



IMF

Business
School

MÁSTER EN DATA SCIENCE Y BUSINESS ANALYTICS
ONLINE

**Integración de IoT y Aprendizaje
Automático para mejorar la detección
temprana de enfermedades.**

TFM elaborado por: Richard Douglas Grijalba

Tutor de TFM: Juan Manuel Moreno Lamparero

Tabla de Contenido

Tabla de Contenido.....	1
Dedicatoria	3
MARCO INTRODUCTORIO	5
Introducción.....	7
Consideraciones personales	9
Justificación del Proyecto	10
Antecedentes del proyecto	12
Contexto Global y Perspectiva de la OMS	14
Objetivos	16
Objetivo General	16
Objetivos Específicos:	16
MARCO METODOLÓGICO	17
Metodología CRISP-DM (Cross-Industry Standard Process for Data Mining).....	18
Métodos de recolección de datos	20
Tipos de variables	21
Análisis Estadístico	21
Distribución Estadísticas de las Características	21
Medidas de tendencia central.....	22
Imputación de Datos	22
Elementos Clave del Proceso ETL y del Proyecto.....	23
Administración de Avances y Organización.....	24
Métodos, materiales y tecnologías de uso potencial	29
Modelos de aprendizaje automático	29
Acceso a los modelos desarrollados	32
Materiales y Conjunto de Datos: Datasets	32
EXPLICACIÓN Y ANÁLISIS DE LOS RESULTADOS.....	33
IoT y el uso de wearables - FitBit Fitness Tracker Data.....	34
Paso 1: Dispositivos Wearables (IoT Devices).....	36
Paso 2: Proceso ETL (Extract, Transform, Load)	37
Paso 3: Bases de Datos (Staging & Data Warehouse).....	38
Paso 4: Visualización de Datos	38
Desarrollo del Proceso ETL – Fitbit Wearables.....	39
1. Captura de Datos desde los Wearables (IoT Devices)	40
2. Extracción, Transformación y Carga (ETL) en Spoon Pentaho	41

3. Almacenamiento en SQL Server (Staging & Data Warehouse)	43
4. Visualización en Power BI.....	50
Predicción de Daños Cardiacos - Heart Disease Dataset	66
EDA - Análisis exploratorio de datos.....	67
Dataset	68
Conocimiento de los datos	69
Descripción Estadística.....	69
Imputación de Datos Nulos	71
Correlaciones.....	79
Modelos Aplicados.....	81
Feature engineering en la característica target.....	89
Modelado Featuring Target	94
Resumen de los Resultados Obtenidos	102
Modelo Final Ejecutable.....	104
CONCLUSIONES.....	105
BIBLIOGRAFÍA	108
ANEXOS	112
Anexo 1 : Proceso de Integración y configuracion Spoon Pentaho	113
Anexo 2 MS SQL Server.....	133
Anexo 3 Power bi dashboards	140
Modelo de Datos Heart Disease Dataset del UCI	143

Dedicatoria

Gracias Dios, gracias a mi esposa Cinthya y familia, gracias a todos los profesores y tutores.

El camino recorrido en este proceso ha llegado a su fin, sin embargo aún quedan muchos viajes por emprender.

Por todos esos días de vacaciones y noches de madrugada obligatorias que fueron necesarios para culminar este proyecto.

Este logro es el resultado de esfuerzo, perseverancia y el apoyo incondicional de quienes me rodean. No hay éxito sin sacrificio.

Richard Douglas Grijalba

Resumen

El uso de las tecnologías en cada uno de los entornos y ambientes en el que nos desarrollamos es una realidad en nuestros días, por lo tanto este proyecto tiene como objetivo demostrar cómo el uso de tecnologías IoT, combinadas con modelos de aprendizaje automático, pueden ser de gran utilidad y contribuir a la prevención y detección temprana de enfermedades graves como las enfermedades cardíacas y algunos patrones y hábitos que pueden incidir en el desarrollo de afecciones. A través de un enfoque estructurado, se utilizó un conjunto de datos que corresponde a información capturada de dispositivos wearables dedicados a la recolección de datos sobre la actividad física y la salud de los usuarios, para lo que se desarrolló un ambiente de simulación de un entorno de un proceso de transformación y carga de los datos (ETL) con el fin de demostrar cómo funcionan estos procesos para análisis de los datos.

Una vez cargados los datos e integrados a una base de datos de staging y transferidos al Data Warehouse elaborado en SQL, se procedió a realizar una serie de visualizaciones en el que se observan los patrones de los datos e información útil, sin embargo este proyecto no se limita solamente a ese conjunto de datos en vista que se utilizó un segundo data set para utilizar técnicas de modelado de datos para generar predicciones sobre la posibilidad de daño cardiaco, abarcando de esta forma un análisis de datos tanto descriptivo como predictivo.

El proyecto aborda la necesidad crítica de mejorar la prevención en el ámbito de la salud, destacando la importancia de la integración de tecnologías innovadoras en el monitoreo continuo de la salud. Se utilizaron modelos de machine learning como la regresión logística, KNN y LDA para predecir la probabilidad de desarrollar enfermedades cardíacas, mostrando resultados que validan la aplicabilidad de estas herramientas en un contexto clínico.

A lo largo del proyecto, se demuestra cómo el uso de tecnologías emergentes puede mejorar la eficiencia y precisión en el diagnóstico de enfermedades, proporcionando a los profesionales de la salud herramientas más avanzadas para la toma de decisiones. Este trabajo contribuye a la comprensión de cómo la tecnología IoT puede integrarse en los sistemas de salud actuales, promoviendo una mejor calidad de vida y reduciendo los costos de atención médica mediante una detección más temprana y eficaz.

MARCO INTRODUCTORIO

Planteamiento del problema

¿Cómo demostrar que el uso de los Modelos de Aprendizaje automático y wearables pueden contribuir en la detección temprana de enfermedades?

En los últimos años, hemos visto un gran avance en el uso de tecnologías IoT y wearables para monitorear de manera continua la salud de las personas, mantenimiento preventivo de maquinaria especializada, control de equipos climáticos, artículos para casas inteligentes. Cada uno de estos instrumentos recopilan datos con un fin específico, en lo que corresponde al tema de la salud, tenemos desde pulseras que cuentan nuestros pasos hasta dispositivos que miden la frecuencia cardíaca, así como básculas inteligentes que nos indican y control el peso e índices de grasa corporal, estos aparatos nos permiten tener un seguimiento constante de algunos de nuestros indicadores de salud más importantes.

Sin embargo, a pesar de su creciente popularidad, todavía hay muchas dudas sobre cuán precisos y confiables son estos dispositivos cuando se trata de detectar enfermedades graves como el cáncer o problemas cardíacos. Esto es un gran problema porque, aunque la tecnología está disponible y es cada vez más accesible, no se está utilizando en todo su potencial dentro del ámbito clínico. Si los médicos y profesionales de la salud no confían plenamente en estos datos, es difícil que se incorporen de manera efectiva en la práctica diaria, lo que limita las posibles mejoras en la salud pública.

Es fundamental abordar este problema porque, si logramos demostrar que los datos proporcionados por los dispositivos IoT y wearables son realmente útiles y confiables, podríamos ver un cambio significativo en la forma en que se manejan los cuidados preventivos. Esto no solo beneficiaría a los pacientes, permitiendo una detección más temprana de enfermedades graves, sino que también podría reducir los costos del sistema de salud en general. Además, si más personas tuvieran acceso a herramientas confiables para monitorear su salud desde casa, estaríamos un paso más cerca de hacer que los cuidados preventivos sean más accesibles para todos, independientemente de su situación económica.

Introducción

La salud es uno de los bienes más preciosos que tenemos, solemos mencionar que nuestro tiempo diario, las 24 horas del día y los siete días de la semana son nuestro mayor activo, pero la salud es el motor que nos permite disfrutar ese tiempo. Por lo tanto la prevención de enfermedades graves como el cáncer y las enfermedades cardíacas no solo puede salvar vidas, sino también mejorar significativamente la calidad de vida de las personas. Mi interés en este tema no es meramente académico; está profundamente influenciado por mi experiencia personal.

El cáncer y las enfermedades cardíacas son problemas de salud globales significativos. El cáncer de piel y el cáncer de seno afectan a millones de personas en todo el mundo cada año, con una alta incidencia y mortalidad, especialmente en regiones con acceso limitado a servicios de salud. Las enfermedades cardiovasculares, por otro lado, son la principal causa de muerte a nivel mundial, afectando tanto a países desarrollados como en desarrollo, hemos visto en noticias como deportistas y figuras de gran renombre han muerto por un ataque cardiaco fulminante.

La Organización Mundial de la Salud (OMS) subraya la importancia de la prevención y el acceso universal a la atención médica. La OMS promueve hábitos saludables, programas de detección temprana y el uso de tecnologías innovadoras para mejorar la precisión en el diagnóstico y tratamiento de enfermedades. Estos esfuerzos reflejan un compromiso global para abordar estos desafíos de salud y mejorar la calidad de vida.

En la actualidad, muchos de nosotros poseemos o utilizamos dispositivos que recopilan información sobre nuestras funciones básicas. Los wearables, como relojes inteligentes y pulseras de fitness, realizan un monitoreo continuo de nuestros parámetros de salud como la frecuencia cardíaca, la actividad física y los patrones de sueño. Esta información es recopilada en tiempo real, agrega un valor significativo al monitoreo continuo de la salud y puede ser crucial para diagnosticar problemas o prevenir situaciones críticas antes de que se conviertan en enfermedades graves. Además, los wearables son un claro ejemplo de cómo el Internet de las Cosas (IoT) está revolucionando la forma en que entendemos y gestionamos la salud, permitiendo un seguimiento constante y preciso de indicadores vitales.

Se puede decir que el uso de modelos de aprendizaje automático (ML) y la inteligencia artificial (IA) se ha vuelto en una herramienta determinante y útiles para el

campo de la salud, cuyo potencial para transformar la forma en que se detectan y previenen estas enfermedades graves está en un nivel incipiente por la capacidad analítica que se agrega año con año, sumado a las habilidades técnicas de los cuerpos médicos. Integrar datos de diversas fuentes, como datasets históricos de enfermedades y datos modernos provenientes de dispositivos de IoT, puede proporcionar una visión más completa y precisa del estado de salud de los individuos.

Mi objetivo con este proyecto es demostrar que los modelos de ML son un factor útil y determinante en la salud actual y futura, y que la IA puede contribuir significativamente a la solución de problemas de salud a través del uso de técnicas avanzadas. La capacidad de prever y prevenir enfermedades basándose en datos personales no solo es una esperanza para un futuro más saludable, sino también una necesidad urgente en la lucha contra el cáncer y las enfermedades cardíacas. Integrar la tecnología en la prevención puede hacer una diferencia significativa, extendiendo las posibilidades médicas a más personas y ayudando a salvar vidas.

Consideraciones personales

La tecnología siempre a través de la historia del ser humano nos ha presentado retos, desafíos y oportunidades. La manera en que elegimos aplicarla y aprovecharla define nuestro desarrollo como sociedad y civilización, anteriormente cada hito de la historia estaba marcado por un avance o desarrollo de alguna técnica en especial, nuestra etapa actual no es la excepción, desde el dominio e integración del fuego, la agricultura y el motor de vapor y la industrialización, tal y como indicó Clive Humby “los datos son el nuevo petróleo, en vista que se Extrae, se Transforma y se Carga”.

Sin embargo cada uno de nosotros tiene el poder de darle sentido a la tecnología, ya sea permitiéndole que nos impulse hacia nuevas alturas o dejándola estancarnos en el lodo del conformismo, es la misma historia que nos permite pensar en ejemplos claros, como lo fue en su momento la salida de los primeros computadores y como las empresas de venta de máquinas de escribir menospreciaban este nuevo aparato, todo radica en cómo reaccionamos ante las amenazas y cómo transformamos nuestro entorno para generar oportunidades, en palabras de John F. Kennedy, “algunos hacen que las cosas sucedan, otros ven que las cosas suceden, y algunos solo se preguntan qué pasó”, ante el avance de la tecnología surgirán detractores y defensores de esta.

Por lo tanto, este proyecto busca demostrar que la IA y el Aprendizaje Automático son una posibilidad: un compromiso con la innovación y la mejora continua, y una invitación a todos a no solo adaptarse, sino a prosperar en el mundo que estamos construyendo. Como dijo Darwin, “no es la especie más fuerte la que sobrevive, ni la más inteligente, sino la que responde mejor al cambio,” y como Ralph Waldo Emerson agregó, “haz lo que temes, y la muerte del temor será segura.”

No debemos temerle a la IA y al Aprendizaje Automático; debemos adaptarnos ahora que la buena ola está llegando, en lugar de esperar a que sea demasiado tarde.

Richard Douglas Grijalba

Justificación del Proyecto

La salud es uno de los valores máspreciados para el ser humano y sus seres queridos, mantener una buena salud no solo contribuye al bienestar individual, sino que también afecta positivamente la calidad de vida de quienes nos rodean. Sin embargo, el acceso a servicios de salud puede ser costoso y limitado en ciertas circunstancias, lo que puede dificultar la prevención y tratamiento de enfermedades.

Personalmente, he vivido de cerca la importancia de la prevención y el tratamiento de enfermedades. Mi padre sufrió un ataque cardíaco hace cuatro años; afortunadamente, sobrevivió, pero no todos tienen la misma suerte, como fue el caso de mi abuelo paterno, quien falleció debido a problemas de presión arterial y sus complicaciones relacionadas. Mi madre ha estado enfrentando problemas relacionados con daños en la piel y ha estado recibiendo tratamiento. En mi propio caso, me he sometido a diagnósticos tempranos y procesos preventivos que colaboraron en la detección precoz de una situación clínica que, aunque no puso en riesgo mi salud más allá de un procedimiento quirúrgico, subrayó la importancia de la intervención oportuna. Sin embargo, he escuchado y visto casos clínicos en los que los diagnósticos y exámenes llegaron tarde, dejando poco o nada por hacer para los pacientes. Estas experiencias resaltan el hecho de que la prevención es crucial y puede salvar vidas, y sumado a las posibilidades tecnológicas que se han vuelto más dinámicas y accesibles hoy en día.

La integración de Machine Learning (ML) y la Inteligencia Artificial (IA) en el campo de la salud tiene el potencial de transformar la manera en que se gestionan y previenen las enfermedades. Estas tecnologías avanzadas pueden democratizar el acceso a la atención médica, haciendo que la prevención y el diagnóstico precoz de enfermedades sean más accesibles y eficaces.

En particular, la incorporación de datos de wearables añade una dimensión importante al análisis de salud. Cada vez más dispositivos permiten monitorear los signos vitales de las personas y sus respectivos parámetros establecidos. Estos dispositivos recopilan información en tiempo real sobre la actividad física, la frecuencia cardíaca y otros indicadores de salud, permitiendo un monitoreo continuo y una intervención temprana. Anteriormente, estos dispositivos estaban constituidos por costosos equipos, mientras que hoy en día, muchos de nosotros tenemos a nuestro alcance algún tipo de estos artefactos que pueden beneficiarnos significativamente. La combinación de estos datos con

información médica tradicional puede mejorar notablemente la capacidad para predecir y prevenir enfermedades cardíacas y cáncer.

¿Y si pudiéramos no solo prevenir, sino también pronosticar y adelantarnos a posibles problemas de salud utilizando nuestra información personal?

La capacidad de pronosticar basándonos en datos personales y antecedentes médicos, incluyendo factores genéticos y hereditarios, podría proporcionar herramientas aún más poderosas para la detección temprana y la intervención proactiva. Esto no solo podría reducir el dolor y la tristeza asociados con enfermedades crónicas y graves, sino también disminuir la mortalidad al permitir intervenciones antes de que se desarrollen problemas graves. Para una familia, un ser querido jamás será reemplazable, y para una sociedad, lograr prolongar la salud de personas en su etapa productiva es crucial. Personajes muy influyentes y útiles para la sociedad han pasado fugazmente debido a problemas de salud, pero si consideramos la prevención y el uso de la tecnología, podríamos llevar calidad de vida a más personas.

Implementar estas tecnologías en el análisis y la predicción de enfermedades cardíacas y otros problemas de salud puede ofrecer una atención médica de mayor calidad a un costo más accesible. Al permitir un enfoque más personalizado y predictivo, se contribuye a mejorar la salud pública global y a promover una vida más saludable y plena.

Antecedentes del proyecto

La integración de tecnologías de Internet de las Cosas (IoT) y Aprendizaje Automático (ML) ha emergido como un campo prometedor para mejorar la detección temprana de enfermedades. Históricamente, el uso de wearables para monitorear la salud se ha limitado a aplicaciones básicas, como el seguimiento de la actividad física y el control de parámetros simples como la frecuencia cardíaca y el peso. Sin embargo, en la última década, hemos sido testigos de avances significativos en la precisión y capacidad de estos dispositivos, lo que ha permitido una recopilación de datos más compleja y detallada, como puede observar en estudios realizados sobre la recopilación de datos y trato de pacientes para procesos de rehabilitación Patel, S., Park, H., Bonato, P., Chan, L., & Rodgers, M. (2012).

En los últimos años, el campo de la salud digital ha evolucionado notablemente. Según Wang et al. (2018), los dispositivos wearables han pasado de ser simples monitores de actividad física a sofisticados sensores capaces de medir una variedad de indicadores biométricos, como la variabilidad de la frecuencia cardíaca, la presión arterial y los niveles de glucosa en sangre. Esta evolución ha sido respaldada por la creciente capacidad de los algoritmos de aprendizaje automático para analizar grandes volúmenes de datos y detectar patrones que podrían indicar problemas de salud.

Por ejemplo, un estudio realizado por Quer et al. (2020) denominado *Wearable sensor data and self-reported symptoms for COVID-19 detection*, corresponde a un intento para el uso de los datos obtenidos por medio de dispositivos *wearables*, combinados con la autoevaluación de síntomas, pueden detectar casos de COVID-19 con una precisión del 80%. Este enfoque permite identificar a individuos presintomáticos o asintomáticos, lo que es crucial para contener brotes virales en sus etapas más tempranas.

Tradicionalmente, el diagnóstico médico se basa en la identificación de síntomas reportados por los pacientes y observados por los médicos. Estos síntomas, que podrían ser por ejemplo dolor de cabeza, fatiga, cambios en el peso, entre otros, son parte de indicadores que son como pequeños faros o luces brindando información. Posteriormente, los profesionales de la salud utilizan esta información para formar una hipótesis sobre la presencia de una enfermedad.

A medida que los síntomas se presentan, generan datos valiosos que pueden ser capturados y analizados mediante dispositivos de monitoreo de salud (actualmente, estos

equipos pueden ser electrocardiogramas, ultrasonidos, exámenes médicos, etc.). Ahora pensemos en estos equipos como *wearables* especializados. Estos dispositivos, al registrar parámetros fisiológicos como la frecuencia cardíaca, la presión arterial y los niveles de actividad física, proporcionan una dimensión objetiva que complementa la evaluación clínica basada en síntomas.

Por ejemplo, un paciente que reporta episodios frecuentes de fatiga puede tener datos de un wearable que muestran patrones de sueño interrumpido y un aumento en los niveles de estrés. Estos datos cuantitativos ofrecen evidencia adicional que puede respaldar la hipótesis del médico y ayudar a refinar y redirigir el diagnóstico a un punto más acertado.

La integración de datos de dispositivos wearables y las tecnologías denominadas IoT (Internet de las Cosas) con técnicas de Aprendizaje Automático (ML) permite una evaluación más profunda y precisa. Al analizar grandes volúmenes de datos recopilados de wearables junto con los síntomas reportados, los modelos de ML pueden identificar patrones y correlaciones que podrían no ser evidentes en un análisis clínico tradicional. Esta combinación de datos objetivos y subjetivos mejora la capacidad de los profesionales de la salud para realizar diagnósticos más precisos y personalizados (Ravi et al., 2017).

Contexto Global y Perspectiva de la OMS

Contexto Global y Perspectiva de la OMS

Las enfermedades como el cáncer y las enfermedades cardíacas representan desafíos significativos a nivel mundial. Estas condiciones no solo afectan a millones de personas, sino que también imponen una carga económica y social considerable sobre los sistemas de salud y las sociedades en general. La magnitud de estas enfermedades resalta la necesidad urgente de encontrar mejores estrategias para su prevención, diagnóstico y tratamiento (*Organización Mundial de la Salud [OMS], 2021*).

Daños y Casos a Nivel Mundial

Cáncer:

Cáncer de piel: El cáncer de piel es uno de los tipos más comunes a nivel global, con el melanoma siendo la forma más grave y prevalente en países desarrollados. Cada año, millones de personas son diagnosticadas con cáncer de piel, lo que lo convierte en una prioridad de salud pública en múltiples regiones del mundo (OMS,2021).

Cáncer de seno: El cáncer de seno se encuentra entre los más frecuentes en mujeres a nivel mundial. En 2020, se reportaron aproximadamente 2.3 millones de nuevos casos. A pesar de los avances en tratamientos y la implementación de programas de detección, la tasa de mortalidad sigue siendo elevada, particularmente en regiones con acceso limitado a servicios de salud adecuados (International Agency for Research on Cancer [IARC], 2021).

Enfermedades Cardíacas:

Enfermedades cardiovasculares: Estas enfermedades constituyen la principal causa de mortalidad global. En 2019, alrededor de 18 millones de muertes fueron atribuidas a enfermedades cardiovasculares, representando el 32% del total de fallecimientos a nivel mundial. Estas enfermedades afectan tanto a países desarrollados como a aquellos en desarrollo, reflejando una preocupación universal (OMS,2021).

Interés de la OMS en la Prevención y Acceso Ampliado

La Organización Mundial de la Salud (OMS) ha intensificado su enfoque en la prevención y predicción de enfermedades, especialmente a raíz de la pandemia de COVID-19. Reconociendo el impacto de las enfermedades crónicas y su alta carga en los sistemas de salud, la OMS ha implementado una serie de estrategias para mejorar la salud pública

a través de la prevención, el acceso a la atención médica y la innovación tecnológica. A continuación, se destacan algunas de las estrategias clave que la OMS emplea para enfrentar estos desafíos:

1. Prevención y Detección Temprana:

Promoción de Estilos de Vida Saludables: La OMS promueve la adopción de hábitos saludables, como una dieta equilibrada y una actividad física regular, para reducir el riesgo de desarrollar enfermedades crónicas. Estas recomendaciones están basadas en la evidencia y buscan mejorar la salud general de la población (OMS,2020).

Programas de Cribado y Detección: La OMS apoya la implementación de programas de detección para el cáncer y enfermedades cardiovasculares. Estos programas están diseñados para identificar condiciones de salud en sus etapas iniciales, cuando los tratamientos son más efectivos y las tasas de éxito son más altas (OMS,2020).

2. Acceso Universal a la Atención Médica:

Expansión de Servicios de Salud: La OMS trabaja para garantizar que todas las personas, independientemente de su ubicación geográfica o situación económica, tengan acceso a servicios de atención médica de calidad. Esta expansión busca reducir la brecha en el acceso a cuidados esenciales y mejorar la equidad en salud (OMS,2021).

Reducción de Desigualdades: La OMS está comprometida en reducir las desigualdades en el acceso a la salud. La organización se enfoca en apoyar a las comunidades desfavorecidas y garantizar que reciban los recursos y el apoyo necesario para la **prevención** y tratamiento de enfermedades (OMS,2021).

3. Innovación y Tecnología:

Incorporación de Tecnologías Avanzadas: La OMS respalda el uso de tecnologías innovadoras, como la inteligencia artificial y el aprendizaje automático, para mejorar la precisión en la predicción, diagnóstico y tratamiento de enfermedades. Estas tecnologías tienen el potencial de transformar la forma en que se aborda la salud pública y los cuidados clínicos (OMS,2021).

Investigación y Desarrollo: La OMS fomenta la investigación en nuevas tecnologías y métodos para hacer la prevención y el tratamiento de enfermedades más **efectivos**. El enfoque en la investigación busca impulsar el desarrollo de soluciones innovadoras que puedan ser implementadas a nivel global (OMS,2021).

Estos esfuerzos reflejan un compromiso global para enfrentar los problemas de salud pública y mejorar la calidad de vida a través de la prevención, el acceso a la atención médica y la innovación tecnológica. Integrar estas estrategias en la vida diaria y en las comunidades puede marcar una diferencia significativa en la lucha contra enfermedades graves, contribuyendo a un futuro más saludable para todos.

Objetivos

Objetivo General

Implementar modelos de aprendizaje automático (ML) para la demostración del uso de las tecnologías y los beneficios que generan en la prevención y el diagnóstico temprano de enfermedades graves como las enfermedades cardíacas, por medio del uso de datasets especializados en el área de salud.

Objetivos Específicos:

1. Explorar cómo el uso de datos recopilados por dispositivos IoT, como wearables, puede optimizar los procesos de detección y tratamiento de enfermedades graves, y contribuir a un enfoque más personalizado y efectivo en la atención médica.
2. Realizar una exploración exhaustiva y preparación de los conjuntos de datos para garantizar su calidad y relevancia en el desarrollo de modelos de ML.
3. Aplicar modelos de ML para predecir el riesgo de desarrollar enfermedades, utilizando técnicas vistas durante el programa de estudio.

MARCO METODOLÓGICO

Metodología CRISP-DM (Cross-Industry Standard Process for Data Mining)

La metodología CRISP-DM es una de las más reconocidas en el campo de la minería de datos y es muy útil para cualquier proyecto relacionado con el análisis de datos. Este enfoque te guía a través de un proceso bien estructurado y paso a paso, asegurando que no te pierdas en el camino. La metodología se divide en seis fases principales:

Comprensión del Negocio: Aquí es en donde se define qué se desea lograr con el proyecto y entiendes el contexto empresarial. Esto incluye identificar los problemas que necesitas resolver y establecer qué éxito significa para ti. Es como poner las bases para todo el trabajo (Chapman et al., 2000).

El proyecto tiene dos objetivos claros. Por un lado, busca mostrar el proceso de captura, transformación y distribución de datos en el contexto de tecnología IoT, mediante una simulación de un pipeline ETL. Por otro lado, se pretende demostrar que los datos generados por dispositivos wearables pueden ofrecer insights e indicadores valiosos sobre la salud de las personas, que podrían ser útiles para diseñar políticas de salud pública e incentivar hábitos de vida más saludables. Además, se quiere poner en práctica los conocimientos adquiridos durante el Máster en Data Science and Business Analytics del IMF.

Comprensión de los Datos: En esta fase se concentra en recolectar y explorar los datos disponibles. Entender la calidad de los datos y ver si hay problemas o patrones interesantes. Es un paso crucial para asegurarte de que tienes todo lo que necesitas para el análisis (Data Science Project Management, n.d.).

Se trabajó con dos datasets: el primero, de Fitbit, que permite analizar la actividad física diaria de los usuarios, y el segundo, de Heart Disease Dataset, centrado en predecir enfermedades cardíacas. Esta fase se centró en explorar la calidad de los datos, identificando posibles inconsistencias o valores nulos, y en buscar patrones relevantes. El análisis busca demostrar que los datos provenientes de dispositivos wearables no solo son útiles, sino que también pueden tener un impacto positivo en la salud pública.

La identificación de inconsistencias y la exploración inicial de los datos proporcionaron una base sólida para preparar los datos y asegurarse de que la información relevante estuviera disponible para el análisis en las siguientes fases.

Preparación de los Datos: En este paso se procede a realizar técnicas de *Feature engineering* y trata de datos, esto con la finalidad de tener datos útiles para el modelado. Esto puede incluir manejar errores o datos nulos, valores faltantes, y combinar datos de diferentes fuentes si es necesario (Chapman et al., 2000).

Se llevó a cabo una simulación del proceso ETL con el dataset de Fitbit, demostrando cómo se capturan, transforman y distribuyen los datos en un entorno de tecnología IoT. Las tareas incluyeron la limpieza de los datos, la creación de nuevas variables y su preparación para un análisis más profundo. En el dataset de Heart Disease Dataset, se trabajó con variables que influyen en el proceso de predicción de la característica de estudio, aplicando distintos modelos de aprendizaje automático, mismo que se encuentran documentados en el archivo Python que puede ser consultado en su totalidad así como las diferentes capturas en los anexos.

Por lo tanto en la ejecución del trabajo, se aplicaron técnicas de feature engineering, buscando un enriquecimiento de los datos, ampliando las posibilidades de análisis y buscando una limpieza de datos, con lo que se buscó que los datos fueran adecuados para el modelado. Esto está directamente alineado con el objetivo de crear un pipeline ETL que capture, transforme y distribuya datos en un entorno IoT.

Modelado: Aplicar técnicas estadísticas y algoritmos para construir modelos que respondan a las preguntas del negocio o solucionen los problemas que identificaste. Es donde realmente entra la parte técnica del proyecto (Data Science Project Management, n.d.).

En esta fase, se aplicaron técnicas descriptivas para el dataset de Fitbit, creando insights y visualizaciones que permiten detectar patrones en la actividad física y el sedentarismo. Para el dataset de Heart Disease Dataset, se aplicaron modelos de Machine Learning como la regresión logística y los árboles de decisión, con el fin de predecir el riesgo de enfermedades cardíacas. La combinación de análisis descriptivo y predictivo demuestra el potencial de los datos capturados por wearables para generar indicadores de salud relevantes.

Evaluación: Revisión de los resultados del modelado para asegurar de que cumplen con los objetivos del negocio. Evaluar si los modelos son efectivos y útiles y hacer ajustes si es necesario (Chapman et al., 2000).

Los modelos predictivos del dataset de Heart Disease Dataset se evaluaron utilizando métricas como precisión, recall y matriz de confusión. En cuanto al análisis descriptivo del dataset de Fitbit, se validaron los insights a través de las visualizaciones en Power BI, asegurando que la información obtenida sobre la actividad física y el sedentarismo tiene valor práctico para diseñar estrategias de prevención de enfermedades. Este enfoque refuerza la idea de que los datos provenientes de wearables pueden ser útiles para la toma de decisiones en salud pública.

Despliegue: Finalmente, implementar los modelos y resultados en el entorno de producción. Asegurarse de que todo funcione bien en la práctica, integrando los modelos. (Data Science Project Management, n.d.).

Una de las mejores cosas de CRISP-DM es que es un proceso iterativo. Eso significa que puede volver a fases anteriores si se encuentra o se descubre que hay algo nuevo o si se requiere hacer ajustes en base a nuevos hallazgos (Chapman et al., 2000; Data Science Project Management, n.d.).

La fase de despliegue se enfocó en la creación de dashboards interactivos en Power BI para visualizar los patrones de actividad física extraídos de los wearables. Estos dashboards están pensados para ser utilizados por usuarios finales o incluso por instituciones de salud, facilitando el monitoreo del comportamiento físico. En cuanto al modelo predictivo de Heart Disease Dataset, se diseñó para la predicción de la posibilidad de desarrollar una enfermedad cardiaca, basado en los datos incorporados al modelo.

Métodos de recolección de datos

En lo que compete al desarrollo y ejecución de este proyecto se consideran los siguientes aspectos importantes, para el uso e implementación de estrategias de análisis de datos, se plantea el desarrollo de un ambiente de simulación de lo que corresponde un proceso de IoT, con el uso de un dataset tomado de un banco de datos en este caso de estudio corresponde a kaggle, posterior a esto se analizan los datos y se realizan transformaciones, así también se tiene un segundo dataset en el que se pretende aplicar modelos de aprendizaje automático para predecir la probabilidad de afectación cardiaca, por lo tanto la recolección de datos está cubierto por medio de la selección de los conjuntos de datos que puedan ser útiles para el desarrollo del presente trabajo final.

Tipos de variables

Se analizan variables discretas, continuas y categóricas, las cuales se definen según Minitab, las variables categóricas están agrupadas en un número finito de categorías, en nuestro caso de análisis de datos corresponden al tipo de object o string, mientras que por otra parte las variables discretas son numéricas y tienen un número contable entre dos valores, para nuestro caso de estudio corresponden a las de tipo integer o int, y finalmente las continuas son aquellas que representan un número infinito de valores entre dos valores, para los casos de estudio de análisis de datos corresponden a las características del tipo float o date.

Análisis Estadístico

En cuanto al análisis estadístico tenemos presente para el desarrollo de este proyecto el uso del análisis descriptivo de características, el mismo lo que busca es denotar el comportamiento e información de los datos en general, sumado a esto tenemos el análisis exploratorio de las características que intervienen en el data set y finalmente tenemos el análisis predictivo en el que una vez que se tienen los datos, se conocen y se interpretan, se procede a la creación de modelos y algoritmos que permiten crear predicciones sobre el resultado de una variable de salida basados en los datos de entrada.

Distribución Estadísticas de las Características

Parte del análisis de los datos, corresponde conocer y/o entender la distribución estadística de las características, como lo son el peso, edad y sexo, cada una de las características responde a un tipo de distribución, y esto nos puede dar información relevante, entre las distribuciones más comunes tenemos la distribución normal la cual es aquella que una vez que se grafica por medio de un histograma la distribución de las barras tiende a parecerse a una campana Triola (2020), mientras que también intervienen otras distribuciones de los datos como la binomial y exponencial.

Es aquí cuando entra el tema de la simetría o asimetría de los datos, la simetría está presente cuando en la distribución de los datos tenemos a la moda, mediana y media son iguales, por otra parte en lo que respecta a la asimetría según Triola (2020), una distribución de datos es asimétrica si se extiende más hacia un lado que hacia el otro, la asimetría puede estar inclinada hacia la derecha o la izquierda.

Medidas de tendencia central

De acuerdo con Mario F. Triola (2020), "una medida de tendencia central es un valor en medio o en el centro de un conjunto de datos" (p. 82). Triola también explica que la media es generalmente la más importante de las mediciones numéricas usadas para describir datos, y comúnmente se le llama promedio. Por otro lado, la mediana se describe como un "valor medio" que divide un conjunto de datos en dos partes iguales, donde aproximadamente la mitad de los valores son menores y la otra mitad son mayores. Finalmente, menciona que la moda, aunque no se usa frecuentemente con datos cuantitativos, es la única medida de tendencia central aplicable a datos cualitativos (p. 83).

Imputación de Datos

Estas tres medidas de tendencia central mencionadas anteriormente, son tan importantes que son utilizadas como medio para procedes a imputar y/o cubrir datos faltantes u omisos, según lo visto en el módulo de Preparación y proceso de datos, se indica que el imputar datos es un proceso que se refiere a asignar valores probables a una variable cuando el valor real no está disponible, este valor puede que no esté disponible por un error en el procesamiento y captura de los datos o del todo faltantes, en ese caso se puede utilizar la media de los valores existentes en lugar del valor más frecuente para realizar la imputación. Por lo tanto aplicando el criterio que corresponde según el tipo de variable y según su distribución, en aquellas características de tipo categóricas la imputación se realiza por medio de la moda, mientras que las características numéricas se realizan para aquellas cuya distribución es simétrica la imputación se realiza por medio de la media y las distribuciones asimétricas se realizan por medio de la mediana.

Elementos Clave del Proceso ETL y del Proyecto

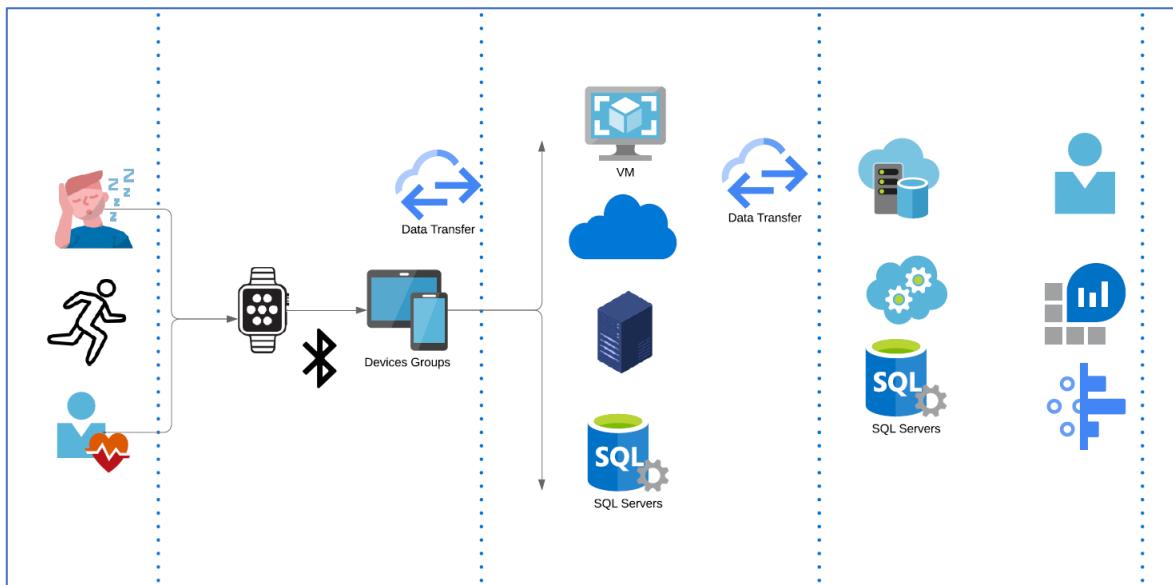


Diagrama de creación propia con lucid.app

Simulación del Proceso ETL: Este proyecto simula cómo se manejan los datos en entornos reales de tecnología IoT, demostrando la capacidad de capturar, transformar y distribuir información proveniente de wearables. Se recreó un pipeline de datos que refleja el manejo de grandes volúmenes de información en tiempo real.

Generación de Insights con Wearables: Los datos obtenidos a partir de dispositivos wearables fueron procesados para generar indicadores útiles sobre la actividad física y el sedentarismo. Estos insights pueden ser aplicados en políticas de salud pública, orientadas a promover hábitos de vida más saludables y a la prevención de enfermedades relacionadas con la inactividad física.

Aplicación de Conocimientos del Máster: A lo largo del proyecto se emplearon múltiples técnicas avanzadas de análisis de datos, incluyendo la preparación de datos mediante procesos ETL, la visualización de datos en Power BI y la creación de modelos predictivos en Python. Todo esto demuestra una sólida aplicación de las herramientas y conceptos aprendidos durante el Máster en Data Science and Business Analytics del IMF.

Relevancia para la Salud Pública: El proyecto no solo aporta valor desde un punto de vista académico o técnico, sino que también tiene implicaciones en políticas de salud pública. Los insights generados sobre la actividad física pueden orientar campañas de concientización y estilos de vida saludable, mientras que los modelos predictivos pueden

ayudar a identificar personas con mayor riesgo de sufrir enfermedades cardíacas, lo cual es clave para la prevención (factores de riesgo).

Combinación de Análisis Descriptivo y Predictivo: La combinación del análisis descriptivo con el dataset de Fitbit y del análisis predictivo con el dataset de Heart Disease Dataset ofrece una visión integral. Se abordan tanto la visualización de datos actuales como la predicción de riesgos futuros, lo que proporciona un análisis estadístico completo con aplicaciones prácticas en la toma de decisiones en salud.

Administración de Avances y Organización

Tomando aspectos relevantes y útiles de la metodología ágil se eligió para la gestión de este proyecto debido a su capacidad para adaptarse rápidamente a los cambios y a su enfoque iterativo, que es crucial en proyectos de ciencia de datos donde los requisitos y los datos pueden evolucionar constantemente. Según Beck et al. (2001), 'la metodología ágil se basa en la capacidad de adaptarse a los cambios y en la entrega continua de valor mediante ciclos de desarrollo iterativos' (p. 1). Aunque el proyecto se realiza de forma individual y no se trabaja en un equipo SCRUM, se han adaptado aspectos claves de la metodología ágil para mantener un orden y flujo de trabajo eficiente.

En el ámbito de la ciencia de datos, los proyectos a menudo enfrentan incertidumbre en cuanto a los datos y los resultados esperados. Como se menciona en Highsmith (2009), 'la flexibilidad y la capacidad de adaptarse a nuevos requerimientos son fundamentales para manejar la incertidumbre en proyectos complejos y dinámicos' (p. 45). La metodología ágil permite una flexibilidad que facilita la adaptación continua del proyecto, la incorporación de nuevos insights a medida que se descubren y la mejora progresiva del modelo basado en los resultados de las iteraciones anteriores.

En este contexto individual, se han utilizado prácticas ágiles para mantener un enfoque centrado en el cliente y en los resultados, asegurando que el proyecto no solo cumpla con los objetivos iniciales, sino que también se adapte a las necesidades emergentes durante el proceso (Schwaber & Sutherland, 2017, p. 63). Este enfoque permite cumplir con los plazos y garantizar la calidad del proyecto, aunque se implemente a nivel individual en lugar de un equipo SCRUM completo."

Para la correcta gestión y control del proyecto, se implementó un cronograma de tareas que siguió una metodología basada en la planificación, permitiendo una organización eficiente del trabajo mediante la división en historias de usuario y tareas específicas. Este

enfoque aseguró que el proyecto se mantuviera alineado con los objetivos y plazos establecidos, facilitando la identificación y resolución de posibles desviaciones a lo largo del proceso, pero lo más importante que no se dejara de lado ninguna tarea y que se asegure que se van completando las tareas que poseen dependencias de otras.

Las tareas se clasificaron en cuatro estados principales:

- **Done** (Completado): Las tareas que se han completado satisfactoriamente.
- **Waiting** (En espera): Tareas que están pendientes de finalización de otras actividades o requieren de recursos adicionales antes de poder iniciarse.
- **Not Started** (No iniciado): Tareas que aún no han comenzado, pero están programadas para fases posteriores del proyecto.
- **In Progress** (En progreso): Tareas que están en proceso de ejecución en la etapa actual del proyecto.

Secciones del Cronograma de Tareas

1. Planning (Planificación):

- **Seleccionar un nombre adecuado para el proyecto:** Como propietario del proyecto, se eligió un nombre representativo del enfoque y objetivos del proyecto.
- **Trabajar en el anteproyecto:** Desarrollo inicial del anteproyecto, detallando los objetivos, metodología, datasets a utilizar y resultados esperados.
- **Someter a revisión el anteproyecto:** Una vez completado el anteproyecto, se sometió a revisión para recibir retroalimentación y realizar las correcciones necesarias.
- **Recibir la revisión y realizar las correcciones:** Revisión del feedback recibido y ejecución de las correcciones pertinentes para asegurar que el anteproyecto cumpla con los estándares requeridos.
- **Mantener la estructura del proyecto y citas bibliográficas:** Asegurar que la estructura del documento final sea coherente y que todas las citas bibliográficas estén correctamente formateadas y documentadas.

2. Milestones (Hitos):

- **Dataset - Wearables:** Organización y preparación del dataset relacionado con dispositivos wearables para su uso en el análisis y modelado.

- **Dataset - Heart Disease Dataset:** Organización y preparación del dataset sobre enfermedades cardíacas, que se utilizará para el análisis exploratorio y desarrollo del modelo predictivo.

3. Next Steps (Próximos pasos):

- **Dataset Wearables - Creación del Pipeline en Pentaho:** Desarrollo de un pipeline en Pentaho para procesar los datos del dataset de **wearables**, asegurando la integración y transformación de los datos necesarios.
- **Dataset Wearables - Feature Engineering:** Aplicación de técnicas de ingeniería de características (feature engineering) para **extraer** y crear nuevas variables que mejoren el rendimiento del modelo.
- **Dataset Wearables - Crear ambiente de simulación en Pentaho - datos Streaming:** Configuración de un entorno de simulación en Pentaho para manejar datos en tiempo real (streaming), permitiendo la evaluación **dinámica** de los datos de wearables.
- **Dataset Wearables - Enlazar con SQL:** Establecimiento de conexiones con bases de datos SQL para integrar y consultar datos adicionales necesarios para el análisis.
- **Dataset Wearables - Exportar datos en Power BI:** Exportación de los datos procesados a Power BI para su visualización y análisis avanzado.
- **Dataset Heart Disease Dataset - Importar y limpiar el dataset en Python:** Importación del dataset de enfermedades cardíacas en Python, **seguido** de un proceso de limpieza de datos para asegurar su calidad y coherencia.
- **Dataset Heart Disease Dataset - Realizar un EDA para identificar patrones y relaciones:** Realización de un Análisis Exploratorio de Datos (EDA) para identificar patrones, correlaciones y relaciones en el dataset, que serán fundamentales para el modelado predictivo.
- **Dataset Heart Disease Dataset - Feature Engineering:** Aplicación de técnicas de feature engineering en el dataset de Heart Disease Dataset para crear variables que optimicen la predicción de resultados.
- **Dataset Heart Disease Dataset - Modelado y entrenamiento ML:** Desarrollo y entrenamiento de modelos de machine learning utilizando el dataset de Heart Disease Dataset, con el objetivo de predecir enfermedades cardíacas de manera precisa.

Sprint	Generalidad	Detalle
Sprint 1	- Completar el pipeline en Pentaho.	<ul style="list-style-type: none"> - Verificar integración de datos. - Realizar pruebas de validación del pipeline. - Documentar el proceso.
Sprint 1	- Iniciar feature engineering para el dataset de wearables.	<ul style="list-style-type: none"> - Identificar variables clave. - Continuar con el feature engineering. - Implementar nuevas variables.
Sprint 1	- Configurar el entorno de simulación para datos en tiempo real.	<ul style="list-style-type: none"> - Instalar herramientas necesarias. - Realizar pruebas del entorno de simulación. - Asegurar la captura y procesamiento de datos en tiempo real.
Sprint 2	- Establecer conexiones con bases de datos SQL.	<ul style="list-style-type: none"> - Configurar accesos y permisos. - Validar la conexión con las bases de datos. - Realizar pruebas de integración.
Sprint 2	- Exportar datos a Power BI.	<ul style="list-style-type: none"> - Preparar los datos para la visualización. - Crear dashboards iniciales en Power BI. - Validar la visualización de datos.
Revisión parcial y documentación del proyecto		<ul style="list-style-type: none"> - Asegurarse mantener el hilo conductor
Sprint 3	- Importar el dataset de Heart Disease Dataset en Python.	<ul style="list-style-type: none"> - Verificar la integridad de los datos. - Limpiar y preprocesar el dataset de Heart Disease Dataset. - Documentar el proceso de limpieza.
Sprint 3	- Realizar el análisis exploratorio de datos (EDA).	<ul style="list-style-type: none"> - Identificar patrones y correlaciones. - Completar el análisis EDA. - Documentar hallazgos y visualizaciones.
Sprint 3	- Aplicar feature engineering al dataset de Heart Disease Dataset.	<ul style="list-style-type: none"> - Crear nuevas variables para el modelo. - Revisar y validar las nuevas variables. - Preparar el dataset para el modelado.
Sprint 3	- Desarrollar el modelo predictivo para Heart Disease Dataset.	<ul style="list-style-type: none"> - Seleccionar algoritmos adecuados. - Entrenar el modelo y ajustar hiperparámetros. - Validar el rendimiento del modelo.
Sprint 3	- Revisión final del proyecto.	<ul style="list-style-type: none"> - Consolidar y verificar resultados finales. - Preparar el documento final del proyecto. - Realizar una revisión exhaustiva y ajustes finales.

Para el presente proyecto final de máster, se utilizó Asana para organizar las tareas y asegurar el cumplimiento del proyecto. Al igual que se hizo en el transcurso del programa de estudio, esta herramienta permitió crear listas de tareas, asignar responsables, definir fechas límite (deadlines), establecer estatus y seguir los avances de manera eficiente.

En el proyecto, Asana fue clave para dividir el trabajo en varias fases, lo que facilitó un enfoque estructurado. Cada tarea fue acompañada de indicadores de estado como Done (Completado), In Progress (En progreso), Waiting (En espera) y Not Started (No iniciado), para tener una visión clara del progreso. Además, se asignaron prioridades como High (Alta) para tareas críticas, asegurando que los elementos más importantes se abordaran primero.

Al final, el uso de Asana aseguró una gestión eficiente del tiempo y de los recursos, proporcionando visibilidad completa sobre el avance del proyecto y garantizando que se cumplieran los plazos establecidos.

El uso de un seguimiento de tareas bajo este esquema, sumado además de la implementación de la herramienta de gestión Asana, fue fundamental para garantizar el avance continuo y ordenado del proyecto, con el fin de no dejar de lado ningún aspecto de importancia y relevante para la culminación del trabajo deseado. La plataforma Asana permitió dividir el trabajo en tareas más manejables y pequeñas, las cuales se les asignó un estatus y cambio del mismo según el avance, facilitando el seguimiento de cada una a través de indicadores claros como Done (completado), In Progress (en progreso), Waiting (en espera) y Not Started (no iniciado), esto permitió que no se trabajara de forma redundante sobre algún aspecto ya cubierto y poder fijar esfuerzos sobre tareas incompleta o en proceso.

Uno de los aspectos clave de la metodología ágil es su capacidad para adaptarse a cambios. A medida que avanzaba el proyecto, se descubrieron nuevos insights y se identificaron áreas de mejora tanto en los procesos de ETL como en los modelos de machine learning aplicados. Gracias a la retroalimentación en tiempo real y la iteración, se ajustaron ciertas etapas del proyecto. Por ejemplo, durante la fase de modelado, el análisis inicial mostró la necesidad de ajustar los parámetros del modelo predictivo para mejorar la precisión. Este descubrimiento fue integrado rápidamente en el ciclo de trabajo mediante Asana, donde se crearon nuevas tareas relacionadas con la recolección y ajuste de datos y modelos.

Asana no solo facilitó la organización del flujo de trabajo, sino que también proporcionó visibilidad constante sobre los hitos clave del proyecto. Gracias a la

herramienta, se pudo realizar un seguimiento detallado de los avances en cada sprint, asegurando que las tareas críticas, como la configuración de pipelines y la creación de visualizaciones en Power BI, se completaran a tiempo. La posibilidad de asignar prioridades y recursos en tiempo real permitió que se ajustaran los tiempos de entrega cuando surgían desafíos imprevistos, manteniendo el proyecto dentro de los plazos acordados.

Métodos, materiales y tecnologías de uso potencial

Aprendizaje Automático: Se implementarán métodos de *machine learning* para mejorar la precisión de los modelos predictivos mediante el uso de técnicas avanzadas de entrenamiento y ajuste.

Análisis Exploratorio de Datos (EDA): Se llevará a cabo un análisis exploratorio exhaustivo para entender mejor la estructura y las características del conjunto de datos. Esto incluirá la identificación de patrones, la detección de valores atípicos y la evaluación de la calidad de los datos. Aplicando técnicas de visualización y estadísticas descriptivas para obtener información valiosa que guiará la toma de decisiones y mejorará la interpretación de los resultados del modelo.

MS SQL Server: Los datos procesados serán almacenados y gestionados en Microsoft SQL Server, que permitirá organizar la información en un Data Warehouse. Esto asegurará que los datos estén optimizados para consultas y análisis posteriores.

Python: Se utilizará Python como lenguaje principal para la implementación de algoritmos de análisis y modelos predictivos. Librerías como Pandas, NumPy, Scikit-learn y TensorFlow facilitarán el procesamiento de datos, la construcción y evaluación de modelos, y la visualización de resultados.

Spoon Pentaho: La herramienta Spoon de Pentaho se utilizará para llevar a cabo los procesos de Extracción, Transformación y Carga (ETL). Con esta herramienta, los datos de los archivos CSV de wearables y otros datasets serán limpiados, transformados y cargados en SQL Server para su posterior análisis.

Modelos de aprendizaje automático

Según el material suministrado parte de la Universidad IMF en el módulo de Modelos Analíticos, material de Ediciones Robles (s.f.), se exponen diversos modelos analíticos y algoritmos empleados en la ciencia de datos. Un modelo se define como una representación simplificada de una porción de la realidad que se busca analizar, estableciendo una serie

de suposiciones sobre cómo funciona esa realidad. Los algoritmos, por otro lado, son instrucciones o recetas utilizadas para realizar cálculos con base en los modelos establecidos. Juntos, modelos y algoritmos son fundamentales para llevar a cabo análisis predictivos, descriptivos y prescriptivos, permitiendo el manejo efectivo de grandes volúmenes de datos y la solución de problemas como la regresión, clasificación y análisis bayesiano, algunos modelos utilizados para el proceso y desarrollo de este proyecto son los siguientes:

K-Nearest Neighbors (KNN): El algoritmo de los vecinos más cercanos se basa en la premisa de que los datos nuevos se clasifican según los valores de sus vecinos más próximos. Según el material suministrado en el módulo de Aprendizaje Automático, Ediciones Roble (s. f.), el KNN es tan útil para temas y problemas de regresión como para clasificación, este modelo trabaja por medio del cálculo de las distancias entre puntos, y prediciendo la etiqueta o valor con base en las instancias más cercanas. Es particularmente efectivo en data sets pequeños y cuando se quiere capturar relaciones no lineales entre las variables.

