706 21  2 exitoso (0.904761905 0.095238095)
        18) azimut>=218.7802 19  0 exitoso (1.000000000 0.000000000) *
        19) azimut< 218.7802 2  0 fallido (0.000000000 1.000000000) *
    5) velocidad< 0.003473583 19  3 exitoso (0.842105263 0.157894737)
      10) rssi>=-114.5 14  0 exitoso (1.000000000 0.000000000) *
      11) rssi< -114.5 5  2 fallido (0.400000000 0.600000000) *
  3) rssi>=-80 9  3 exitoso (0.666666667 0.333333333) *
```