Linear Discriminant Analysis (LDA): según el material suministrado en el curso de Aprendizaje automático, Ediciones Roble (s. f.), el análisis discriminante lineal es una técnica que maximiza la separación entre las clases al proyectar los datos en un espacio de menor dimensión. Aunque en el documento no se cubre de manera exclusiva, se explica cómo los modelos estadísticos lineales son capaces de identificar límites de decisión claros entre diferentes clases.

Logistic Regression (Regresión Logística): Siguiendo la información y material visto en el módulo de Aprendizaje automático, Ediciones Roble (s. f.), el modelo de regresión logística es uno de los más utilizados comúnmente en lo que respecta para problemas de clasificación binaria, donde la variable de respuesta es categórica. La regresión logística calcula la probabilidad de que una instancia pertenezca a una clase u otra utilizando una función sigmoide para transformar los resultados de la regresión lineal en probabilidades.

Decision Tree Classifier (CART) Arboles de Decisión: En lo que respecta a este modelo, según Ediciones Roble (s. f.) este tipo de modelos caen en el tipo de modelos que solucionan problemas de regresión y clasificación, funcionando por medio de que clasifican los datos dividiendo el espacio de características en regiones homogéneas. Según Ediciones Roble (s. f.) Este proceso se realiza mediante la creación de reglas basadas en

la variable que mejor separa las clases en cada nodo del árbol, lo que resulta en un modelo fácil de interpretar y visualizar.

Naive Bayes (Gaussian Naive Bayes): El algoritmo Naive Bayes, basado en el teorema de Bayes, asume que todas las características son condicionalmente independientes entre sí. Aunque esta suposición puede no ser realista en algunos casos, el algoritmo es sorprendentemente eficaz para muchos problemas de clasificación (Ediciones Robles).

Support Vector Classifier (SVC): Según Ediciones Roble (s. f.) El SVC busca el hiperplano que mejor separa dos clases en un conjunto de datos, maximizando la distancia entre el hiperplano y los puntos más cercanos de cada clase, conocidos como vectores de soporte.

Redes Neuronales Artificiales (ANN): Según Ediciones Roble (s. f.) material para el módulo de Aprendizaje automático, las redes neuronales corresponden a los algoritmos que se utilizan como parte de modelos más profundos, estas mismas se les llama así porque se crearon inspiradas en el comportamiento y estructura de las neuronas humanas por medio de la neurociencia, esto es un indicador que estos algoritmos y modelos buscan imitar el comportamiento del cerebro a través de neuronas artificiales interconectadas. Estas redes son especialmente útiles para problemas de clasificación y regresión cuando hay una gran cantidad de datos. Las redes neuronales trabajan por medio de capas, y conforme se agregan capas y capacidad analítica, sumado a un grupo extenso de datos logran un mayor rendimiento que otros modelos.

Random Forest (Bosques Aleatorios): Según Ediciones Roble (s. f.) Este modelo combina múltiples árboles de decisión, por lo tanto forma parte de los modelos que utilizan los denominados algoritmos ensamblados, son aquellos que actual en función de usar algoritmos simples, por lo tanto el Random Forest usa estos algoritmos ensamblados y cada bosque es entrenado con diferentes subconjuntos de los datos y variables. Al combinar los resultados de todos los árboles, el modelo reduce el riesgo de sobreajuste y mejora la precisión en comparación con un solo árbol de decisión.

Gradient Boosting (XGBoost, LightGBM): Los algoritmos de boosting, como XGBoost y LightGBM, construyen modelos de forma secuencial. Cada modelo nuevo corrige los errores de los modelos anteriores, lo que los convierte en herramientas poderosas para problemas de clasificación y regresión complejos, según Ediciones Roble (s. f.).

Acceso a los modelos desarrollados

Todo lo referente al desarrollo y modelos necesarios para este proyecto final de máster, se encuentra contenido en la siguiente carpeta con libre acceso.

https://drive.google.com/drive/folders/1tAeWofqjrC3-VBZm2KEBC1hlDmzQYLz4?usp=drive_link

Materiales y Conjunto de Datos: Datasets

El presente trabajo se basa en el análisis de dos conjuntos de datos provenientes de diferentes fuentes que permiten estudiar tanto la actividad física diaria de usuarios de dispositivos IoT como el riesgo de enfermedades cardíacas. Estos conjuntos de datos son fundamentales para capturar y analizar patrones de comportamiento en salud y bienestar.

El primer conjunto de datos proviene del FitBit Fitness Tracker Data, un dataset disponible en Kaggle que recopila información detallada sobre la actividad física de usuarios de rastreadores Fitbit. Este conjunto de datos se enfoca en la recopilación automática de métricas relacionadas con la actividad física, lo que resulta crucial en el contexto de los wearables y su capacidad para monitorear indicadores de salud. El dataset está estructurado en varias tablas que abarcan diferentes aspectos del comportamiento físico diario, incluyendo la actividad total, las calorías quemadas, la intensidad de la actividad y el tiempo sedentario.

El segundo conjunto de datos proviene del Cleveland Heart Disease Dataset del UCI Machine Learning Repository, utilizado para la predicción y clasificación de enfermedades cardíacas. Este dataset incluye datos clínicos que permiten identificar factores de riesgo cardiovascular, proporcionando características esenciales como la edad, sexo, presión arterial, niveles de colesterol y resultados de pruebas cardíacas. Dicho dataset es ampliamente utilizado en modelos de predicción de enfermedades cardíacas, aportando información crítica sobre la salud cardíaca de los pacientes.

Ambos conjuntos de datos son complementarios y permiten realizar un análisis integral del impacto de la actividad física monitorizada por dispositivos IoT y el riesgo de enfermedades cardíacas, contribuyendo a una mejor comprensión de los factores que influyen en la salud general de los individuos.

EXPLICACIÓN Y ANÁLISIS DE LOS RESULTADOS

IoT y el uso de wearables - FitBit Fitness Tracker Data

El dataset de Fitbit ofrece un análisis exhaustivo de la actividad física de un usuario a través de varias métricas clave recopiladas por un rastreador de fitness Fitbit, es de mucha importancia en lo que respecta en el uso de la captura de los datos provenientes de wearables dedicados al control y seguimiento de indicadores de salud. Este conjunto de datos está diseñado para proporcionar una visión integral del comportamiento físico diario y las tendencias a lo largo del tiempo, y está dividido en varias tablas con diferentes aspectos de la actividad física del usuario.

El acceso en que puede ser consultado este conjunto de datos es el siguiente:

<https://www.kaggle.com/datasets/arashnic/fitbit>

1. Daily Activity:

- **Descripción:** Registra datos diarios sobre la actividad física general del usuario. Incluye métricas como el número de pasos dados, la distancia recorrida, las calorías quemadas, y la cantidad de minutos dedicados a actividades de diferentes intensidades.
- **Columnas Principales:**
 - Date: Fecha del registro.
 - Calories: Total de calorías quemadas durante el día.
 - Distance: Distancia total recorrida en kilómetros.
 - Steps: Número total de pasos dados.
 - SedentaryMinutes, LightlyActiveMinutes, FairlyActiveMinutes, VeryActiveMinutes: Minutos dedicados a diferentes niveles de actividad.

2. Daily Intensity:

- **Descripción:** Similar a la tabla de Daily Activity, pero centrado en la intensidad de la actividad física diaria. Permite un análisis detallado de cómo varía la intensidad de las actividades a lo largo del tiempo.

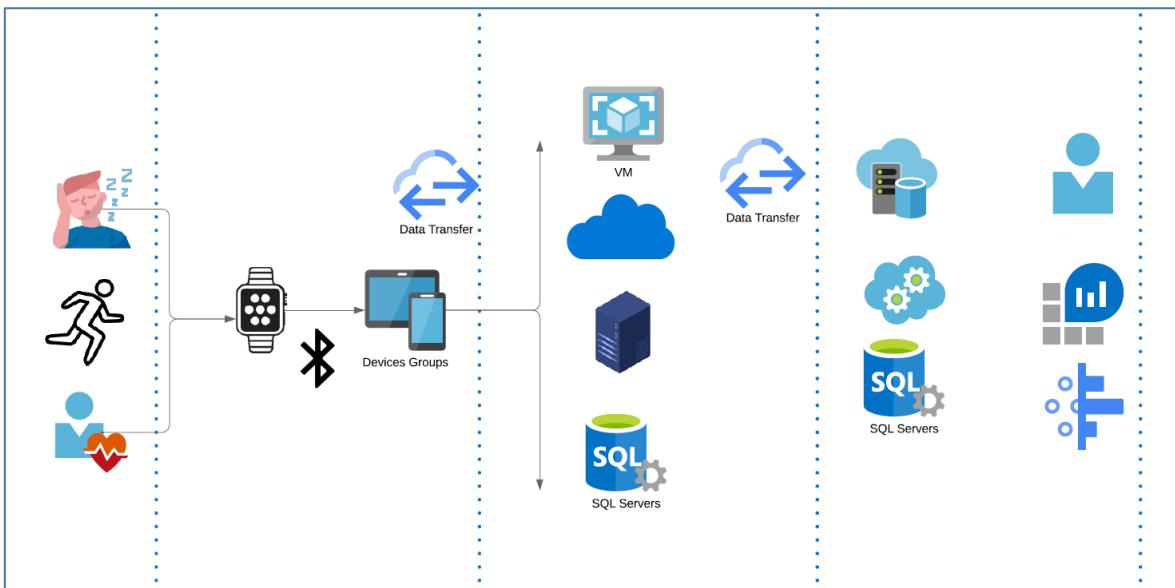


Diagrama de creación propia con lucid.app

Descripción del proceso de interacción de los dispositivos de IoT *wearables* dedicados al fitness o monitoreo continuo de actividades diarias, estos dispositivos pueden ser desde un smartwatch o una banda inteligente o el mismo smartphone que los usuarios tienen consigo prácticamente todo el día.



Imagen tomada de freepik
Designed by macrovector / Freepik

Paso 1: Dispositivos Wearables (IoT Devices)

Los dispositivos wearables como smartwatches y smartbands son los dispositivos que se encuentran más a disposición del público en general, por lo tanto en el anterior diagrama de proceso de uso de wearables se utiliza un icono del tipo de smartwatch, el mismo se conecta con otros dispositivos para trasladar la data de las rutinas o ejercicios del usuario, estos datos información o detalles biométricos tales como pulso cardiaco, capacidad de oxigenación, respiración , presión arterial, pasos, distancia recorrida, entre otros. Lo interesante de estos dispositivos modernos radia en que poseen sensores lo suficientemente pequeños pero potentes para capturar esos datos y permitir que se puedan convertir en información.

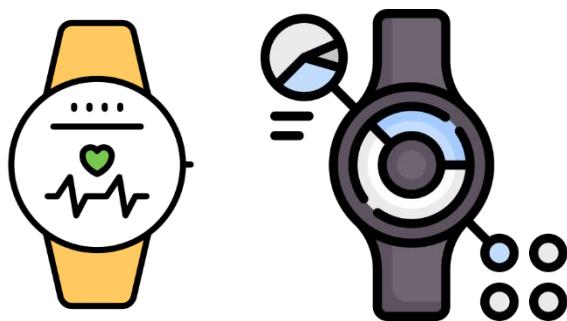


Imagen tomada de freepik
Designed by macrovector / Freepik

Transferencia de datos: Si los datos o bits generados por parte de los dispositivos inteligentes (wearables) se quedan en el dispositivo serán de poca utilidad, la importancia radica en que los datos se pueden o tienen la capacidad de ser transferidos de un dispositivo a otro, ya sea por comodidad del usuario o para ser vistos o tratados por medio de una app específica para cada marca o para control y seguimiento específico de los indicadores generados durante un periodo específico. Por lo tanto esta transferencia de datos se conoce como una sincronización o sharing de datos entre dispositivos, por medio del uso de tecnología de conexiones Bluetooth



Imagen tomada de freepik
Designed by macrovector / Freepik

Paso 2: Proceso ETL (Extract, Transform, Load)

A partir de este punto es cuando los datos comienzan a cobrar mayor relevancia aportando información, por lo tanto este proceso ETL es cuando se le da un valor agregado a los datos, se extraen o recopilan de las fuentes de los datos, posteriormente se transforman y finalmente se cargan (Load), lo importante del ETL y procesos de Pipelines, corresponde a que se establecen reglas sobre los orígenes de los datos y el tratamiento de los mismos, para procesarlos de manera automatizada cuando estos corresponden a grandes cantidades y volúmenes , características propias del big data..

Extracción: La extracción de datos implica recuperar los datos acumulados desde fuentes estructuradas como archivos CSV o JSON, que son exportados desde las aplicaciones móviles asociadas a los dispositivos wearables.

Transformación: En esta fase, se realizan operaciones de limpieza y preparación de datos, incluyendo la eliminación de duplicados y la corrección de errores de formato. Además, el 'feature engineering' se emplea para衍生 nuevas variables que puedan enriquecer el análisis, como la conversión de fechas a días de la semana para observar patrones comportamentales.

Carga: Los datos limpios y transformados se cargan finalmente en sistemas de gestión de bases de datos como SQL Server, preparándolos para análisis y consultas subsiguientes.

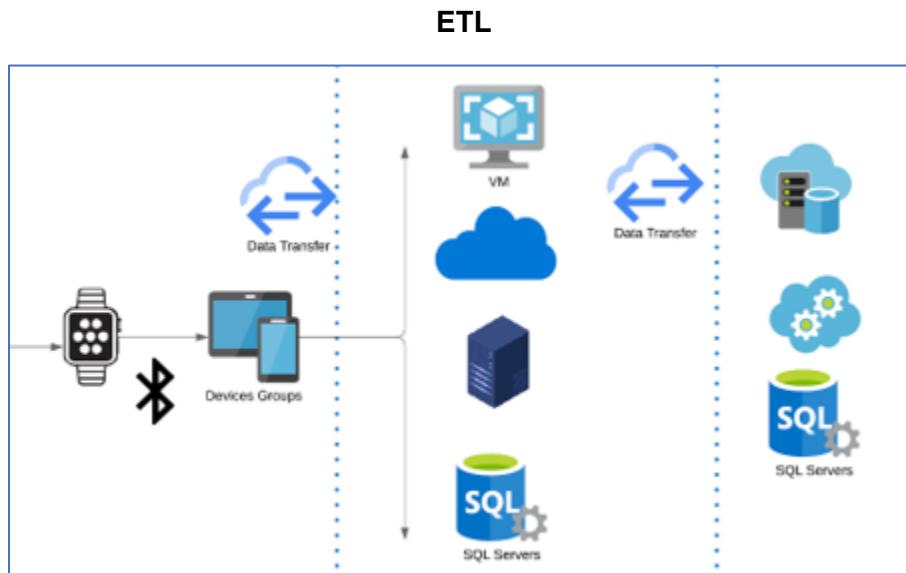


Diagrama de creación propia con lucid.app

Paso 3: Bases de Datos (Staging & Data Warehouse)

Staging: Según la explicación y contenido del material IMF el área de *Staging* corresponde a la parte de carga de los datos y su almacenamiento en formato crudo o tal y como vienen los datos, estos datos tienden a mantenerse de forma temporal para posteriormente ser trasladados al Data Warehouse.

Data Warehouse: Esta parte del proceso su función es de almacenar los datos preparados y procesados para estar listos para el trámite y uso de los siguientes pasos del proceso (IMF, n.d.), por lo tanto se puede decir que los datos que se encuentran en este “almacen de datos” ya han pasado por ciertos niveles de transformaciones para permitir que las herramientas de visualización e inteligencia de negocios puedan extraer información útil.

Paso 4: Visualización de Datos

Conexión y Extracción de Datos: Utilizando herramientas como SQL Server Management Studio o servicios en la nube como Google BigQuery, los analistas pueden extraer datos organizados desde el Data Warehouse.

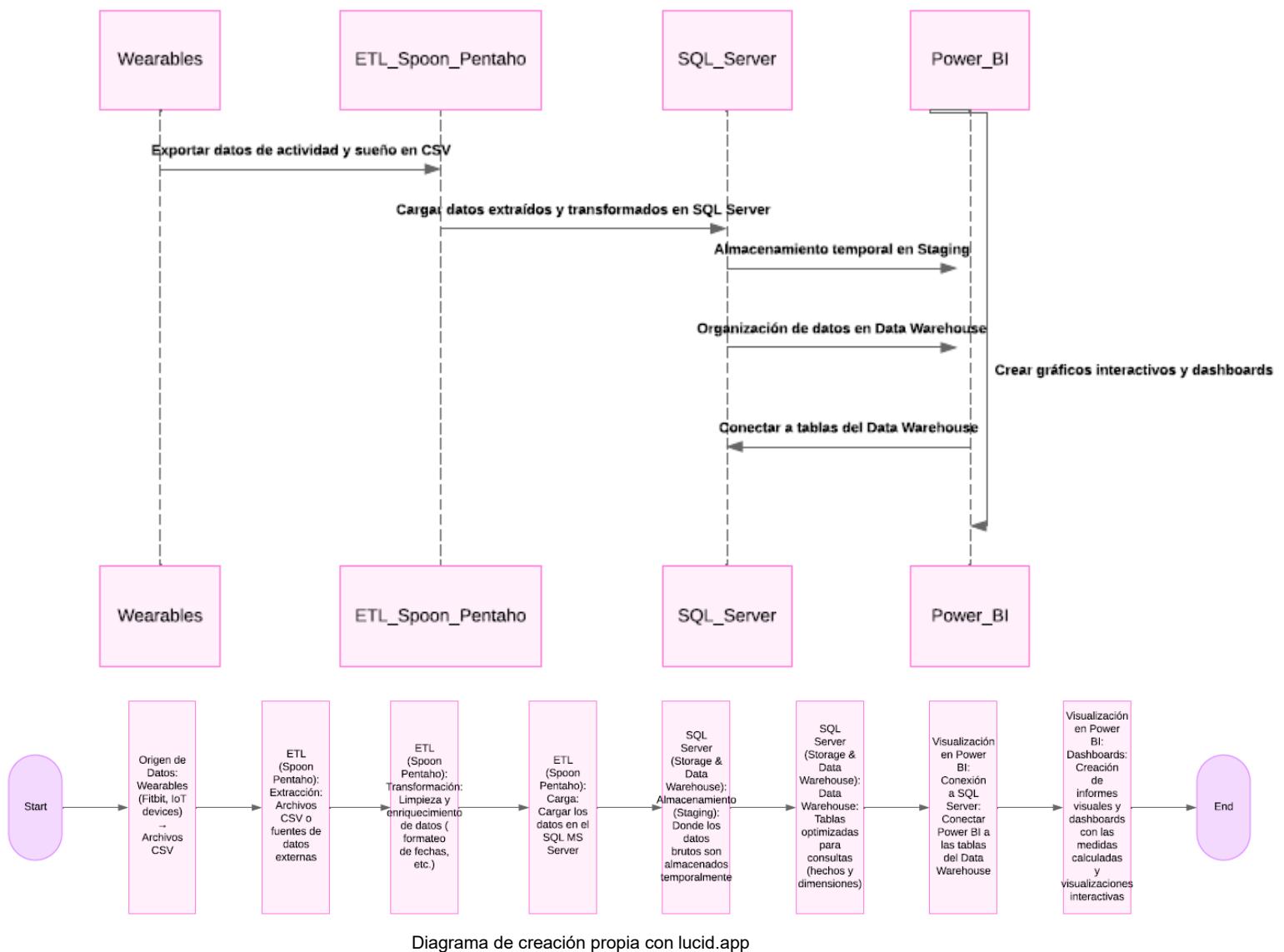
Creación de Dashboards: La visualización de los datos es tan importante como los pasos anteriores, en vista que los pasos técnicos culminan con el paso de permitir que los

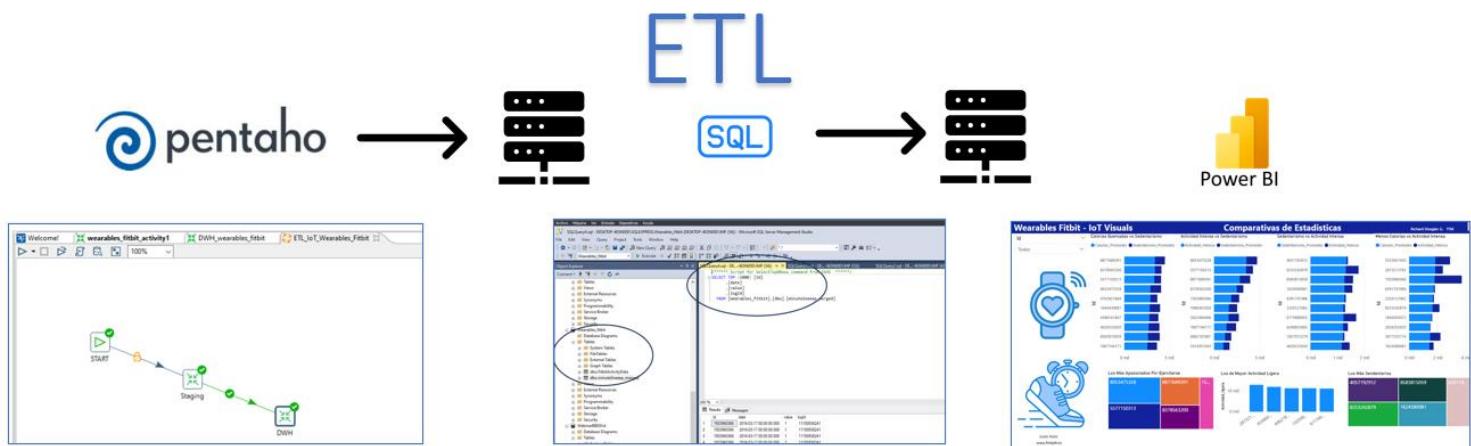
usuarios finales y quienes toman decisiones puedan tener claridad sobre los datos y ver patrones en estos. Las herramientas de visualización tales como Tableau, Microsoft Power BI, o Qlik Sense permiten crear dashboards interactivos.

Cada uno de estos pasos no solo es fundamental para el proceso de análisis de datos, sino que también requiere una consideración cuidadosa de las mejores prácticas y estándares en la industria para asegurar resultados precisos y útiles.

Desarrollo del Proceso ETL – Fitbit Wearables

Diagrama de Proceso de Captura y transformación de datos





En lo que respecta al dataset de Fitbit Wearables, para el proceso de análisis y transformaciones se ha optado por crear un ambiente de simulación de un proceso ETL, en el que se integran los aspectos tales como la captura de datos, staging, storage, distribución y visualización de datos, por medio de un ambiente de máquina virtual tal y como se desarrolló durante el programa de formación.

Por lo tanto se toma en consideración los procesos que intervienen en relación a IoT relacionado a los wearables destinados a fitness y como estos generan y capturan la información para posteriormente ser almacenados y distribuidos para generar insights.

El propósito de este primer dataset es ser útil para la explicación del funcionamiento de los procesos de IoT en la captura de información de las diferentes actividades de las personas, finalmente crear un ambiente en el se genera un Pipeline que entrelaza una serie de sistemas que interactúan entre sí para la generación y transformación de los datos.

1. Captura de Datos desde los Wearables (IoT Devices)

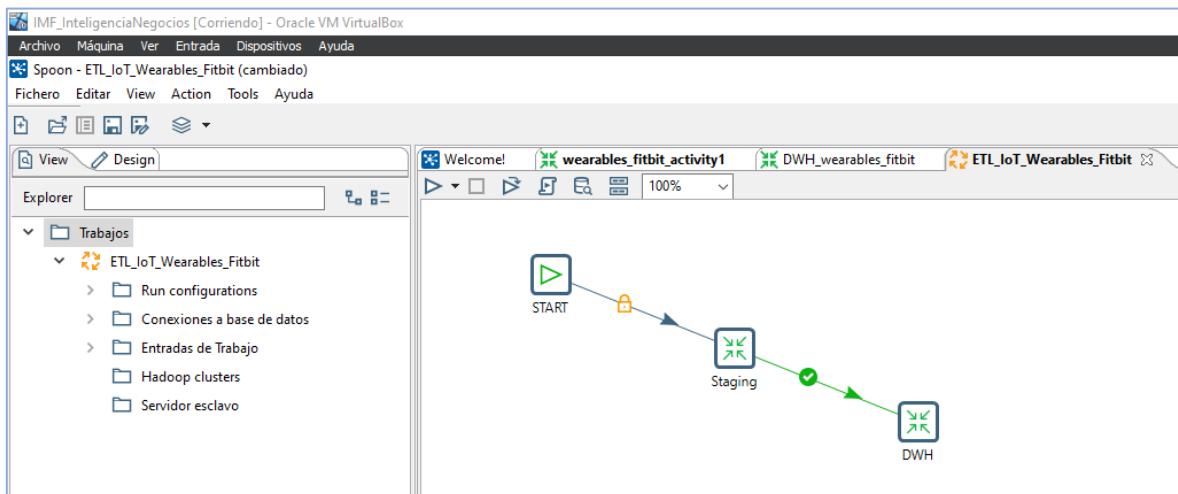
El proceso de

Comportamiento del usuario: El usuario lleva consigo un dispositivo wearable que recopila información sobre su actividad física, como los pasos, las calorías quemadas, los minutos activos y los datos de sueño.

Exportación de datos: Estos dispositivos recogen los datos y los sincronizan a través de una aplicación móvil. Los datos se exportan a archivos CSV u otros formatos.

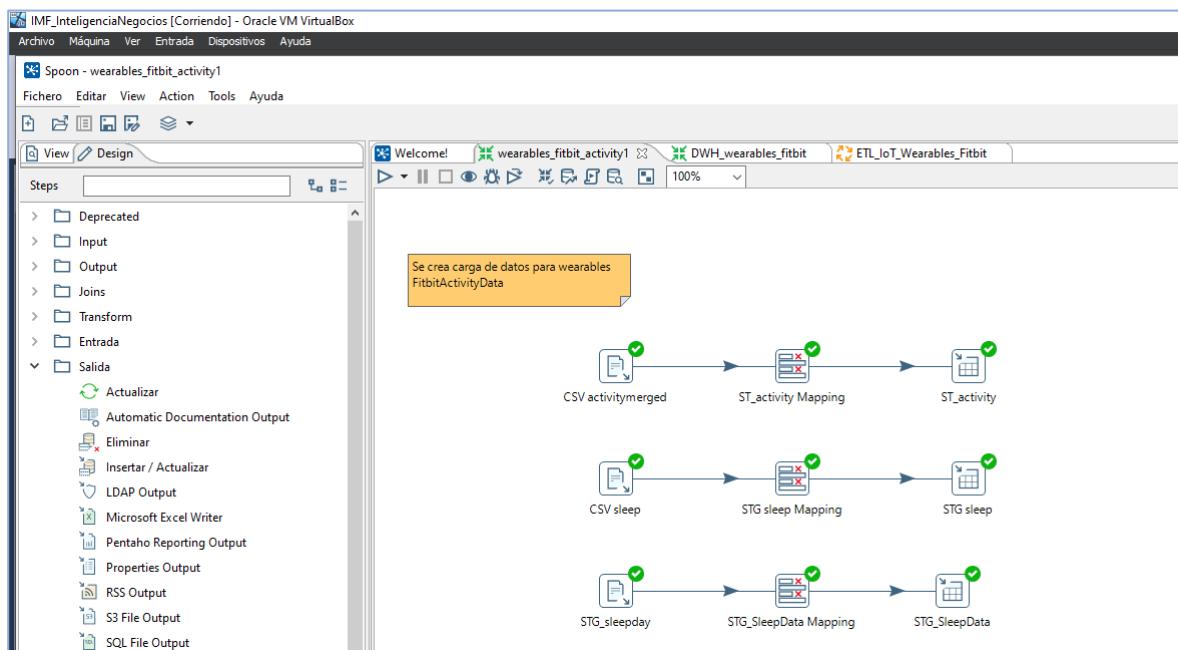
Nombre	Estado	Fecha de modificación	Tipo	Tamaño
dailyActivity_merged1.csv	✓	5/9/2024 16:03	Archivo de valores...	51 KB
heartrate_seconds_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	40 108 KB
hourlyCalories_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	853 KB
hourlyIntensities_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	949 KB
hourlySteps_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	846 KB
minuteCaloriesNarrow_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	70 764 KB
minuteIntensitiesNarrow_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	49 340 KB
minuteMETsNarrow_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	50 753 KB
minuteSleep_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	9 098 KB
minuteStepsNarrow_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	49 505 KB
weightLogInfo_merged.csv	✓	2/3/2024 20:47	Archivo de valores...	4 KB

2. Extracción, Transformación y Carga (ETL) en Spoon Pentaho



- Extracción: Los datos en formato CSV se extraen para ser procesados mediante Spoon Pentaho, una herramienta ETL que gestiona la automatización del flujo de datos.
- Transformación: Los datos brutos extraídos se limpian, se eliminan duplicados o inconsistencias, asegurando que los datos sean consistentes y estén listos para su carga.
- Carga: Los datos transformados se cargan en un servidor SQL Server, donde pasarán por una fase de almacenamiento temporal.

Esta captura demuestra el proceso de trabajo en Spoon Pentaho, en el que se crea la carga de los datos.



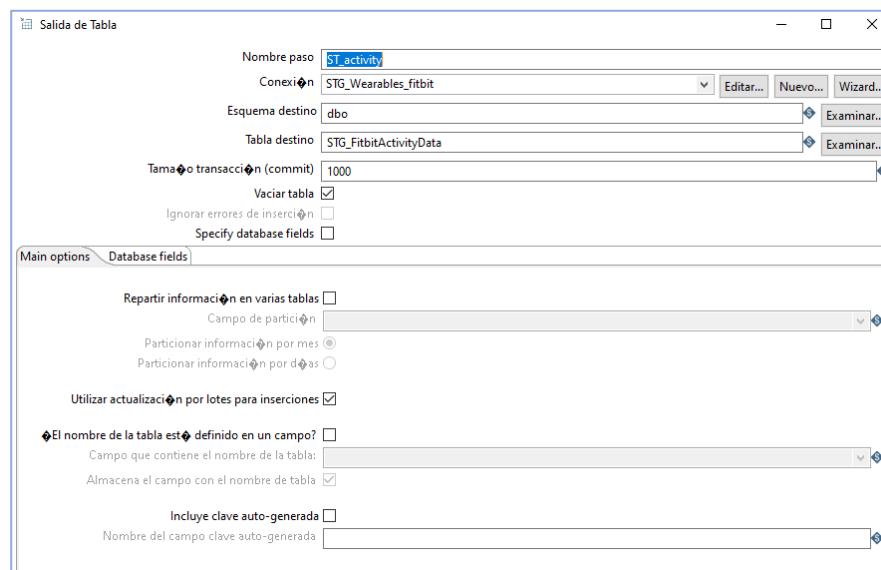
CSV activitymerged

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Id	Number	#	20	0	€	.	.	ninguno
2	ActivityDate	Date	MM/dd/yyyy	10	0	€	.	.	ninguno
3	TotalSteps	Integer	#	20	0	€	.	.	ninguno
4	TotalDistance	Number	#.#	20	0	€	.	.	ninguno
5	TrackerDistance	Number	#.#	20	0	€	.	.	ninguno
6	LoggedActivitiesDistance	Integer	#	20	0	€	.	.	ninguno
7	VeryActiveDistance	Number	#.#	20	0	€	.	.	ninguno
8	ModeratelyActiveDistance	Number	#.#	20	0	€	.	.	ninguno
9	LightActiveDistance	Number	#.#	20	0	€	.	.	ninguno
10	SedentaryActiveDistance	Number	#.#	20	0	€	.	.	ninguno
11	VeryActiveMinutes	Integer	#	20	0	€	.	.	ninguno
12	FairlyActiveMinutes	Integer	#	20	0	€	.	.	ninguno
13	LightActiveMinutes	Integer	#	20	0	€	.	.	ninguno
14	SedentaryMinutes	Integer	#	20	0	€	.	.	ninguno
15	Calories	Integer	#	20	0	€	.	.	ninguno

Esta captura muestra el proceso de staging, la cual envía los datos en bruto hacia la base de datos staging en MS SQL Server.



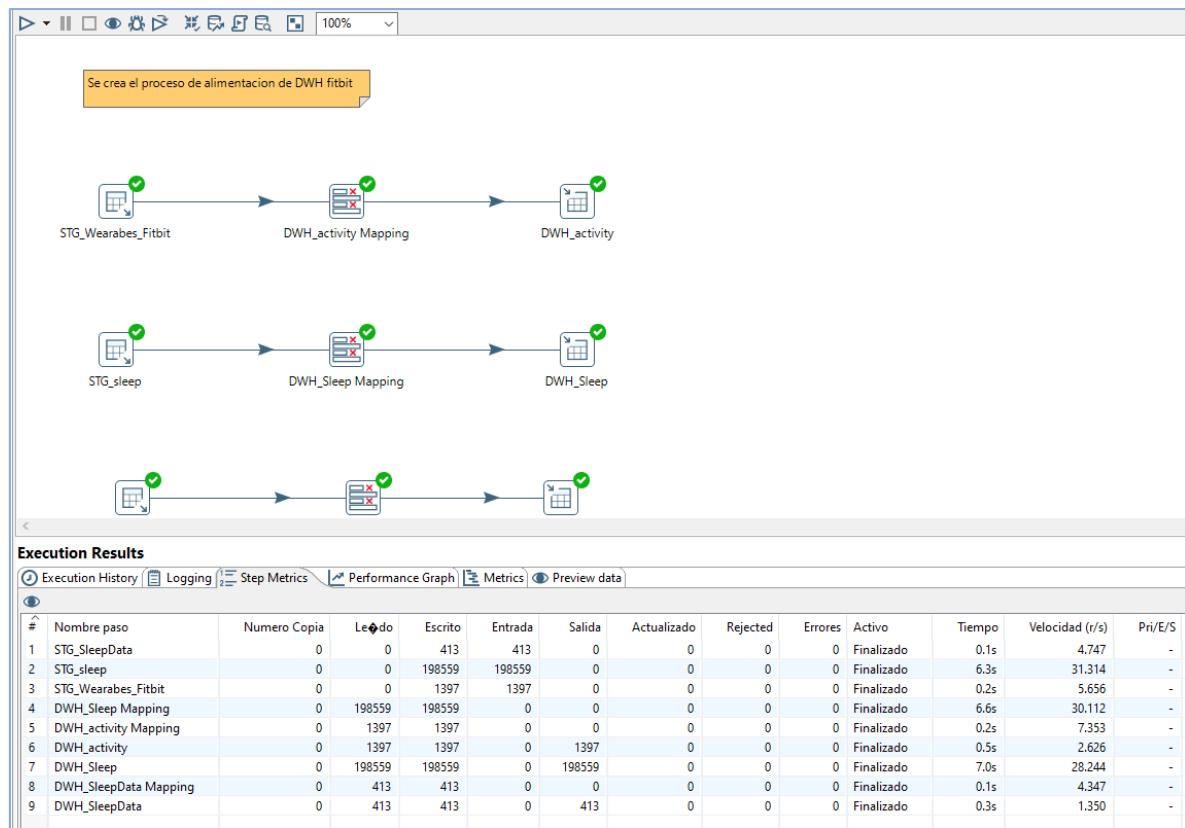
ST_activity



3. Almacenamiento en SQL Server (Staging & Data Warehouse)

Staging: Los datos procesados se almacenan inicialmente en un área de Staging, una zona temporal donde los datos brutos son guardados antes de ser organizados.

Data Warehouse: Posteriormente, los datos son trasladados al **Data Warehouse**.



Esta captura muestra la base de datos staging en SQL Server.

```
SQLQuery2.sql - DE...-4ESN00D\IMF (55)*  X  SQLQuery1.sql - DE...-4ESN00D\IMF (53)*

CREATE DATABASE STG_Wearables_fitbit;

-- Paso 2: Seleccionar la base de datos para usarla
USE STG_Wearables_fitbit;

-- Paso 3: Crear la tabla minuteSneep_merged
CREATE TABLE minuteSneep_merged (
    Id BIGINT,
    date DATETIME,
    value INT,
    logId BIGINT
);

-- Paso 4: Crear la tabla FitbitActivityData
CREATE TABLE FitbitActivityData (
    Id BIGINT,
    ActivityDate DATE,
    TotalSteps INT,
    TotalDistance DECIMAL(10, 5),
    TrackerDistance DECIMAL(10, 5),
    LoggedActivitiesDistance DECIMAL(10, 5),
    VeryActiveDistance DECIMAL(10, 5),
    ModeratelyActiveDistance DECIMAL(10, 5),
    LightActiveDistance DECIMAL(10, 5),
    SedentaryActiveDistance DECIMAL(10, 5),
    VeryActiveMinutes INT,
    FairlyActiveMinutes INT,
    LightlyActiveMinutes INT,
    SedentaryMinutes INT,
    Calories INT
);
```

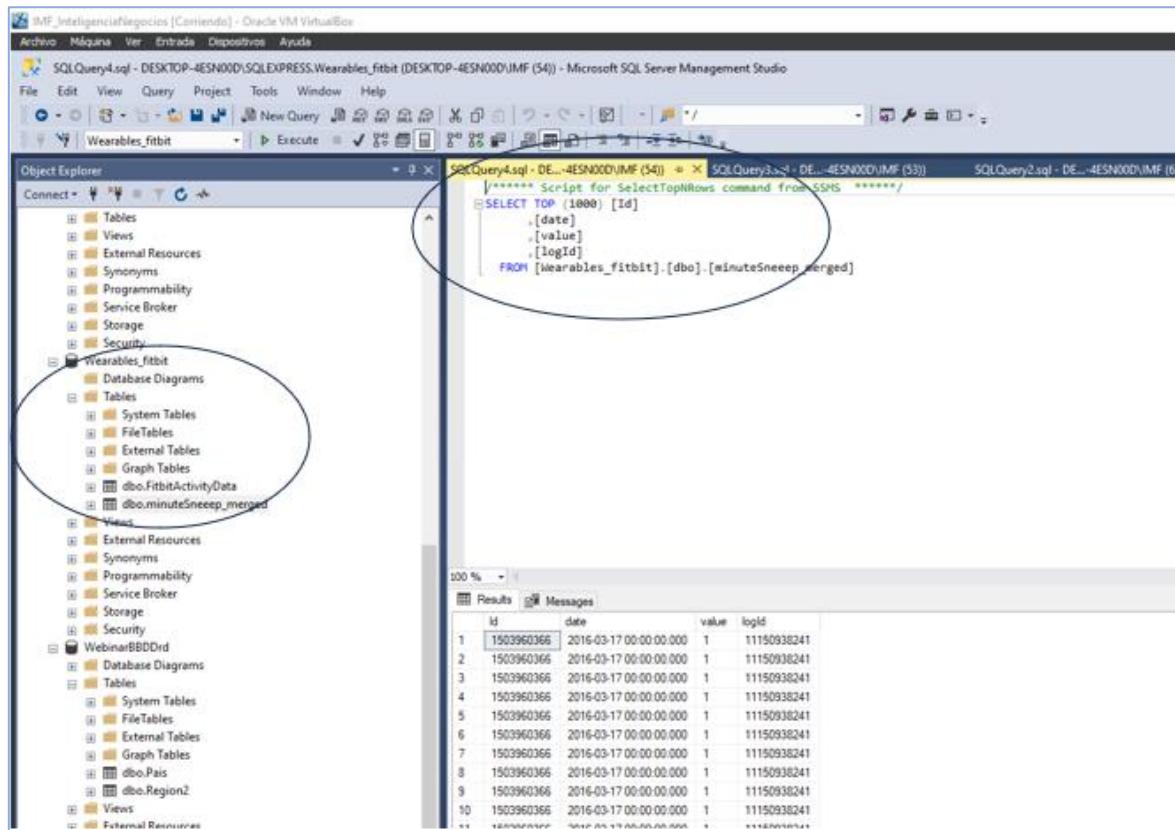
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left lists databases such as DESKTOP-4ESN00D\SQLEXPRESS, DWH_WWBI, Hoteles_db, Northwind, Northwind_DW, and STG_Wearables_fitbit. Under STG_Wearables_fitbit, the Tables node is expanded, showing System Tables, FileTables, External Tables, Graph Tables, and Views. A circled area highlights the Views node. The central pane contains a query window titled 'SQLQuery2.sql - DESKTOP-4ESN00D\IMF (63)' with the following T-SQL script:

```
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [Id]
      ,[ActivityDate]
      ,[TotalSteps]
      ,[TotalDistance]
      ,[TrackerDistance]
      ,[LoggedActivitiesDistance]
      ,[VeryActiveDistance]
      ,[ModeratelyActiveDistance]
      ,[LightActiveDistance]
      ,[SedentaryActiveDistance]
      ,[VeryActiveMinutes]
      ,[FairlyActiveMinutes]
      ,[LightlyActiveMinutes]
      ,[SedentaryMinutes]
      ,[Calories]
  FROM [STG_Wearables_fitbit].[dbo].[STG_FitbitActivityData]
```

The Results pane at the bottom shows the output of the query, displaying 8 rows of data:

	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance
1	1503960366	2016-03-25	11004	7.00000	7.00000	0.00000	3.00000
2	1503960366	2016-03-26	17609	12.00000	12.00000	0.00000	7.00000
3	1503960366	2016-03-27	12736	9.00000	9.00000	0.00000	5.00000
4	1503960366	2016-03-28	13231	9.00000	9.00000	0.00000	3.00000
5	1503960366	2016-03-29	12041	8.00000	8.00000	0.00000	2.00000
6	1503960366	2016-03-30	10970	7.00000	7.00000	0.00000	2.00000
7	1503960366	2016-03-31	12256	8.00000	8.00000	0.00000	2.00000
8	1503960366	2016-04-01	12262	8.00000	8.00000	0.00000	3.00000

Esta captura muestra la base de datos ware house, la cual corresponde a la base de datos que exporta los datos.



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left shows the database structure, with a callout circle highlighting the 'Wearables_fitbit' database. The central pane displays a T-SQL script for a 'SelectTopNRows' command from SSMS:

```
/*===== Script for SelectTopNRows command from SSMS =====*/
SELECT TOP (1000) [Id]
      ,[date]
      ,[value]
      ,[logId]
  FROM [Wearables_fitbit].[dbo].[minuteSleep_merged]
```

The Results pane at the bottom shows the output of the query, which consists of 10 rows of data:

	Id	date	value	logId
1	1503960366	2016-03-17 00:00:00.000	1	11150938241
2	1503960366	2016-03-17 00:00:00.000	1	11150938241
3	1503960366	2016-03-17 00:00:00.000	1	11150938241
4	1503960366	2016-03-17 00:00:00.000	1	11150938241
5	1503960366	2016-03-17 00:00:00.000	1	11150938241
6	1503960366	2016-03-17 00:00:00.000	1	11150938241
7	1503960366	2016-03-17 00:00:00.000	1	11150938241
8	1503960366	2016-03-17 00:00:00.000	1	11150938241
9	1503960366	2016-03-17 00:00:00.000	1	11150938241
10	1503960366	2016-03-17 00:00:00.000	1	11150938241

Agregar información de valor a una columna : crear el día de la semana -Dayofweek

The screenshot shows a SQL Server Management Studio window with two tabs: 'SQLQuery9.sqi' and 'SQLQuery8.sqi'. The code in 'SQLQuery9.sqi' is as follows:

```
--Feature Engineering
--Crear una columna con el dia de la semana

--agregar una nueva columna
ALTER TABLE [Wearables_fitbit].[dbo].[FitbitActivityData]
ADD DayOfWeek VARCHAR(10);

--agregar la inofrmacion del dia de la semana
UPDATE [Wearables_fitbit].[dbo].[FitbitActivityData]
SET DayOfWeek = DATENAME(weekday, [ActivityDate]);

--revision de la informacion cargada
SELECT TOP (10) [Id], [ActivityDate], DayOfWeek
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The results tab displays the following data:

	Id	ActivityDate	DayOfWeek
1	1503960366	2016-03-25	Viernes
2	1503960366	2016-03-26	Sábado
3	1503960366	2016-03-27	Domingo
4	1503960366	2016-03-28	Lunes
5	1503960366	2016-03-29	Martes
6	1503960366	2016-03-30	Miércoles
7	1503960366	2016-03-31	Jueves
8	1503960366	2016-04-01	Viernes
9	1503960366	2016-04-02	Sábado
10	1503960366	2016-04-03	Domingo

Exploración de la característica Steps

The screenshot shows a SQL Server Management Studio window with three tabs at the top: 'SQLQuery4.sql - DE...-4ESN00D\IMF (64)*' (selected), 'SQLQuery3.sql - DE...-4ESN00D\IMF (59)*', and 'SQLQuery2.sql - DE...-4ESN00D\IMF (60)*'. The code in the selected query window is as follows:

```
--Conocer Informacion relevante sobre el dataset
-- Por ejemplo las caracterista de Steps
SELECT
    COUNT(*) AS total_registros,
    AVG(TotalSteps) AS promedio_pasos,
    MIN(TotalSteps) AS minimo_pasos,
    MAX(TotalSteps) AS maximo_pasos
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The results pane shows a single row of data:

	total_registros	promedio_pasos	minimo_pasos	maximo_pasos
1	1397	7280	0	36019

Exploración de la característica Calorías

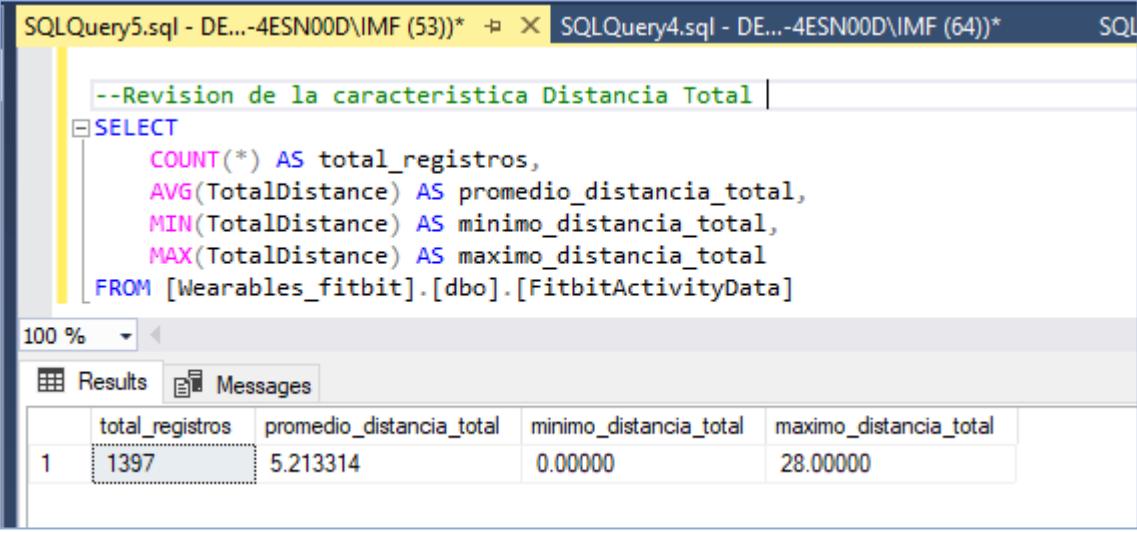
The screenshot shows a SQL Server Management Studio window with three tabs at the top: 'SQLQuery4.sql - DE...-4ESN00D\IMF (64)*' (selected), 'SQLQuery3.sql - DE...-4ESN00D\IMF (59)*', and 'SQLQuery2.sql - DE...-4ESN00D\IMF (60)*'. The code in the selected query window is as follows:

```
--Conocer Informacion relevante sobre el dataset
-- Por ejemplo las caracterista de Calorias
SELECT
    COUNT(*) AS total_registros,
    AVG(Calories) AS promedio_calorias,
    MIN(Calories) AS minimo_calorias,
    MAX(Calories) AS maximo_calorias
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The results pane shows a single row of data:

	total_registros	promedio_calorias	minimo_calorias	maximo_calorias
1	1397	2266	0	4900

Exploración de la característica Distancia Total



The screenshot shows a SQL Server Management Studio (SSMS) window with two tabs: 'SQLQuery5.sql' and 'SQLQuery4.sql'. The 'SQLQuery5.sql' tab contains a SQL query to analyze the 'TotalDistance' column in the 'FitbitActivityData' table. The query includes a comment '--Revision de la caracteristica Distancia Total', a SELECT statement with four aggregate functions (COUNT, AVG, MIN, MAX), and a FROM clause pointing to the table. The results pane shows a single row with four columns: 'total_registros' (1397), 'promedio_distancia_total' (5.213314), 'minimo_distancia_total' (0.00000), and 'maximo_distancia_total' (28.00000).

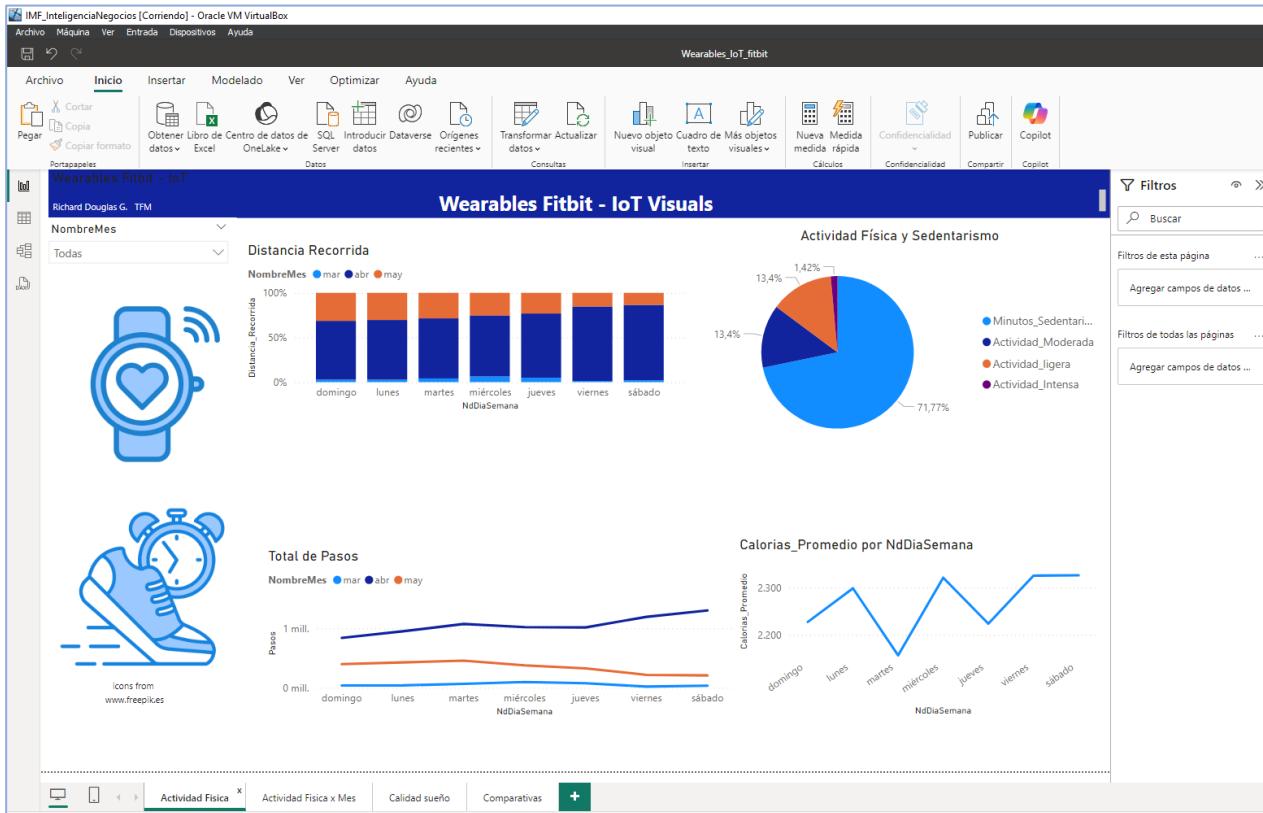
```
--Revision de la caracteristica Distancia Total
SELECT
    COUNT(*) AS total_registros,
    AVG(TotalDistance) AS promedio_distancia_total,
    MIN(TotalDistance) AS minimo_distancia_total,
    MAX(TotalDistance) AS maximo_distancia_total
FROM [Wearables_fitbit].[dbo].[FitbitActivityData]
```

	total_registros	promedio_distancia_total	minimo_distancia_total	maximo_distancia_total
1	1397	5.213314	0.00000	28.00000

4. Visualización en Power BI

Finalmente, llega el apartado de visualización de los datos, donde el análisis se materializa en formas que son más fáciles de entender y comunicar a otros. Este proceso permite visualizar parámetros e información clave que ayuda en la toma de decisiones y en la identificación de patrones significativos en el comportamiento relacionado con la salud y el bienestar.

- **Conexión a SQL Server:** Power BI se conecta a las bases de datos almacenadas en el Data Warehouse para extraer los datos procesados y organizados.
- **Creación de Dashboards:** Se utilizan los datos organizados para crear informes y dashboards interactivos, donde se visualizan las tendencias de actividad física y sueño del usuario.



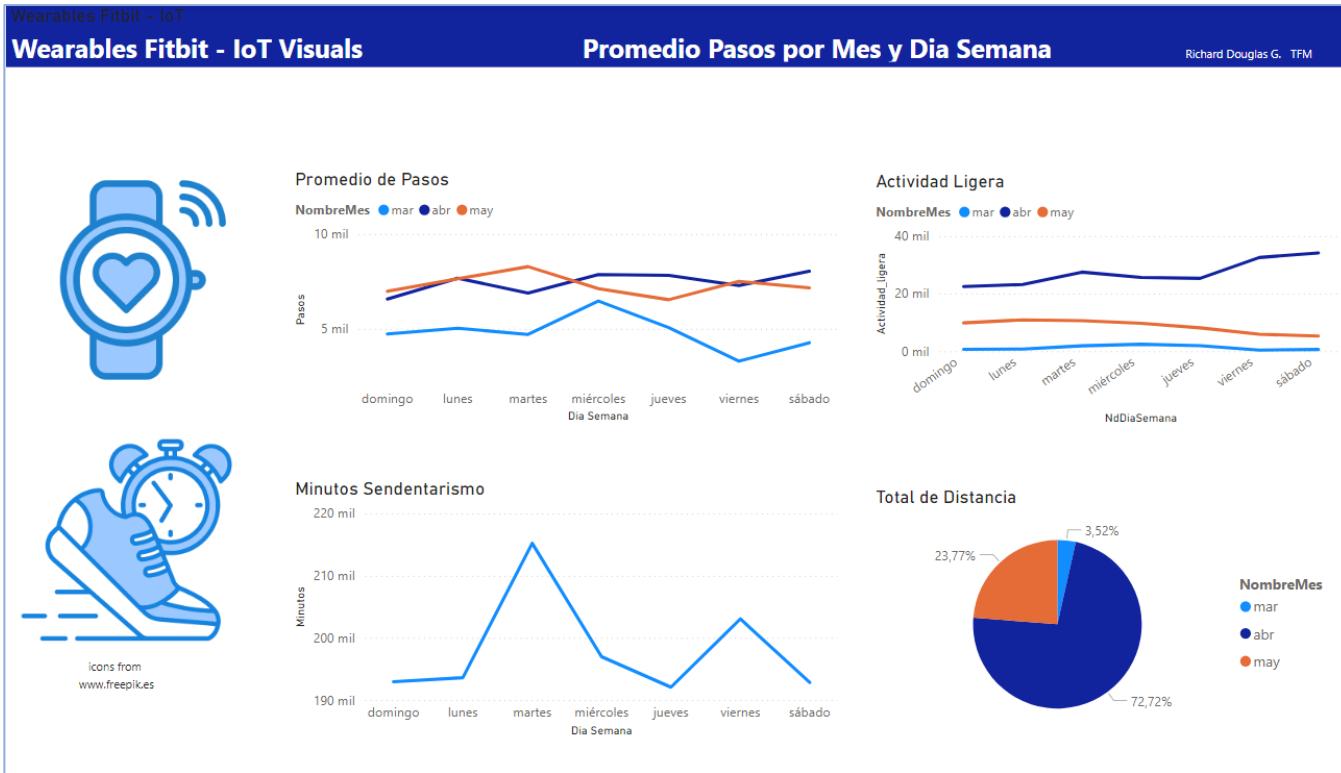
Captura del dashboard. Creación propia

Distancia Recorrida: Un gráfico de barras muestra el porcentaje de distancia recorrida por día de la semana, comparando dos meses (abril y mayo).

Actividad Física y Sedentarismo: Un gráfico circular desglosa el tiempo en actividad sedentaria, moderada, ligera e intensa.

Total de Pasos: Gráfico de líneas que compara los pasos totales durante la semana entre abril y mayo.

Calorías Promedio por Día de la Semana: Visualización de la tendencia de calorías quemadas en promedio por día.



Captura del dashboard. Creación propia

Promedio de Pasos: Un gráfico de líneas compara los pasos promedio por día de la semana en tres meses (marzo, abril y mayo), mostrando fluctuaciones en la actividad física.

Actividad Ligera: También en formato de líneas, muestra la actividad física ligera por día y mes.

Minutos de Sedentarismo: Un gráfico muestra el tiempo de sedentarismo diario, con un aumento notable a mitad de semana.

Total de Distancia: Un gráfico circular muestra que mayo tiene la mayor proporción de distancia recorrida (72.72%).



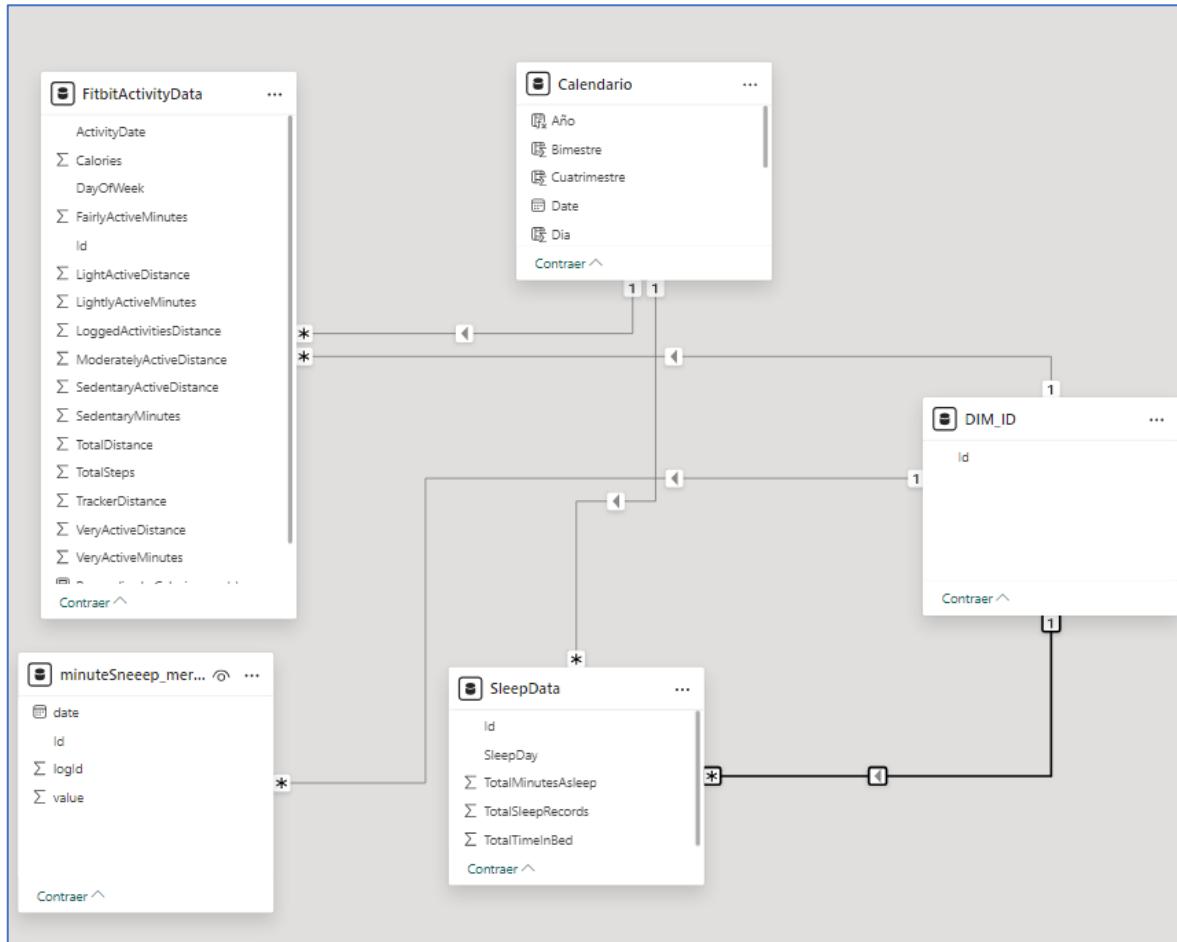
Captura del dashboard. Creación propia

Total Horas de Sueño: Un gráfico de barras compara la cantidad de horas de sueño diarias durante la semana, con un valor cercano a las 3 mil horas en cada día.

Sueño Semanal: Un gráfico de líneas presenta la tendencia de las horas de sueño acumuladas por semana, manteniéndose constante alrededor de 3 mil horas.

Las gráficas que muestran reflejan las horas totales de sueño por día de la semana y la tendencia semanal de sueño acumulado.

Modelo estrella en Power BI



Como se puede ver en la captura, se generan las relaciones entre tablas que contienen los datos, por medio de las claves o llaves principales. Estas relaciones permiten realizar análisis multidimensionales, donde los datos de la actividad física pueden combinarse con la información del sueño y desglosarse por tiempo utilizando la tabla calendario.

Este diseño de modelo estrella optimiza el análisis en Power BI, proporcionando una estructura clara y eficiente para realizar consultas rápidas y construir dashboards interactivos que ayuden en la toma de decisiones basadas en los datos obtenidos de los wearables.

Tabla calendario que es necesaria para el uso y visualizaciones que implican fechas

The screenshot shows the Microsoft Power BI Data Editor interface. The title bar reads "IMF_InteligenciaNegocios [Corriendo] - Oracle VM VirtualBox". The menu bar includes Archivo, Máquina, Ver, Entrada, Dispositivos, and Ayuda. The ribbon tabs are "Herramientas de tablas" and "Herramientas de columnas", with "Herramientas de columnas" currently selected. The main area displays a table titled "1 Calendario = CALENDARAUTO()". The table has 13 columns: Date, Año, Mes, Dia, Cuatrimestre, Trimestre, Bimestre, Semana, DiaSemana, NdDiaSemana, and NombreMes. The data starts from 01/01/2016 and continues through 24/01/2016. The "NombreMes" column shows abbreviations like "ene" for January.

Date	Año	Mes	Dia	Cuatrimestre	Trimestre	Bimestre	Semana	DiaSemana	NdDiaSemana	NombreMes
01/01/2016 0:00:00	2016	1	1	1	1	1	1	1	6	viernes
02/01/2016 0:00:00	2016	1	2	1	1	1	1	1	7	sábado
03/01/2016 0:00:00	2016	1	3	1	1	1	2	1	domingo	ene
04/01/2016 0:00:00	2016	1	4	1	1	1	1	2	2	lunes
05/01/2016 0:00:00	2016	1	5	1	1	1	1	2	3	martes
06/01/2016 0:00:00	2016	1	6	1	1	1	1	2	4	miércoles
07/01/2016 0:00:00	2016	1	7	1	1	1	1	2	5	jueves
08/01/2016 0:00:00	2016	1	8	1	1	1	1	2	6	viernes
09/01/2016 0:00:00	2016	1	9	1	1	1	1	2	7	sábado
10/01/2016 0:00:00	2016	1	10	1	1	1	1	3	1	domingo
11/01/2016 0:00:00	2016	1	11	1	1	1	1	3	2	lunes
12/01/2016 0:00:00	2016	1	12	1	1	1	1	3	3	martes
13/01/2016 0:00:00	2016	1	13	1	1	1	1	3	4	miércoles
14/01/2016 0:00:00	2016	1	14	1	1	1	1	3	5	jueves
15/01/2016 0:00:00	2016	1	15	1	1	1	1	3	6	viernes
16/01/2016 0:00:00	2016	1	16	1	1	1	1	3	7	sábado
17/01/2016 0:00:00	2016	1	17	1	1	1	1	4	1	domingo
18/01/2016 0:00:00	2016	1	18	1	1	1	1	4	2	lunes
19/01/2016 0:00:00	2016	1	19	1	1	1	1	4	3	martes
20/01/2016 0:00:00	2016	1	20	1	1	1	1	4	4	miércoles
21/01/2016 0:00:00	2016	1	21	1	1	1	1	4	5	jueves
22/01/2016 0:00:00	2016	1	22	1	1	1	1	4	6	viernes
23/01/2016 0:00:00	2016	1	23	1	1	1	1	4	7	sábado
24/01/2016 0:00:00	2016	1	24	1	1	1	1	5	1	domingo

Creación de la tabla de medidas “features”

Medidas

- Actividad_Intensa
- Actividad_ligera
- Actividad_Moderada
- Calorias_Promedio
- Distancia_Recorrida
- Minutos_Sedentarios_Suma
- Pasos
- Sedentarismo_Promedio
- Sueno_semanal
- Total_Horas_Sueno
- Total_Minutos_Sueno

Dashboard de análisis y relaciones entre características



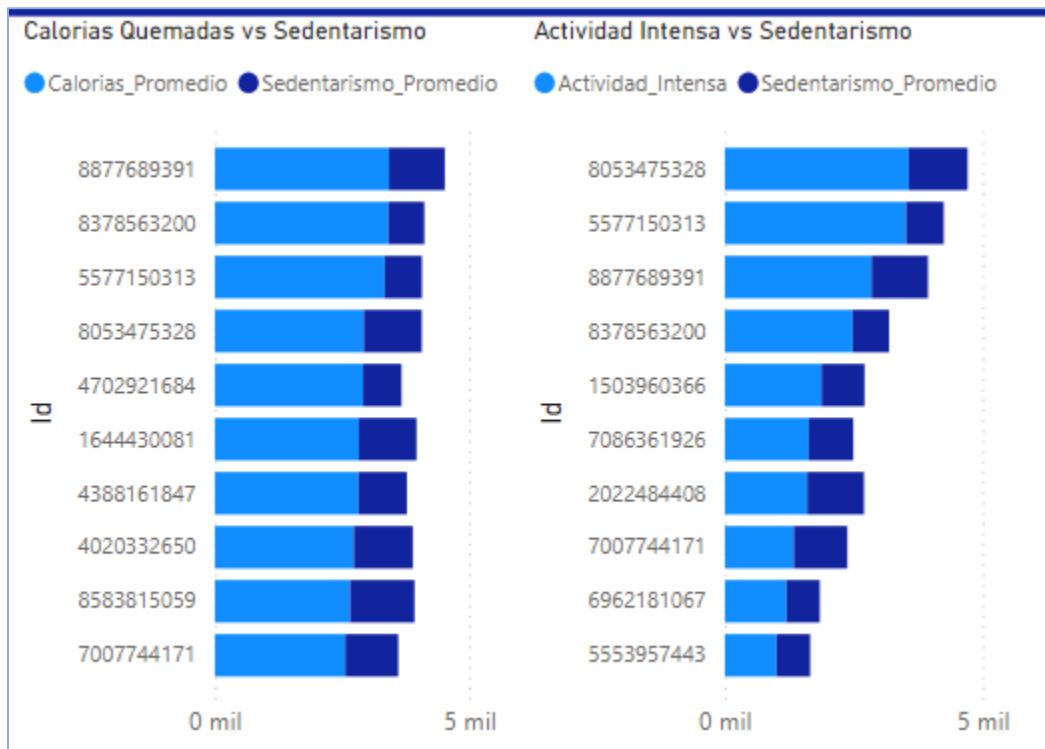
Captura del dashboard. Creación propia

Análisis de la Relación Inversa:

1. Mayor actividad intensa, menor sedentarismo:

En el gráfico de barras comparativo, se puede ver que los usuarios con mayor nivel de actividad intensa tienden a tener menores niveles de sedentarismo. Esto es esperable, ya que los minutos dedicados a actividades físicas de alta intensidad (como correr, hacer ejercicios de alta demanda física) ocupan una parte importante del día, desplazando el tiempo que podrían dedicar a actividades sedentarias.

Ejemplo: Los usuarios representados por los Ids 8053475328 y 8877689391 tienen altos niveles de actividad intensa, lo que coincide con niveles relativamente bajos de sedentarismo.



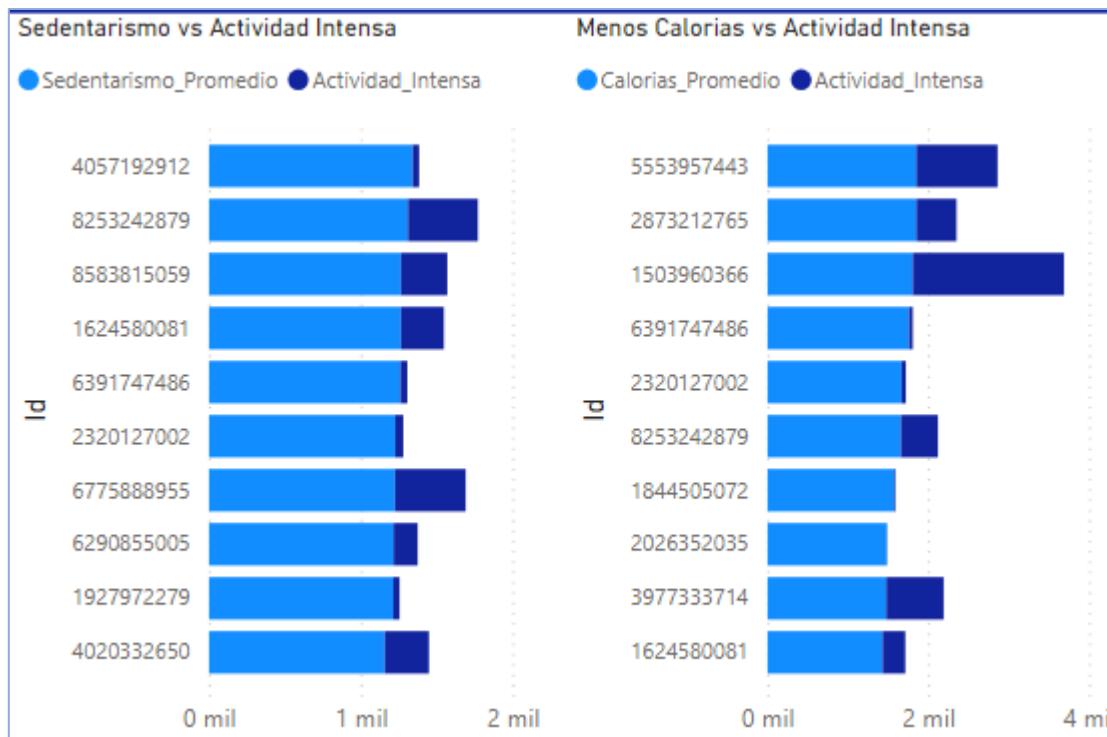
Captura del dashboard. Creación propia

2. Mayor sedentarismo, menor actividad intensa:

En el gráfico opuesto, aquellos usuarios con altos niveles de sedentarismo tienden a tener bajos niveles de actividad intensa. Este comportamiento es preocupante, ya que el

sedentarismo prolongado está vinculado a una serie de riesgos de salud, como problemas cardiovasculares, diabetes tipo 2 y obesidad.

Ejemplo: Los usuarios representados por los Ids 4057192912 y 8583815059 muestran altos niveles de sedentarismo, mientras que sus niveles de actividad intensa son significativamente menores.



Captura del dashboard. Creación propia

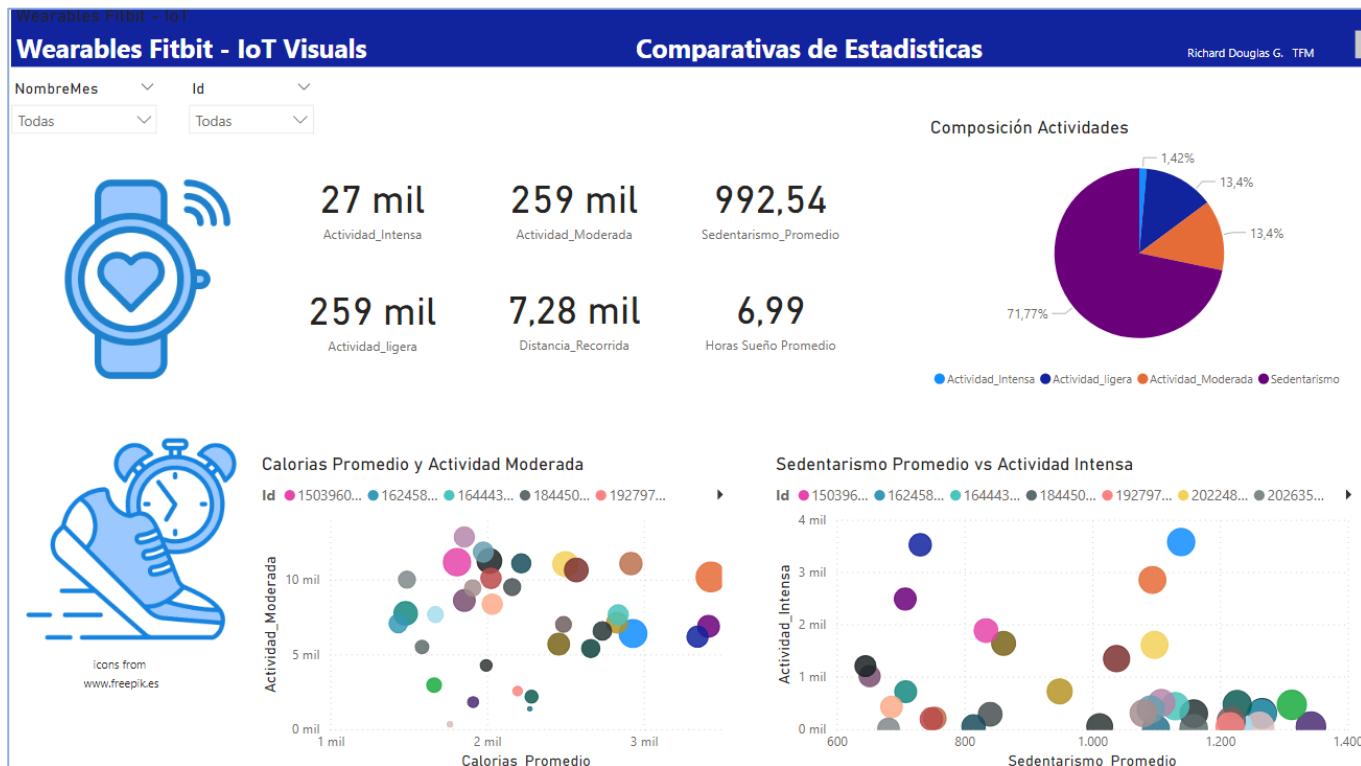
Implicaciones para la Salud:

Riesgos de salud asociados al sedentarismo:

El sedentarismo se ha relacionado con una serie de problemas de salud, como aumento del riesgo de enfermedades cardíacas, hipertensión, obesidad y otros trastornos metabólicos. Las personas que pasan más tiempo inactivas físicamente y dedicadas a actividades sedentarias, como estar sentadas o tumbadas durante gran parte del día, son más propensas a sufrir estas condiciones.

Los datos visualizados en Power BI resaltan que aquellos usuarios con altos niveles de sedentarismo son precisamente los que tienen menores niveles de actividad física

intensa, lo que sugiere que no están compensando el tiempo sedentario con suficientes minutos de ejercicio físico intenso o moderado.



Captura del dashboard. Creación propia

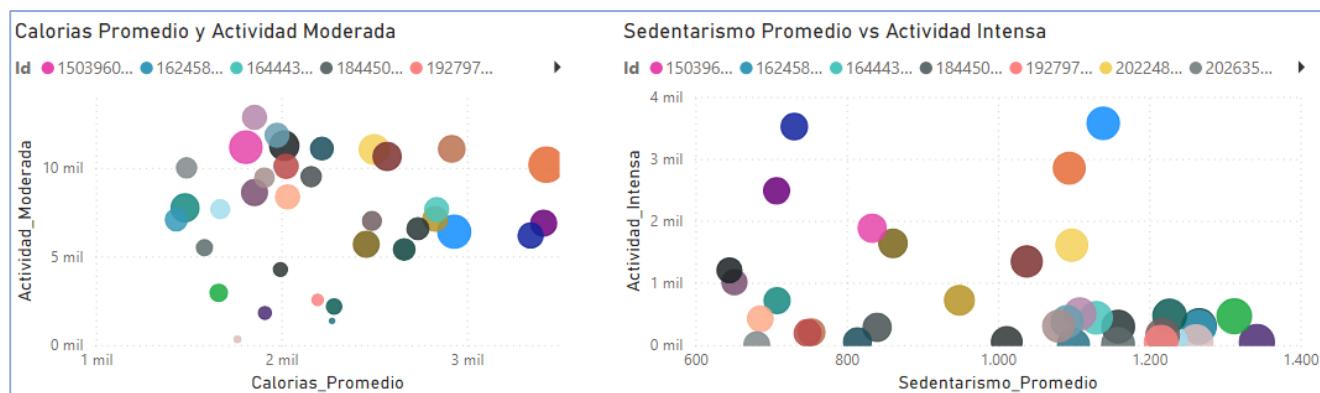


Captura del dashboard. Creación propia

El gráfico de pastel muestra la **distribución porcentual de las actividades**:

- 71.77% del tiempo se pasa en sedentarismo, lo que refleja que la mayor parte del día no está activa.
- 13.4% del tiempo en actividad moderada y actividad ligera, respectivamente.
- Solo el 1.42% del tiempo en actividad intensa.

Esto indica que la actividad física intensa es muy baja, y la mayoría del tiempo de los usuarios está caracterizada por sedentarismo, lo cual podría representar un riesgo para la salud si no se mantiene un equilibrio entre actividad y reposo.



Captura del dashboard. Creación propia

Gráfico de Dispersión: Calorías Promedio y Actividad Moderada:

En este gráfico de dispersión, se observa la relación entre calorías promedio y actividad moderada.

Relación Positiva: A medida que aumenta la cantidad de actividad moderada, también aumenta el número de calorías quemadas. Este es un indicador claro de que la actividad moderada tiene un impacto positivo en el gasto calórico.

Los usuarios con más actividad moderada tienden a quemar más calorías, lo que demuestra la importancia de este tipo de actividad para la salud y la quema de energía.

5. Gráfico de Dispersión: Sedentarismo vs. Actividad Intensa:

- En este gráfico se observa una relación inversa entre el sedentarismo y la actividad intensa.
- **Relación Inversa:** Los usuarios que tienen altos niveles de sedentarismo tienden a tener bajos niveles de actividad intensa.

Esto demuestra que los usuarios con menos actividad intensa son aquellos que pasan más tiempo en un estado sedentario, lo que puede ser una señal de un estilo de vida menos activo y un riesgo potencial para la salud.



Captura del dashboard. Creación propia

Los **KPI Cards** en la parte superior muestran los totales y promedios de cada tipo de actividad:

- **Actividad intensa:** 27 mil minutos.
- **Actividad moderada:** 259 mil minutos.
- **Actividad ligera:** 259 mil minutos.
- **Sedentarismo promedio:** 992.54 minutos.

Esto indica que la mayoría de los usuarios pasan una cantidad significativa de tiempo en actividad moderada y ligera, con un alto promedio de sedentarismo que sugiere que, en general, hay una gran cantidad de tiempo sedentario, lo que podría reflejar un estilo de vida menos activo en muchos usuarios. El KPI muestra que los usuarios tienen un promedio de 6.99 horas de sueño. Este valor es cercano al umbral mínimo recomendado (7 horas), lo que sugiere que la mayoría de los usuarios se acercan a los estándares de sueño saludables.

Este dashboard permite identificar algunos patrones clave en los datos de actividad y sueño:

- **Más actividad moderada = más calorías quemadas:** Lo que sugiere que los usuarios que realizan actividad física moderada pueden controlar mejor su gasto calórico.
- **Más sedentarismo = menos actividad intensa:** Esta relación inversa destaca la importancia de reducir el tiempo sedentario para aumentar los niveles de actividad física, especialmente la actividad intensa.

- **Sueño promedio adecuado:** Los usuarios están durmiendo en promedio 6.99 horas, lo cual es un buen indicador de un descanso saludable, aunque con una ligera desviación por debajo del estándar recomendado.

Resultados del Proceso de IoT Wearables

El trabajo realizado con este dataset demostró de manera efectiva el flujo completo de los datos provenientes de dispositivos wearables. El objetivo corresponde a demostrar que los datos que se generan por medio de estos dispositivos se pueden integrar a modelos avanzados de procesamiento de datos (ETL), con los cuales se llevan terminales o cubos de datos, los mismos a su vez son consultados para tomar información de ahí y evaluar las condiciones del usuario. Desde los datos capturados sobre la actividad diaria del usuario, como el número de pasos, la distancia recorrida, los minutos activos y las calorías quemadas, se llevó a cabo el procesamiento y análisis mediante las herramientas seleccionadas.

Aplicando los conocimientos adquiridos durante el programa de formación del master de IMF, se procedió a utilizar herramientas que permiten realizar la simulación de procesos de ETL que normalmente se aplican en ambientes de Big Data, los datos fueron transformados y limpiados en Spoon Pentaho, en el cual se establecieron reglas de captura de datos, transformación y aplicación de formatos de datos, para posteriormente realizar la conexión y envío de los datos al Staging y Data Ware House, almacenados eficientemente en SQL Server, en el cual se crearon los ambientes Staging de Datos y el Data Warehouse por medio de código y scripts, y se alimentaron por medio de un JOB de Spoon Pentaho, y finalmente los datos contenidos en SQL Server fueron enviados a Power Bi para tener la posibilidad de crear visualizaciones.

Este proceso permitió realizar un análisis detallado sobre los patrones de comportamiento físico del usuario, como las variaciones en los pasos diarios, la comparación entre días activos y sedentarios, y la relación entre la actividad física y la quema de calorías. Todo ello confirma la viabilidad y el valor de los dispositivos wearables para el monitoreo continuo y la mejora del bienestar a través de la actividad física.

Aplicación Práctica de Conocimientos Adquiridos

El desarrollo de este análisis permitió poner en práctica los conocimientos teóricos y técnicos adquiridos a lo largo del programa de estudio, aplicando las siguientes tecnologías y técnicas:

- Spoon Pentaho: Se utilizó para la extracción, transformación y carga (ETL) de los datos, realizando procesos de limpieza y transformación, como la normalización de fechas y la eliminación de duplicados.
- SQL Server: Los datos transformados fueron almacenados en un Data Warehouse para permitir consultas eficientes y estructuradas. Este almacenamiento optimizado facilitó la ejecución de análisis avanzados y la recuperación rápida de grandes volúmenes de datos.
- Power BI: En esta herramienta se crearon dashboards interactivos que permitieron visualizar los resultados del análisis de una manera clara y comprensible. Se emplearon gráficos de líneas para visualizar la evolución diaria de la actividad física, gráficos de barras apiladas para comparar la intensidad de la actividad, y gráficos circulares para analizar la distribución de los datos entre diferentes períodos.

Además, se creó una tabla calendario en Power BI que permitió realizar análisis temporales detallados, comparando la actividad física entre diferentes días de la semana, meses, o trimestres, aplicando técnicas de segmentación y agregación de datos vistas durante el programa.

Uso de Tecnologías Disponibles

Las tecnologías seleccionadas, como Spoon Pentaho, SQL Server y Power BI, no solo facilitaron el procesamiento y análisis de los datos, sino que también demostraron ser altamente eficientes para el manejo de grandes volúmenes de información generados por los dispositivos wearables. Además, reflejan el estado actual de las herramientas utilizadas en entornos profesionales para proyectos de análisis de datos y IoT.

Estas herramientas permitieron gestionar de manera efectiva los datos obtenidos de los wearables, asegurando que el proceso de análisis fuera robusto y eficiente. La integración de tecnologías como Spoon Pentaho, SQL Server y Power BI facilitó la realización de análisis avanzados y la generación de informes interactivos que proporcionan insights valiosos sobre la actividad física diaria. Además, el uso de estas herramientas

permitió poner en práctica los conocimientos adquiridos durante la formación universitaria. La máquina virtual proporcionada por la universidad, en la cual se encuentran disponibles las herramientas necesarias para correr los procesos que se requieren, en lo que respecta al desarrollo del proyecto, este apartado fue posible por la disponibilidad de la suite de herramientas en un solo ambiente de trabajo.

Predicción de Daños Cardiacos - Heart Disease Dataset

Heart Disease Dataset del UCI Machine Learning Repository, también conocido como el Cleveland Heart Disease Dataset. Este dataset es ampliamente utilizado para la predicción de enfermedades cardíacas y contiene información valiosa sobre factores de riesgo cardiovascular.

El dataset se utiliza principalmente para la clasificación y predicción de la presencia o ausencia de enfermedades cardíacas en pacientes. Los datos se basan en características clínicas y de salud de los pacientes.

Los enlaces en los que el dataset puede ser consultado así como la información relevante sobre el conjunto de datos:

- <https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data>
- <https://archive.ics.uci.edu/dataset/45/heart+disease>

Características: El dataset incluye varios atributos que son relevantes para predecir enfermedades cardíacas. A continuación se detallan las características principales:

1. **Edad:** Edad del paciente en años.
2. **Sexo:** Sexo del paciente (1 = masculino, 0 = femenino).
3. **Tipo de dolor en el pecho:**
 - 1: Dolor en el pecho típico angina
 - 2: Dolor atípico angina
 - 3: Dolor no anginoso
 - 4: Asintomático
4. **Presión arterial en reposo (en mm Hg):** Medida de la presión arterial cuando el paciente está en reposo.
5. **Colesterol en suero (en mg/dl):** Nivel de colesterol en la sangre del paciente.
6. **Electrocardiograma en reposo:**
 - 0: Normal, 1: Anomalías de onda ST-T, 2: Hipertrofia ventricular izquierda
7. **Frecuencia cardíaca máxima alcanzada (en bpm):** Frecuencia cardíaca más alta alcanzada durante una prueba de esfuerzo.

EDA - Análisis exploratorio de datos

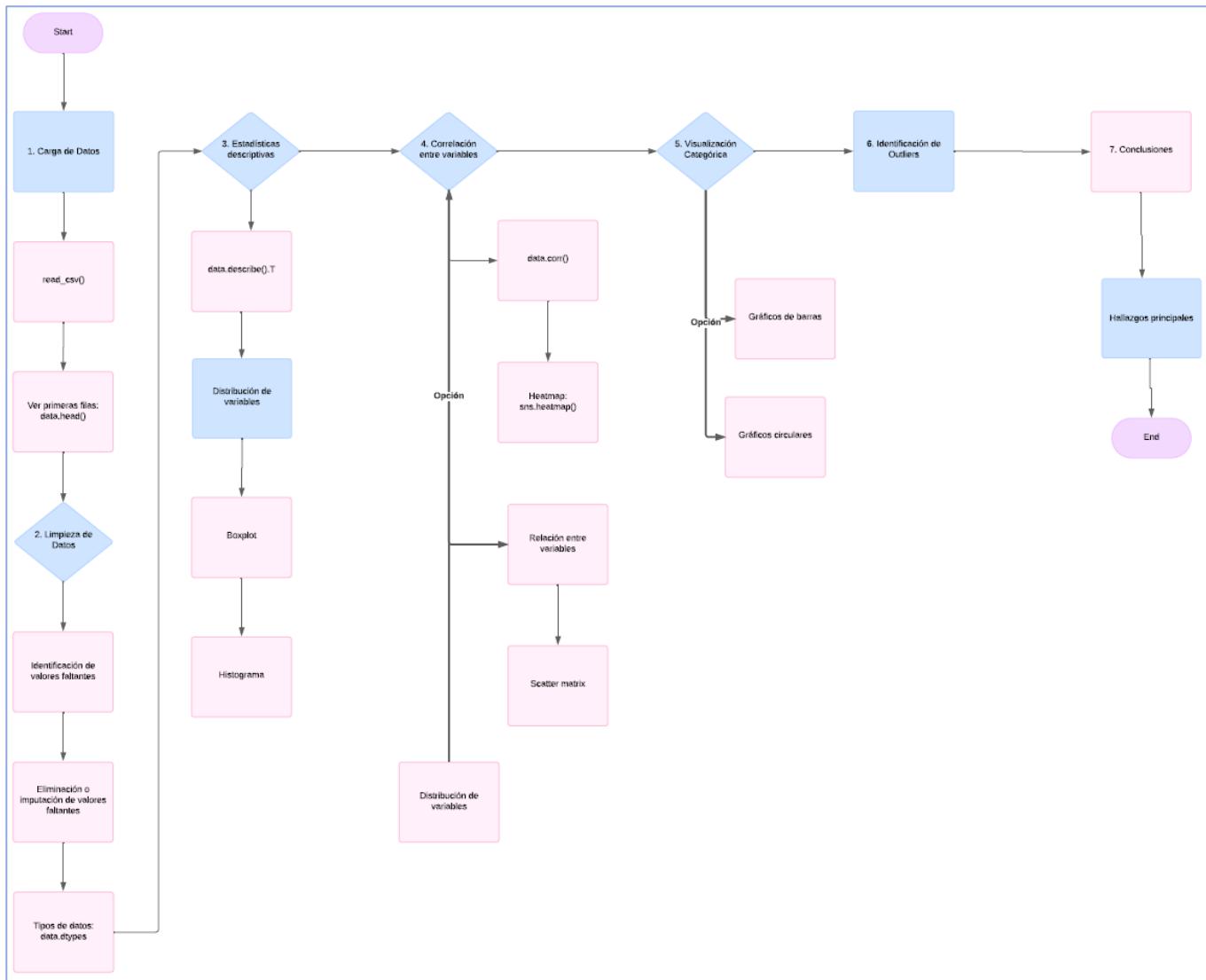


Diagrama de flujo del Proceso EDA
Creación propia con lucid.app

El diagrama anterior corresponde a una guía estructurada para llevar a cabo un proceso de Análisis Exploratorio de Datos (EDA). Este procedimiento está diseñado para facilitar la comprensión y el manejo inicial de un conjunto de datos, asegurando que se tomen los pasos necesarios antes de profundizar en el modelado.

El proceso comienza con la carga de datos, obtener una visión general de las primeras filas del conjunto de datos, se lleva a cabo la limpieza de los datos, identificando y eliminando o imputando valores faltantes, y verificando los tipos de datos.

El siguiente paso se centra en el análisis de las estadísticas descriptivas, donde se exploran las distribuciones de las variables a través de diagramas como boxplots e histogramas. Esta fase se complementa con un análisis de la correlación entre variables, utilizando visualizaciones como mapas de calor (heatmaps) y scatter plots para identificar las relaciones entre las distintas variables del conjunto de datos.

Posteriormente, se lleva a cabo una visualización categórica, donde se opta por gráficos de barras o circulares según la naturaleza de las variables. El proceso concluye con la identificación de outliers y la obtención de los principales hallazgos, que luego son consolidados en las conclusiones.

Este enfoque sistemático asegura que los datos estén preparados y completamente comprendidos antes de cualquier análisis avanzado o construcción de modelos predictivos.

Dataset

Dataset heart-disease-data

- Este dataset corresponde a un estudio realizado sobre pacientes y las diferentes características que pueden determinar en la predicción o no de un posible daño cardíaco.
- El dataset se utiliza principalmente para la clasificación y predicción de la presencia o ausencia de enfermedades cardíacas en pacientes. Los datos se basan en características clínicas y de salud de los pacientes.

se encuentra en las siguientes fuentes:

- <https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data>
- <https://archive.ics.uci.edu/dataset/45/heart+disease>

Información General de los Datos

- id: Identificador único.
- age: Edad del paciente.
- sex: Sexo (Male/Female).
- dataset: Fuente o ubicación del dataset (por ejemplo, "Cleveland").
- cp: Tipo de dolor en el pecho (como "typical angina", "asymptomatic", etc.).
- trestbps: Presión arterial en reposo (en mm Hg).
- chol: Nivel de colesterol en sangre (mg/dL).
- fbs: Glucosa en ayunas (TRUE si es mayor a 120 mg/dL, FALSE si es menor).
- restecg: Resultados del electrocardiograma en reposo (por ejemplo, "lv hypertrophy").
- thalch: Frecuencia cardíaca máxima alcanzada.
- exang: Angina inducida por ejercicio (TRUE o FALSE).
- oldpeak: Depresión del ST inducida por ejercicio en relación al reposo.
- slope: Pendiente del segmento ST (upsloping, flat, downsloping).
- ca: Número de vasos principales coloreados por fluoroscopia.
- thal: Tipo de defecto talámico (por ejemplo, "fixed defect", "normal", etc.).
- num: Etiqueta que indica la presencia de enfermedad cardíaca (0 para ninguna, * otros valores indican presencia de enfermedad en mayor severidad).

Característica adicional : target se agregan las características 0 = Representa la cantidad de casos para la clase 0 y 1= la suma de las clases 1, 2, 3, y 4.

Conocimiento de los datos

IMPORTACION DEL DATASET

```
data = pd.read_csv('/content/heart_disease_uci.csv', delimiter= ',')
```

Análisis Exploratorio de Datos EDA

Exploración Inicial de los Datos

```
# con esto se conoce un breve detalle de los datos a ser tratados
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 920 entries, 0 to 919
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   id          920 non-null    int64  
 1   age         920 non-null    int64  
 2   sex         920 non-null    object  
 3   dataset     920 non-null    object  
 4   cp          920 non-null    object  
 5   trestbps    861 non-null    float64 
 6   chol        898 non-null    float64 
 7   fbs         830 non-null    object  
 8   restecg     918 non-null    object  
 9   thalch      865 non-null    float64 
 10  exang       865 non-null    object  
 11  oldpeak     858 non-null    float64 
 12  slope       611 non-null    object  
 13  ca          309 non-null    float64 
 14  thal        434 non-null    object  
 15  num         920 non-null    int64  
dtypes: float64(5), int64(3), object(8)
memory usage: 115.1+ KB
```

Descripción Estadística

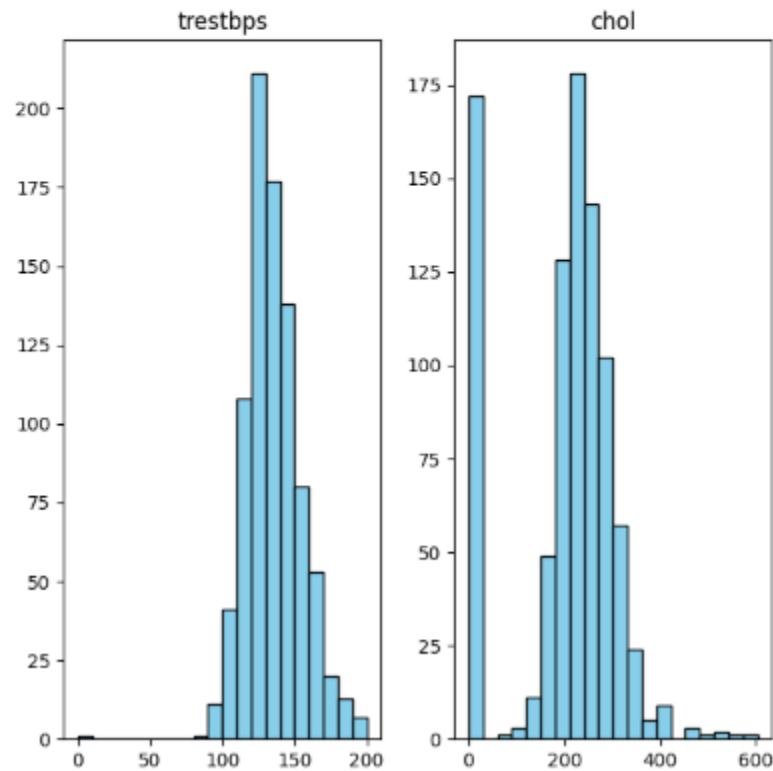
# Descripción estadística del DataFrame								
	id	age	trestbps	chol	thalch	oldpeak	ca	num
count	920.000000	920.000000	861.000000	890.000000	865.000000	858.000000	309.000000	920.000000
mean	460.500000	53.510870	132.132404	199.130337	137.545665	0.878788	0.676375	0.995652
std	265.725422	9.424685	19.066070	110.780810	25.926276	1.091226	0.935653	1.142693
min	1.000000	28.000000	0.000000	0.000000	60.000000	-2.600000	0.000000	0.000000
25%	230.750000	47.000000	120.000000	175.000000	120.000000	0.000000	0.000000	0.000000
50%	460.500000	54.000000	130.000000	223.000000	140.000000	0.500000	0.000000	1.000000
75%	690.250000	60.000000	140.000000	268.000000	157.000000	1.500000	1.000000	2.000000
max	920.000000	77.000000	200.000000	603.000000	202.000000	6.200000	3.000000	4.000000

```
# Crear una figura y ejes para los 10 histogramas
fig, axs = plt.subplots(1, 2, figsize=(6, 6))

# Nombres de las columnas
columnas = ['trestbps', 'chol']

# Iterar sobre cada columna y crear un histograma en el eje correspondiente
for i, columna in enumerate(columnas):
    axs[i].hist(df[columna], bins=20, color='skyblue', edgecolor='black')
    axs[i].set_title(columna)

# Ajustar el diseño de los subgráficos
plt.tight_layout()
plt.show()
```



Imputación de Datos Nulos

La imagen muestra cómo se completaron los datos faltantes en el conjunto. Para las variables numéricas, se utilizó la mediana, y para las variables categóricas, se recurrió a la moda, esta técnica corresponde a un procedimiento normal y necesario cuando se está trabajando con datos, debido a que permite manejar los valores nulos de una manera eficiente sin alterar demasiado los patrones de los datos. Así, los datos mantienen su coherencia, lo que es esencial para evitar problemas durante el análisis o el uso en modelos de aprendizaje automático. La idea detrás de esto es mejorar la calidad del dataset para que los resultados sean más confiables.

Imagen: Imputación de Valores Nulos

Tratamiento de Valores Nulos o Faltantes

```
#realizar una copia del dataset original antes de iniciar las transformaciones
df = data.copy()
```

En una distribución simétrica el valor de la media aritmética, la mediana y la moda coinciden.

Asimetría positiva

Moda Mediana Media

Distribución simétrica

Media = mediana = moda

Asimetría negativa

Media Mediana Moda

- En el caso de una distribución con un comportamiento Simétrico se toma el valor de la Media
- En el caso de una distribución con comportamiento Asimétrico se toma el valor de la Mediana
- En el caso de alguna característica de tipo Categorica se utiliza la Moda

Se va a tratar los campos con valores nulos o faltantes segun lo indicado

- numericos por la mediana
- caracteristicas objeto por la moda

```
df['nombre_columna'] = df['nombre_columna'].fillna(valor)
```

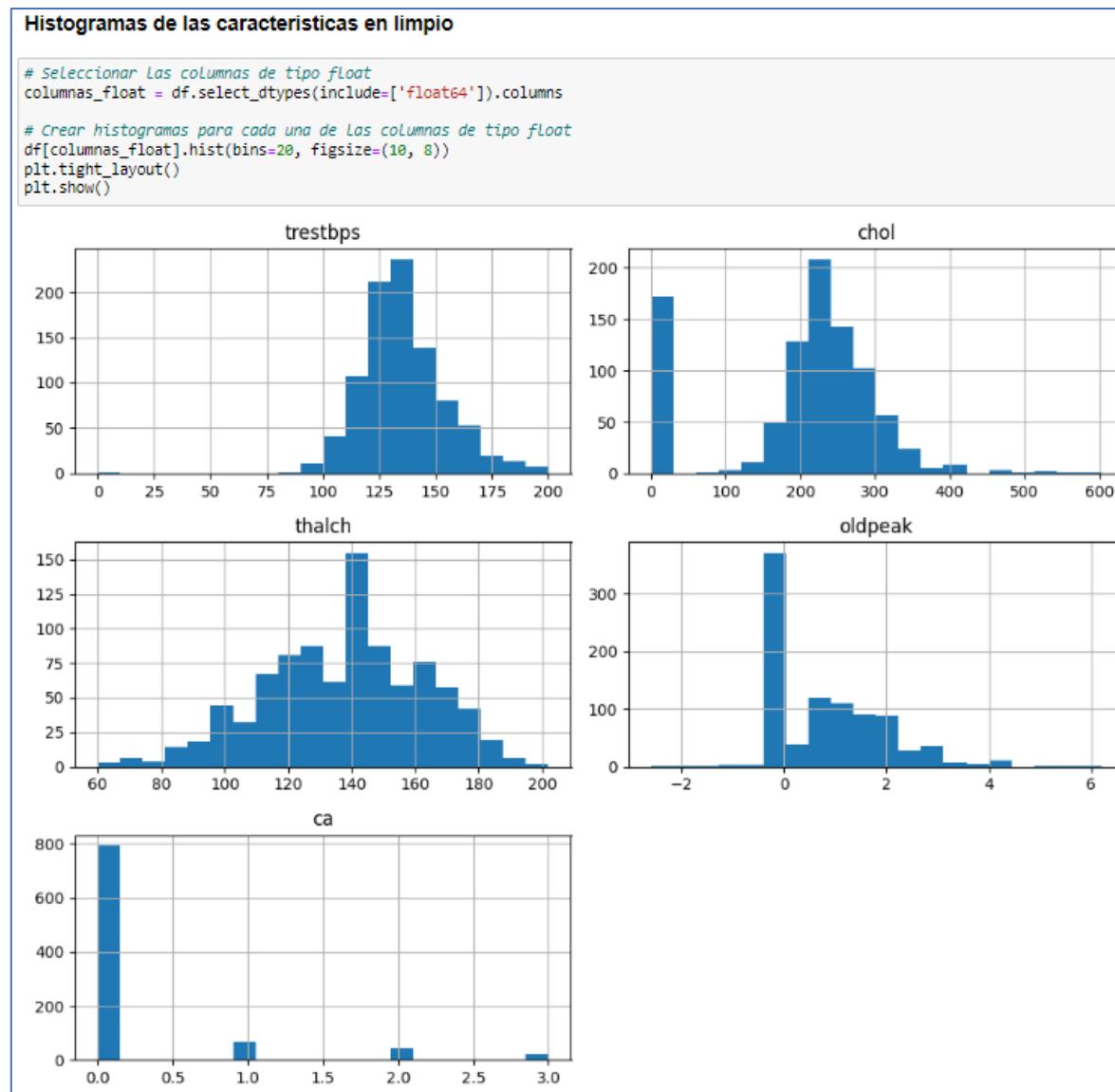
```
# Rellenar valores nulos para características numéricas (tipo float) con La mediana
for columna in df.select_dtypes(include=['float64']).columns:
    mediana = df[columna].median()
    df[columna] = df[columna].fillna(mediana)
```

```
# Rellenar valores nulos para características de tipo objeto con La moda
for columna in df.select_dtypes(include=['object']).columns:
    moda = df[columna].mode().iloc[0]
    df[columna] = df[columna].fillna(moda)
```

```
# Mostrar DataFrame modificado
print(df)
```

```
      id  age   sex dataset       cp trestbps  chol  fbs \
0     1   63  Male  Cleveland  typical angina  145.0  233.0  True
1     2   67  Male  Cleveland  asymptomatic  160.0  286.0 False
2     3   67  Male  Cleveland  asymptomatic  120.0  229.0 False
3     4   37  Male  Cleveland  non-anginal   130.0  250.0 False
4     5   41 Female  Cleveland atypical angina  130.0  204.0 False
..   ...
915  916   54 Female  VA Long Beach  asymptomatic  127.0  333.0  True
916  917   62  Male  VA Long Beach  typical angina  130.0  139.0 False
917  918   55  Male  VA Long Beach  asymptomatic  122.0  223.0  True
918  919   58  Male  VA Long Beach  asymptomatic  130.0  385.0  True
919  920   62  Male  VA Long Beach atypical angina  120.0  254.0 False
```

Seguidamente se corren los gráficos tipo histograma sobre las características posteriormente a la aplicación de la imputación de los datos, estos gráficos proporcionan una visión general del comportamiento de las características numéricas y permiten identificar posibles tendencias que podrían influir en el análisis posterior, así como la distribución de los datos después del proceso de limpieza.



Trestbps La mayoría de las personas en el estudio tienen una presión arterial en reposo que se sitúa entre 120 y 130 mm Hg. Esto sugiere que la distribución sigue un patrón normal, aunque tiende a mostrar una ligera inclinación hacia valores más altos.

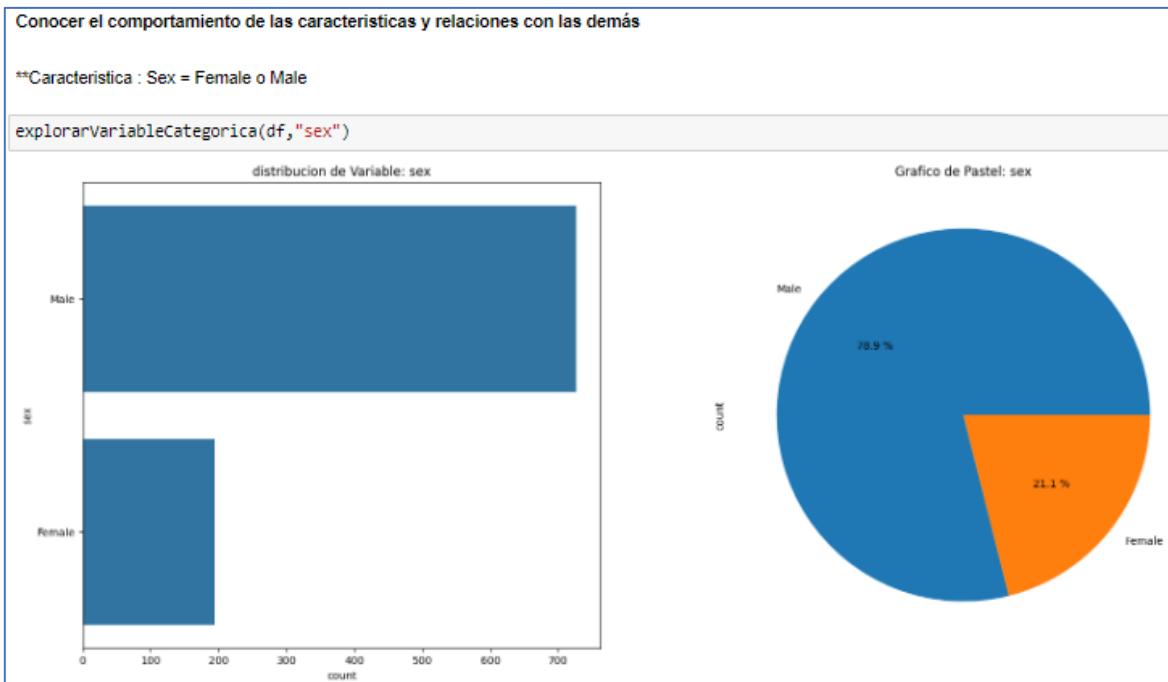
Chol Los niveles de colesterol de la mayoría de los pacientes se encuentran entre 200 y 250 mg/dl, con algunos casos que superan los 300 mg/dl. Esto indica que la mayoría tiene colesterol en un rango moderado a alto.

Thalach La frecuencia cardíaca máxima alcanzada muestra una distribución casi normal, con la mayoría de las personas alcanzando alrededor de 140 latidos por minuto durante el ejercicio. Las frecuencias muy altas o muy bajas son menos comunes.

Oldpeak La mayor parte de los pacientes no presenta una depresión significativa en el segmento ST durante el ejercicio, ya que los valores se concentran cerca de 0. Sin embargo, algunos muestran una leve a moderada depresión, lo que podría indicar problemas cardíacos.

Ca La gran mayoría de los pacientes no tiene vasos coloreados, como lo indica la concentración de valores en 0. Los pacientes que tienen entre 1 y 3 vasos coloreados son mucho menos comunes, lo que podría señalar casos específicos o más graves en la población analizada.

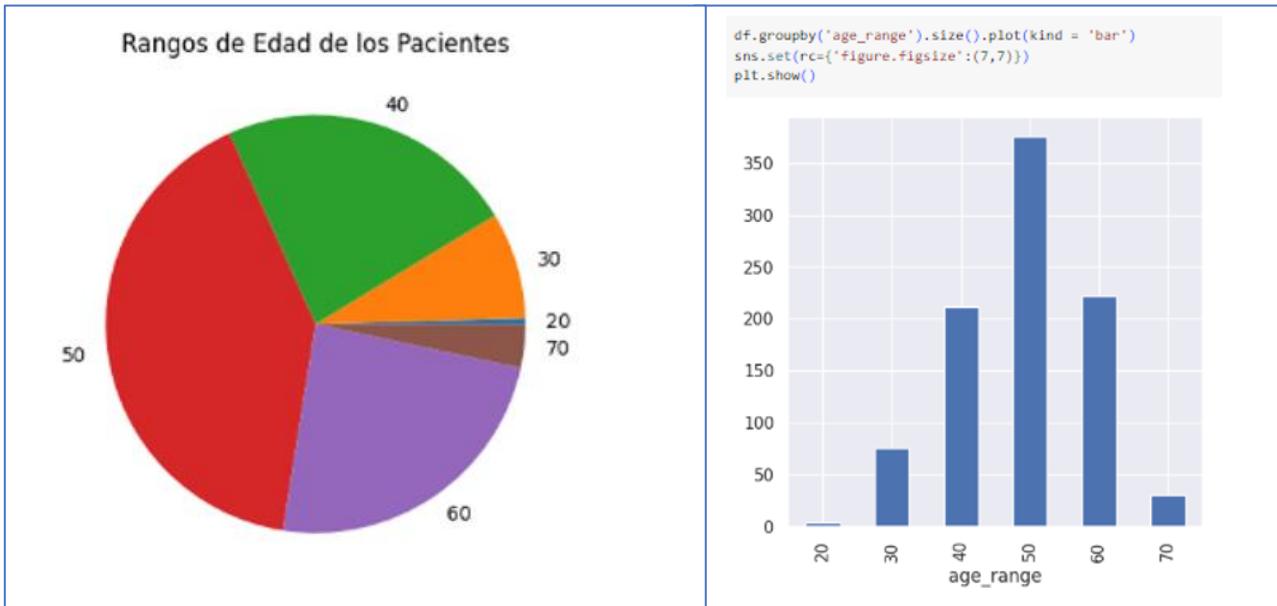
Distribución: Característica Sex



Los gráficos muestran la distribución de la variable sexo (sex) en el conjunto de datos. A la izquierda, un gráfico de barras ilustra la cantidad de observaciones categorizadas como "Male" (hombres) y "Female" (mujeres). Se observa que la mayoría de las observaciones corresponden a hombres, con más de 700 registros, mientras que las mujeres representan poco más de 200 registros.

En lo que respecta al gráfico circular, nos presenta la información en términos de porcentajes, de forma que actúa de complementariamente al mostrar la proporción de cada categoría. Se puede ver que los hombres constituyen el 78.9% de las observaciones, mientras que las mujeres representan el 21.1%.

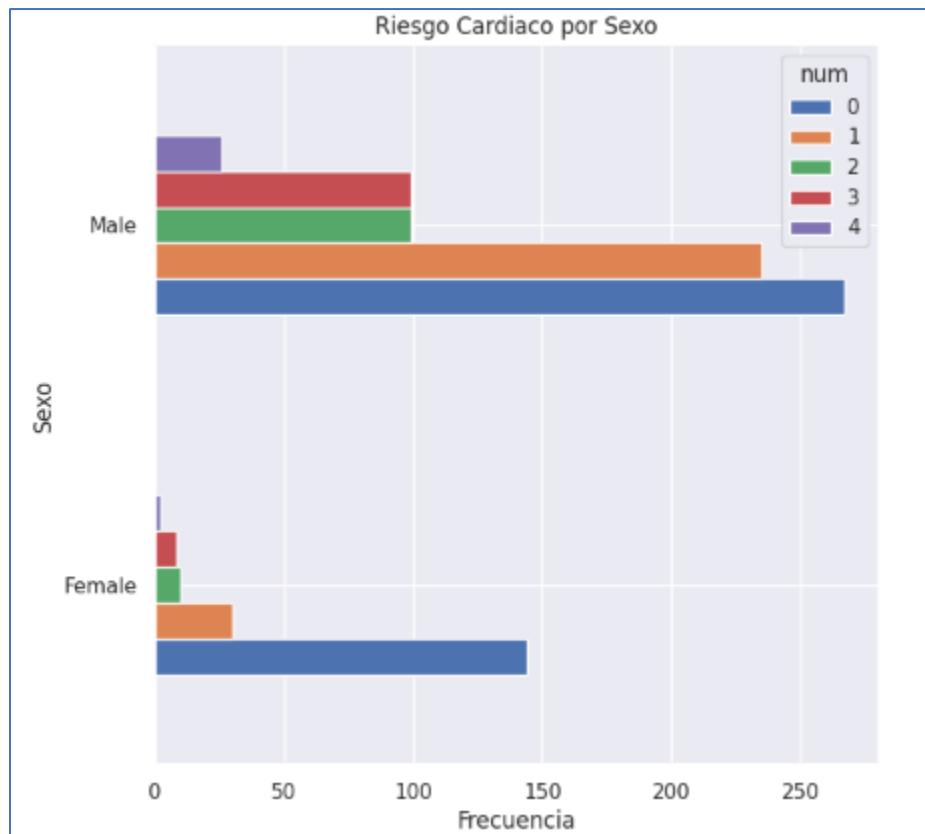
Característica Edad



Como parte del análisis, se creó una columna con las edades, lo que permite agrupar a las personas por rangos de edad para facilitar el análisis mismo. Por lo tanto, los siguientes gráficos nos dicen:

El gráfico de pastel ofrece una representación porcentual de los diferentes grupos etarios, destacando el rango de 50 años como el más representado, seguido de los rangos de 60, 40 y 30 años. Los grupos de 20 y 70 años presentan una representación significativamente menor. Este gráfico facilita la visualización de la proporción de cada rango de edad en la población total, mostrando que la mayoría de los pacientes se concentran en edades comprendidas entre los 40 y 60 años.

Por otro lado, el gráfico de barras proporciona una representación cuantitativa más precisa de los mismos datos. Se observa claramente que el rango de 50 años cuenta con el mayor número de pacientes, con más de 350 individuos, seguido por los rangos de 60 y 40 años, que agrupan aproximadamente a 200 pacientes cada uno. Los rangos de 20 y 70 años son los menos frecuentes, con menos de 50 personas en cada grupo.

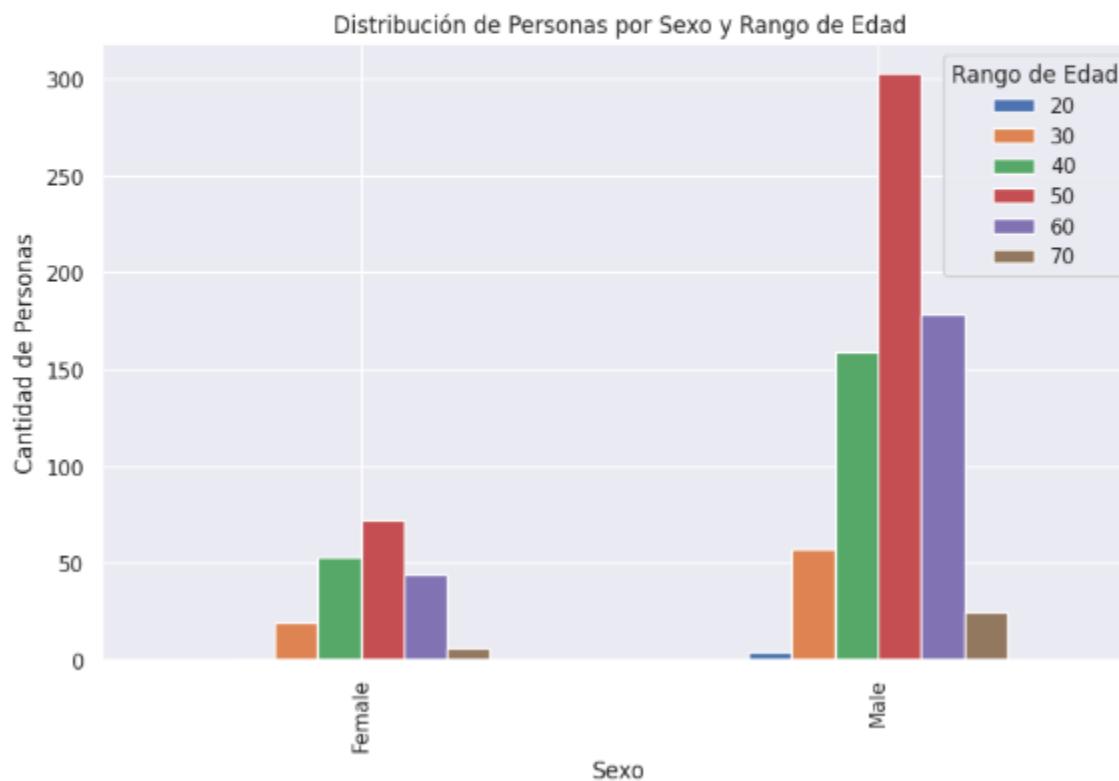


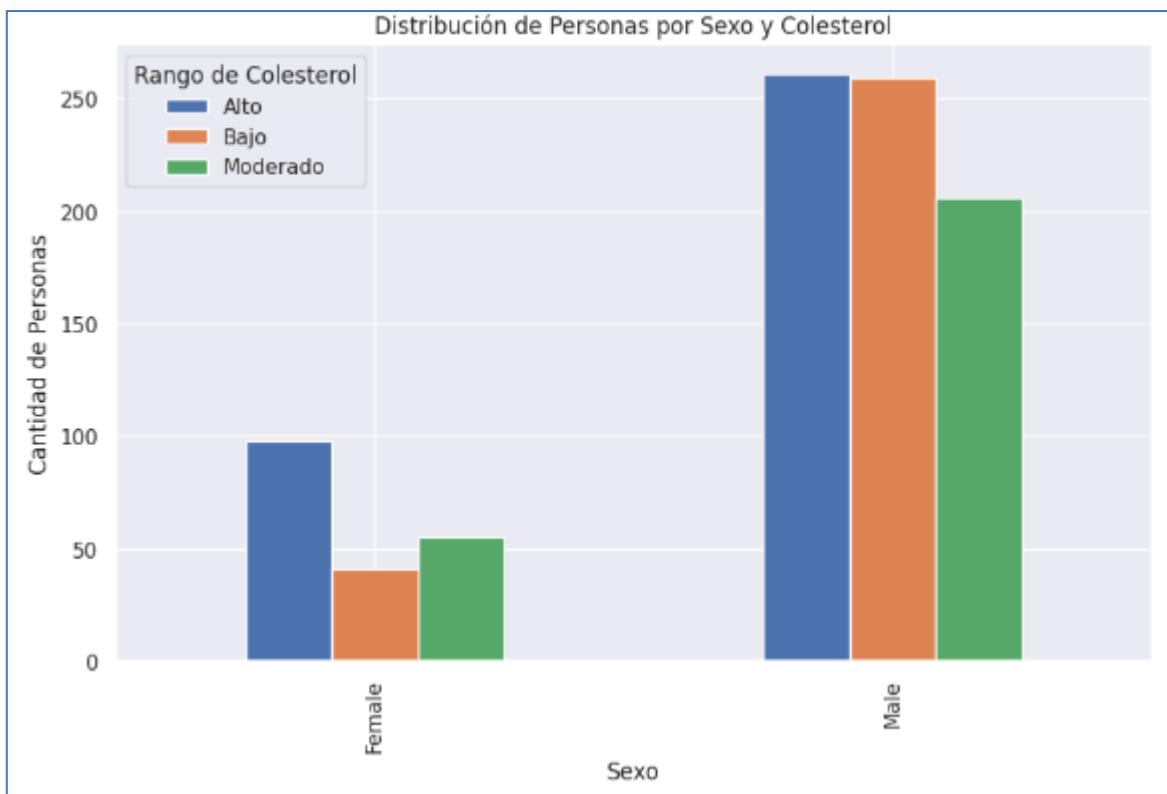
El gráfico presentado es significativo, ya que muestra la distribución de la característica num, la cual es la característica objetivo en el análisis, desglosada por sexo. Se puede observar que los hombres (Male) tienen una mayor representación en todas las clases de riesgo cardíaco en comparación con las mujeres (Female). En particular, la clase 0 (sin enfermedad cardíaca) es predominante en ambos sexos, pero más en hombres. Para las clases de mayor riesgo (1 a 4), la tendencia sigue siendo más alta en los hombres, lo que sugiere una diferencia significativa en la incidencia de enfermedades cardíacas entre los géneros. Este tipo de análisis es crucial para identificar patrones y posibles factores de riesgo relacionados con el sexo en la aparición de enfermedades cardíacas.

```
graficos2 = pd.crosstab(df['sex'], df['age_range'])

## Crear el gráfico de barras agrupado
graficos2.plot(kind='bar', figsize=(10, 6))

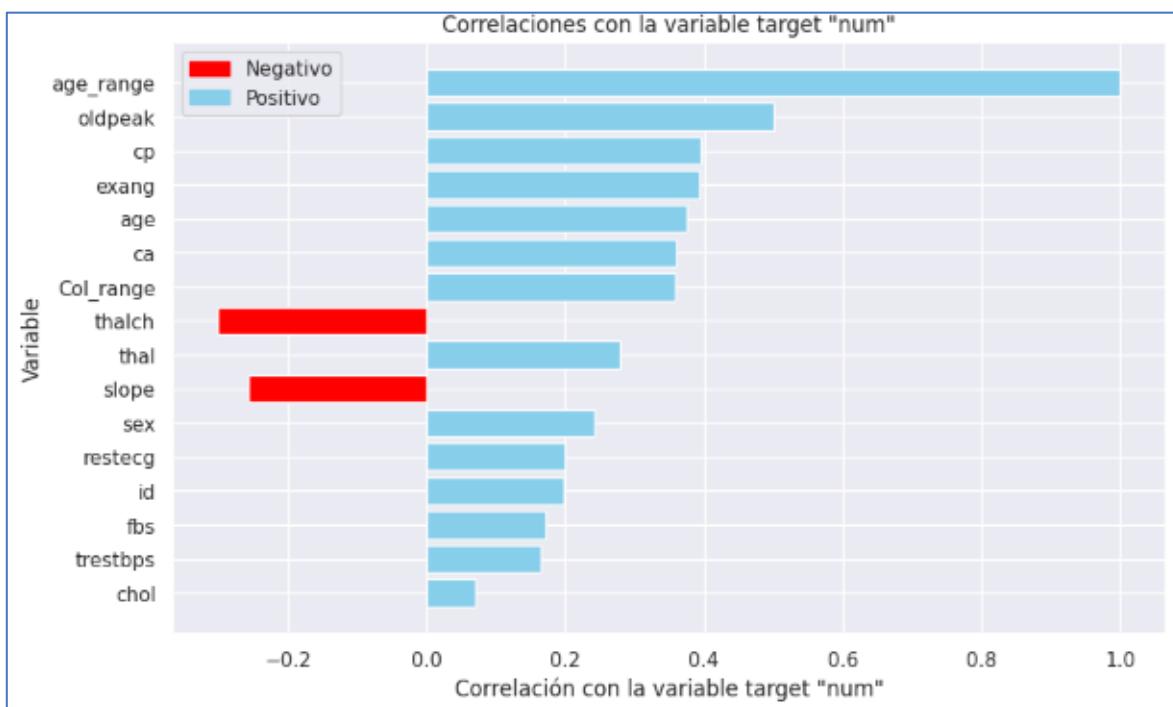
## Agregar título y etiquetas
plt.title('Distribución de Personas por Sexo y Rango de Edad')
plt.xlabel('Sexo')
plt.ylabel('Cantidad de Personas')
plt.legend(title='Rango de Edad')
plt.show()
```



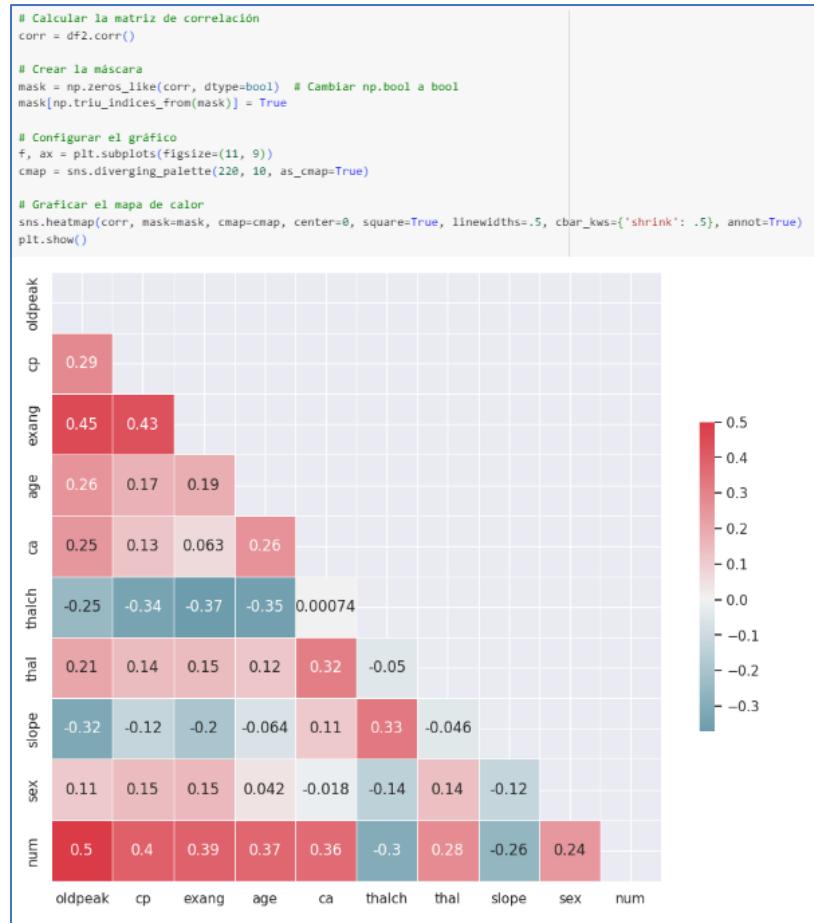


Es notorio que la cantidad de hombres que fueron tomados para la muestra de estudio que da origen al dataset es mayor que la proporción de mujeres, sin embargo si es importante señalar que los niveles de colesterol son elevados en las poblaciones presentando una alta proporción de hombres con el colesterol alto, importante indicar que la categoría de Rango de Colesterol se creó como parte del proceso de feature engineering para facilitar la creación de gráficos y análisis de los mismos.

Correlaciones



Este gráfico de barras muestra las correlaciones entre las diferentes variables del dataset y la variable objetivo, denominada "num", que representa el riesgo cardíaco. Las barras azules indican correlaciones positivas, es decir, aquellas variables que aumentan junto con el riesgo cardíaco. Por otro lado, las barras rojas representan correlaciones negativas, donde un aumento en el valor de la variable está asociado con una disminución en el riesgo cardíaco.



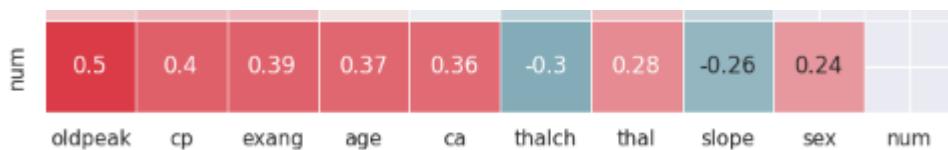
El gráfico de mapa de calor de correlación muestra las relaciones entre las diferentes características del dataset. Las correlaciones están representadas en un espectro de colores, donde los tonos más cercanos al rojo indican una correlación positiva fuerte, mientras que los tonos más cercanos al azul representan una correlación negativa.

Oldpeak y Num tienen una correlación positiva significativa (0.50), lo que indica que a medida que aumenta el valor de "oldpeak", también lo hace la variable objetivo "num" (relacionada con la presencia de enfermedades cardíacas).

Exang y Num también están correlacionadas positivamente (0.45), lo que sugiere que la presencia de dolor inducido por el ejercicio (exang) está asociada con un mayor riesgo de enfermedad cardíaca.

Cp y Num (0.40) muestran una correlación moderada, lo que indica que el tipo de dolor en el pecho podría ser relevante para el diagnóstico.

Thalach y Num tienen una correlación negativa (-0.30), lo que indica que en cuanto una disminuye la otra aumenta, tienen una correlación inversa, mientras una disminuye la otra tenderá a aumentar.



Este análisis permite identificar las características más relevantes para el proceso de análisis, como "oldpeak", "exang" y "cp", así como age, que son importantes para los modelos de machine learning, ya que tienen una relación más fuerte con el resultado que estamos buscando predecir (la variable objetivo "num").

Modelos Aplicados

A lo largo del proceso de análisis, se aplicaron diversos modelos de machine learning para la predicción de la presencia de daño cardíaco, optimizando el rendimiento de cada uno mediante técnicas de ajuste y selección de características relevantes, la plataforma o sistema utilizado para correr estos modelos corresponde a python, la selección del mismo en vista que Python es un lenguaje de programación ampliamente utilizado en el ámbito de la ciencia de datos y el machine learning debido a su flexibilidad, simplicidad y la vasta cantidad de bibliotecas disponibles para análisis de datos, además durante el programa de estudio la herramienta que se utilizó para el uso, análisis y corrida de Modelos de Aprendizaje Automático fue Python.

Para el desarrollo de este proyecto se tomaron en cuenta varios modelos de ML, con el fin de determinar y/o buscar diferentes rendimientos en el tratamiento del dataset. Vemos por lo tanto de forma inicial el K-Nearest Neighbors (KNN). Este modelo se basa en la cercanía de los datos, donde un nuevo dato es clasificado según los vecinos más cercanos en el espacio de características. KNN ha demostrado ser útil en tareas de clasificación, ya que asume que las instancias similares estarán cercanas entre sí.

Se utilizó también el Linear Discriminant Analysis (LDA), un método estadístico que busca maximizar la separación entre las clases al proyectar los datos en un espacio de menor dimensión. Este algoritmo es particularmente efectivo cuando las clases están bien diferenciadas y las relaciones entre las variables siguen una distribución normal.

En cuanto a la Regresión Logística (Logistic Regression), se aplicó para abordar la predicción de una variable categórica binaria, especialmente relevante en este estudio para clasificar la presencia o ausencia de enfermedad cardíaca. La regresión logística es un modelo ampliamente utilizado en problemas de clasificación binaria y ha mostrado un buen rendimiento en este caso.

El árbol de decisión (CART) también se utilizó como parte del análisis. Este modelo funciona dividiendo el espacio de características en diferentes ramas hasta alcanzar nodos puros, donde todas las instancias pertenecen a una sola clase. Aunque este modelo es intuitivo y fácil de interpretar, puede ser propenso al sobreajuste, por lo que se aplicaron técnicas de optimización y validación cruzada para mitigar este riesgo.

Otro enfoque que se exploró fue el Naive Bayes (Gaussian Naive Bayes), un algoritmo basado en el teorema de Bayes que asume la independencia condicional entre las características. Aunque esta suposición puede ser restrictiva en ciertos contextos, el modelo Naive Bayes ha demostrado ser eficiente y efectivo en problemas de clasificación binaria, como el que se aborda en este estudio.

El Support Vector Classifier (SVC) fue otro de los modelos aplicados, y su enfoque consiste en encontrar un hiperplano que maximiza el margen entre las clases en el espacio de características. Este modelo es particularmente útil en problemas donde las clases no son linealmente separables, aunque su rendimiento depende fuertemente del ajuste de hiperparámetros como el coeficiente C y gamma, los cuales fueron ajustados para mejorar los resultados.

Asimismo, se implementaron Redes Neuronales Artificiales (ANN), que son modelos inspirados en el funcionamiento de las neuronas biológicas. Las redes neuronales son capaces de capturar patrones complejos en los datos, lo que las hace adecuadas para problemas con interacciones no lineales entre las características. Para este estudio, se aplicaron redes con diferentes configuraciones y capas ocultas, con el fin de obtener una clasificación más precisa.

El Random Forest (Bosques Aleatorios) fue otro modelo de gran relevancia en el análisis. Este enfoque de ensemble combina múltiples árboles de decisión, reduciendo el riesgo de sobreajuste y proporcionando predicciones más robustas. El modelo de bosque aleatorio es especialmente efectivo cuando se trabaja con datos de alta dimensionalidad y ruido.

Finalmente, el Gradient Boosting (incluyendo algoritmos como XGBoost y LightGBM) se utilizó como un enfoque adicional de ensemble. Este modelo construye iterativamente nuevos árboles que corrigen los errores de los anteriores, lo que lo hace muy eficaz en tareas de clasificación y regresión complejas.

Además de la implementación de estos modelos, se realizaron esfuerzos considerables para mejorar el rendimiento de los mismos mediante técnicas de optimización de hiperparámetros como GridSearchCV y el balanceo del dataset. Se ajustaron las características más relevantes mediante técnicas de selección, y se reestructuró el dataset, pasando de una clasificación multiclase a una binaria para simplificar el problema y mejorar la capacidad predictiva de los modelos. Asimismo, se aplicaron técnicas de estandarización y reducción de dimensionalidad para garantizar que las características estuvieran bien ajustadas y escaladas, lo que resultó en una mejora significativa del rendimiento de los modelos.

En resumen, la combinación de diversos modelos, junto con la optimización de sus parámetros y el preprocesamiento adecuado de los datos, contribuyó significativamente a mejorar la precisión de las predicciones sobre la probabilidad de daño cardíaco, alcanzando resultados de alto nivel con modelos como LDA, KNN, SVC y Redes Neuronales.

Después de realizar tres corridas de testing y varios intentos intermedios entre los modelos, se lograron los siguientes resultados, mismos que se encuentran detallados más adelante.

	Accuracy_Test1	Accuracy_Test2	Accuracy_Test3
LR	0.640621	0.642232	0.824492
LDA	0.647316	0.640508	0.831215
KNN	0.618785	0.598729	0.831158
CART	0.555311	0.561921	0.794379
NB	0.618955	0.617090	0.817740
SVC	0.643983	0.640593	0.847881
ANN	0.573333	0.566667	0.826667

Librerías de Modelado

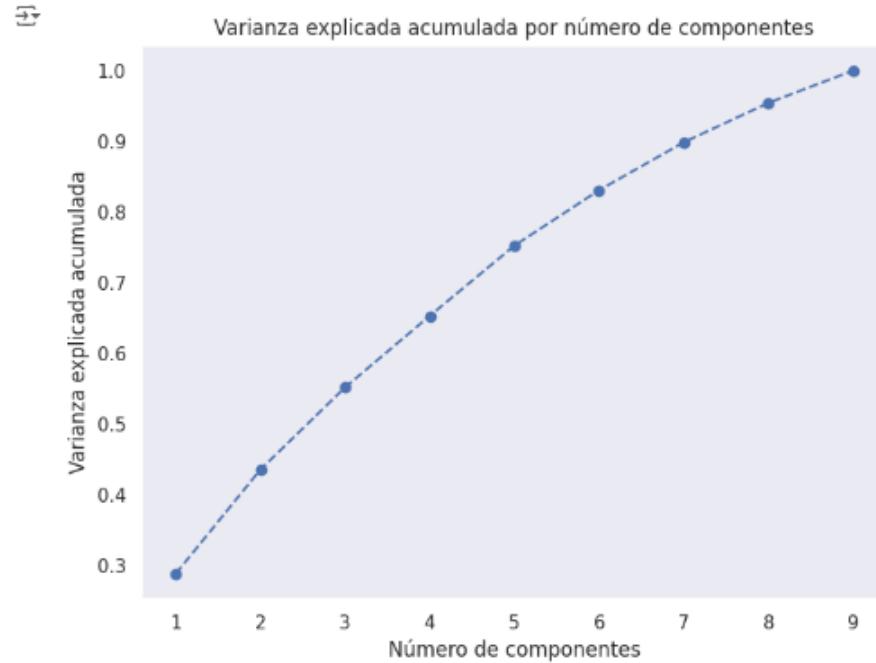
```
[75] from sklearn.model_selection import StratifiedKFold, cross_val_score
     from sklearn.linear_model import LogisticRegression
     from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.naive_bayes import GaussianNB
     from sklearn.svm import SVC

[76] from sklearn.decomposition import PCA
     import matplotlib.pyplot as plt

     # Ajustar PCA
     pca = PCA().fit(X_train)

     # Varianza explicada acumulada
     varianza_acumulada = pca.explained_variance_ratio_.cumsum()

     # Graficar la varianza acumulada
     plt.figure(figsize=(8,6))
     plt.plot(range(1, len(varianza_acumulada)+1), varianza_acumulada, marker='o', linestyle='--')
     plt.title('Varianza explicada acumulada por número de componentes')
     plt.xlabel('Número de componentes')
     plt.ylabel('Varianza explicada acumulada')
     plt.grid()
     plt.show()
```

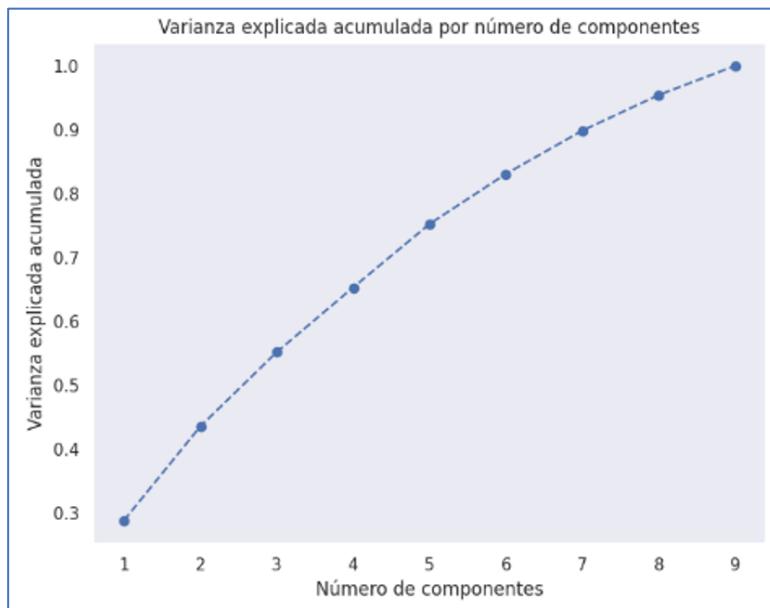


Esta gráfica ilustra la "Varianza explicada acumulada por número de componentes" y es fundamental en el análisis de componentes principales (PCA, por sus siglas en inglés). El objetivo de este gráfico es mostrar cuántos componentes principales son necesarios para explicar un porcentaje significativo de la varianza total de los datos.

En este caso, el gráfico revela que con aproximadamente 7 componentes principales se puede capturar más del 90% de la varianza acumulada en los datos. Esto significa que

al reducir el número de dimensiones de los datos originales a 7 componentes, se puede retener la mayor parte de la información, lo que optimiza el rendimiento de los modelos y reduce la complejidad del análisis sin perder precisión significativa.

Este tipo de análisis es crucial cuando se busca simplificar los datos para mejorar la eficiencia de los modelos de machine learning, especialmente en conjuntos de datos con muchas variables, manteniendo una alta capacidad predictiva y generalización.



LR-LDA-KNN-CART-NB-SVC

En la siguiente captura se puede ver como se nombran y establecen los modelos para ser ejecutados. En la captura presentada se aprecia la ejecución de varios modelos de aprendizaje automático, evaluados mediante validación cruzada. Los modelos incluidos en el análisis son: Logistic Regression (LR), Linear Discriminant Analysis (LDA), K-Nearest Neighbors (KNN), Decision Tree Classifier (CART), Naive Bayes (NB) y Support Vector Classifier (SVC).

Los valores de accuracy (precisión) varían según el modelo, el ajuste de hiperparámetros, así también por cada corrida de datos los porcentajes pueden variar por la naturaleza propia de los modelos y los algoritmos, sin embargo se toman los datos según los resultados obtenidos y se plasman a continuación.

```
# Lista de modelos
models = []
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVC', SVC()))

# Inicializar resultados y nombres
resultados = []
names = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    resultados.append(cv_resultados)
    names.append(name)
    print('%s: Accuracy = %f (Std = %f)' % (name, cv_resultados.mean(), cv_resultados.std()))

# Inicializar nombres y resultados de accuracy
names = []
accuracies = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    accuracies.append(cv_resultados.mean())
    names.append(name)

# Crear un DataFrame para mostrar Los resultados de accuracy
accuracies_test1 = pd.DataFrame(accuracies, columns=['Accuracy_Test1'], index=names)

# Mostrar La tabla de resultados de accuracy
# se almacena el resultado para realizar un acompañamiento con Las diferentes corridas y pruebas
print(accuracies_test1)

LR: Accuracy = 0.640621 (Std = 0.039760)
LDA: Accuracy = 0.647316 (Std = 0.049254)
KNN: Accuracy = 0.618785 (Std = 0.038755)
CART: Accuracy = 0.558644 (Std = 0.054614)
NB: Accuracy = 0.618955 (Std = 0.072338)
SVC: Accuracy = 0.643983 (Std = 0.050137)
```

En cuanto a los resultados obtenidos, tanto Logistic Regression como LDA muestran un rendimiento similar, con una precisión alrededor del 64%, lo que sugiere que estos modelos son capaces de capturar relaciones lineales en los datos. Por otro lado, KNN y Naive Bayes presentan un desempeño inferior, con precisiones en torno al 61%, mientras que CART registra la menor precisión, con un 55.86 %.

Estos resultados dan a entender que los modelos deben ser ajustados o el conjunto de los datos se deben tratar para buscar mejores resultados, situación que será tratado más adelante por medio de técnicas de feature engineering – ingeniería de características.

Redes Neuronales

En la captura presentada, se visualiza el proceso de implementación y evaluación de un modelo de red neuronal utilizando la clase MLPClassifier de la biblioteca sklearn. Inicialmente, se realiza la estandarización de los datos a través de StandardScaler para garantizar que todas las variables predictoras se encuentren en la misma escala, lo cual es fundamental para mejorar la eficiencia del modelo.

se define la arquitectura de la red neuronal, especificando parámetros como el tamaño de la capa oculta (100 neuronas), la función de activación ReLU, el optimizador Adam y el número máximo de iteraciones fijado en 500. El modelo es entrenado utilizando el conjunto de entrenamiento escalado, permitiendo a la red neuronal ajustar sus pesos y sesgos con base en los datos.

Aplicación de un modelo de redes neuronales

Entrenar el modelo de red neuronal:

```
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Iniciar el modelo
ann_model = MLPClassifier(hidden_layer_sizes=(100,), max_iter=500, activation='relu', solver='adam', random_state=1)
# Iniciar el modelo
ann_model.fit(X_train_scaled, Y_train)

# Entrenar el modelo con Los datos
ann_model.fit(X_train, Y_train)
```

MLPClassifier(max_iter=500, random_state=1)
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Evaluar el modelo

```
accuracy_ann = ann_model.score(X_test_scaled, Y_test)

print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)
```

Accuracy del modelo de redes neuronales (ANN): 0.573333333333334

Finalmente, se evalúa el rendimiento del modelo sobre el conjunto de prueba, mostrando un valor de precisión (accuracy) del 57.33%. Este resultado indica que el modelo logró clasificar correctamente aproximadamente el 57% de los casos, lo que sugiere la necesidad de ajustes adicionales en los hiperparámetros o en el preprocesamiento de los datos para mejorar la capacidad predictiva de la red neuronal.

Evaluación de Redes Neuronales

Evaluar el modelo

```
[93] accuracy_ann = ann_model.score(X_test_scaled, Y_test)
print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)

→ Accuracy del modelo de redes neuronales (ANN):  0.573333333333334

▼ Aumentar la complejidad de la red:

[94] from sklearn.neural_network import MLPClassifier
    # Aumentar capas ocultas y el número de iteraciones
    ann_model = MLPClassifier(hidden_layer_sizes=(150, 100, 50), max_iter=1000, activation='relu', solver='adam', random_state=1)
    ann_model.fit(X_train_scaled, Y_train)
    accuracy_ann = ann_model.score(X_test_scaled, Y_test)
    print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)

→ Accuracy del modelo de redes neuronales (ANN):  0.52
```

Este ajuste y modificaciones no surgen efecto positivo, en lugar de mejorar el resultado, se puede notar que el rendimiento de las redes neuronales se redujo.

Optimización de Red Neuronal

Optimización de Hiperparámetros:

```
: from sklearn.model_selection import GridSearchCV
# Parámetros para optimizar
param_grid = {
    'hidden_layer_sizes': [(100), (150, 100, 50)],
    'activation': ['relu', 'tanh'],
    'alpha': [0.0001, 0.001, 0.01],
    'solver': ['adam', 'lbfgs']
}

# GridSearchCV para optimizar hiperparámetros
grid_ann = GridSearchCV(MLPClassifier(max_iter=1000), param_grid, cv=5, verbose=2)
grid_ann.fit(X_train_scaled, Y_train)

# Evaluar el mejor modelo
mejor_ann = grid_ann.best_estimator_
accuracy_ann_opt = mejor_ann.score(X_test_scaled, Y_test)
print("Accuracy del mejor modelo ANN con GridSearchCV: ", accuracy_ann_opt)

[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=adam; total time= 5.1s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=adam; total time= 5.0s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=adam; total time= 5.4s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=lbfgs; total time= 6.2s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=lbfgs; total time= 4.3s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=lbfgs; total time= 7.6s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=lbfgs; total time= 5.0s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(100,), solver=lbfgs; total time= 5.2s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 13.6s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 19.8s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 19.4s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 19.6s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 20.3s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 52.7s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 52.0s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 51.0s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 53.3s
[CV] END activation=tanh, alpha=0.01, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 52.1s
Accuracy del mejor modelo ANN con GridSearchCV:  0.5666666666666667
```

En el caso de las modificaciones de los mejores hiperparámetros se obtiene un 56 % de rendimiento en contraste con el resultado de 52%, sin embargo estos resultados siguen siendo no satisfactorios.

Feature engineering en la característica target

El proceso de feature engineering fue crucial para mejorar la capacidad predictiva de los modelos en el análisis de enfermedades cardíacas. Un paso clave fue la creación de la característica target, la cual simplificó el problema al agrupar varias clases de daño cardíaco en una sola variable binaria. Originalmente, la variable num clasificaba el daño cardíaco en cinco niveles: desde 0 (sin daño cardíaco) hasta 4 (daño cardíaco crítico).

Sin embargo, dado que la mayoría de los casos correspondían a la clase "sin daño cardíaco", esto generaba un desbalance significativo en el dataset, con la finalidad de encontrar una solución y no dañar la calidad de los datos, se decidió combinar las clases 1, 2, 3 y 4 en una sola clase que representara cualquier tipo de daño cardíaco, y mantener la clase 0 como la ausencia de daño. Esto transformó el problema de clasificación multiclasa en una clasificación binaria, facilitando la tarea de los modelos y mejorando su rendimiento.

Después de crear la nueva variable target, se volvieron a entrenar los modelos de clasificación, lo que permitió obtener una notable mejora en los resultados. El modelo Linear Discriminant Analysis (LDA) logró un accuracy de 0.929831 con una desviación estándar de 0.040630, lo que demuestra una alta capacidad de separación entre las dos clases. Por otro lado, el modelo de K-Nearest Neighbors (KNN) alcanzó una precisión de 0.872994 con una desviación estándar de 0.034089, lo que también refleja un buen rendimiento en la clasificación binaria de la nueva variable target.

Además de la creación de la variable target, el feature engineering incluyó la selección de características relevantes mediante el análisis de correlaciones. Se identificaron variables como oldpeak, cp, exang, age, ca, thal y thalch, las cuales mostraban una mayor correlación con la variable target y, por lo tanto, fueron mantenidas en el análisis. Estas decisiones permitieron reducir el ruido en los datos y centrarse en las variables más predictivas, lo que contribuyó a la mejora en los resultados de los modelos.

Este enfoque de feature engineering no solo simplificó el modelo, sino que también equilibró mejor las clases en el dataset, evitando problemas asociados al desbalance de clases. Al aplicar este conjunto de técnicas, se logró mejorar significativamente el rendimiento de los modelos predictivos, lo que indica que los modelos se comportan mejor en la predicción de variables binarias.

Feature engineering en la característica target

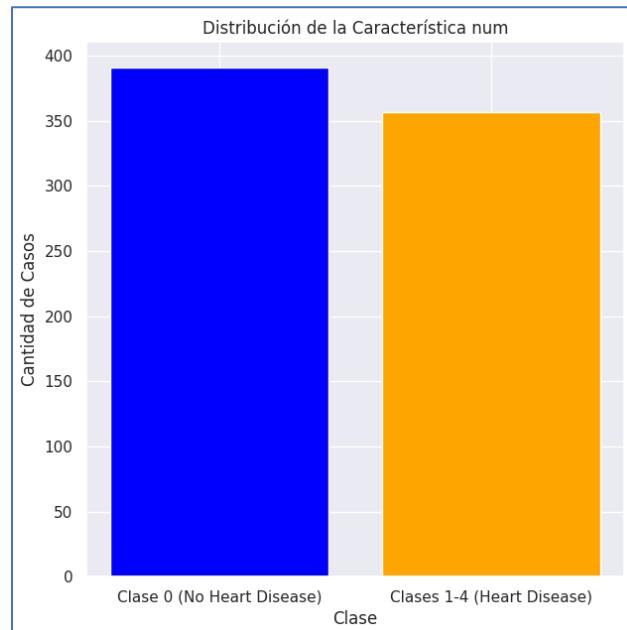
Decisión de Conversión de la Variable Objetivo Motivación: La variable num original del dataset contenía cinco clases que representaban distintos grados de daño cardíaco (0 = no heart disease, 1 = mild, 2 = moderate, 3 = severe, 4 = critical). Sin embargo, al evaluar el desempeño de los modelos predictivos con esta variable multiclas, observamos que los resultados en términos de precisión fueron moderados, con valores de accuracy entre 50% y 71%. Este rendimiento puede estar relacionado con el desequilibrio de clases, donde la clase "no heart disease" representaba la mayoría de los casos (411 de un total de 920).

Decisión: Para mejorar la calidad de la predicción y simplificar el problema, se tomó la decisión de convertir la variable num en una variable binaria. Esta nueva variable, llamada target, tiene los siguientes valores:

0: No presenta daño cardíaco (igual que la clase 0 de la variable original num). 1: Presenta algún grado de daño cardíaco (esto agrupa las clases 1, 2, 3 y 4 de la variable original).

df4.head()

	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	num	age_range	Col_range
0	1	63	1	0	145.0	233.0	1	1	150.0	0	2.3	0	0.0	2	0	60	1
1	2	67	1	3	160.0	288.0	0	1	108.0	1	1.5	1	3.0	0	2	60	2
2	3	67	1	3	120.0	229.0	0	1	129.0	1	2.6	1	2.0	1	1	60	1
3	4	37	1	2	130.0	250.0	0	0	187.0	0	3.5	0	0.0	0	0	30	2
4	5	41	0	1	130.0	204.0	0	1	172.0	0	1.4	2	0.0	0	0	40	1



Este gráfico refleja claramente cómo, después de modificar la característica target, las clases de "No Heart Disease" (clase 0) y "Heart Disease" (clases 1-4 combinadas) están ahora mucho más equilibradas, se deja de lado el desbalanceo de pesos entre las observaciones de una y otra. Cuando estamos en un proceso de creación y entrenamiento de modelos de aprendizaje automático es importante tomar en cuenta estos aspectos, ya que los datos desbalanceados suelen hacer que los modelos favorezcan la clase con más ejemplos, reduciendo su capacidad para predecir correctamente las clases con menos observaciones.

Los procesos de balanceo y equilibrio entre las observaciones, especialmente cuando una de ellas estaba claramente subrepresentada, los algoritmos pueden aprender de manera más eficaz las diferencias entre cada grupo. Como resultado, deberíamos ver una mejora en métricas como el recall y el F1-score para la clase "Heart Disease". Esto es especialmente importante en un contexto de predicción de enfermedades, donde es más valioso identificar correctamente los casos positivos que simplemente tener un alto nivel de precisión general.

En resumen, esta estrategia de *feature engineering* y balanceo de clases permite que los modelos trabajen de forma más justa, reduciendo el sesgo hacia la clase dominante y aumentando las probabilidades de obtener mejores resultados al predecir enfermedades cardíacas.

Correlaciones con nueva característica "target"

Correlaciones con nueva característica "target"

```
# Calcular la matriz de correlaciones para el DataFrame df4
matriz_correlaciones = df4.corr()

# Correlación de las variables con la target 'target'
# Aquí obtenemos directamente las correlaciones con 'target', sin necesidad de usar índices
correlaciones_target = matriz_correlaciones['target'].values[:-1] # Excluir la última que sería 'target' en sí misma

# Obtener los índices de las correlaciones ordenados de mayor a menor en valor absoluto
indices_inversos = abs(correlaciones_target).argsort()[:-1]

# Crear un diccionario para almacenar las correlaciones con la target 'target'
diccionario = {}
# Aquí utilizamos X como el DataFrame sin la columna 'target'
X = df4.drop(columns=['target']) # Excluimos 'target' para no calcular su correlación consigo misma
for nombre, correlacion in zip(X.columns[indices_inversos], list(correlaciones_target[indices_inversos])):
    diccionario[nombre] = correlacion

# Mostrar el resultado como DataFrame
resultado = pd.DataFrame.from_dict(diccionario, orient='index', columns=['Correlación con la target'])
print(resultado)
```

	Correlación con la target
num	0.792644
exang	0.482860
cp	0.463495
oldpeak	0.454458
thalch	-0.353850
age	0.299401
sex	0.294758
age_range	0.281230
id	0.278671
thal	0.264463
ca	0.241381
slope	-0.239280
trestbps	0.160526
fbs	0.150489
restecg	0.135910
chol	0.119271
Col_range	0.117590



Modelado Featuring Target

Modelado featuring Target : booleano 1 y 0

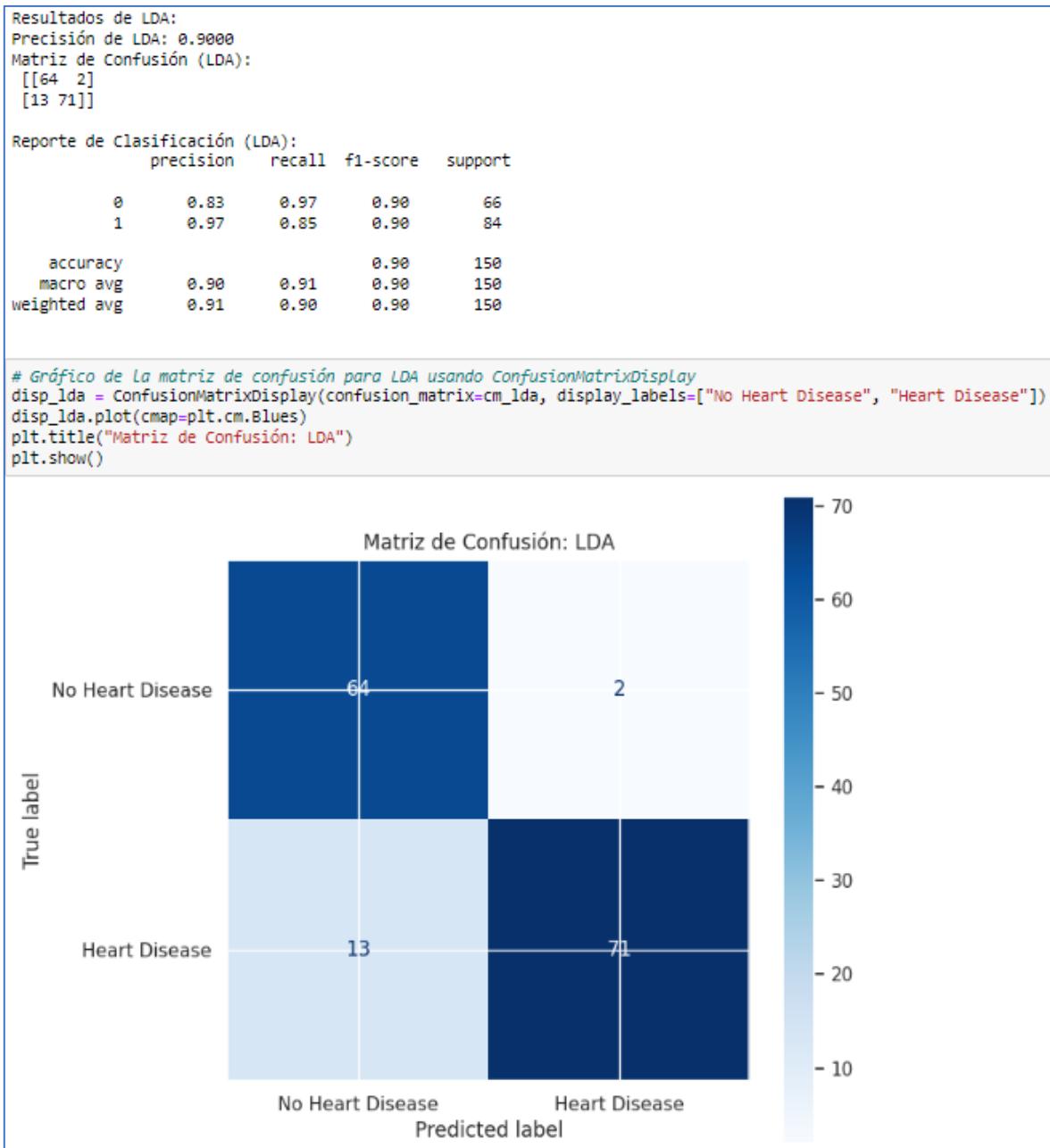
Se excluye num en vista que podria generar un sobre ajuste en los resultados, en vista que target se origina de los valores de 'num'.

Se procede a probar el resultado de los modelos una vez que se han consolidado los valores en una caracteristica del tipo booleana.

```
# Definir Las variables predictoras excluyendo 'target' y 'num'  
X = df4.drop(columns=['target', 'num'])  
  
# Definir La columna 'target' como La variable objetivo  
Y = df4['target']  
  
# Estandarizar Las características  
obj_escalar = StandardScaler()  
X_estandarizado = obj_escalar.fit_transform(X)  
  
# División de Los datos en conjunto de entrenamiento y prueba  
X_train, X_test, Y_train, Y_test = train_test_split(X_estandarizado, Y, test_size=0.2, random_state=0)  
  
# Lista de modelos  
models = []  
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))  
models.append(('LDA', LinearDiscriminantAnalysis()))  
models.append(('KNN', KNeighborsClassifier()))  
models.append(('CART', DecisionTreeClassifier()))  
models.append(('NB', GaussianNB()))  
models.append(('SVC', SVC()))  
  
# Inicializar resultados y nombres  
resultados = []  
names = []  
  
# Evaluar Los modelos  
for name, model in models:  
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)  
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')  
    resultados.append(cv_resultados)  
    names.append(name)  
    print('%s: Accuracy = %f (Std = %f)' % (name, cv_resultados.mean(), cv_resultados.std()))  
  
# Inicializar nombres y resultados de accuracy  
names = []  
accuracies = []  
  
# Evaluar Los modelos  
for name, model in models:  
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)  
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')  
    accuracies.append(cv_resultados.mean())  
    names.append(name)  
  
# Crear un DataFrame para mostrar Los resultados de accuracy  
accuracies_test3 = pd.DataFrame(accuracies, columns=['Accuracy_Test3'], index=names)  
  
# Mostrar La tabla de resultados de accuracy  
# se almacena el resultado para realizar un acoparacion con Las diferentes corridas y pruebas  
print(accuracies_test3)  
  
LR: Accuracy = 0.824492 (Std = 0.046455)  
LDA: Accuracy = 0.831215 (Std = 0.051061)  
KNN: Accuracy = 0.831158 (Std = 0.027046)  
CART: Accuracy = 0.797768 (Std = 0.036359)  
NB: Accuracy = 0.817740 (Std = 0.055654)  
SVC: Accuracy = 0.847881 (Std = 0.042925)
```

Evaluación de Resultados

Matriz de Confusión LDA:



Matriz de Confusión KNN:

Resultados de KNN:

Precisión de KNN: 0.8667

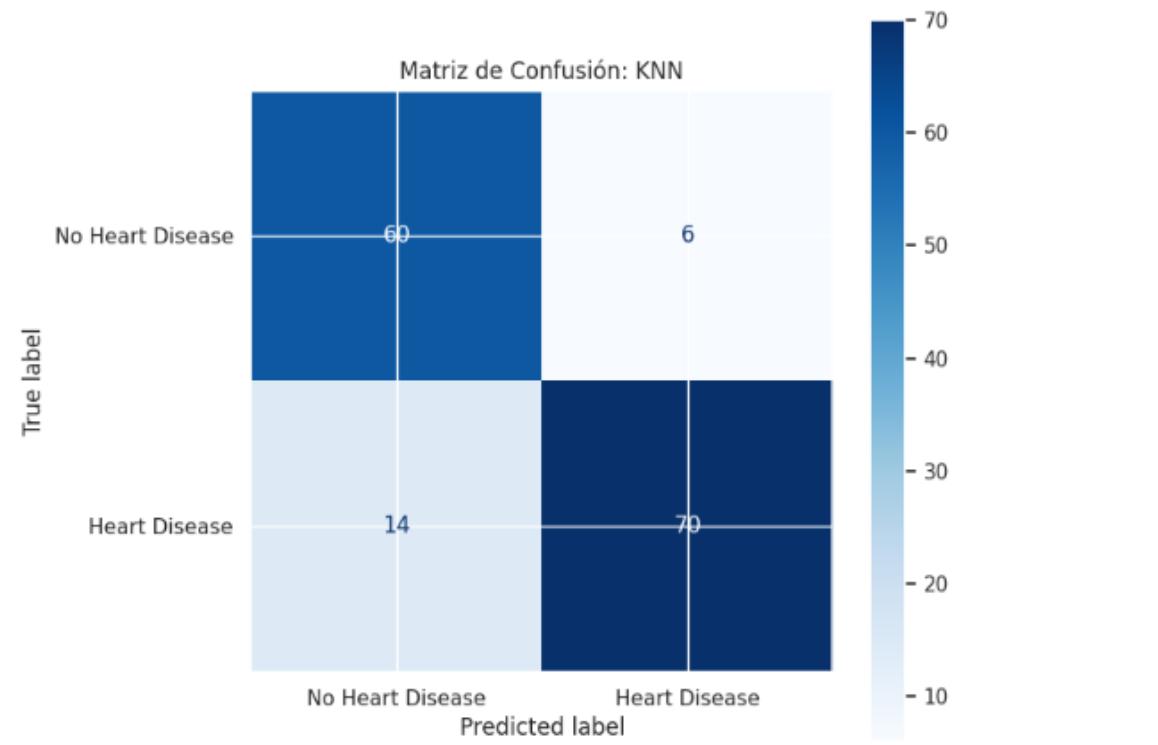
Matriz de Confusión (KNN):

```
[[60  6]
 [14 70]]
```

Reporte de Clasificación (KNN):

	precision	recall	f1-score	support
0	0.81	0.91	0.86	66
1	0.92	0.83	0.88	84
accuracy			0.87	150
macro avg	0.87	0.87	0.87	150
weighted avg	0.87	0.87	0.87	150

```
# Gráfico de La matriz de confusión para KNN usando ConfusionMatrixDisplay
disp_knn = ConfusionMatrixDisplay(confusion_matrix=cm_knn, display_labels=["No Heart Disease", "Heart Disease"])
disp_knn.plot(cmap=plt.cm.Blues)
plt.title("Matriz de Confusión: KNN")
plt.show()
```



Matriz de Confusión SVC:

```
# Matriz de confusión y reporte de clasificación para SVC
print("Resultados de SVC:")
cm_svc = confusion_matrix(y_test, y_pred_svc)
ac_svc = accuracy_score(y_test, y_pred_svc)
print(f"Precisión de SVC: {ac_svc:.4f}")
print("Matriz de Confusión (SVC):\n", cm_svc)
print("\nReporte de Clasificación (SVC):\n", classification_report(y_test, y_pred_svc))
```

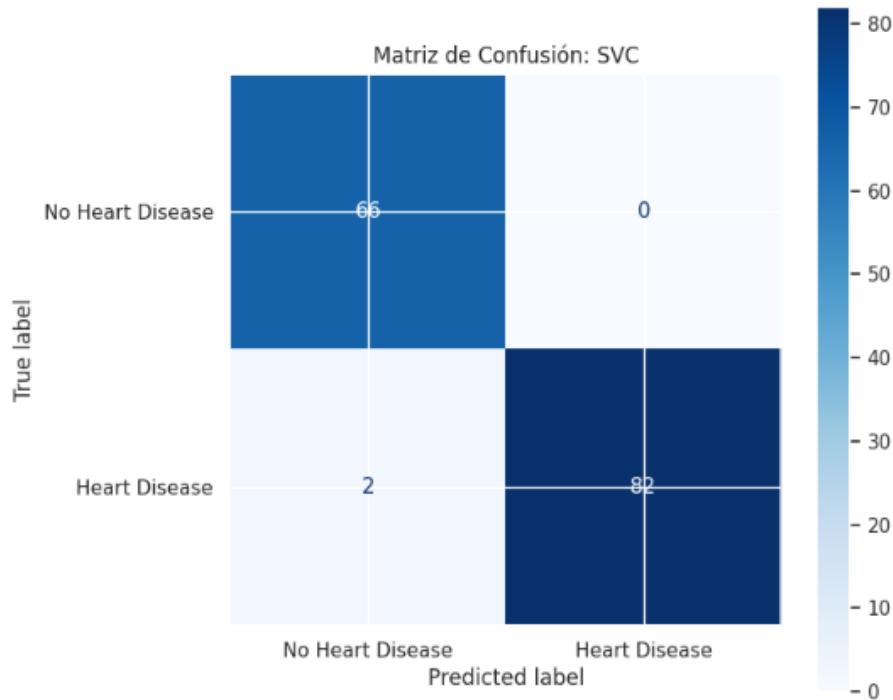
Resultados de SVC:

Precisión de SVC: 0.9867
Matriz de Confusión (SVC):
[[66 0]
 [2 82]]

Reporte de Clasificación (SVC):

	precision	recall	f1-score	support
0	0.97	1.00	0.99	66
1	1.00	0.98	0.99	84
accuracy			0.99	150
macro avg	0.99	0.99	0.99	150
weighted avg	0.99	0.99	0.99	150

```
# Gráfico de La matriz de confusión para SVC usando ConfusionMatrixDisplay
disp_svc = ConfusionMatrixDisplay(confusion_matrix=cm_svc, display_labels=["No Heart Disease", "Heart Disease"])
disp_svc.plot(cmap=plt.cm.Blues)
plt.title("Matriz de Confusión: SVC")
plt.show()
```



Redes Neuronales sobre Característica Target

Se procedió también a realizar una corrida de código con una red neuronal sencilla en la que se busca probar la posibilidad de una mejora en la predicción de la característica target, como se comentó anteriormente, se realizó una transformación de la característica objetivo, creando una nueva variable binaria denominada "Target", que clasifica a los pacientes en dos grupos: aquellos sin enfermedad cardíaca y aquellos con algún tipo de afección cardíaca. Se definió el conjunto de variables predictoras excluyendo las características que no aportaban al análisis, como 'num', y se procedió a dividir el conjunto de datos en entrenamiento y prueba.

```
# Definir las variables predictoras excluyendo 'target' y 'num'  
X = df4.drop(columns=['target', 'num'])  
  
# Definir la columna 'target' como la variable objetivo  
Y = df4['target']  
  
# División de los datos en conjunto de entrenamiento y prueba  
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)  
  
from sklearn.neural_network import MLPClassifier  
from sklearn.preprocessing import StandardScaler  
  
scaler = StandardScaler()  
X_train_scaled = scaler.fit_transform(X_train)  
X_test_scaled = scaler.transform(X_test)  
# Inicializar el modelo  
ann_model = MLPClassifier(hidden_layer_sizes=(100,), max_iter=500, activation='relu', solver='adam', random_state=1)  
# Inicializar el modelo  
ann_model.fit(X_train_scaled, Y_train)  
  
# Entrenar el modelo con los datos  
ann_model.fit(X_train, Y_train)  
  
*      MLPClassifier  
MLPClassifier(max_iter=500, random_state=1)
```

Evaluación del Modelo ANN

```
accuracy_ann = ann_model.score(X_test_scaled, Y_test)
print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)

Accuracy del modelo de redes neuronales (ANN):  0.7533333333333333

# Definir la red neuronal (ANN) con un espacio de búsqueda reducido
param_grid = {
    'hidden_layer_sizes': [(50,), (100,)],  # Menos configuraciones para capas ocultas
    'activation': ['relu'],  # Solo 'relu'
    'solver': ['adam'],  # Solo el optimizador 'adam'
    'alpha': [0.0001, 0.001],  # Menos valores de regularización
    'learning_rate': ['constant'],  # Solo tasa de aprendizaje constante
    'max_iter': [500]  # Reducir el número de iteraciones
}

# Inicializar el modelo de ANN
mlp = MLPClassifier(random_state=42)

# Utilizar GridSearchCV para encontrar los mejores hiperparámetros
grid_search = GridSearchCV(mlp, param_grid, cv=5, scoring='accuracy', verbose=3, n_jobs=-1)
grid_search.fit(X_train, Y_train)

# Mostrar los mejores hiperparámetros encontrados
print("Mejores hiperparámetros encontrados:")
print(grid_search.best_params_)

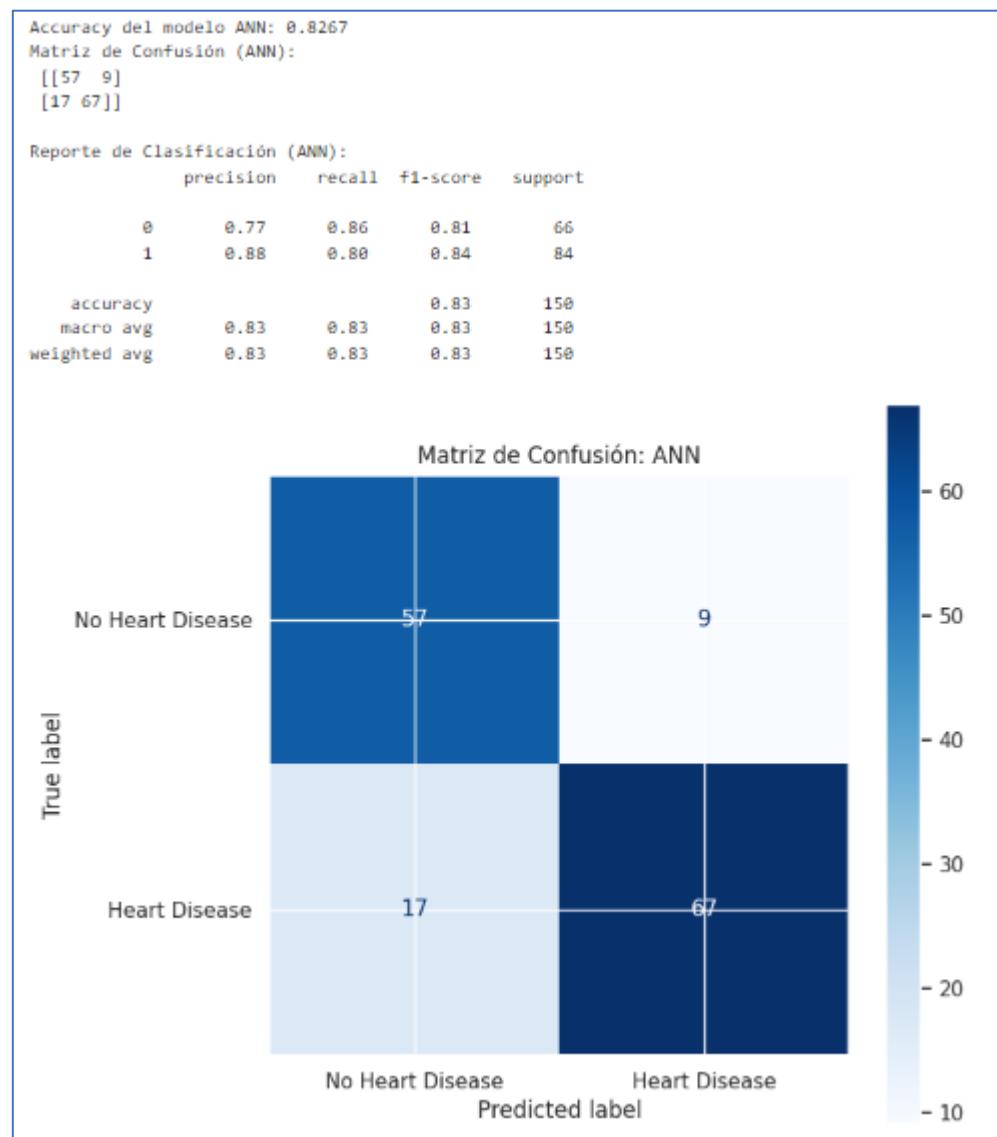
Fitting 5 folds for each of 4 candidates, totalling 20 fits
Mejores hiperparámetros encontrados:
{'activation': 'relu', 'alpha': 0.0001, 'hidden_layer_sizes': (50,), 'learning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}

# Entrenar el modelo con los mejores hiperparámetros
mejor_ann = grid_search.best_estimator_

# Evaluar el modelo con los datos de prueba
accuracy_ann_opt = mejor_ann.score(X_test, Y_test)
print(f"Accuracy del mejor modelo ANN con GridSearchCV: {accuracy_ann_opt:.4f}")

Accuracy del mejor modelo ANN con GridSearchCV: 0.8267
```

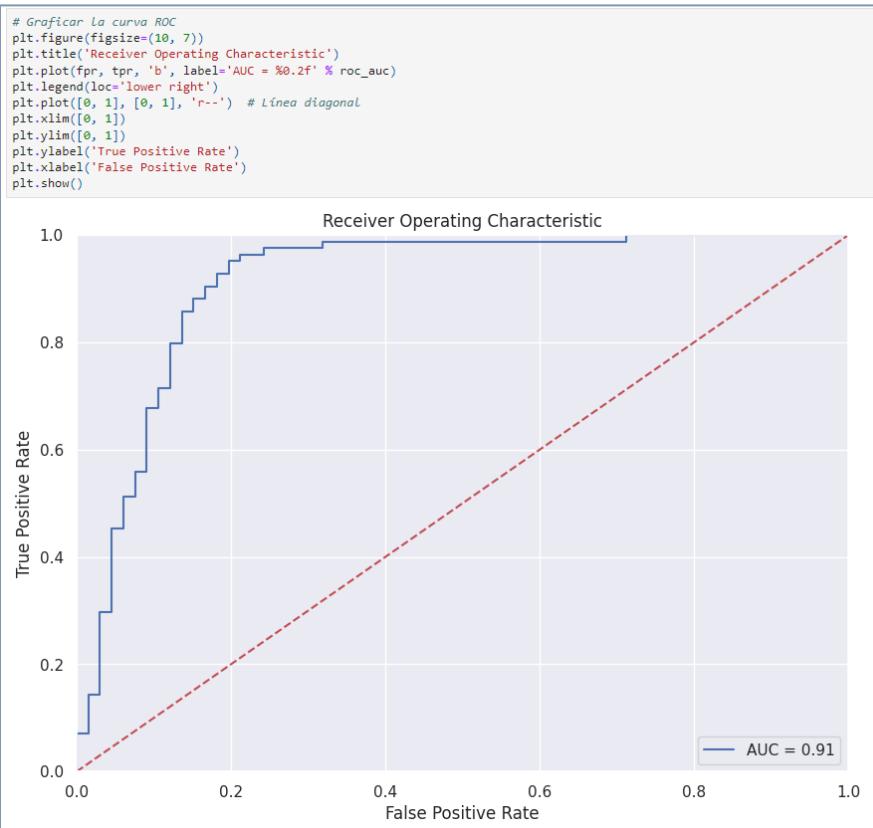
Matriz de Confusión Redes Neuronales:



Un resultado satisfactorio, a mejorado con los ultimo ajustes aplicados.

- El modelo ha clasificado correctamente 57 casos como "No Heart Disease"
- clasificado incorrectamente 9 casos de "No Heart Disease" como "Heart Disease", lo que indica falsos positivos
- Ha clasificado correctamente 67 casos como "Heart Disease".

Gráfica ROC



Tenemos la curva de evaluación ROC, la cual explica que Cuanto más cerca esté la curva ROC de la esquina superior izquierda, mejor será el clasificador, ya que indicará más verdaderos positivos y menos falsos positivos.

Por lo tanto, si la curva está más cerca de la esquina superior izquierda, el modelo está logrando identificar correctamente más verdaderos positivos (TP) mientras mantiene los falsos positivos (FP) bajos, lo que indica un buen rendimiento, por otra parte, cuanto más se aleja la curva de la diagonal, mejor es el modelo, ya que puede identificar correctamente los casos positivos con pocos errores.

Según la gráfica anterior sobre los resultados calculados, se denota que el resultado que está generando el modelo es un buen resultado, presentando un AUC = 0.91: El área bajo la curva (AUC) es una métrica que mide qué tan bien el modelo distingue entre clases.

Resumen de los Resultados Obtenidos

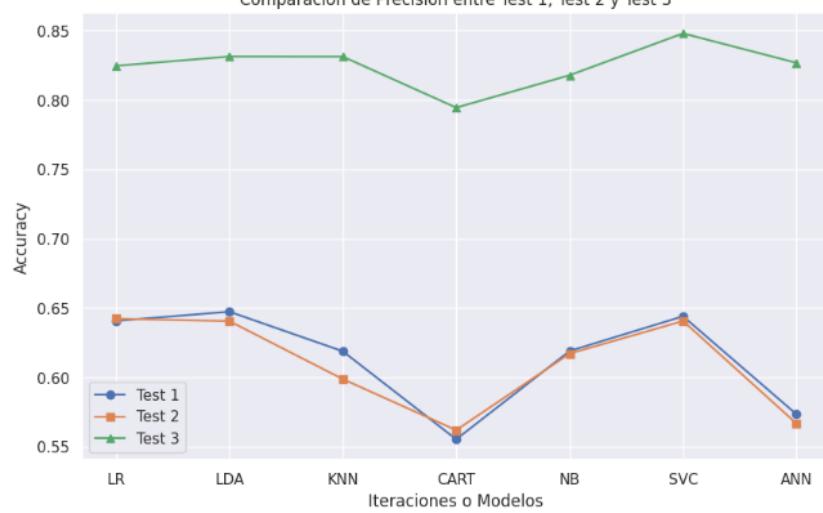
Se capturaron los datos de los diferentes testeos realizados y se resumieron en una tabla, lo cual permite apreciar los diferentes resultados obtenidos.

	Accuracy_Test1	Accuracy_Test2	Accuracy_Test3
LR	0.640621	0.642232	0.824492
LDA	0.647316	0.640508	0.831215
KNN	0.618785	0.598729	0.831158
CART	0.555311	0.561921	0.794379
NB	0.618955	0.617090	0.817740
SVC	0.643983	0.640593	0.847881
ANN	0.573333	0.566667	0.826667

```
# Concatenar Los DataFrames por columnas
df_comparacion = pd.concat([accuracias_test1, accuracias_test2, accuracias_test3], axis=1)

# Mostrar La tabla resultante
print(df_comparacion)
```

Comparación de Precisión entre Test 1, Test 2 y Test 3



De forma exhaustiva se realizaron varios intentos y testeos sobre el conjunto de datos, por medio de la implementación de los diferentes modelos de aprendizaje automático, con esto se lograron diferentes resultados, observando el anterior gráfico comparativo se pueden apreciar como el rendimiento de los modelos varían según cada una de las pruebas. (Test 1, Test 2 y Test 3). Al principio, los resultados de Test 1 y Test 2 presentan rendimientos conservadores, con niveles de precisión (accuracy) que varían

entre los modelos, resultado muy bajos para decir que se está logrando una predicción de la situación. Sin embargo, con la implementación de mejoras como el ajuste de hiperparámetros y técnicas de feature engineering, se observa una mejora notable en los resultados de Test 3, especialmente en modelos como Logistic Regression (LR), SVC y ANN.

El comportamiento de los modelos refleja una evolución en su capacidad predictiva. Modelos como ANN (redes neuronales) muestran una curva ascendente significativa, lo que mejora la capacidad predictiva de la característica una vez que se trabaja el target a predecir, además de las mejoras en los parámetros, lo que sugiere que los ajustes realizados fueron particularmente efectivos en este modelo. Por otro lado, modelos como KNN y CART presentan una mejora más gradual, lo que podría indicar que estos modelos requieren ajustes más específicos para alcanzar rendimientos superiores.

La visualización de las tres pruebas permite identificar fácilmente las diferencias entre los modelos y cómo sus resultados mejoran o se mantienen consistentes a lo largo de las iteraciones. Esta gráfica, junto con los resultados de la tabla de precisión, subraya la importancia de los ajustes y las pruebas múltiples para obtener modelos más robustos y precisos.

Modelo Final Ejecutable

Crear el Modelo Final en un ejecutable

```
import joblib

# Guardar el modelo SVC entrenado en un archivo
joblib.dump(modelo_final, 'modelo_SVC_entrenado.pkl')

['modelo_SVC_entrenado.pkl']

Llamar el Modelo Entrenado

heart_disease = df4.copy()

heart_disease.columns

Index(['id', 'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'num', 'age_range',
       'Col_range', 'target'],
      dtype='object')

heart_disease = heart_disease.drop(columns=['num', 'target'])

heart_disease.columns

Index(['id', 'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'age_range',
       'Col_range'],
      dtype='object')

# Cargar el modelo guardado posteriormente
modelo_cargado = joblib.load('modelo_SVC_entrenado.pkl')

# Usar el modelo cargado para hacer predicciones
predicciones = modelo_cargado.predict(heart_disease)

# Añadir las predicciones al DataFrame en una nueva columna llamada 'HeartDisease_Pred'
heart_disease['HeartDisease_Pred'] = predicciones

# Mostrar las primeras filas del DataFrame con las predicciones añadidas
heart_disease.head()
```

	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	age range	Col range	HeartDisease Pred
0	1	63	1	0	145.0	233.0	1	1	150.0	0	2.3	0	0.0	2	60	1	1
1	2	67	1	3	160.0	286.0	0	1	108.0	1	1.5	1	3.0	0	60	2	1
2	3	67	1	3	120.0	229.0	0	1	129.0	1	2.6	1	2.0	1	60	1	1
3	4	37	1	2	130.0	250.0	0	0	187.0	0	3.5	0	0.0	0	30	2	1
4	5	41	0	1	130.0	204.0	0	1	172.0	0	1.4	2	0.0	0	40	1	1

Se procedió a capturar uno de los modelos como parte de las pruebas, según los resultados de predicción se tomó el modelo SVC como el modelo_final, el cual se procedió a cargar por medio de joblib, con esto se crea un ejecutable u objeto que se invoca que contiene los parámetros del modelo entrenado, se le carga el dataset original y se calculan las predicciones, finalmente estas predicciones se convierten en la nueva columna, e este caso denominada HeartDiseases Pred, este paso concluye todo el análisis como una primera etapa en vista que en sí los modelos sufren de constante desactualización por lo tanto deben ser sometidos a seguimiento y entrenamiento periódico, este último paso es de suma importancia en los ambientes de aprendizaje automático, cuando se busca lanzar un modelo en producción o prueba.

CONCLUSIONES

El análisis de los datos obtenidos a través del proceso de captura con los wearables, y su posterior visualización en los dashboards de Power BI, ha demostrado la enorme utilidad de estos dispositivos para el seguimiento continuo de la salud. Los resultados reflejan de manera clara cómo métricas clave como el número de pasos diarios, las calorías quemadas, el sedentarismo y los minutos de actividad intensa pueden ser aprovechadas no solo para obtener una visión detallada del estado físico de los usuarios, sino también para ser utilizadas en un contexto clínico.

Uno de los principales beneficios que se derivan de este análisis es la capacidad de monitorear a distancia la salud de los pacientes mediante dispositivos wearables. Esto permitiría a los médicos, con el consentimiento del paciente, realizar un seguimiento continuo y detallado de parámetros como la calidad del sueño, el ritmo cardíaco, y los niveles de actividad física, sin la necesidad de una consulta presencial.

La exploración de los conjuntos de datos llevada a cabo fue meticulosa y completa, abarcando no solo la identificación de valores faltantes, sino también la detección y tratamiento de valores atípicos o inconsistentes. A lo largo del proceso, se implementaron técnicas de imputación para sustituir valores nulos mediante la mediana para variables numéricas y la moda para variables categóricas. Este enfoque no solo mejoró la calidad del dataset, sino que garantizó su coherencia, evitando sesgos que pudieran distorsionar los resultados.

Si bien los resultados obtenidos en este proyecto son prometedores, es importante considerar algunas limitaciones que podrían haber influido en la precisión de los modelos de machine learning aplicados. En primer lugar, uno de los desafíos más significativos fue el desbalance de clases en los datasets utilizados, particularmente en aquellos relacionados con la predicción de enfermedades cardíacas, para lo cual se implementaron varios modelos de aprendizaje automático tales como KNN, Random Forest, SVC y Redes Neuronales, entre otros, los mismos fueron seleccionados por su capacidad analítica.

Otro aspecto relevante es la calidad variable de los datos recopilados a través de los dispositivos wearables. Los wearables, aunque útiles, no siempre generan datos de alta precisión, ya que pueden estar influenciados por factores externos como la posición del dispositivo, la conectividad y el comportamiento del usuario. Estos factores pueden introducir ruido en el dataset, afectando la calidad de los resultados de los modelos predictivos. Para mitigar estos problemas, se implementaron técnicas de imputación y

detección de valores atípicos; sin embargo, la imprecisión inherente a los datos sigue siendo una limitación.

Además se incorporan mediciones de la calidad de predicción de los modelos, una vez que se detectaron fallos en la calidad se tomaron decisiones para mejorar los resultados sin sacrificar la calidad de los datos, por lo tanto la integración de la matriz de confusión así como el indicador de accuracy y finalmente la curva ROC, permite evaluar la calidad de los modelos aplicados, basados en los indicadores de calidad de predicción fue posible ir ajustando y tomar decisiones en pro de mejorar la capacidad de predicción.

Se llevó a cabo con el objetivo de incorporar los datos recopilados de dispositivos wearables y aplicar tecnologías avanzadas como el aprendizaje automático, así como análisis descriptivos y predictivos, para identificar patrones y obtener información valiosa de los datos. Estos objetivos se cumplieron mediante el uso exhaustivo de los conjuntos de datos seleccionados. Además, los constantes avances en los dispositivos médicos, junto con el aumento en la capacidad de análisis y la reducción en el tamaño y costo de los dispositivos, nos sitúan en una etapa en la que es posible reducir significativamente los costos asociados al diagnóstico médico.

En conclusión, este proyecto demuestra cómo la integración de datos de wearables y técnicas avanzadas de análisis puede revolucionar la medicina preventiva y personalizada, ofreciendo un enfoque más accesible y eficiente para el diagnóstico y monitoreo de la salud.

BIBLIOGRAFÍA

- Abbott. (2023). *Freestyle Libre*. Recuperado de <https://www.freestylelibre.us>
- Alam, T., Biswas, A., & Dawar, R. (2020). IoT-based smart health monitoring system: A review. *IEEE Sensors Journal*, 20(1), 22-30.
- Amwell. (2023). *Telehealth Solutions*. Recuperado de <https://www.amwell.com>
- Apple Inc. (2023). *Apple Watch*. Recuperado de <https://www.apple.com/watch>
- Aprendizaje Automatico. Ediciones Roble, S.L. (2023). Modalidades y técnicas de deep learning. Ediciones Roble, S.L.
- Cerner Corporation. (2023). *Cerner Millennium*. Recuperado de <https://www.cerner.com>
- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. Recuperado de <https://www.kde.cs.uni-kassel.de/wp-content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf>
- Data Science Project Management. (n.d.). *CRISP-DM 2.0*. Recuperado el 28 de agosto de 2024, de <https://www.datascience-pm.com/crisp-dm-2/>
- Epic Systems Corporation. (2023). *Epic EHR*. Recuperado en <https://www.epic.com>
- Fitbit. (2023). *Fitbit Devices*. Disponible en <https://www.fitbit.com>
- Hardy, R. (2020). Fitness tracker and smartwatch data used to predict COVID-19 cases. *New Atlas*. Recuperado de <https://newatlas.com/health-wellbeing/fitness-tracker-smartwatch-data-coronavirus-diagnose-predict/>
- International Agency for Research on Cancer. (2021). *Global cancer observatory: Breast cancer*. Recuperado de <https://gco.iarc.fr/today/data/factsheets/cancers/19-Breast-factsheet.pdf>
- IMF. (n.d.). Material suministrado parte del módulo: *El Almacén De Datos - Data Warehouse*. Ediciones Roble, S.L.
- Jiang, F., Jiang, Y., Wang, Y., Zhi, H., Dong, Y., Li, H., ... & Wang, X. (2020). Artificial Intelligence in Healthcare: Past, Present and Future. *Seminars in Cancer Biology*, 31, 1-11.
- Khatri, V., & Brown, M. (2019). The transformation of healthcare through the integration of IoT. *Journal of Healthcare Informatics Research*, 3(4), 233-245.
- Lee, J., Kim, S., & Kim, K. (2021). Personalized Health Management Using Wearable Devices and Data Analysis. *Journal of Biomedical Informatics*, 115, 103676.

- macrovector. (n.d.). *Diagrama de flujo infografía pulsera fitness isométrico* [Imagen]. Freepik. Recuperado el 28 de agosto de 2024, de https://www.freepik.es/vector-gratis/diagrama-flujo-infografia-pulsera-fitness-isometrico_10155699.htm#fromView=search&page=1&position=37&uuid=3cce3f37-6208-43e3-a07c-e4476cb454d7
- macrovector. (n.d.). *Reloj pulsera* [Icono]. Freepik. Recuperado el 28 de agosto de 2024, de https://www.freepik.es/icono/reloj-pulsera_977414
- macrovector. (n.d.). *Smartphone* [Icono]. Freepik. Recuperado el 28 de agosto de 2024, de https://www.freepik.es/icono/smartphone_2235300
- Medtronic. (2023). *Pacemakers and ICDs*. Disponible de <https://www.medtronic.com>
- Minitab. (s. f.). *¿Qué son las variables categóricas, discretas y continuas?* Recuperado el 28 de agosto de 2024, en <https://support.minitab.com/es-mx/minitab/help-and-how-to/statistical-modeling/regression/supporting-topics/basics/what-are-categorical-discrete-and-continuous-variables/>
- Modelos BI. (s. f.). *Preparación y proceso de datos: Imputación*. Material Suministrado en el curso. Ediciones Roble, S. L. IMF.
- Organización Mundial de la Salud. (2020). *Enfermedades no transmisibles: Prevención y control*. Recuperado de <https://www.who.int/activities/noncommunicable-diseases>
- Organización Mundial de la Salud. (2021). *Enfermedades cardiovasculares (ECV)*. Recuperado de [https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds))
- Organización Mundial de la Salud. (2021). *Salud digital: Estrategia global sobre salud digital 2020-2025*. Recuperado de <https://www.who.int/publications/i/item/9789240062884>
- Organización Mundial de la Salud. (2021). *Cáncer de piel*. Recuperado de <https://www.who.int/news-room/fact-sheets/detail/skin-cancer>
- Patel, S., Park, H., Bonato, P., Chan, L., & Rodgers, M. (2012). A review of wearable sensors and systems with application in rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 9, 21. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3354997/>
- Quer, G., Radin, J. M., Gadaleta, M., Baca-Motes, K., Ariniello, L., Ramos, E., Kheterpal, V., Topol, E. J., & Steinhubl, S. R. (2020). Wearable sensor data and self-reported symptoms for COVID-19 detection. *Nature Medicine*, 26(10), 1623-1625. <https://doi.org/10.1038/s41591-020-1123-x>

- Smith, J., & Jones, A. (2015). Advanced diagnostic tools in modern medicine. *Medical Journal of Technology*, 45(3), 157-172.
- Teladoc Health. (2023). *Telemedicine Solutions*. Disponible en <https://www.teladoc.com>
- Triola, M. F. (2020). *Estadística* (12.^a ed.). Pearson.
- Yun, S. M., Kim, M., Kwon, Y. W., Kim, H., Kim, M. J., Park, Y.-G., & Park, J.-U. (2021). Recent advances in wearable devices for non-invasive sensing. *Applied Sciences*, 11(3), Article 1235. <https://doi.org/10.3390/app11031235>

ANEXOS

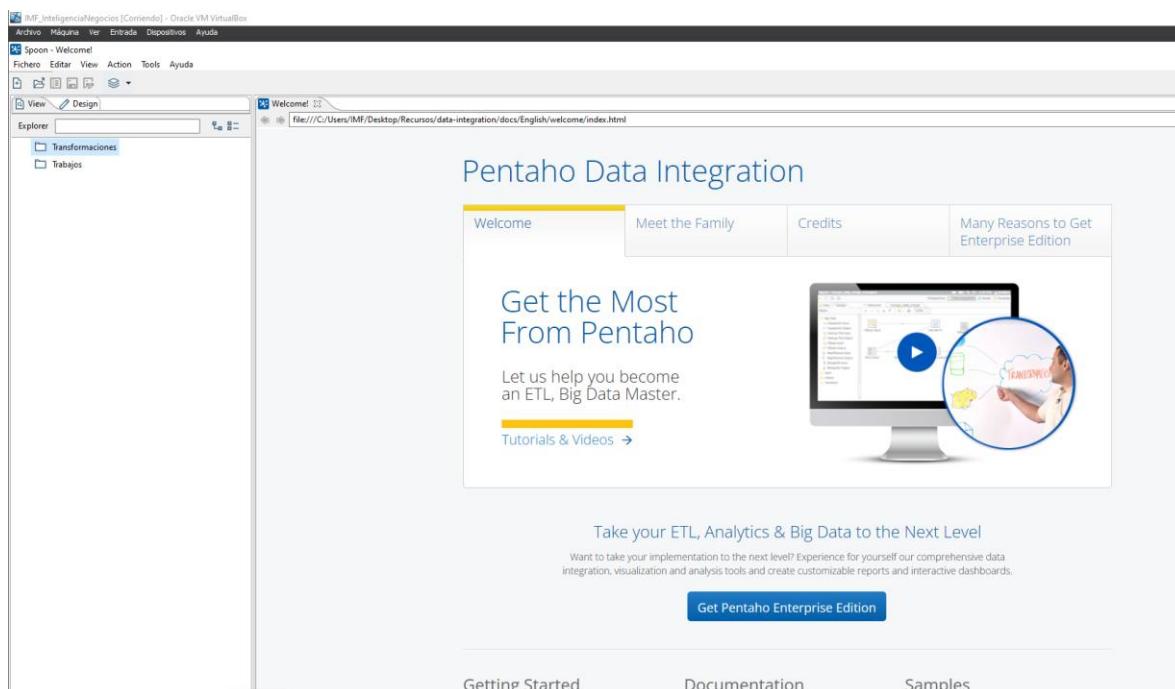
Anexo 1 : Proceso de Integración y configuracion Spoon Pentaho

Todo el proceso realizado para el análisis del dataset de Fitbit Fitness Tracker Data, puede ser consultado en este link

https://drive.google.com/drive/folders/1tAeWofqjrC3-VBZm2KEBC1hIDmzQYLz4?usp=drive_link

Las siguientes capturas muestran el proceso detallado de la elaboración y configuración del proceso de ETL para tratar el dataset de Fitbit Fitness Tracker Data.

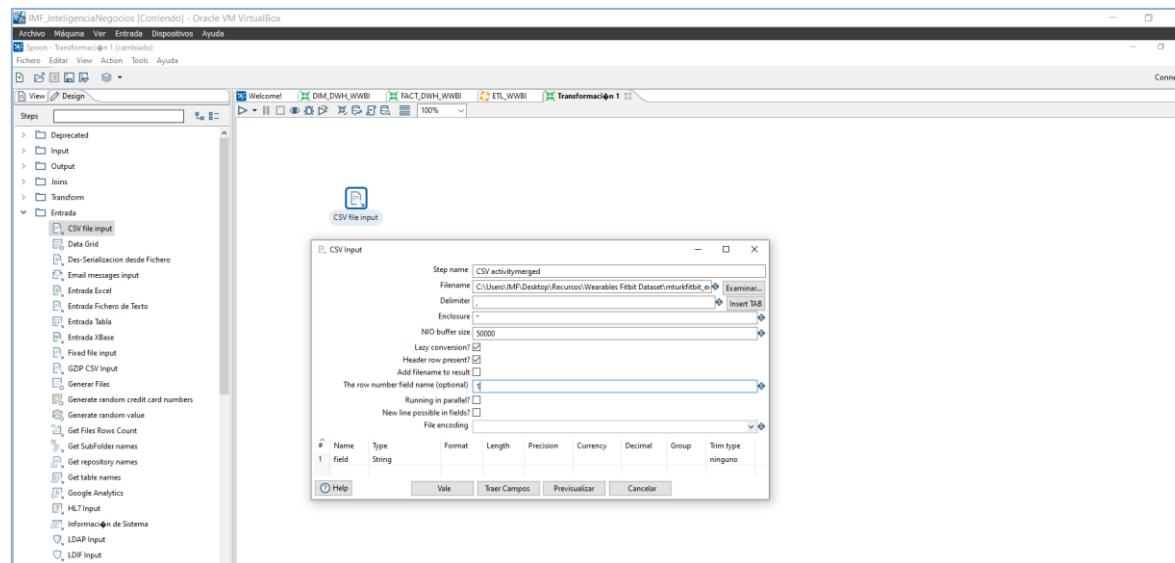




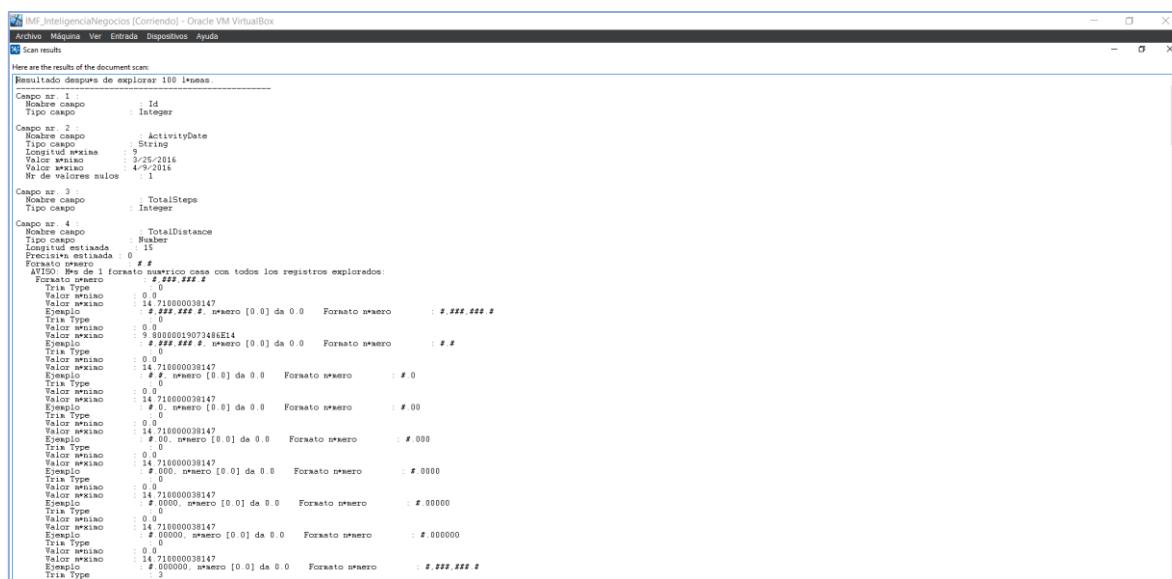
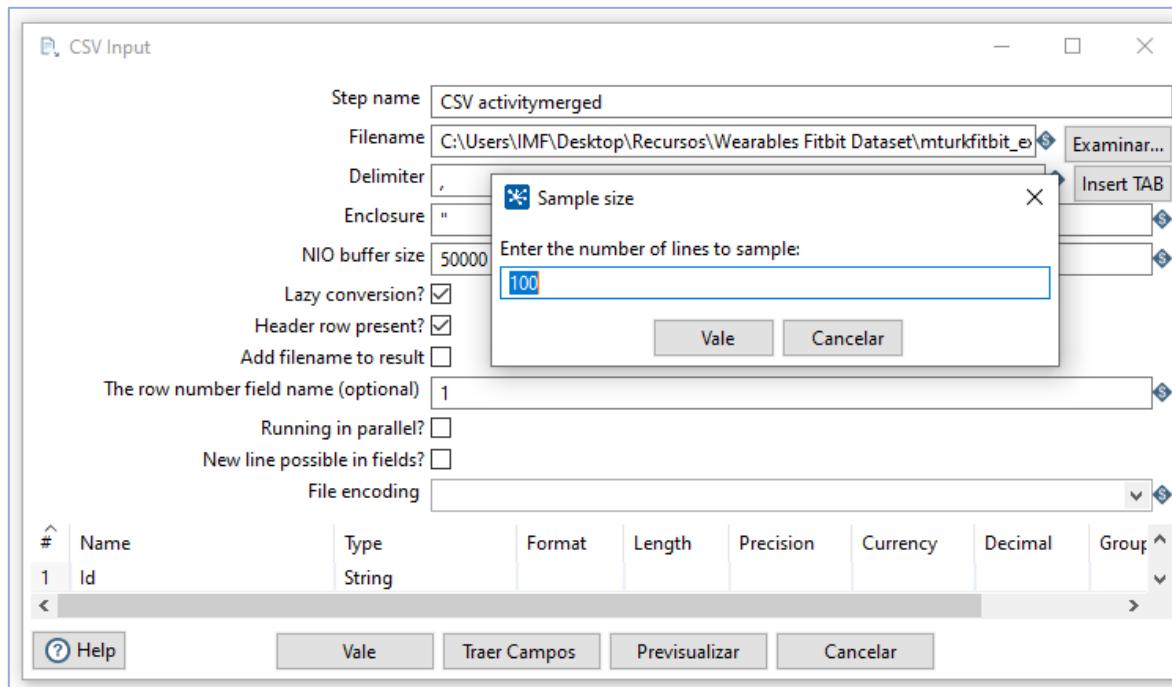
Getting Started

Documentation

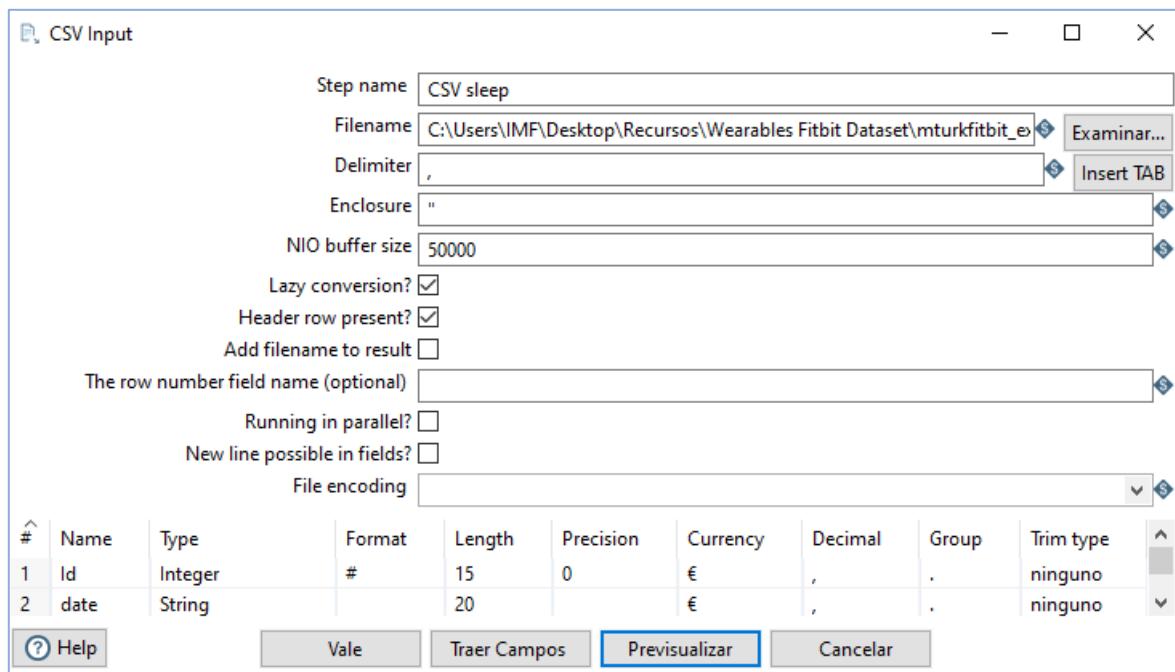
Samples



Traer los campos



Configuración del input de los datos



Previsualización de los datos que se van a traer

Rows of step: CSV sleep (1000 rows)					
#	Id	date	value	logId	
1	1503960366	3/13/2016 2:39:30 AM	1	11114919637	
2	1503960366	3/13/2016 2:40:30 AM	1	11114919637	
3	1503960366	3/13/2016 2:41:30 AM	1	11114919637	
4	1503960366	3/13/2016 2:42:30 AM	1	11114919637	
5	1503960366	3/13/2016 2:43:30 AM	1	11114919637	
6	1503960366	3/13/2016 2:44:30 AM	1	11114919637	
7	1503960366	3/13/2016 2:45:30 AM	2	11114919637	
8	1503960366	3/13/2016 2:46:30 AM	2	11114919637	
9	1503960366	3/13/2016 2:47:30 AM	1	11114919637	
10	1503960366	3/13/2016 2:48:30 AM	1	11114919637	
11	1503960366	3/13/2016 2:49:30 AM	2	11114919637	
12	1503960366	3/13/2016 2:50:30 AM	2	11114919637	
13	1503960366	3/13/2016 2:51:30 AM	1	11114919637	
14	1503960366	3/13/2016 2:52:30 AM	1	11114919637	
15	1503960366	3/13/2016 2:53:30 AM	1	11114919637	
16	1503960366	3/13/2016 2:54:30 AM	1	11114919637	
17	1503960366	3/13/2016 2:55:30 AM	1	11114919637	

IMF_InteligenciaNegocios [Comiendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

CSV Input

Step name: STG_heartrate

Filename: C:\Users\IMF\Desktop\Recursos\Wearables Fitbit Dataset\mturkfitbit_export_3.12.16-4.11.16\Fitbase Data 3.12.16-4.11.16\heartrate_seconds_merged.csv

Delimited , Enclosure "

NIO buffer size: 50000

Lazy conversion?

Header row present?

Add filename to result

The row number field name (optional)

Running in parallel?

New line possible in fields?

File encoding: UTF-8

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Id	Integer	#	15	0	€	.	.	ninguno
2	Time	Date	MM/dd/yyyy	19		€	.	.	ninguno
3	Value	Integer	#	15	0	€	.	.	ninguno

Examine preview data

Rows of step: STG_heartrate (1000 rows)

#	Id	Time	Value
1	20224848408	04/01/2016	93
2	20224848408	04/01/2016	91
3	20224848408	04/01/2016	96
4	20224848408	04/01/2016	66
5	20224848408	04/01/2016	100
6	20224848408	04/01/2016	101
7	20224848408	04/01/2016	104

Previsualizar los datos

Examine preview data

Rows of step: CSV activitiymerged (457 rows)

#	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightlyActiveDistance	SedentaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedentaryMinutes	Calories	
1	1503960366	3/25/2016	11004	7.1	7.1	0	2.6	0.5	4.1	0	33	12	205	804	181	
2	1503960366	3/26/2016	17609	11.6	11.6	0	6.9	0.7	3.9	0	89	17	274	588	215	
3	1503960366	3/27/2016	17236	8.5	8.5	0	4.7	0.2	3.7	0	56	5	268	605	194	
4	1503960366	3/28/2016	13231	8.9	8.9	0	3.2	0.8	4.9	0	39	20	224	1000	193	
5	1503960366	3/29/2016	10241	7.8	7.8	0	2.2	1.1	4.6	0	28	28	28	540	188	
6	1503960366	3/30/2016	10970	7.2	7.2	0	2.4	0.3	4.3	0	30	13	223	763	188	
7	1503960366	3/31/2016	12256	7.9	7.9	0	2.3	0.5	5	0	33	12	239	820	188	
8	1503960366	4/1/2016	12286	7.9	7.9	0	3.3	0.8	3.6	0	47	21	209	866	186	
9	1503960366	4/2/2016	11248	7.2	7.2	0	3	0.4	3.7	0	40	11	244	636	184	
10	1503960366	4/3/2016	10016	6.4	6.4	0	0.9	1.3	4.2	0	15	30	314	655	185	
11	1503960366	4/4/2016	14557	9.8	9.8	0	3.4	0.7	5.7	0	43	18	285	757	203	
12	1503960366	4/5/2016	10544	5.9	5.9	0	2.9	0.8	5	0	36	18	257	256	206	
13	1503960366	4/6/2016	11974	7.7	7.7	0	2	0.3	52	0	27	12	228	1170	186	
14	1503960366	4/7/2016	10198	6.4	6.4	0	1.2	0.8	4.4	0	17	20	195	1208	185	
15	1503960366	4/8/2016	12521	7.9	7.9	0	3.3	0.9	3.7	0	46	22	212	1160	189	
16	1503960366	4/9/2016	12432	8.1	8.1	0	2.6	0.6	4.9	0	32	15	248	738	188	
17	1503960366	4/10/2016	10557	7	7	0	4	0.5	2.5	0	44	13	168	737	175	
18	1503960366	4/11/2016	10990	7.3	7.3	0	2	0.6	4.7	0	26	14	216	855	181	
19	1503960366	4/12/2016	224	0.1	0.1	0	0	0	0	0	0	0	0	9	32	5
20	1424580081	3/25/2016	1810	1.2	1.2	0	0	0	1.1	0	0	0	0	121	133	139
21	1424580081	3/26/2016	815	0.5	0.5	0	0	0	0	0	0	0	0	47	193	126
22	1424580081	3/27/2016	1985	1.3	1.3	0	0	0	0	1.3	0	0	0	112	1328	135
23	1424580081	3/28/2016	1905	1.2	1.2	0	0	0	0	1.2	0	0	0	95	1345	134
24	1424580081	3/29/2016	1552	1	1	0	0	0	0	1	0	0	0	66	1374	130
25	1424580081	3/30/2016	1675	1.1	1.1	0	0	0	0	1.1	0	0	0	84	1356	131
26	1424580081	3/31/2016	4596	2.9	2.9	0	0.5	0.3	2.2	0	7	4	144	1295	149	
27	1424580081	4/1/2016	9218	6	6	0	0	0	0	6	0	0	0	221	1278	154
28	1424580081	4/2/2016	1556	1	1	0	0	0	0	1	0	0	0	88	1352	132
29	1424580081	4/3/2016	2910	1.9	1.9	0	0	0	0	1.9	0	0	0	157	1283	145
30	1424580081	4/4/2016	18464	12	12	0	0	0	0	12	0	0	0	270	1170	157
31	1424580081	4/5/2016	1335	0.9	0.9	0	0	0	0	0.9	0	0	0	74	1366	130
32	1424580081	4/6/2016	1004	0.6	0.6	0	0	0	0	0.6	0	0	0	55	1385	127

CSV Input

Step name: CSV activitiymerged

Filename: C:\Users\IMF\Desktop\Recursos\Wearables Fitbit Dataset\mturkfitbit_export_3.12.16-4.11.16\Fitbase Data 3.12.16-4.11.16\dailyactivity_merged.csv

Delimited , Enclosure "

NIO buffer size: 50000

Lazy conversion?

Header row present?

Add filename to result

The row number field name (optional): 1

Running in parallel?

New line possible in fields?

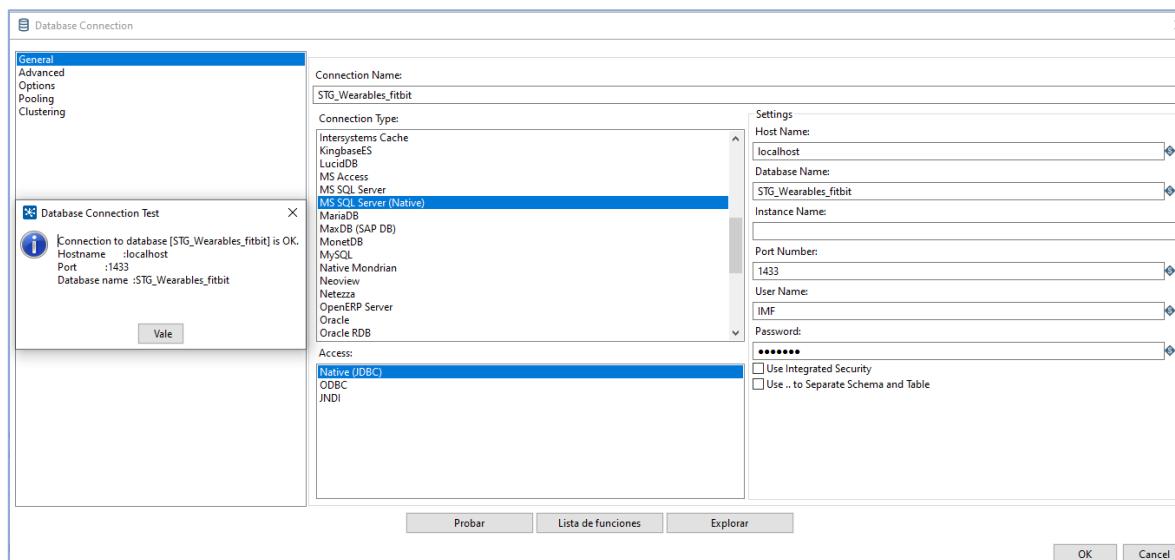
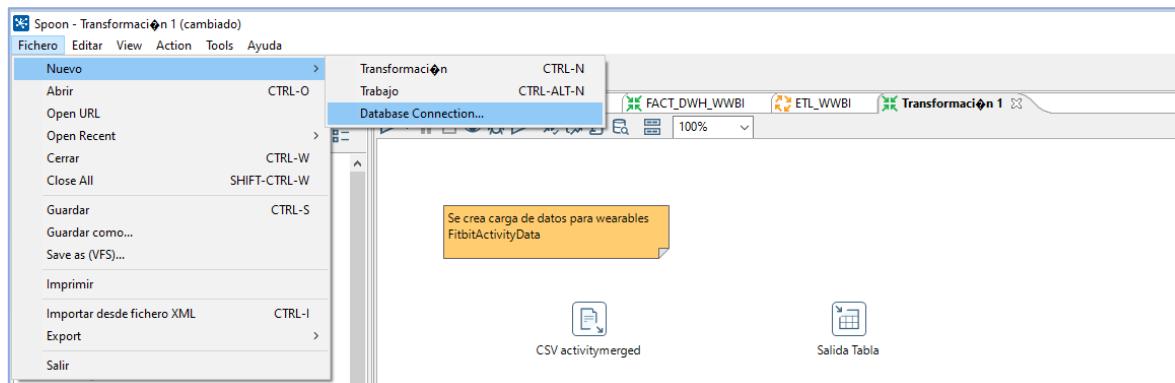
File encoding: UTF-8

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Id	Number	#	20	0	€	.	.	ninguno
2	ActivityDate	Date	MM/dd/yyyy	10	0	€	.	.	ninguno
3	TotalSteps	Integer	#	20	0	€	.	.	ninguno
4	TotalDistance	Number	#.#	20	0	€	.	.	ninguno
5	TrackerDistance	Number	#.#	20	0	€	.	.	ninguno
6	LoggedActivitiesDistance	Integer	#	20	0	€	.	.	ninguno
7	VeryActiveDistance	Number	#.#	20	0	€	.	.	ninguno
8	ModeratelyActiveDistance	Number	#.#	20	0	€	.	.	ninguno
9	SedentaryActiveDistance	Number	#.#	20	0	€	.	.	ninguno
10	SedentaryActiveDistance	Number	#.#	20	0	€	.	.	ninguno
11	VeryActiveMinutes	Integer	#	20	0	€	.	.	ninguno
12	FairlyActiveMinutes	Integer	#	20	0	€	.	.	ninguno
13	LightlyActiveMinutes	Integer	#	20	0	€	.	.	ninguno
14	SedentaryMinutes	Integer	#	20	0	€	.	.	ninguno
15	Calories	Integer	#	20	0	€	.	.	ninguno

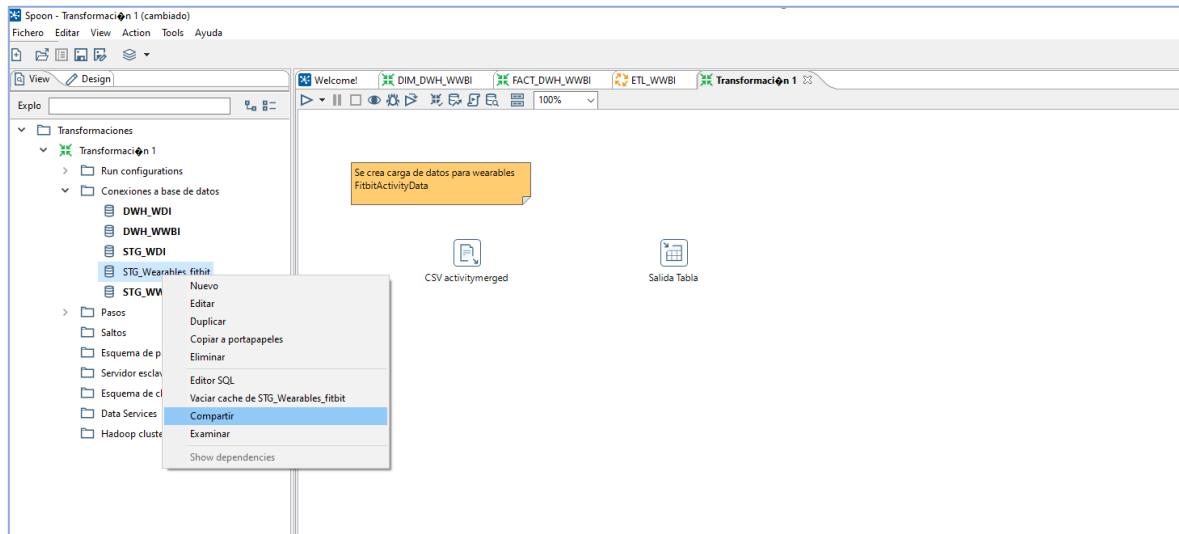
Examine preview data

Insert

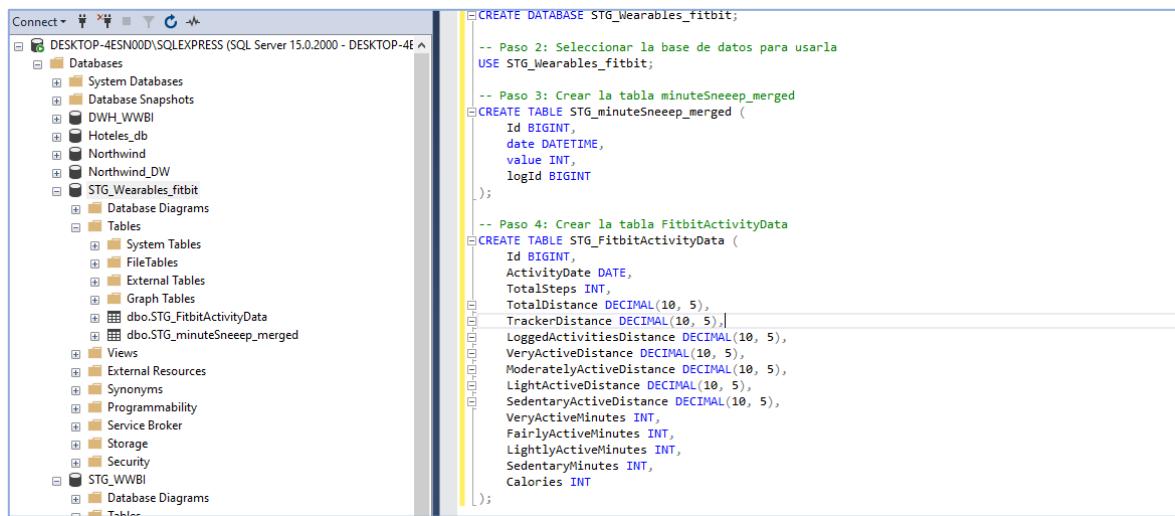
Crear la conexión a base de datos, la cual es necesaria para que los datos se envíen y carguen a la base de datos staging.



Permitir que la conexión esté en forma compartida para permitir el uso en otros Jobs / tareas



Crear la base de datos Staging (SQL)



The screenshot shows the Object Explorer on the left with the database **STG_Wearables_fitbit** selected. The center pane displays the following T-SQL script:

```

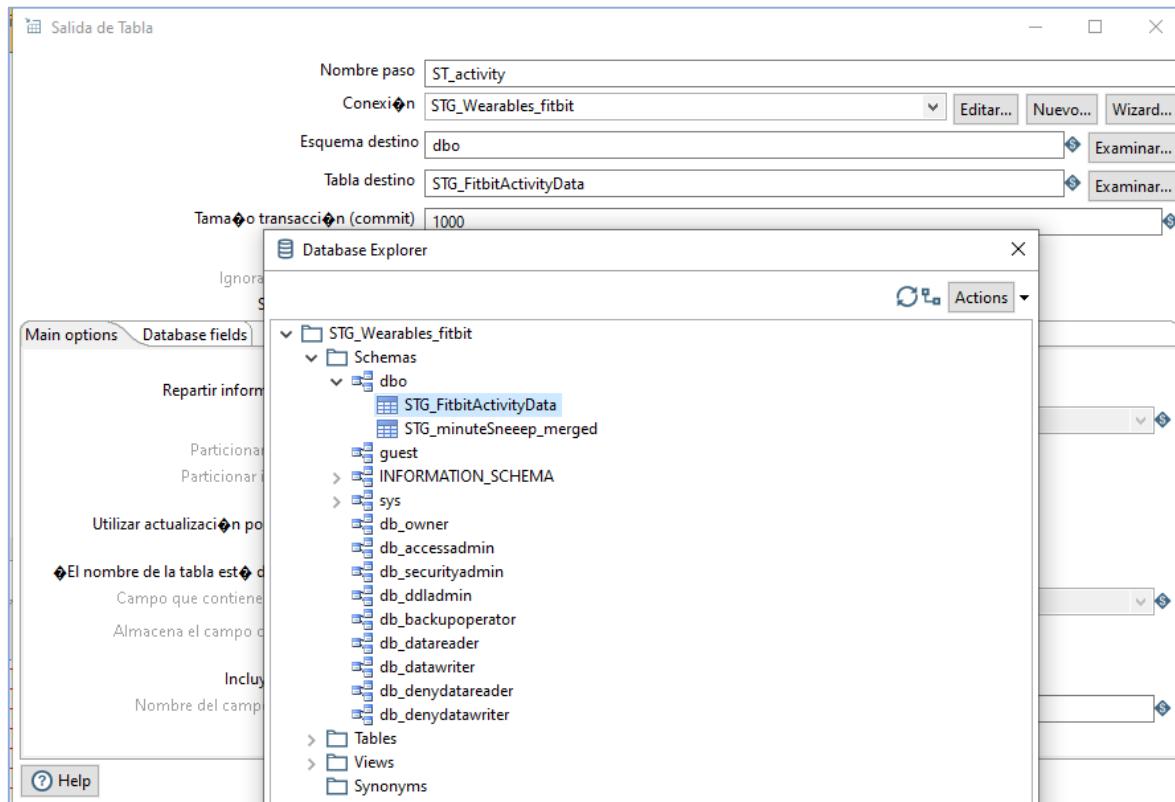
CREATE DATABASE STG_Wearables_fitbit;
-- Paso 2: Seleccionar la base de datos para usarla
USE STG_Wearables_fitbit;

-- Paso 3: Crear la tabla minuteSneep_merged
CREATE TABLE STG_minuteSneep_merged (
    Id BIGINT,
    date DATETIME,
    value INT,
    logId BIGINT
);

-- Paso 4: Crear la tabla FitbitActivityData
CREATE TABLE STG_FitbitActivityData (
    Id BIGINT,
    ActivityDate DATE,
    TotalSteps INT,
    TotalDistance DECIMAL(10, 5),
    TrackerDistance DECIMAL(10, 5),
    LoggedActivitiesDistance DECIMAL(10, 5),
    VeryActiveDistance DECIMAL(10, 5),
    ModeratelyActiveDistance DECIMAL(10, 5),
    LightActiveDistance DECIMAL(10, 5),
    SedentaryActiveDistance DECIMAL(10, 5),
    VeryActiveMinutes INT,
    FairlyActiveMinutes INT,
    LightlyActiveMinutes INT,
    SedentaryMinutes INT,
    Calories INT
);

```

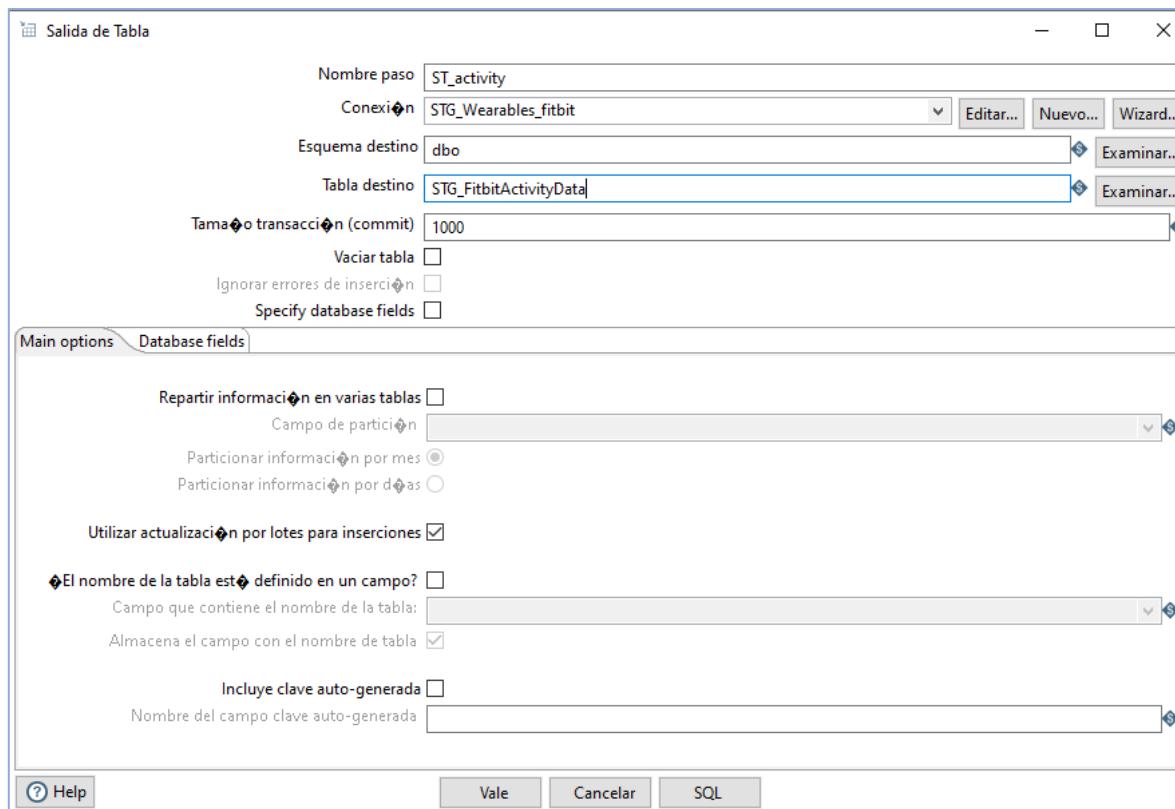
Configuración de la salida de la tabla



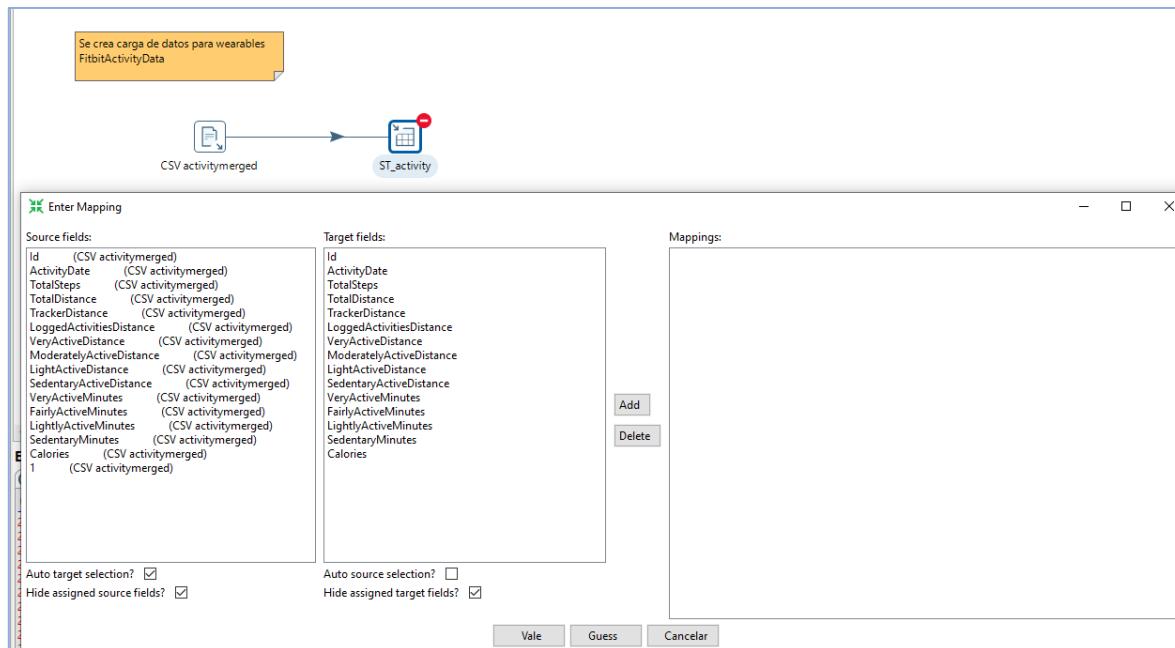
The screenshot shows the **Salida de Tabla** (Output) dialog box open in the foreground. The configuration fields are as follows:

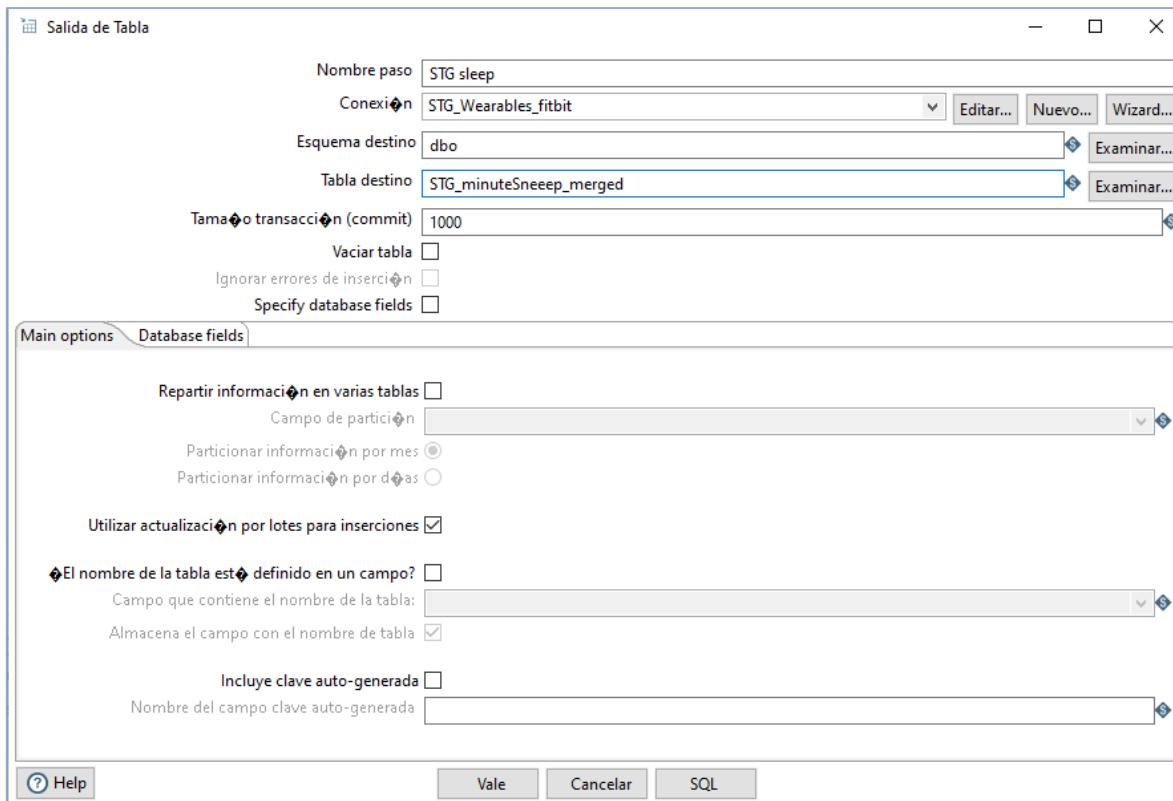
- Nombre paso:** ST_activity
- Conexión:** STG_Wearables_fitbit
- Esquema destino:** dbo
- Tabla destino:** STG_FitbitActivityData
- Tamaño de transacción (commit):** 1000

In the background, the **Database Explorer** window is visible, showing the structure of the **STG_Wearables_fitbit** database, including the **dbo** schema which contains the **STG_FitbitActivityData** and **STG_minuteSneep_merged** tables.

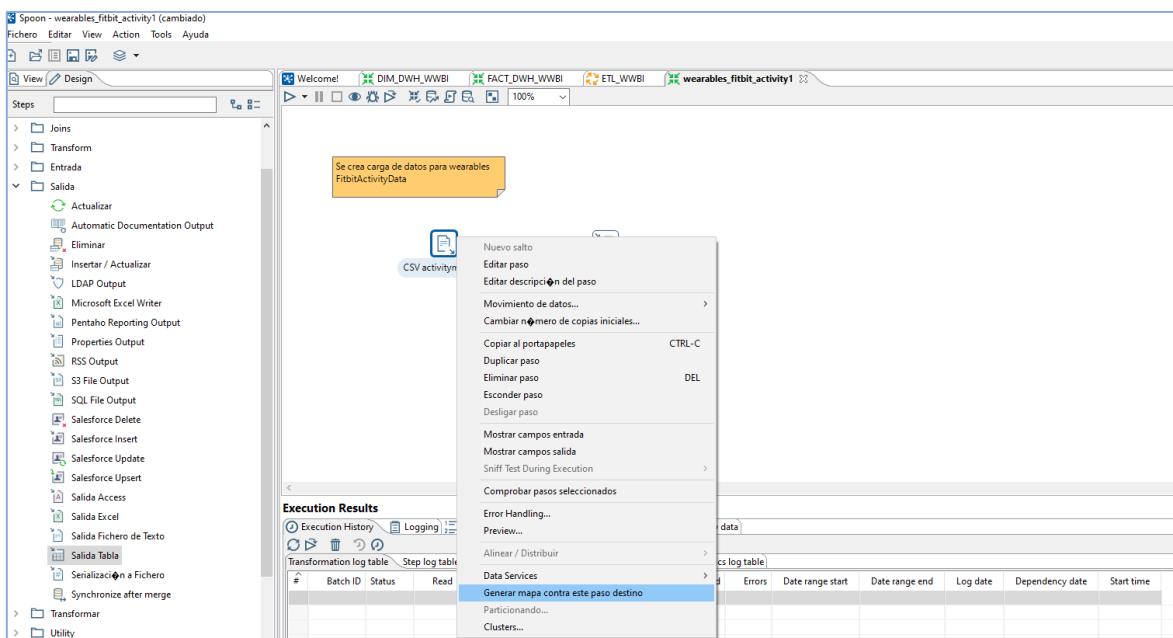


Mapeo de la salida

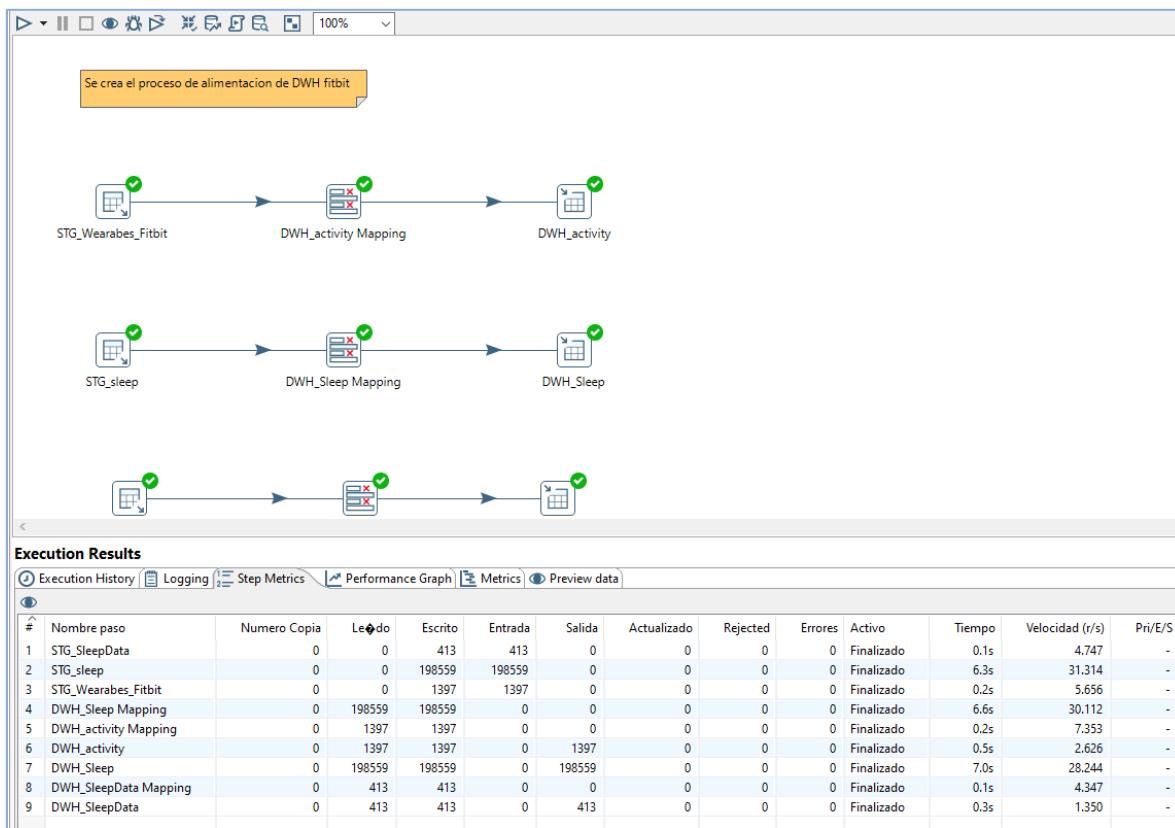




Revisar el mapeo entre la carga del archivo y la tabla de salida



Primer proceso



Carga de datos en SQL

Object Explorer

```
SELECT * FROM [STG_Wearables_Fitbit] [dbo].[STG_FitbitActivityData]
```

Results

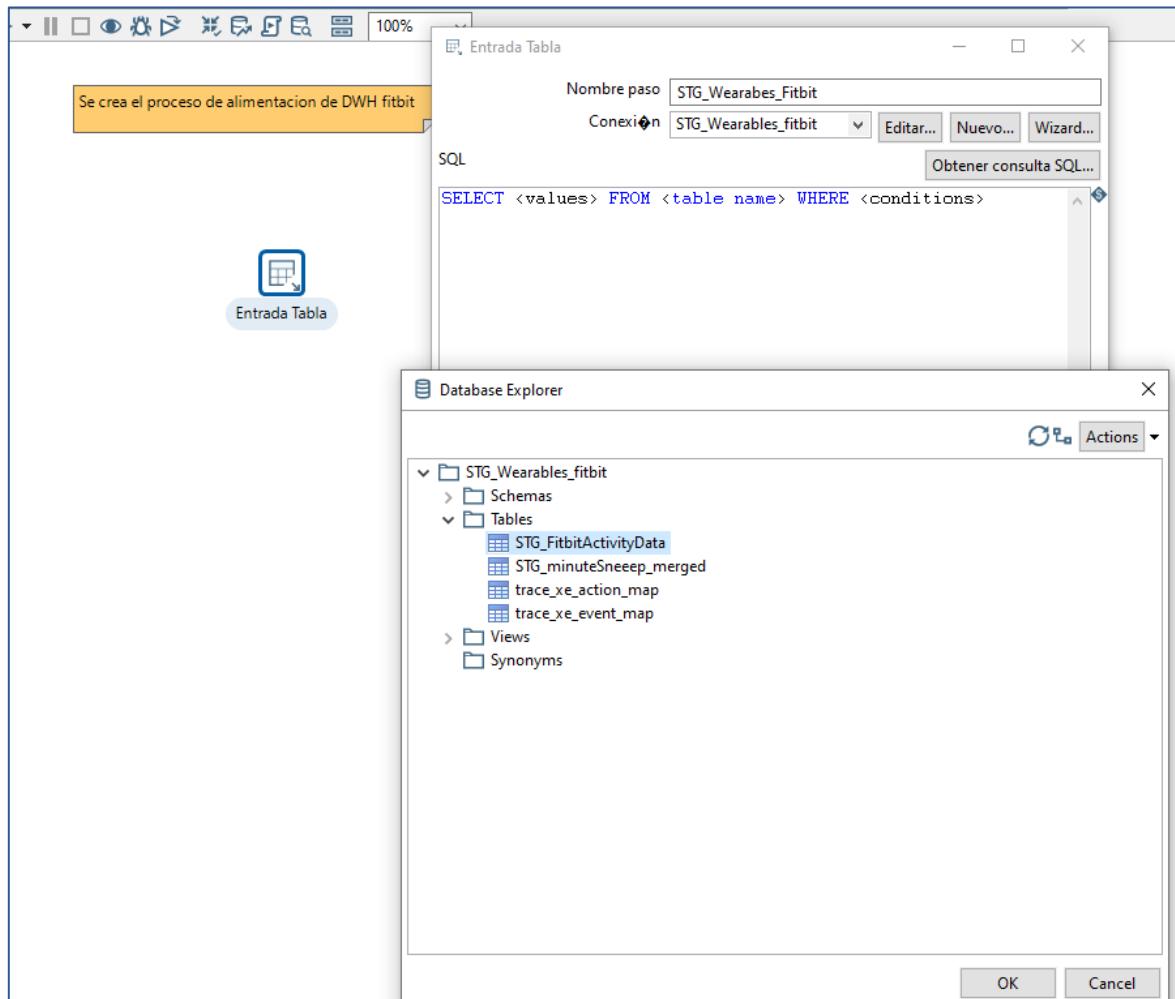
U	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LapsedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedentaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedentaryMinutes	
172	162450001	2016-04-08	6344	4.12000	4.12000	0.00000	0.00000	4.10000	0.00000	0	0	143	1297	
173	162450001	2016-04-09	3572	2.32000	2.32000	0.00000	0.00000	2.29000	0.00000	0	0	122	1318	
174	162450001	2016-04-10	3910	2.54000	2.54000	0.00000	0.47000	1.81000	0.01000	7	7	186	1240	
175	162450001	2016-04-11	10000	6.50000	6.50000	0.00000	0.00000	6.50000	0.00000	0	0	172	1268	
176	162450001	2016-04-12	4627	4.31000	4.31000	0.00000	0.00000	4.31000	0.00000	0	0	89	604	
177	164400001	2016-04-13	4548	3.44000	3.44000	0.00000	0.00000	2.88000	0.00000	0	16	586	1587	
178	164400001	2016-04-13	2037	14.71000	14.71000	0.00000	2.37000	6.73000	0.01000	34	141	347	918	
179	164400001	2016-04-13	12912	9.41000	9.41000	0.00000	4.83000	1.00000	3.79000	0.02000	59	16	283	1092
180	164400001	2016-04-14	2819	2.05000	2.05000	0.00000	0.31000	1.56000	0.01000	4	4	87	1345	
181	164400001	2016-04-15	921	7.21000	7.21000	0.00000	0.34000	5.36000	0.00000	5	35	219	1181	
182	164400001	2016-04-16	8046	5.85000	5.85000	0.00000	2.00000	2.45000	0.00000	18	41	137	1244	
183	164400001	2016-04-17	7248	8.12000	8.12000	0.00000	0.21000	5.98000	0.00000	3	50	230	1492	
184	164400001	2016-04-18	7942	5.70000	5.70000	0.00000	0.48000	2.21000	0.00000	7	43	161	1229	
185	164400001	2016-04-19	13940	10.06000	10.06000	0.00000	1.31000	4.00000	5.03000	0.00000	18	81	192	1149
186	164400001	2016-04-10	1329	0.97000	0.97000	0.00000	0.00000	0.97000	0.00000	0	0	35	207	
187	184450072	2016-04-01	6847	4.53000	4.53000	0.00000	0.61000	0.00000	3.55000	0.00000	9	9	251	1171
188	184450072	2016-04-02	5367	3.55000	3.55000	0.00000	0.00000	3.55000	0.00000	0	0	263	1177	
189	184450072	2016-04-03	2841	1.88000	1.88000	0.00000	0.00000	1.88000	0.00000	0	0	136	1114	
190	184450072	2016-04-04	8	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0	0	0	669	

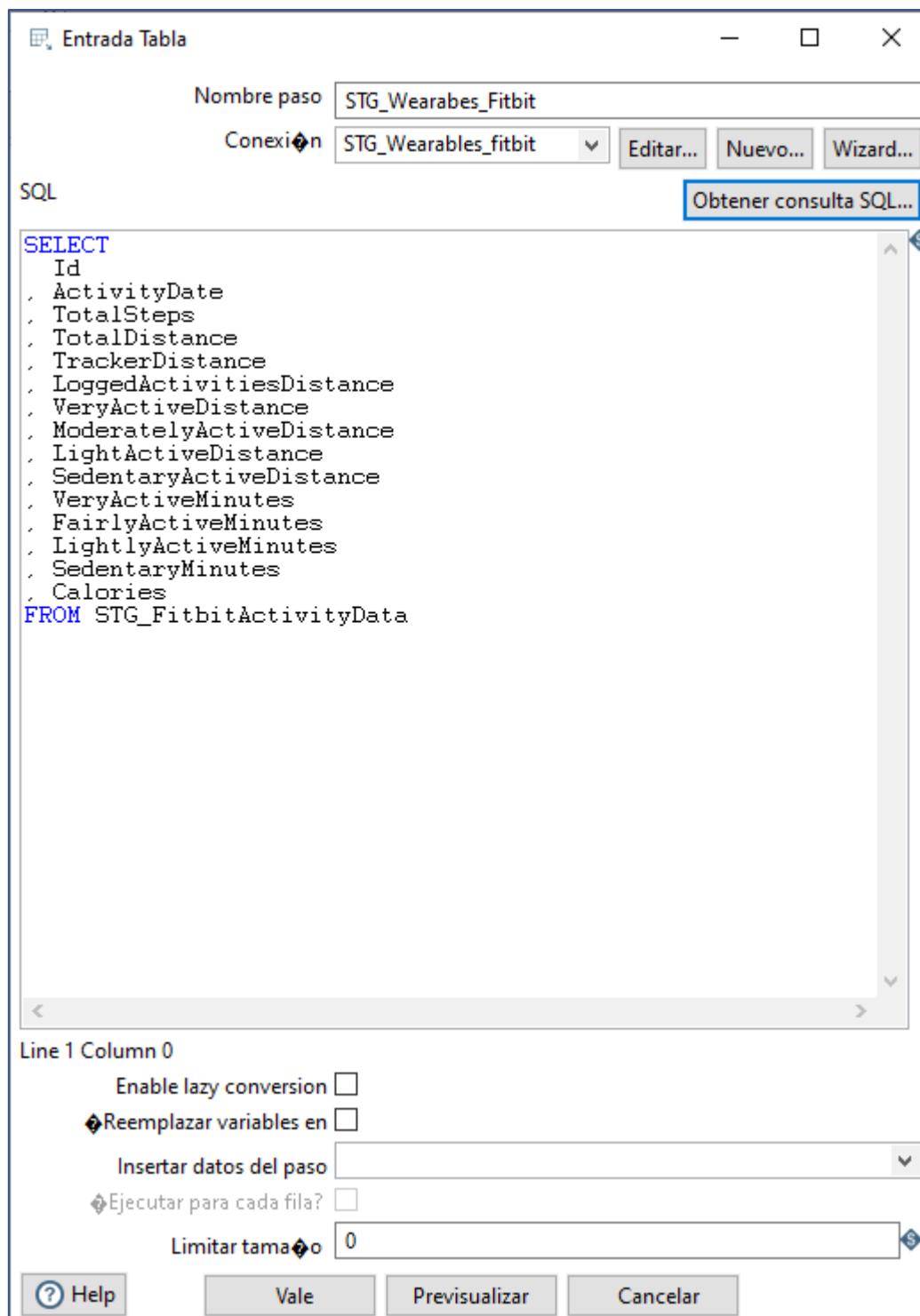
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer pane displays a tree view of database objects for the 'DESKTOP-4ESN00D\SQLEXPRESS' instance, including databases like 'System Databases', 'DWH_WWBI', 'Hoteles_db', 'Northwind', 'Northwind_DW', and 'STG_Wearables_fitbit'. The 'STG_Wearables_fitbit' database is expanded to show its schema, including tables like 'STG_FitbitActivityData' and 'STG_minuteSneeeep_merged'. On the right, the main workspace contains two panes: a 'SQLQuery2.sql - DES...es_fitbit (IMF (58))' pane at the top with a script for 'SelectTopNRows' command from SSMS, and a 'Results' pane below it displaying a table of data. The table has columns: Id, date, value, and logId. The data consists of 88 rows, each with an 'Id' of 1503960366, a 'date' of 2016-03-13 00:00:00.000, a 'value' of 1, and a 'logId' of 11114919637.

	Id	date	value	logId
61	1503960366	2016-03-13 00:00:00.000	1	11114919637
62	1503960366	2016-03-13 00:00:00.000	1	11114919637
63	1503960366	2016-03-13 00:00:00.000	1	11114919637
64	1503960366	2016-03-13 00:00:00.000	1	11114919637
65	1503960366	2016-03-13 00:00:00.000	1	11114919637
66	1503960366	2016-03-13 00:00:00.000	1	11114919637
67	1503960366	2016-03-13 00:00:00.000	1	11114919637
68	1503960366	2016-03-13 00:00:00.000	1	11114919637
69	1503960366	2016-03-13 00:00:00.000	1	11114919637
70	1503960366	2016-03-13 00:00:00.000	1	11114919637
71	1503960366	2016-03-13 00:00:00.000	1	11114919637
72	1503960366	2016-03-13 00:00:00.000	1	11114919637
73	1503960366	2016-03-13 00:00:00.000	1	11114919637
74	1503960366	2016-03-13 00:00:00.000	1	11114919637
75	1503960366	2016-03-13 00:00:00.000	1	11114919637
76	1503960366	2016-03-13 00:00:00.000	1	11114919637
77	1503960366	2016-03-13 00:00:00.000	1	11114919637
78	1503960366	2016-03-13 00:00:00.000	1	11114919637
79	1503960366	2016-03-13 00:00:00.000	1	11114919637
80	1503960366	2016-03-13 00:00:00.000	1	11114919637
81	1503960366	2016-03-13 00:00:00.000	1	11114919637
82	1503960366	2016-03-13 00:00:00.000	1	11114919637
83	1503960366	2016-03-13 00:00:00.000	1	11114919637
84	1503960366	2016-03-13 00:00:00.000	1	11114919637
85	1503960366	2016-03-13 00:00:00.000	1	11114919637
86	1503960366	2016-03-13 00:00:00.000	1	11114919637
87	1503960366	2016-03-13 00:00:00.000	1	11114919637
88	1503960366	2016-03-13 00:00:00.000	1	11114919637

Creación de DWH

Conexiones entre las tablas de Sting y las de Data Ware House





Previsualización de los datos

Spoon - DWH_wearables_fitbit (cambiado)

Archivo Herramientas Entrada Dispositivos Ayuda

Excepciones Editor View Action Tools Ayuda

View Design

Rows of step: STG_Wearables_Fitbit (552 rows)

#	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedentaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedentaryMinutes
1	150396366	2016/03/29 00:00:00.000	11004	7,0	0,0	0,0	0,0	0,0	4,0	0,0	33	12	205	804
2	150396366	2016/03/29 00:00:00.000	17659	12,0	0,0	0,0	0,0	0,0	4,0	0,0	99	17	274	369
3	150396366	2016/03/28 00:00:00.000	12738	8,0	0,0	0,0	0,0	0,0	4,0	0,0	54	5	230	405
4	150396366	2016/03/28 00:00:00.000	13231	9,0	0,0	0,0	0,0	1,0	5,0	0,0	39	20	224	1080
5	150396366	2016/03/29 00:00:00.000	12041	8,0	0,0	0,0	2,0	1,0	5,0	0,0	28	28	243	763
6	150396366	2016/03/30 00:00:00.000	10970	7,0	0,0	0,0	2,0	1,0	4,0	0,0	30	13	223	1174
7	150396366	2016/03/31 00:00:00.000	12256	8,0	0,0	0,0	0,0	0,0	5,0	0,0	33	12	239	820
8	150396366	2016/04/01 00:00:00.000	12252	8,0	0,0	0,0	0,0	0,0	4,0	0,0	47	21	230	669
9	150396366	2016/04/02 00:00:00.000	11248	7,0	0,0	0,0	3,0	0,0	4,0	0,0	40	11	244	636
10	150396366	2016/04/03 00:00:00.000	10016	6,0	0,0	0,0	1,0	1,0	4,0	0,0	15	30	314	655
11	150396366	2016/04/04 00:00:00.000	14557	10,0	0,0	0,0	3,0	1,0	6,0	0,0	43	18	285	757
12	150396366	2016/04/05 00:00:00.000	14844	10,0	0,0	0,0	3,0	1,0	6,0	0,0	36	18	341	736
13	150396366	2016/04/06 00:00:00.000	11974	8,0	0,0	0,0	2,0	0,0	5,0	0,0	27	12	228	1173
14	150396366	2016/04/07 00:00:00.000	10179	6,0	0,0	0,0	0,0	0,0	4,0	0,0	17	20	199	1028
15	150396366	2016/04/08 00:00:00.000	12321	8,0	0,0	0,0	3,0	1,0	4,0	0,0	46	22	212	1160
16	150396366	2016/04/09 00:00:00.000	13432	8,0	0,0	0,0	3,0	1,0	5,0	0,0	32	15	240	738
17	150396366	2016/04/10 00:00:00.000	10057	7,0	0,0	0,0	4,0	0,0	2,0	0,0	44	13	168	737
18	150396366	2016/04/11 00:00:00.000	10990	7,0	0,0	0,0	2,0	1,0	5,0	0,0	26	14	216	855
19	150396366	2016/04/12 00:00:00.000	224	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0	0	9	32
20	150396366	2016/04/13 00:00:00.000	1110	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0	0	121	191
21	150396366	2016/04/14 00:00:00.000	815	1,0	0,0	0,0	0,0	0,0	1,0	0,0	0	0	47	1393
22	162458001	2016/03/27 00:00:00.000	1985	1,0	1,0	0,0	0,0	0,0	1,0	0,0	0	0	112	1328
23	162458001	2016/03/28 00:00:00.000	1905	1,0	1,0	0,0	0,0	0,0	0,0	0,0	0	0	95	1345
24	162458001	2016/03/29 00:00:00.000	1552	1,0	1,0	0,0	0,0	0,0	0,0	0,0	0	0	66	1374
25	162458001	2016/03/30 00:00:00.000	1675	1,0	0,0	0,0	0,0	0,0	0,0	0,0	0	0	84	1584
26	162458001	2016/03/31 00:00:00.000	4266	3,0	3,0	0,0	0,0	0,0	2,0	0,0	7	4	144	1265
27	162458001	2016/04/01 00:00:00.000	9218	4,0	0,0	0,0	0,0	0,0	4,0	0,0	0	0	221	1219
28	162458001	2016/04/02 00:00:00.000	1556	1,0	1,0	0,0	0,0	0,0	1,0	0,0	0	0	88	1352
29	162458001	2016/04/03 00:00:00.000	2010	2,0	0,0	0,0	0,0	0,0	2,0	0,0	0	0	157	1283
30	162458001	2016/04/04 00:00:00.000	18464	12,0	0,0	0,0	0,0	0,0	12,0	0,0	0	0	270	1170
31	162458001	2016/04/05 00:00:00.000	1335	1,0	1,0	0,0	0,0	0,0	1,0	0,0	0	0	74	1366
32	162458001	2016/04/06 00:00:00.000	1004	1,0	1,0	0,0	0,0	0,0	0,0	0,0	0	0	55	1383

Cerrar Show Log

Se crea el proceso de alimentación de DWH fitbit

Salida de Tabla

Nombre paso: DWH_activity

Conexión: DWH_Wearables_Fitbit_IoT

Tamaño transacción (commit): 1000

Database Explorer

Main options Database fields

Repartir informe

Utilizar actualización por defecto

El nombre de la tabla es:

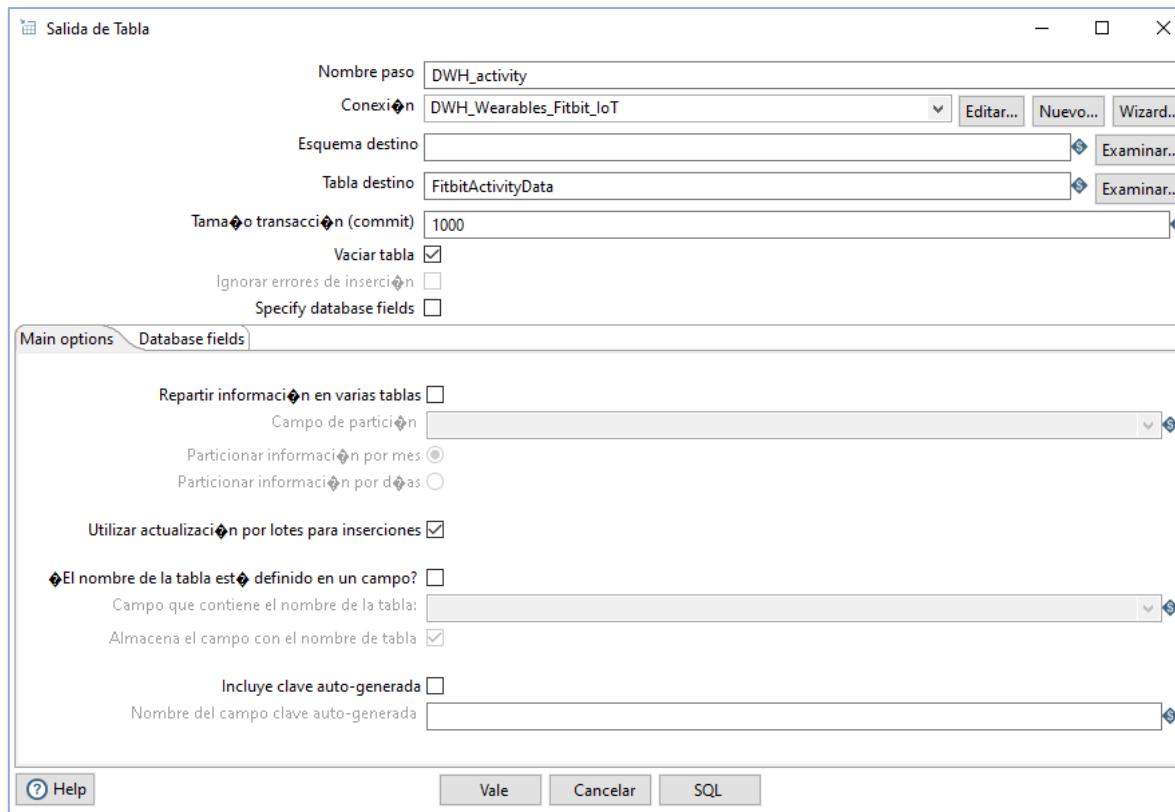
Incluir

OK Cancel

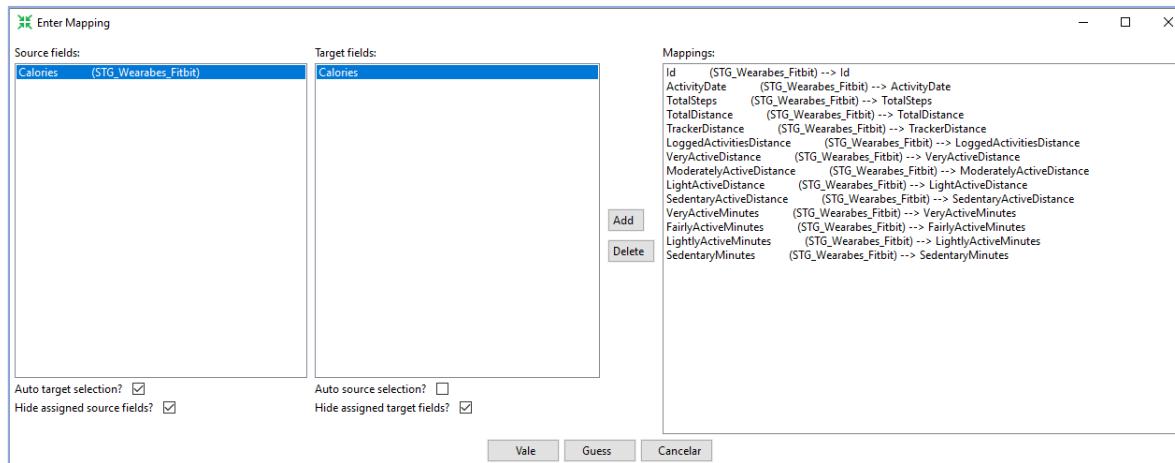
DWH_Wearables_Fitbit_IoT

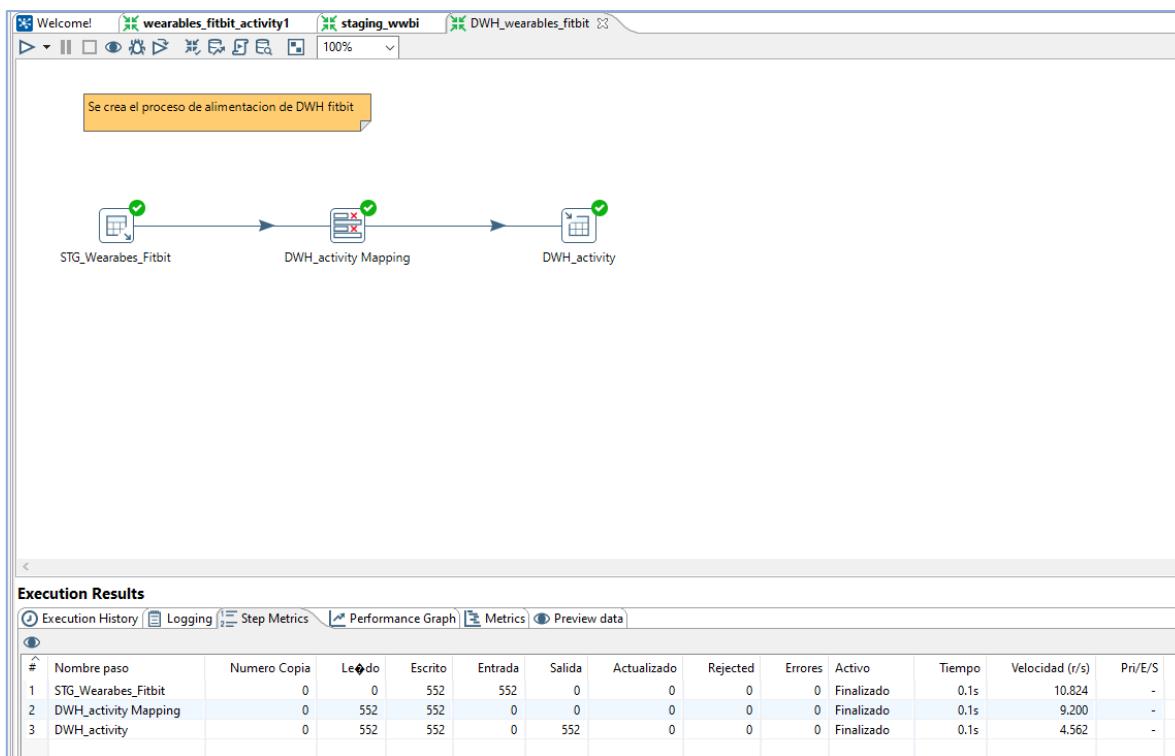
Tables

- FitbitActivityData
- minuteSneeeep_merged
- trace_xe_action_map
- trace_xe_event_map



Esta captura muestra como se crea un mapping entre las dos tablas staging a ware house

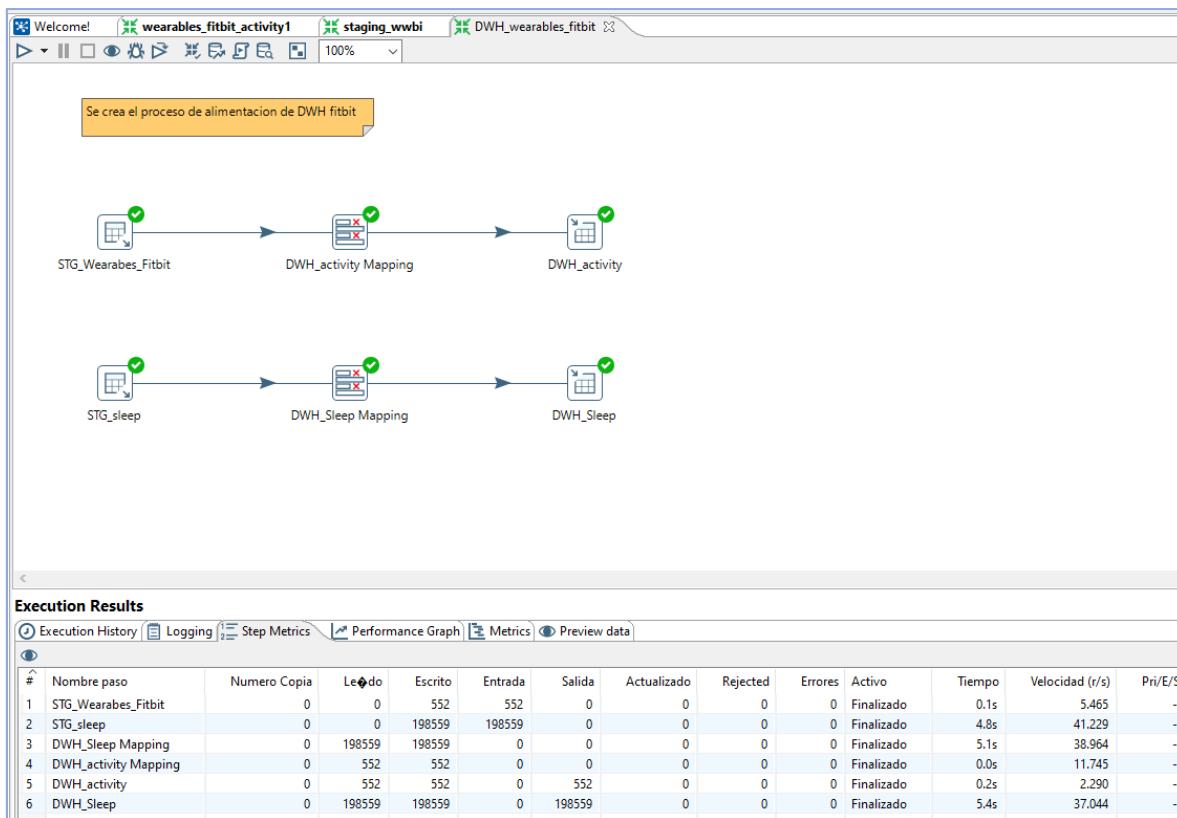




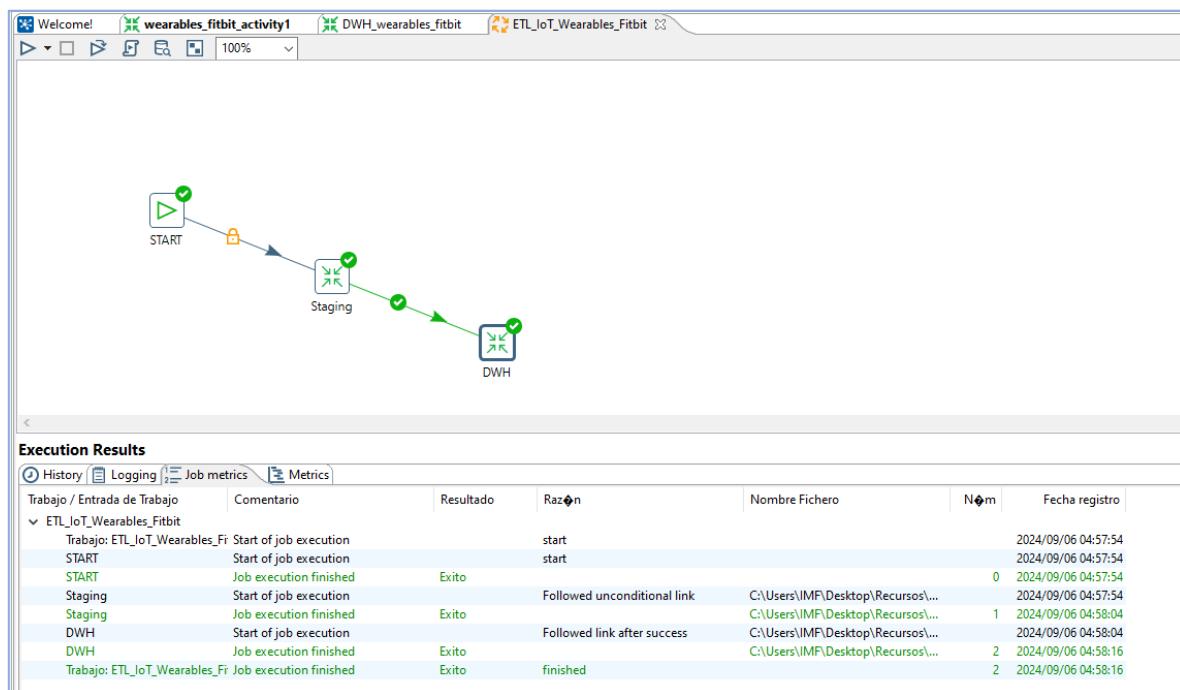
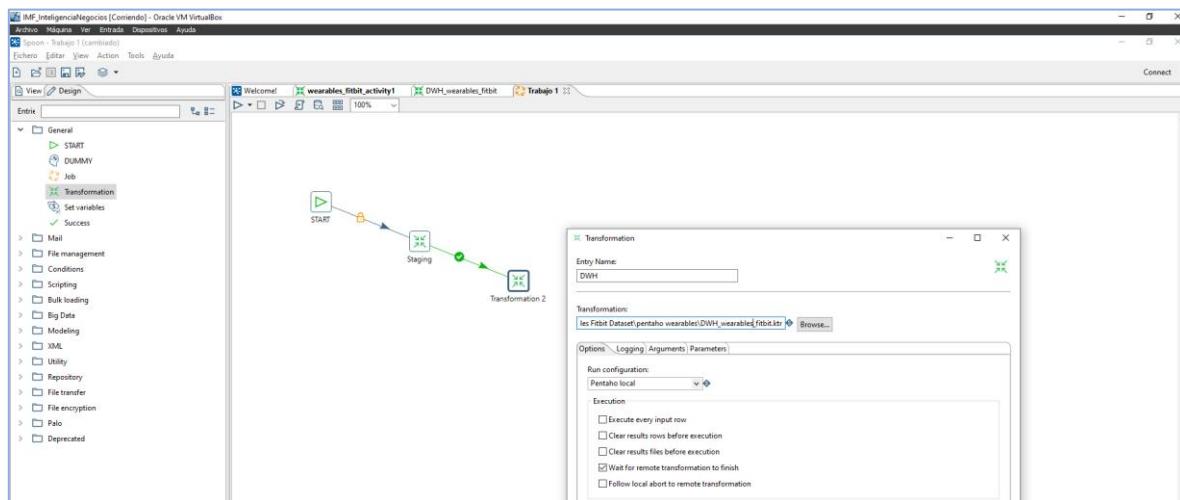
The dialog box is titled "Salida de Tabla" and contains the following settings:

- Nombre paso:** DWH_Sleep
- Conexión:** DWH_Wearables_Fitbit_IoT
- Esquema destino:** (empty)
- Tabla destino:** minuteSleep_merged
- Tamaño transacción (commit):** 1000
- Opciones:**
 - Vaciar tabla:
 - Ignorar errores de inserción:
 - Specify database fields:
- Main options:**
 - Repartir información en varias tablas:
 - Campo de partición: (dropdown menu)
 - Partitionar información por mes:
 - Partitionar información por días:
 - Utilizar actualización por lotes para inserciones:
 - El nombre de la tabla está definido en un campo?
 - Campo que contiene el nombre de la tabla: (dropdown menu)
 - Almacena el campo con el nombre de tabla:
 - Incluye clave auto-generada:
 - Nombre del campo clave auto-generada: (dropdown menu)

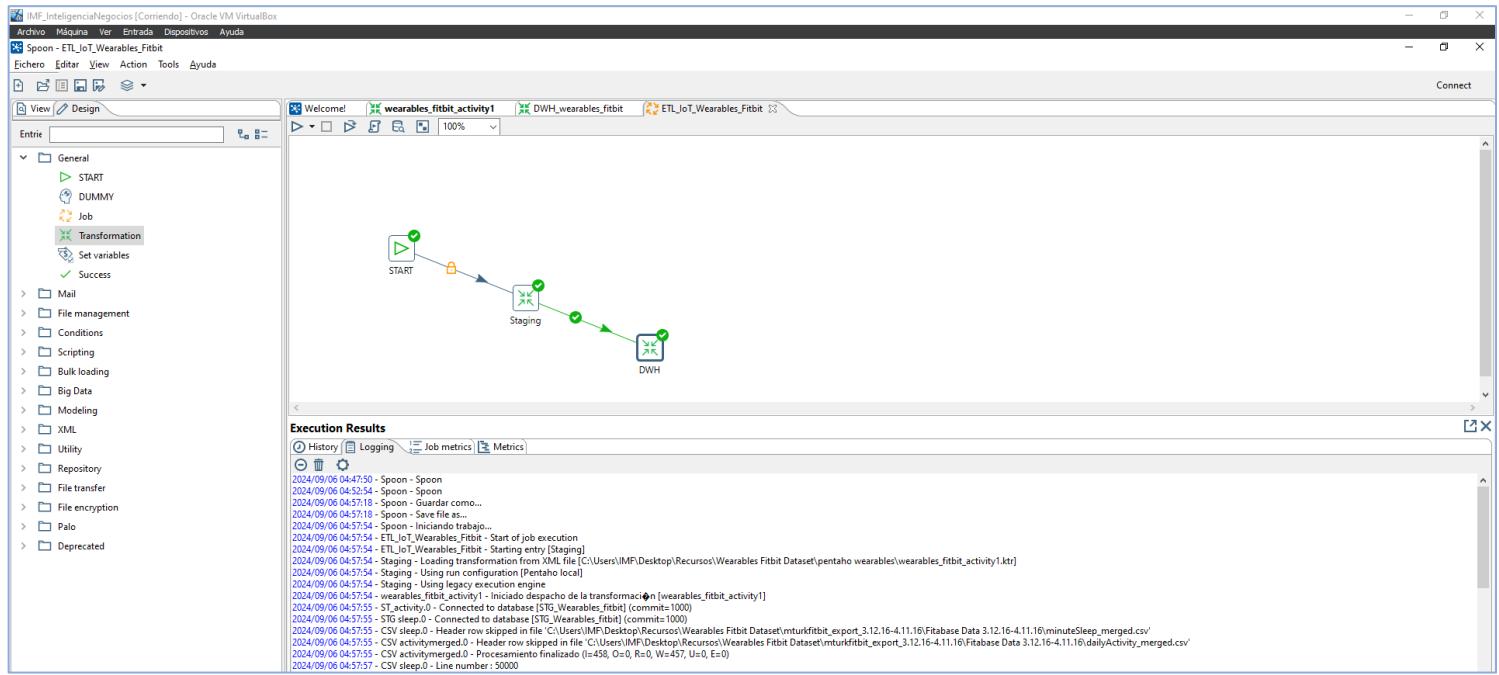
At the bottom are buttons: ? Help, Vale, Cancelar, and SQL.



Creación del Job – ETL Pipeline

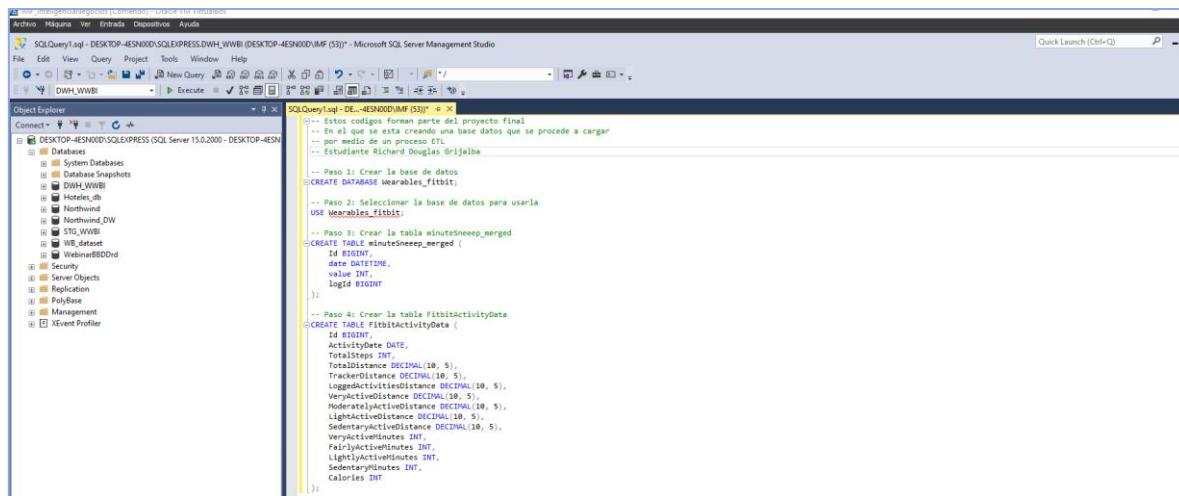


Esta captura muestra como el Pipeline o Job de datos, comienza a trabajar de forma que los loggings dan un check verde y además no existe ningún mensaje de error.



Anexo 2 MS SQL Server

Creación de la base de datos



```

--> Este código forma parte del proyecto final
--> En el que se está creando una base de datos que se procede a cargar
--> por medio de un proceso ETL
--> Estudiante Richard Douglas Grimalba

-- Paso 1: Crear la base de datos
CREATE DATABASE Wearables_fitbit;

-- Paso 2: Seleccionar la base de datos para usarla
USE Wearables_fitbit;

-- Paso 3: Crear la tabla minuteSleep_merged
CREATE TABLE minuteSleep_merged (
    Id BIGINT,
    date DATETIME,
    value INT,
    logId BIGINT
);

-- Paso 4: Crear la tabla FitbitActivityData
CREATE TABLE FitbitActivityData (
    Id BIGINT,
    ActivityDate DATE,
    TotalSteps INT,
    TotalDistance DECIMAL(10, 5),
    TrackerDistance DECIMAL(10, 5),
    LoggedActivitiesDistance DECIMAL(10, 5),
    VeryActiveDistance DECIMAL(10, 5),
    ModeratelyActiveDistance DECIMAL(10, 5),
    LightActiveDistance DECIMAL(10, 5),
    SedentaryActiveDistance DECIMAL(10, 5),
    VeryActiveMinutes INT,
    FairlyActiveMinutes INT,
    LightlyActiveMinutes INT,
    SedentaryMinutes INT,
    Calories INT
);

```

-- Paso 1: Crear la base de datos

```
CREATE DATABASE Wearables_fitbit;
```

-- Paso 2: Seleccionar la base de datos para usarla

```
USE Wearables_fitbit;
```

-- Paso 3: Crear la tabla minuteSleep_merged

```
CREATE TABLE minuteSleep_merged (
```

```
    Id BIGINT,  
    date DATETIME,  
    value INT,  
    logId BIGINT  
);
```

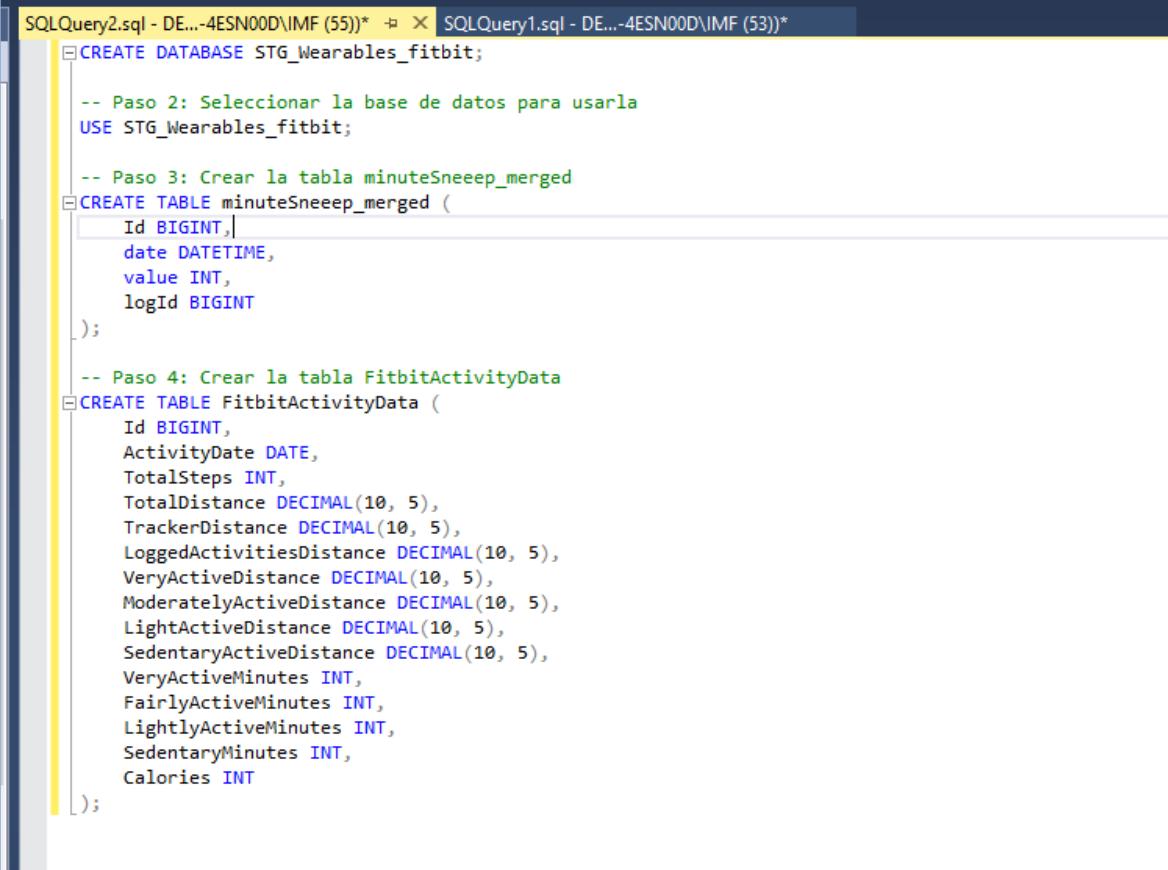
-- Paso 4: Crear la tabla FitbitActivityData

```
CREATE TABLE FitbitActivityData (
```

```
    Id BIGINT,  
    ActivityDate DATE,  
    TotalSteps INT,  
    TotalDistance DECIMAL(10, 5),  
    TrackerDistance DECIMAL(10, 5),  
    LoggedActivitiesDistance DECIMAL(10, 5),  
    VeryActiveDistance DECIMAL(10, 5),  
    ModeratelyActiveDistance DECIMAL(10, 5),  
    LightActiveDistance DECIMAL(10, 5),  
    SedentaryActiveDistance DECIMAL(10, 5),  
    VeryActiveMinutes INT,  
    FairlyActiveMinutes INT,  
    LightlyActiveMinutes INT,  
    SedentaryMinutes INT,  
    Calories INT
```

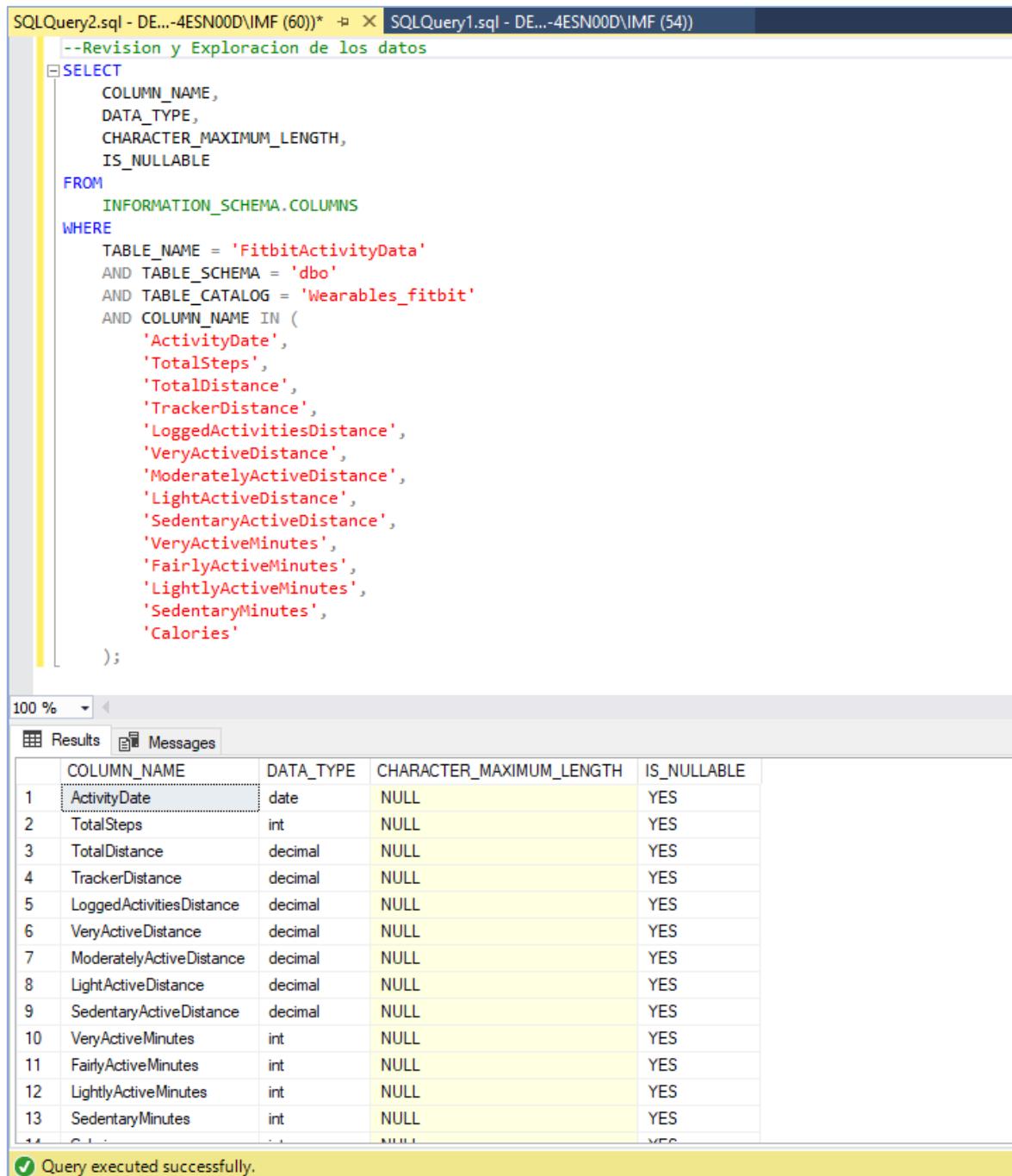
```
LightlyActiveMinutes INT,  
SedentaryMinutes INT,  
Calories INT  
);
```

Crear la base de datos staging



```
SQLQuery2.sql - DE...-4ESN00D\IMF (55)*  X  SQLQuery1.sql - DE...-4ESN00D\IMF (53)*  
CREATE DATABASE STG_Wearables_fitbit;  
  
-- Paso 2: Seleccionar la base de datos para usarla  
USE STG_Wearables_fitbit;  
  
-- Paso 3: Crear la tabla minuteSleep_merged  
CREATE TABLE minuteSleep_merged (  
    Id BIGINT,  
    date DATETIME,  
    value INT,  
    logId BIGINT  
);  
  
-- Paso 4: Crear la tabla FitbitActivityData  
CREATE TABLE FitbitActivityData (  
    Id BIGINT,  
    ActivityDate DATE,  
    TotalSteps INT,  
    TotalDistance DECIMAL(10, 5),  
    TrackerDistance DECIMAL(10, 5),  
    LoggedActivitiesDistance DECIMAL(10, 5),  
    VeryActiveDistance DECIMAL(10, 5),  
    ModeratelyActiveDistance DECIMAL(10, 5),  
    LightActiveDistance DECIMAL(10, 5),  
    SedentaryActiveDistance DECIMAL(10, 5),  
    VeryActiveMinutes INT,  
    FairlyActiveMinutes INT,  
    LightlyActiveMinutes INT,  
    SedentaryMinutes INT,  
    Calories INT  
);
```

Consulta y exploración de los datos



The screenshot shows a SQL Server Management Studio (SSMS) window. The top tab bar has two tabs: 'SQLQuery2.sql - DE...-4ESN00D\IMF (60)*' and 'SQLQuery1.sql - DE...-4ESN00D\IMF (54)'. The main area contains a SQL query titled '-Revisión y Exploración de los datos'. The query selects columns from the 'INFORMATION_SCHEMA.COLUMNS' table where the table name is 'FitbitActivityData', schema is 'dbo', catalog is 'Wearables_fitbit', and column names are listed in a subquery. The results grid below shows 13 columns with the following data:

	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH	IS_NULLABLE
1	ActivityDate	date	NULL	YES
2	TotalSteps	int	NULL	YES
3	TotalDistance	decimal	NULL	YES
4	TrackerDistance	decimal	NULL	YES
5	LoggedActivitiesDistance	decimal	NULL	YES
6	VeryActiveDistance	decimal	NULL	YES
7	ModeratelyActiveDistance	decimal	NULL	YES
8	LightActiveDistance	decimal	NULL	YES
9	SedentaryActiveDistance	decimal	NULL	YES
10	VeryActiveMinutes	int	NULL	YES
11	FairlyActiveMinutes	int	NULL	YES
12	LightlyActiveMinutes	int	NULL	YES
13	SedentaryMinutes	int	NULL	YES

At the bottom of the results grid, a message indicates: 'Query executed successfully.'

```
--Revisión y Exploración de los datos
SELECT
    COLUMN_NAME,
    DATA_TYPE,
    CHARACTER_MAXIMUM_LENGTH,
    IS_NULLABLE
FROM
    INFORMATION_SCHEMA.COLUMNS
WHERE
    TABLE_NAME = 'FitbitActivityData'
    AND TABLE_SCHEMA = 'dbo'
    AND TABLE_CATALOG = 'Wearables_fitbit'
    AND COLUMN_NAME IN (
        'ActivityDate',
        'TotalSteps',
        'TotalDistance',
        'TrackerDistance',
        'LoggedActivitiesDistance',
        'VeryActiveDistance',
        'ModeratelyActiveDistance',
        'LightActiveDistance',
        'SedentaryActiveDistance',
        'VeryActiveMinutes',
        'FairlyActiveMinutes',
        'LightlyActiveMinutes',
        'SedentaryMinutes',
        'Calories'
    );

```

Exploración de la característica Steps

The screenshot shows a SQL Server Management Studio window with three tabs at the top: 'SQLQuery4.sql - DE...-4ESN00D\IMF (64)*' (selected), 'SQLQuery3.sql - DE...-4ESN00D\IMF (59)*', and 'SQLQuery2.sql - DE...-4ESN00D\IMF (60)*'. The main pane displays a SQL query to calculate statistics for the 'FitbitActivityData' table:

```
--Conocer Informacion relevante sobre el dataset
-- Por ejemplo las caracterista de Steps
SELECT
    COUNT(*) AS total_registros,
    AVG(TotalSteps) AS promedio_pasos,
    MIN(TotalSteps) AS minimo_pasos,
    MAX(TotalSteps) AS maximo_pasos
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The results pane shows the following data:

	total_registros	promedio_pasos	minimo_pasos	maximo_pasos
1	1397	7280	0	36019

Característica Calorías

The screenshot shows a SQL Server Management Studio window with three tabs at the top: 'SQLQuery4.sql - DE...-4ESN00D\IMF (64)*' (selected), 'SQLQuery3.sql - DE...-4ESN00D\IMF (59)*', and 'SQLQuery2.sql - DE...-4ESN00D\IMF (60)*'. The main pane displays a SQL query to calculate statistics for the 'FitbitActivityData' table:

```
--Conocer Informacion relevante sobre el dataset
-- Por ejemplo las caracterista de Calorias
SELECT
    COUNT(*) AS total_registros,
    AVG(Calories) AS promedio_calorias,
    MIN(Calories) AS minimo_calorias,
    MAX(Calories) AS maximo_calorias
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The results pane shows the following data:

	total_registros	promedio_calorias	minimo_calorias	maximo_calorias
1	1397	2266	0	4900

Distancia Total

SQLQuery5.sql - DE...-4ESN00D\IMF (53)* SQLQuery4.sql - DE...-4ESN00D\IMF (64)* SQL

```
--Revisión de la característica Distancia Total |
SELECT
    COUNT(*) AS total_registros,
    AVG(TotalDistance) AS promedio_distancia_total,
    MIN(TotalDistance) AS minimo_distancia_total,
    MAX(TotalDistance) AS maximo_distancia_total
FROM [Wearables_fitbit].[dbo].[FitbitActivityData]
```

100 %

	total_registros	promedio_distancia_total	minimo_distancia_total	maximo_distancia_total
1	1397	5.213314	0.00000	28.00000

IMF_InteligenciaNegocios [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

SQLQuery4.sql - DESKTOP-4ESN00D\SQLEXPRESS.Wearables_fitbit (DESKTOP-4ESN00D\IMF (54)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Wearables_fitbit | Execute | Results | Messages

Object Explorer

```
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [Id]
    ,[date]
    ,[value]
    ,[logId]
FROM [Wearables_fitbit].[dbo].[minuteSneep_merged]
```

	Id	date	value	logId
1	1503960366	2016-03-17 00:00:00.000	1	11150938241
2	1503960366	2016-03-17 00:00:00.000	1	11150938241
3	1503960366	2016-03-17 00:00:00.000	1	11150938241
4	1503960366	2016-03-17 00:00:00.000	1	11150938241
5	1503960366	2016-03-17 00:00:00.000	1	11150938241
6	1503960366	2016-03-17 00:00:00.000	1	11150938241
7	1503960366	2016-03-17 00:00:00.000	1	11150938241
8	1503960366	2016-03-17 00:00:00.000	1	11150938241
9	1503960366	2016-03-17 00:00:00.000	1	11150938241
10	1503960366	2016-03-17 00:00:00.000	1	11150938241
11	1503960366	2016-03-17 00:00:00.000	1	11150938241

Agregar información de valor a una columna : crear el día de la semana

The screenshot shows a SQL Server Management Studio window with two tabs: 'SQLQuery9.sql - DE...-4E5N00D\IMIF (00)' and 'SQLQuery8.sql - DE...-4E5N00D\IMIF (57)'. The 'SQLQuery8.sql' tab is active, displaying the following T-SQL code:

```
--Feature Engineering
--Crear una columna con el dia de la semana

--agregar una nueva columna
ALTER TABLE [Wearables_fitbit].[dbo].[FitbitActivityData]
ADD DayOfWeek VARCHAR(10);

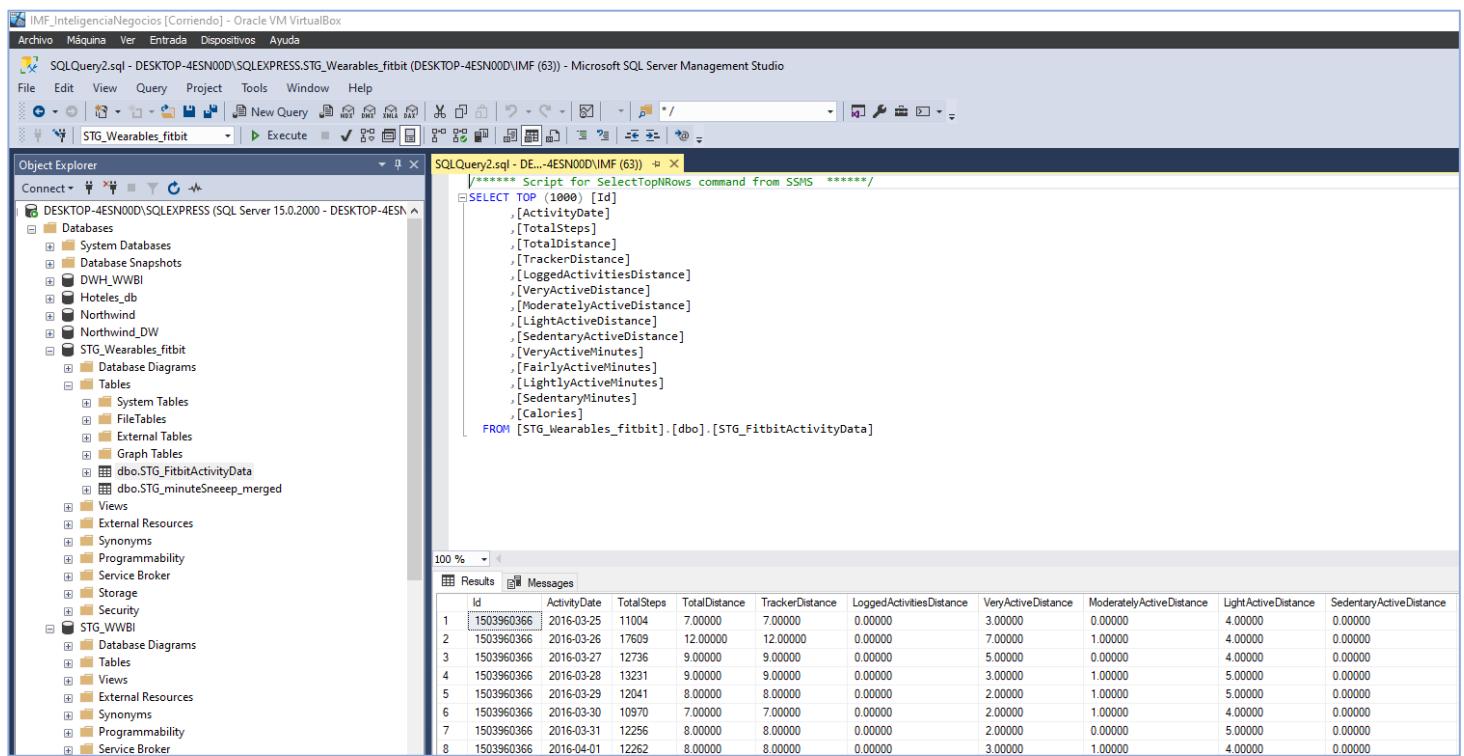
--agregar la inofrmacion del dia de la semana
UPDATE [Wearables_fitbit].[dbo].[FitbitActivityData]
SET DayOfWeek = DATENAME(weekday, [ActivityDate]);

--revision de la informacion cargada

SELECT TOP (10) [Id], [ActivityDate], DayOfWeek
FROM [Wearables_fitbit].[dbo].[FitbitActivityData];
```

The 'Results' tab shows the output of the last query, which is a table with 10 rows of data:

	Id	ActivityDate	DayOfWeek
1	1503960366	2016-03-25	Viernes
2	1503960366	2016-03-26	Sábado
3	1503960366	2016-03-27	Domingo
4	1503960366	2016-03-28	Lunes
5	1503960366	2016-03-29	Martes
6	1503960366	2016-03-30	Miércoles
7	1503960366	2016-03-31	Jueves
8	1503960366	2016-04-01	Viernes
9	1503960366	2016-04-02	Sábado
10	1503960366	2016-04-03	Domingo



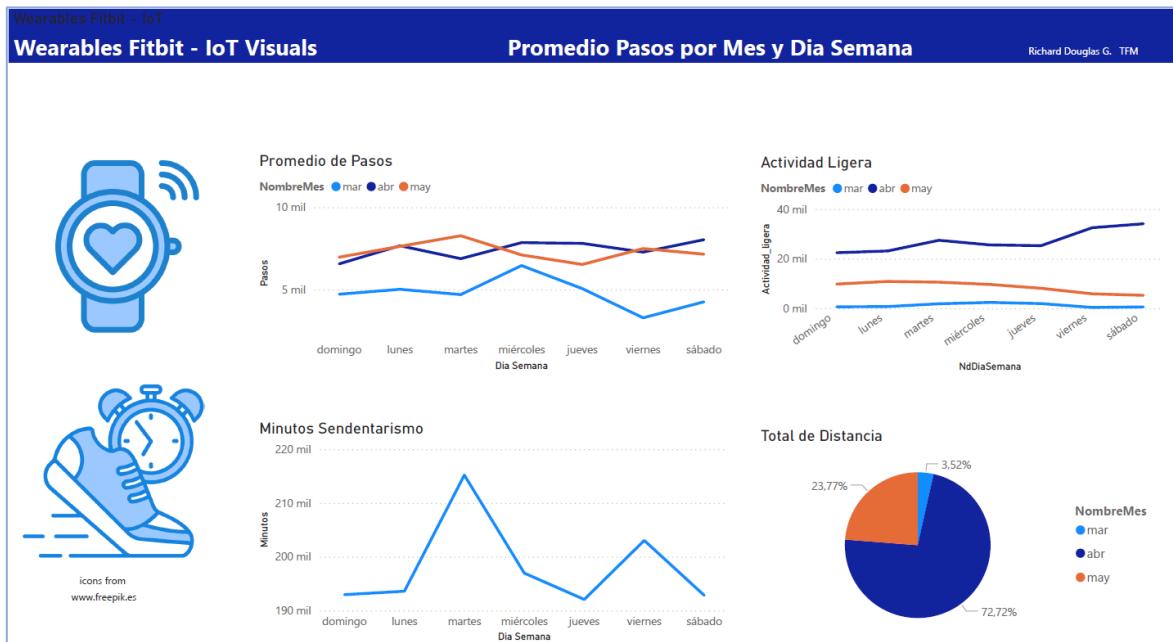
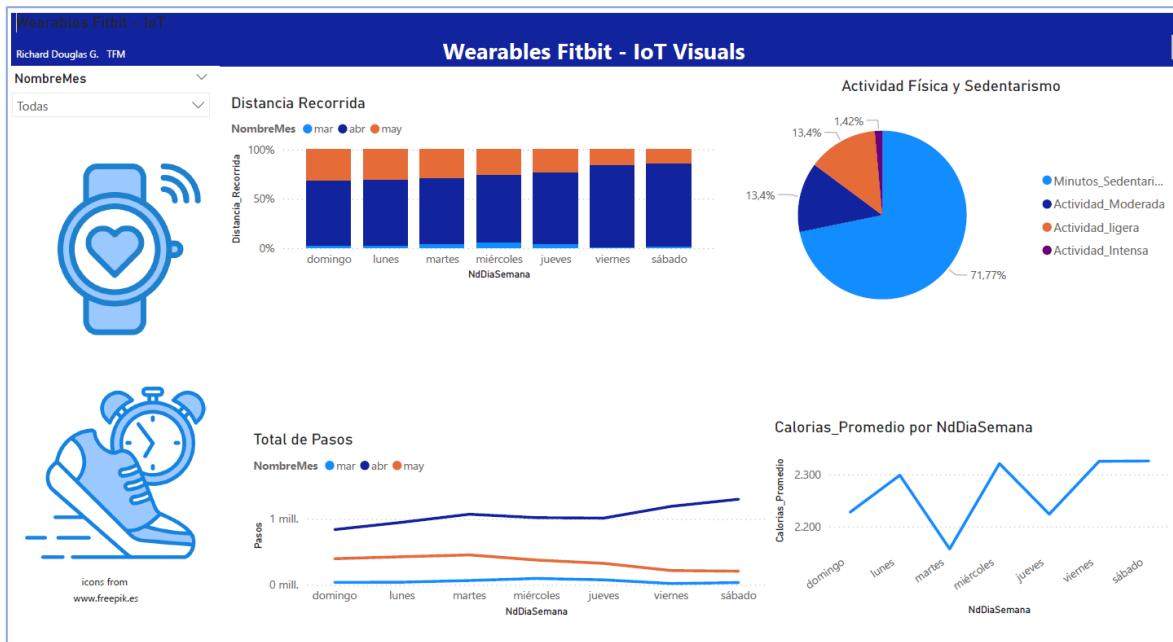
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The left pane displays the Object Explorer with a tree view of database objects for the 'DESKTOP-4ESN00D\SQLEXPRESS' instance. The 'Tables' node under the 'STG_Wearables_fitbit' database is expanded, showing tables like 'STG_FitbitActivityData' and 'STG_minuteSleep_merged'. The right pane contains a query window titled 'SQLQuery2.sql - DESKTOP-4ESN00D\SQLEXPRESS.STG_Wearables_fitbit (DESKTOP-4ESN00D\IMF (63)) - Microsoft SQL Server Management Studio'. The query itself is:

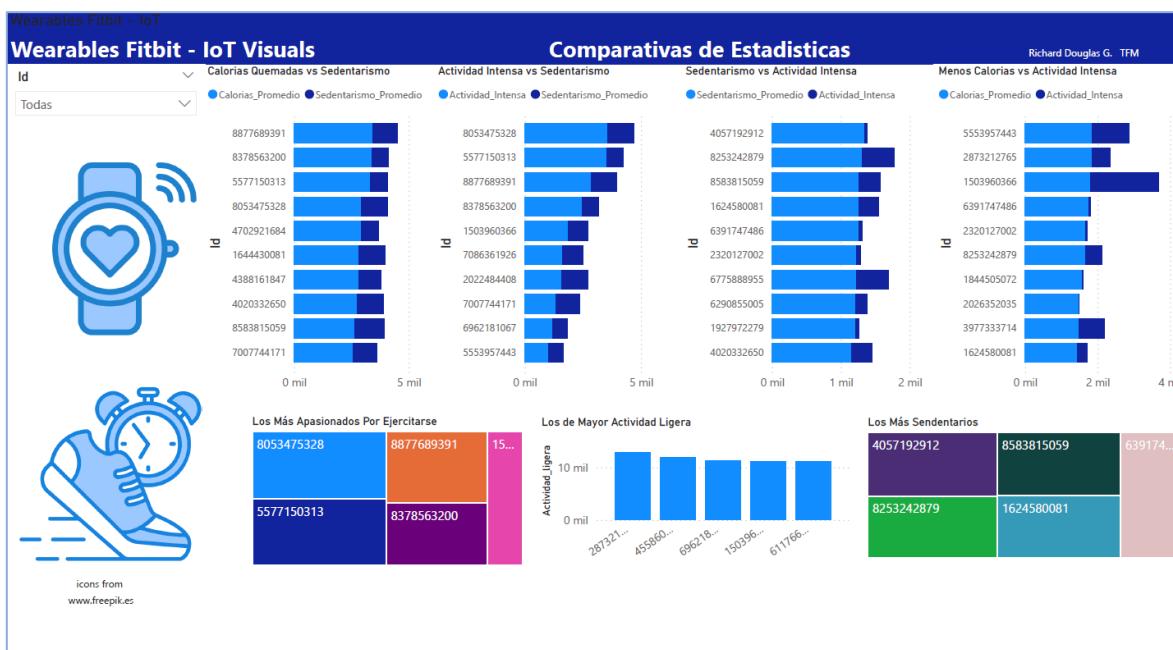
```
===== Script for SelectTopNRows command from SSMS =====
SELECT TOP (1000) [Id]
    ,[ActivityDate]
    ,[TotalSteps]
    ,[TotalDistance]
    ,[TrackerDistance]
    ,[LoggedActivitiesDistance]
    ,[VeryActiveDistance]
    ,[ModeratelyActiveDistance]
    ,[LightActiveDistance]
    ,[SedentaryActiveDistance]
    ,[VeryActiveMinutes]
    ,[FairlyActiveMinutes]
    ,[LightlyActiveMinutes]
    ,[SedentaryMinutes]
    ,[Calories]
FROM [STG_Wearables_fitbit].[dbo].[STG_FitbitActivityData]
```

Below the query window is a results grid titled 'Results'. The grid shows 8 rows of data corresponding to the executed query. The columns are: Id, ActivityDate, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesDistance, VeryActiveDistance, ModeratelyActiveDistance, LightActiveDistance, and SedentaryActiveDistance. The data is as follows:

	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedentaryActiveDistance
1	1503960366	2016-03-25	11004	7.00000	7.00000	0.00000	3.00000	0.00000	4.00000	0.00000
2	1503960366	2016-03-26	17609	12.00000	12.00000	0.00000	7.00000	1.00000	4.00000	0.00000
3	1503960366	2016-03-27	12736	9.00000	9.00000	0.00000	5.00000	0.00000	4.00000	0.00000
4	1503960366	2016-03-28	12321	9.00000	9.00000	0.00000	3.00000	1.00000	5.00000	0.00000
5	1503960366	2016-03-29	12041	8.00000	8.00000	0.00000	2.00000	1.00000	5.00000	0.00000
6	1503960366	2016-03-30	10970	7.00000	7.00000	0.00000	2.00000	1.00000	4.00000	0.00000
7	1503960366	2016-03-31	12256	8.00000	8.00000	0.00000	2.00000	0.00000	5.00000	0.00000
8	1503960366	2016-04-01	12262	8.00000	8.00000	0.00000	3.00000	1.00000	4.00000	0.00000

Anexo 3 Power bi dashboards





Medidas creadas

- Actividad_Intensa = SUM(FitbitActivityData[VeryActiveMinutes])
- Actividad_ligera = SUM(FitbitActivityData[LightlyActiveMinutes])
- Actividad_Moderada = SUM(FitbitActivityData[LightlyActiveMinutes])
- Calorias_Promedio = AVERAGE(FitbitActivityData[Calories])
- Distancia_Recorrida = SUM(FitbitActivityData[TotalDistance])
- Sedentarismo = SUM(FitbitActivityData[SedentaryMinutes])
- Sedentarismo_Promedio = AVERAGE(FitbitActivityData[SedentaryMinutes])
- Pasos = SUM(FitbitActivityData[TotalSteps])
- Horas_Sueno_Promedio = DIVIDE(AVERAGE(SleepData[TotalMinutesAsleep]),60)

The screenshot shows a data catalog interface with a search bar at the top. Below the search bar, there is a tree view of data assets. The 'Data' node is expanded, showing the following items:

- ✓ **Medidas** ...
 - Actividad_Intensa**
 - Actividad_ligera**
 - Actividad_Moderada**
 - Calorias_Promedio**
 - Distancia_Recorrida**
 - Horas_Sueno_Promedio**
 - Minutos_Sedentarios_Suma**
 - Minutos_Sueno_Promedio**
 - Minutos_Sueño_Total**
 - Pasos**
 - Sedentarismo**
 - Sedentarismo_Promedio**
 - Sueno_semanal**
 - Total_Horas_Sueno**
 - Total_Minutos_Sueno**
- > **Calendario**
- > **DIM_ID**
- > **FitbitActivityData**
- > **minuteSleep_merged**
- > **SleepData**

Modelo de Datos Heart Disease Dataset del UCI

Todo el proceso realizado para el análisis del dataset de Heart Disease Dataset del UCI, puede ser consultado en este link

https://drive.google.com/drive/folders/1tAeWofqjrC3-VBZm2KEBC1hIDmzQYLz4?usp=drive_link



Proyecto Final del Master TFM

MÁSTER EN DATA SCIENCE Y BUSINESS ANALYTICS

Estudiante : Richard Douglas Grijalba

Lograr un aporte a la salud por medio de los ML debe ser un nuevo hito en la historia de la humanidad. Ahora no solo podemos diagnosticarnos, sino también adelantarnos a algunos sucesos de salud que pueden acontecer en un futuro cercano o a mediano plazo.

Como dijo Neil Armstrong: "Este es un pequeño paso para un hombre, un gran salto para la humanidad."

En mi experiencia personal, desarrollar este proyecto final me ha dado la oportunidad de valorar muchos aspectos, como el tiempo de calidad en familia, y entender que cada minuto cuenta y vale. Para ello, necesitamos estar en buenas condiciones. La salud no es solo lo que comemos, sino también lo que pensamos y decimos.

Que la salud de todos esté en las manos de Dios y en la sabiduría y conocimiento de los médicos

Programa en modalidad virtual

▼ Dataset heart-disease-data

- Este dataset corresponde a un estudio realizado sobre pacientes y las diferentes características que pueden determinar en la predicción o no de un posible daño cardíaco.
- El dataset se utiliza principalmente para la clasificación y predicción de la presencia o ausencia de enfermedades cardíacas en pacientes. Los datos se basan en características clínicas y de salud de los pacientes.

se encuentra en las siguientes fuentes:

- <https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data>
- <https://archive.ics.uci.edu/dataset/45/heart+disease>

Información General de los Datos

- id: Identificador único.
- age: Edad del paciente.
- sex: Sexo (Male/Female).
- dataset: Fuente o ubicación del dataset (por ejemplo, "Cleveland").
- cp: Tipo de dolor en el pecho (como "typical angina", "asymptomatic", etc.).
- trestbps: Presión arterial en reposo (en mm Hg).
- chol: Nivel de colesterol en sangre (mg/dL).
- fbs: Glucosa en ayunas (TRUE si es mayor a 120 mg/dL, FALSE si es menor).
- restecg: Resultados del electrocardiograma en reposo (por ejemplo, "lv hypertrophy").
- thalch: Frecuencia cardíaca máxima alcanzada.
- exang: Angina inducida por ejercicio (TRUE o FALSE).
- oldpeak: Depresión del ST inducida por ejercicio en relación al reposo.
- slope: Pendiente del segmento ST (upsloping, flat, downsloping).
- ca: Número de vasos principales coloreados por fluoroscopia.
- thal: Tipo de defecto talámico (por ejemplo, "fixed defect", "normal", etc.).
- num: Etiqueta que indica la presencia de enfermedad cardíaca (0 para ninguna, * otros valores indican presencia de enfermedad en mayor severidad).

Característica adicional : target se agregan las características 0 = Representa la cantidad de casos para la clase 0 y 1= la suma de las clases 1, 2, 3, y 4.

Análisis Exploratorio de datos de las variables numéricas y categóricas

Según los resultados tenemos un dataset inicial heart-disease-data con un total de 920 y 16 características (columnas)

```
print("Número de filas: " + str(data.shape[0]))
print("Número de Columnas: " + str(data.shape[1]))
```

Número de filas: 920
Número de Columnas: 16

se exploran las primeras 5 filas del dataset y se observa que los datos tienen las columnas
Exploración inicial del dataset y sus características
data.head(5)

	id	age	sex	dataset	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	num
0	1	63	Male	Cleveland	typical angina	145.0	233.0	True	Iv hypertrophy	150.0	False	2.3	downsloping	0.0	fixed defect	0
1	2	67	Male	Cleveland	asymptomatic	160.0	286.0	False	Iv hypertrophy	108.0	True	1.5	flat	3.0	normal	2
2	3	67	Male	Cleveland	asymptomatic	120.0	229.0	False	Iv hypertrophy	129.0	True	2.6	flat	2.0	reversible defect	1
3	4	37	Male	Cleveland	non-anginal	130.0	260.0	False	normal	187.0	False	3.5	downsloping	0.0	normal	0
4	5	41	Female	Cleveland	atypical angina	130.0	204.0	False	Iv hypertrophy	172.0	False	1.4	upsloping	0.0	normal	0

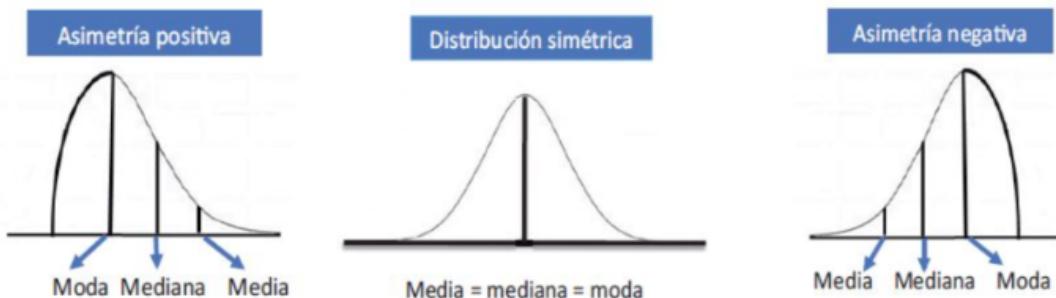
Verificar los nombres de las columnas
data.columns

```
Index(['id', 'age', 'sex', 'dataset', 'cp', 'trestbps', 'chol', 'fbs',
       'restecg', 'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'num'],
      dtype='object')
```

Imputación de Valores Nulos**Tratamiento de Valores Nulos o Faltantes**

```
# realizar una copia del dataset original antes de iniciar las transformaciones
df = data.copy()
```

En una distribución simétrica el valor de la media aritmética, la mediana y la moda coinciden.



- En el caso de una distribución con un comportamiento Simétrico se toma el valor de la Media
- En el caso de una distribución con comportamiento Asimétrico se toma el valor de la Mediana
- En el caso de alguna característica de tipo categórica se utiliza la Moda

```
# Crear una figura y ejes para los 10 histogramas
fig, axs = plt.subplots(1, 2, figsize=(6, 6))

# Nombres de las columnas
columnas = ['trestbps', 'chol']

# Iterar sobre cada columna y crear un histograma en el eje correspondiente
for i, columna in enumerate(columnas):
    axs[i].hist(df[columna], bins=20, color='skyblue', edgecolor='black')
    axs[i].set_title(columna)

# Ajustar el diseño de los subgráficos
plt.tight_layout()
plt.show()
```

```
dtype: object

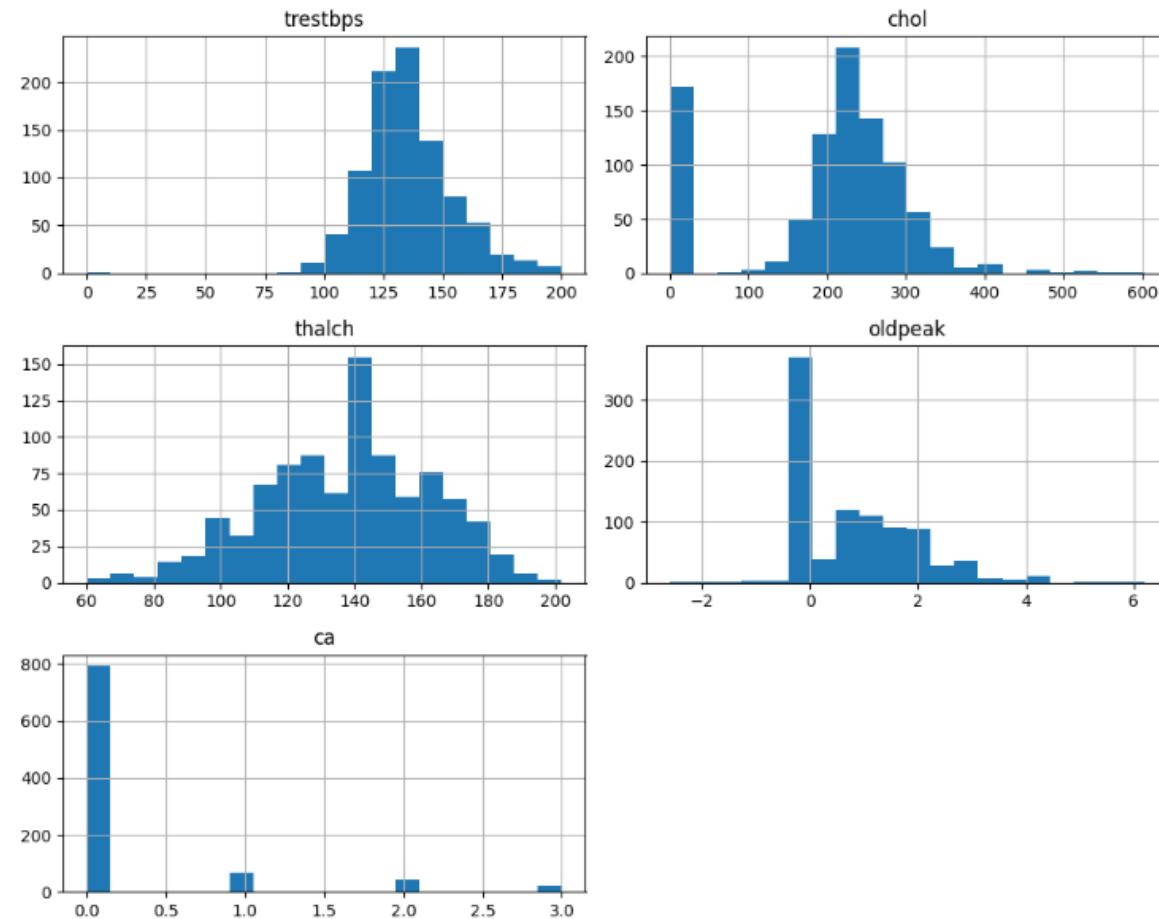
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 920 entries, 0 to 919
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   id          920 non-null    int64  
 1   age         920 non-null    int64  
 2   sex         920 non-null    object  
 3   dataset     920 non-null    object  
 4   cp          920 non-null    object  
 5   trestbps   920 non-null    float64 
 6   chol        920 non-null    float64 
 7   fbs         920 non-null    bool   
 8   restecg    920 non-null    object  
 9   thalch     920 non-null    float64 
 10  exang       920 non-null    bool   
 11  oldpeak    920 non-null    float64 
 12  slope       920 non-null    object  
 13  ca          920 non-null    float64 
 14  thal        920 non-null    object  
 15  num         920 non-null    int64  
dtypes: bool(2), float64(5), int64(3), object(6)
memory usage: 102.5+ KB
```

Histogramas de las características en limpio

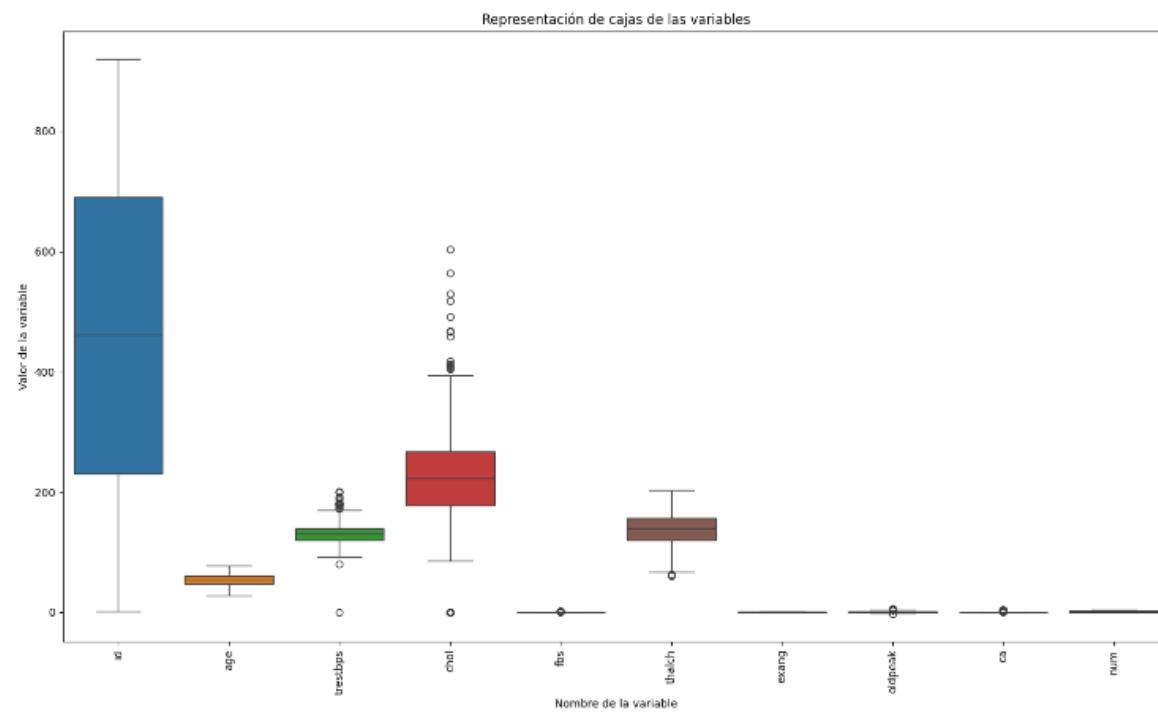
```
# Seleccionar las columnas de tipo float
columnas_float = df.select_dtypes(include=['float64']).columns

# Crear histogramas para cada una de las columnas de tipo float
df[columnas_float].hist(bins=20, figsize=(10, 8))
plt.tight_layout()
plt.show()
```



Generar Graficos Boxplots

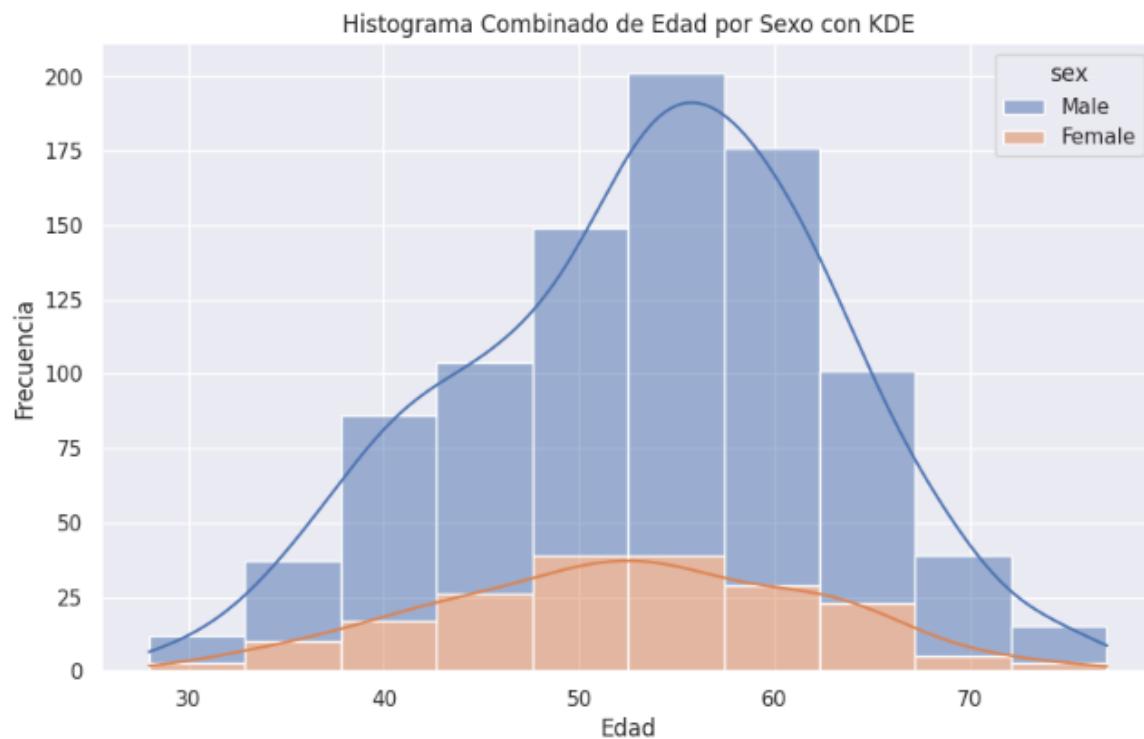
```
#Se generan Los graficos de caja/bigotes Boxplots para ver Los datos
plt.figure(figsize=(18,10))
ax = sns.boxplot(data=df)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
plt.title(u'Representación de cajas de las variables')
plt.ylabel('Valor de la variable')
_ = plt.xlabel('Nombre de la variable')
```



```
# Crear un histograma combinado para la edad, diferenciado por sexo
plt.figure(figsize=(10, 6))
sns.histplot(df, x='age', hue='sex', kde=True, bins=10, multiple='stack')

# Agregar título y etiquetas
plt.title('Histograma Combinado de Edad por Sexo con KDE')
plt.xlabel('Edad')
plt.ylabel('Frecuencia')

# Mostrar el gráfico
plt.show()
```



Visualización de Características

```

df1 = df.copy() # una copia del DF antes de realizar los cambios

# Convertir la columna 'sex' a numérica
df1['sex'] = df1['sex'].replace({'Male': 1, 'Female': 0})

# Convertir la columna 'cp' a numérica
df1['cp'] = df1['cp'].replace({'asymptomatic': 3, 'non-anginal': 2, 'atypical angina': 1, 'typical angina': 0})

# Convertir la columna 'restecg' a numérica
df1['restecg'] = df1['restecg'].replace({'normal': 0, 'lv hypertrophy': 1, 'st-t abnormality': 2})

# Convertir la columna 'slope' a numérica
df1['slope'] = df1['slope'].replace({'flat': 1, 'upsloping': 2, 'downsloping': 0})

# Convertir la columna 'thal' a numérica
df1['thal'] = df1['thal'].replace({'normal': 0, 'reversible defect': 1, 'fixed defect': 2})

# Convertir la columna 'col_range' a numérica
df1['Col_range'] = df1['Col_range'].replace({'Bajo': 0, 'Moderado': 1, 'Alto': 2})

# Convertir la columna 'fbs' a binario
df1['fbs'] = df1['fbs'].replace({True: 1, False: 0})

# Convertir la columna 'exang' a binario
df1['exang'] = df1['exang'].replace({True: 1, False: 0})

```

Verificar que la columna fue eliminada
df1.head()

	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	num	age_range	Col_range
0	1	63	1	0	145.0	233.0	1	1	150.0	0	2.3	0	0.0	2	0	60	1
1	2	67	1	3	160.0	286.0	0	1	108.0	1	1.5	1	3.0	0	2	60	2
2	3	67	1	3	120.0	229.0	0	1	129.0	1	2.6	1	2.0	1	1	60	1
3	4	37	1	2	130.0	250.0	0	0	187.0	0	3.5	0	0.0	0	0	30	2
4	5	41	0	1	130.0	204.0	0	1	172.0	0	1.4	2	0.0	0	0	40	1

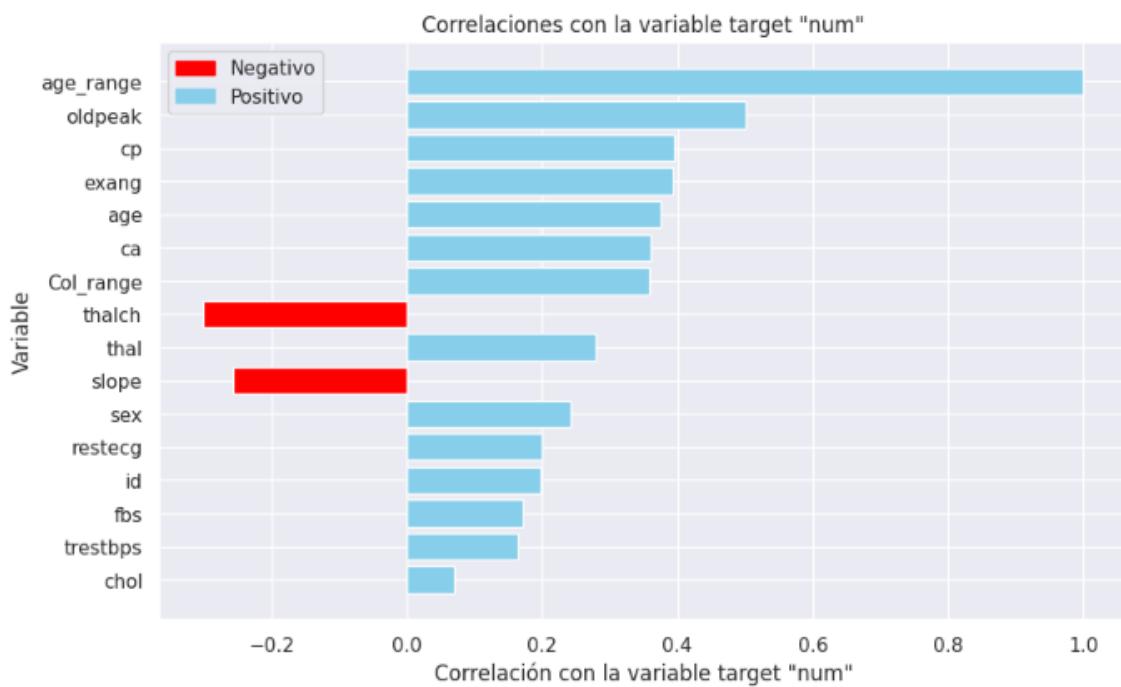
```
# Definir Los datos para el gráfico de barras
nombres_columnas = list(diccionario.keys())
correlaciones = list(diccionario.values())

# Crear el gráfico de barras
plt.figure(figsize=(10, 6))
# Usar 'red' para Las correlaciones negativas y 'skyblue' para Las positivas
colors = ['red' if c < 0 else 'skyblue' for c in correlaciones]
bars = plt.barh(nombres_columnas, correlaciones, color=colors)

plt.xlabel('Correlación con la variable target "num"')
plt.ylabel('Variable')
plt.title('Correlaciones con la variable target "num"')
plt.gca().invert_yaxis() # Invertir el eje y para mostrar La variable más correlacionada en La parte superior

# Crear Los parches para La Leyenda, uno para rojo y otro para azul claro
import matplotlib.patches as mpatches
red_patch = mpatches.Patch(color='red', label='Negativo')
blue_patch = mpatches.Patch(color='skyblue', label='Positivo')
plt.legend(handles=[red_patch, blue_patch])

plt.show()
```

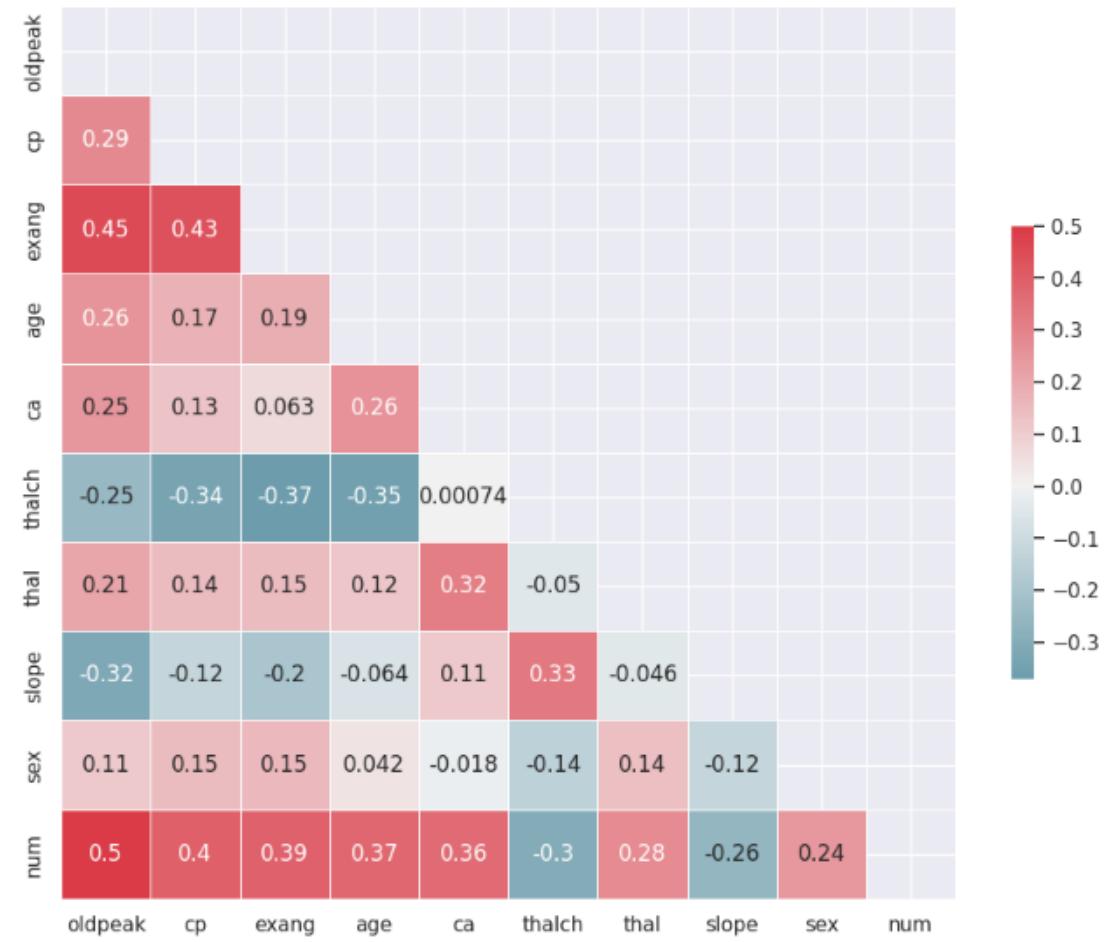


```
# Calcular La matriz de correlación
corr = df2.corr()

# Crear La máscara
mask = np.zeros_like(corr, dtype=bool) # Cambiar np.bool a bool
mask[np.triu_indices_from(mask)] = True

# Configurar el gráfico
f, ax = plt.subplots(figsize=(11, 9))
cmap = sns.diverging_palette(220, 10, as_cmap=True)

# Graficar el mapa de calor
sns.heatmap(corr, mask=mask, cmap=cmap, center=0, square=True, linewidths=.5, cbar_kws={'shrink': .5}, annot=True)
plt.show()
```



Librerías de Modelado

```
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC

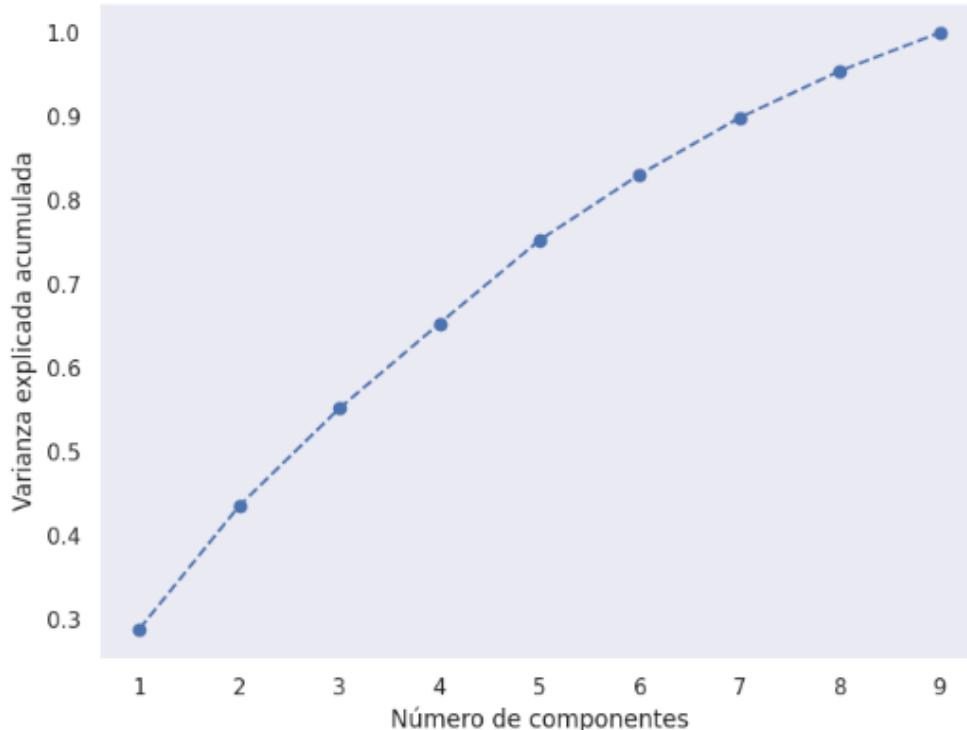
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Ajustar PCA
pca = PCA().fit(X_train)

# Varianza explicada acumulada
varianza_acumulada = pca.explained_variance_ratio_.cumsum()

# Graficar La varianza acumulada
plt.figure(figsize=(8,6))
plt.plot(range(1, len(varianza_acumulada)+1), varianza_acumulada, marker='o', linestyle='--')
plt.title('Varianza explicada acumulada por número de componentes')
plt.xlabel('Número de componentes')
plt.ylabel('Varianza explicada acumulada')
plt.grid()
plt.show()
```

Varianza explicada acumulada por número de componentes



```
# Lista de modelos
models = []
models.append(("LR", LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(("LDA", LinearDiscriminantAnalysis()))
models.append(("KNN", KNeighborsClassifier()))
models.append(("CART", DecisionTreeClassifier()))
models.append(("NB", GaussianNB()))
models.append(("SVC", SVC()))

# Inicializar resultados y nombres
resultados = []
names = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    resultados.append(cv_resultados)
    names.append(name)
    print('%s: Accuracy = %f (Std = %f)' % (name, cv_resultados.mean(), cv_resultados.std()))

# Inicializar nombres y resultados de accuracy
names = []
accuracies = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    accuracies.append(cv_resultados.mean())
    names.append(name)

# Crear un DataFrame para mostrar los resultados de accuracy
accuracies_test1 = pd.DataFrame(accuracies, columns=['Accuracy_Test1'], index=names)

# Mostrar la tabla de resultados de accuracy
# se almacena el resultado para realizar un acompañamiento con las diferentes corridas y pruebas
print(accuracies_test1)

LR: Accuracy = 0.640621 (Std = 0.039760)
LDA: Accuracy = 0.647316 (Std = 0.049254)
KNN: Accuracy = 0.618785 (Std = 0.038755)
CART: Accuracy = 0.558644 (Std = 0.054614)
NB: Accuracy = 0.618955 (Std = 0.072338)
SVC: Accuracy = 0.643983 (Std = 0.050137)

    Accuracy_Test1
LR          0.640621
LDA         0.647316
KNN         0.618785
CART        0.555311
NB          0.618955
SVC         0.643983
```

Aplicación de un modelo de redes neuronales

Entrenar el modelo de red neuronal:

```
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Inicializar el modelo
ann_model = MLPClassifier(hidden_layer_sizes=(100,), max_iter=500, activation='relu', solver='adam', random_state=1)
# Inicializar el modelo
ann_model.fit(X_train_scaled, Y_train)

# Entrenar el modelo con los datos
ann_model.fit(X_train, Y_train)
```

MLPClassifier(max_iter=500, random_state=1)

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Evaluar el modelo

```
accuracy_ann = ann_model.score(X_test_scaled, Y_test)

print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)
```

Accuracy del modelo de redes neuronales (ANN): 0.5733333333333334

```
# Crear un DataFrame con el resultado de la red neuronal
df_ann = pd.DataFrame({'Accuracy_Test1': [accuracy_ann]}, index=['ANN'])

# Concatenar el DataFrame de la red neuronal con accuracies_test1
accuracies_test1 = pd.concat([accuracies_test1, df_ann])

# Mostrar la tabla con los resultados actualizados
print(accuracies_test1)
```

	Accuracy_Test1
LR	0.640621
LDA	0.647316
KNN	0.618785
CART	0.555311
NB	0.618955
SVC	0.643983
ANN	0.573333

```
# Lista de modelos
models = []
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVC', SVC()))

# Inicializar resultados y nombres
resultados = []
names = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    resultados.append(cv_resultados)
    names.append(name)
    print('%s: Accuracy = %f (Std = %f)' % (name, cv_resultados.mean(), cv_resultados.std()))

# Inicializar nombres y resultados de accuracy
names = []
accuracies = []

# Evaluar Los modelos
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    accuracies.append(cv_resultados.mean())
    names.append(name)

# Crear un DataFrame para mostrar los resultados de accuracy
accuracies_test2 = pd.DataFrame(accuracies, columns=['Accuracy_Test2'], index=names)

# Mostrar la tabla de resultados de accuracy
# se almacena el resultado para realizar unacomparacion con las diferentes corridas y pruebas
print(accuracies_test2)
```

```
LR: Accuracy = 0.642232 (Std = 0.040166)
LDA: Accuracy = 0.640508 (Std = 0.046385)
KNN: Accuracy = 0.598729 (Std = 0.050542)
CART: Accuracy = 0.558559 (Std = 0.059464)
NB: Accuracy = 0.617090 (Std = 0.063650)
SVC: Accuracy = 0.640593 (Std = 0.042003)
    Accuracy_Test2
LR          0.642232
LDA         0.640508
KNN         0.598729
CART        0.561921
NB          0.617090
SVC         0.640593
```

Optimización de Hiperparámetros:

```

from sklearn.model_selection import GridSearchCV

# Parámetros para optimizar
param_grid = {
    'hidden_layer_sizes': [(100,), (150, 100, 50)],
    'activation': ['relu', 'tanh'],
    'alpha': [0.0001, 0.001, 0.01],
    'solver': ['adam', 'lbfgs']
}

# GridSearchCV para optimizar hiperparámetros
grid_ann = GridSearchCV(MLPClassifier(max_iter=1000), param_grid, cv=5, verbose=2)
grid_ann.fit(X_train_scaled, Y_train)

# Evaluar el mejor modelo
mejor_ann = grid_ann.best_estimator_
accuracy_ann_opt = mejor_ann.score(X_test_scaled, Y_test)
print("Accuracy del mejor modelo ANN con GridsearchCV: ", accuracy_ann_opt)

Fitting 5 folds for each of 24 candidates, totalling 120 fits
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=adam; total time= 2.0s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=adam; total time= 1.95s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=adam; total time= 2.8s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=adam; total time= 2.75s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=adam; total time= 1.95s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=lbfgs; total time= 2.1s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=lbfgs; total time= 2.0s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=lbfgs; total time= 2.3s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=lbfgs; total time= 2.1s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(100,), solver=lbfgs; total time= 4.7s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 3.95s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 4.1s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 8.25s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 3.05s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=adam; total time= 3.85s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 20.45s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 19.65s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 13.65s
[CV] END activation=relu, alpha=0.0001, hidden_layer_sizes=(150, 100, 50), solver=lbfgs; total time= 24.45s

# Crear un DataFrame con el resultado de La red neuronal
df_ann = pd.DataFrame({'Accuracy_Test2': [accuracy_ann_opt]}, index=['ANN'])

# Concatenar el DataFrame de la red neuronal con accuracies_test1
accuracies_test2 = pd.concat([accuracies_test2, df_ann])

# Mostrar La tabla con los resultados actualizados
print(accuracies_test2)

```

	Accuracy_Test2
LR	0.642232
LDA	0.640508
KNN	0.598729
CART	0.561921
NB	0.617090
SVC	0.640593
ANN	0.566667

Feature engineering en la característica target

Decisión de Conversión de la Variable Objetivo Motivación: La variable num original del dataset contenía cinco clases que representaban distintos grados de daño cardíaco (0 = no heart disease, 1 = mild, 2 = moderate, 3 = severe, 4 = critical). Sin embargo, al evaluar el desempeño de los modelos predictivos con esta variable multiclas, observamos que los resultados en términos de precisión fueron moderados, con valores de accuracy entre 50% y 71%. Este rendimiento puede estar relacionado con el desequilibrio de clases, donde la clase "no heart disease" representaba la mayoría de los casos (411 de un total de 920).

Decisión: Para mejorar la calidad de la predicción y simplificar el problema, se tomó la decisión de convertir la variable num en una variable binaria. Esta nueva variable, llamada target, tiene los siguientes valores:

0: No presenta daño cardíaco (igual que la clase 0 de la variable original num). 1: Presenta algún grado de daño cardíaco (esto agrupa las clases 1, 2, 3 y 4 de la variable original).

df4.head()																	
	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	num	age_range	Col_range
0	1	63	1	0	146.0	233.0	1	1	150.0	0	2.3	0	0.0	2	0	60	1
1	2	67	1	3	160.0	286.0	0	1	108.0	1	1.5	1	3.0	0	2	60	2
2	3	67	1	3	120.0	229.0	0	1	129.0	1	2.6	1	2.0	1	1	60	1
3	4	37	1	2	130.0	250.0	0	0	187.0	0	3.5	0	0.0	0	0	30	2
4	5	41	0	1	130.0	204.0	0	1	172.0	0	1.4	2	0.0	0	0	40	1

Crear la nueva columna 'target'

- 0 411
- 1 265
- 2 109
- 3 107
- 4 28

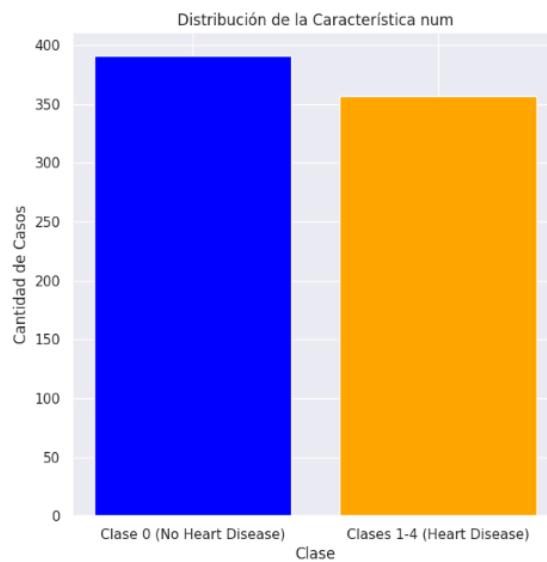
```
# Contar las instancias de la columna 'num'
conteo_num = df4['num'].value_counts()

# Definir las categorías: clase 0 y la suma de clases 1, 2, 3, 4
valores = [conteo_num[0], conteo_num[1] + conteo_num[2] + conteo_num[3] + conteo_num[4]]
etiquetas = ['Clase 0 (No Heart Disease)', 'Clases 1-4 (Heart Disease)']

# Crear el gráfico de barras
plt.bar(etiquetas, valores, color=['blue', 'orange'])

# Añadir etiquetas y título
plt.xlabel('Clase')
plt.ylabel('Cantidad de Casos')
plt.title('Distribución de la Característica num')

# Mostrar el gráfico
plt.show()
```



Modelado feauring Target : booleano 1 y 0

Se excluye num en vista que podria generar un sobre ajuste en los resultados, en vista que target se origina de los valores de 'num'.

Se procede a probar el resultado de los modelos una vez que se han consolidado los valores en una caracteristica del tipo booleana.

```
# Definir Las variables predictoras excluyendo 'target' y 'num'  
X = df4.drop(columns=['target', 'num'])  
  
# Definir La columna 'target' como La variable objetivo  
Y = df4['target']  
  
# Estandarizar Las características  
obj_escalador = StandardScaler()  
X_estandarizado = obj_escalador.fit_transform(X)  
  
# División de Los datos en conjunto de entrenamiento y prueba  
X_train, X_test, Y_train, Y_test = train_test_split(X_estandarizado, Y, test_size=0.2, random_state=0)  
  
# Lista de modelos  
models = []  
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))  
models.append(('LDA', LinearDiscriminantAnalysis()))  
models.append(('KNN', KNeighborsClassifier()))  
models.append(('CART', DecisionTreeClassifier()))  
models.append(('NB', GaussianNB()))  
models.append(('SVC', SVC()))  
  
# Inicializar resultados y nombres  
resultados = []  
names = []  
  
# Evaluar Los modelos  
for name, model in models:  
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)  
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')  
    resultados.append(cv_resultados)  
    names.append(name)  
    print('%s: Accuracy = %f (Std = %f)' % (name, cv_resultados.mean(), cv_resultados.std()))  
  
# Inicializar nombres y resultados de accuracy  
names = []  
accuracias = []  
  
# Evaluar Los modelos  
for name, model in models:  
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)  
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')  
    accuracias.append(cv_resultados.mean())  
    names.append(name)  
  
# Crear un DataFrame para mostrar los resultados de accuracy  
accuracias_test3 = pd.DataFrame(accuracias, columns=['Accuracy_Test3'], index=names)  
  
# Mostrar La tabla de resultados de accuracy  
# se almacena el resultado para realizar un acomparacion con Las diferentes corridas y pruebas  
print(accuracias_test3)  
  
LR: Accuracy = 0.824492 (Std = 0.046455)  
LDA: Accuracy = 0.831215 (Std = 0.051061)  
KNN: Accuracy = 0.831158 (Std = 0.027046)  
CART: Accuracy = 0.797768 (Std = 0.036359)  
NB: Accuracy = 0.817740 (Std = 0.055654)  
SVC: Accuracy = 0.847881 (Std = 0.042925)
```

Redes Neuronales sobre Target

Sobre la nueva característica Target Clase 0 (No Heart Disease) Clase (Heart Disease)

modelo de redes neuronales

```
# Definir las variables predictoras excluyendo 'target' y 'num'  
X = df4.drop(columns=['target', 'num'])  
  
# Definir la columna 'target' como la variable objetivo  
Y = df4['target']  
  
# División de los datos en conjunto de entrenamiento y prueba  
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)  
  
from sklearn.neural_network import MLPClassifier  
from sklearn.preprocessing import StandardScaler  
  
scaler = StandardScaler()  
X_train_scaled = scaler.fit_transform(X_train)  
X_test_scaled = scaler.transform(X_test)  
# Inicializar el modelo  
ann_model = MLPClassifier(hidden_layer_sizes=(100,), max_iter=500, activation='relu', solver='adam', random_state=1)  
# Inicializar el modelo  
ann_model.fit(X_train_scaled, Y_train)  
  
# Entrenar el modelo con los datos  
ann_model.fit(X_train, Y_train)  
  
MLPClassifier(max_iter=500, random_state=1)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Evaluar el modelo

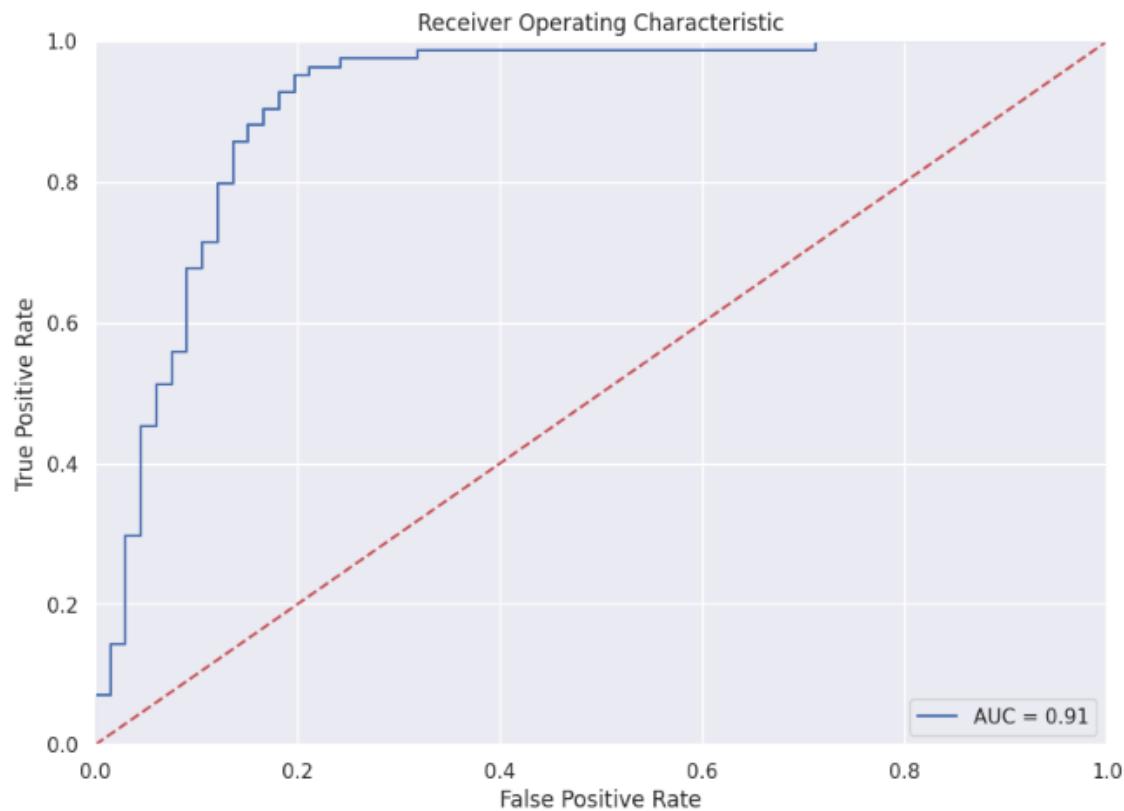
```
accuracy_ann = ann_model.score(X_test_scaled, Y_test)  
print("Accuracy del modelo de redes neuronales (ANN): ", accuracy_ann)
```

Accuracy del modelo de redes neuronales (ANN): 0.7533333333333333

```
# Obtener las predicciones para calcular el ROC AUC
preds = y_test_pred_prob[:, 1] # Probabilidades de la clase positiva (Heart Disease)
```

```
# Calcular la curva ROC y el AUC
fpr, tpr, threshold = metrics.roc_curve(Y_test, preds)
roc_auc = metrics.auc(fpr, tpr)
```

```
# Graficar la curva ROC
plt.figure(figsize=(10, 7))
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc='lower right')
plt.plot([0, 1], [0, 1], 'r--') # Línea diagonal
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



```
from sklearn import metrics

# Umbralizar las predicciones con un umbral de 0.5 (o el que decidas)
umbral = 0.5
y_umbralizadas = 1 * (y_test_pred_prob[:, 1] > umbral)

# Imprimir la matriz de confusión
print(u"Matriz de confusión\n", metrics.confusion_matrix(Y_test, y_umbralizadas))

# Imprimir métricas de rendimiento
print("\nAccuracy\t{}".format(round(metrics.accuracy_score(Y_test, y_umbralizadas), 2)))
print("Sensibilidad (Recall)\t{}".format(round(metrics.recall_score(Y_test, y_umbralizadas), 2)))
print(u"Precisión\t{}".format(round(metrics.precision_score(Y_test, y_umbralizadas), 2)))
```

Matriz de confusión

```
[[57  9]
 [17 67]]
```

```
Accuracy      0.83
Sensibilidad (Recall)  0.8
Precisión      0.88
```

```
represento_doble_hist(y_test_pred_prob_pos[:, 1], y_test_pred_prob_neg[:, 1], n_bins=21)
```



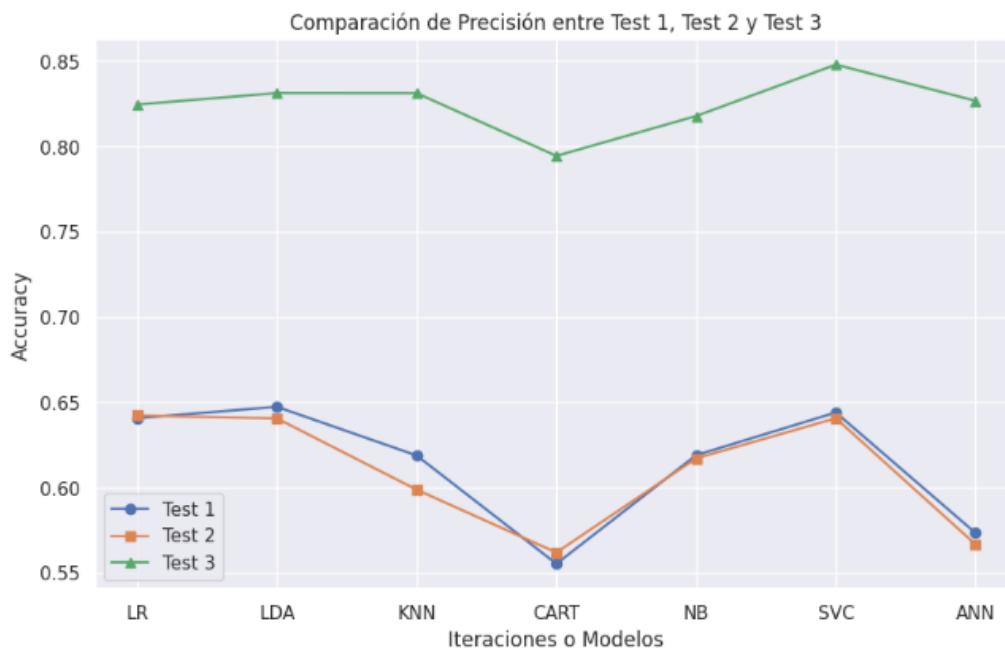


Grafico: Resultados de Modelos

El gráfico comparativo muestra el rendimiento de varios modelos de machine learning en tres pruebas diferentes (Test 1, Test 2 y Test 3). Al principio, los resultados de Test 1 y Test 2 presentan rendimientos conservadores, con niveles de precisión (accuracy) que varían entre los modelos. Sin embargo, con la implementación de mejoras como el ajuste de hiperparámetros y técnicas de feature engineering, se observa una mejora notable en los resultados de Test 3, especialmente en modelos como Logistic Regression (LR), SVC y ANN.

El comportamiento de los modelos refleja una evolución en su capacidad predictiva. Modelos como ANN muestran una curva ascendente significativa, lo que sugiere que los ajustes realizados fueron particularmente efectivos en este modelo. Por otro lado, modelos como KNN y CART presentan una mejora más gradual, lo que podría indicar que estos modelos requieren ajustes más específicos para alcanzar rendimientos superiores.

La visualización de las tres pruebas permite identificar fácilmente las diferencias entre los modelos y cómo sus resultados mejoran o se mantienen consistentes a lo largo de las iteraciones. Esta gráfica, junto con los resultados de la tabla de precisión, subraya la importancia de los ajustes y las pruebas múltiples para obtener modelos más robustos y precisos.

Crear el Modelo Final en un ejecutable

```
import joblib

# Guardar el modelo SVC entrenado en un archivo
joblib.dump(modelo_final, 'modelo_SVC_entrenado.pkl')

['modelo_SVC_entrenado.pkl']
```

Llamar el Modelo Entrenado

```
heart_disease = df4.copy()

heart_disease.columns

Index(['id', 'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'num', 'age_range',
       'Col_range', 'target'],
      dtype='object')
```

```
heart_disease = heart_disease.drop(columns=['num', 'target'])

heart_disease.columns

Index(['id', 'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'age_range',
       'col_range'],
      dtype='object')
```

```
#cargar el modelo guardado posteriormente
modelo_cargado = joblib.load('modelo_SVC_entrenado.pkl')
```

```
# Usar el modelo cargado para hacer predicciones
predicciones = modelo_cargado.predict(heart_disease)
```

```
# Añadir las predicciones al DataFrame en una nueva columna llamada 'HeartDisease_Pred'
heart_disease['HeartDisease_Pred'] = predicciones
```

```
# Mostrar las primeras filas del DataFrame con las predicciones añadidas
heart_disease.head()
```

	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	age_range	Col_range	HeartDisease_Pred
0	1	63	1	0	145.0	233.0	1	1	150.0	0	2.3	0	0.0	2	60	1	1
1	2	67	1	3	160.0	286.0	0	1	108.0	1	1.5	1	3.0	0	60	2	1
2	3	67	1	3	120.0	229.0	0	1	129.0	1	2.6	1	2.0	1	60	1	1
3	4	37	1	2	130.0	250.0	0	0	187.0	0	3.5	0	0.0	0	30	2	1
4	5	41	0	1	130.0	204.0	0	1	172.0	0	1.4	2	0.0	0	40	1	1