TRABAJO DE GRADO
Opción Seminario-Diplomado.

ANÁLISIS Y TOMA DE DECISIONES INFORMADAS A PARTIR DE DATOS DE RIESGO DE OBESIDAD, UTILIZANDO ESTRATEGIAS DE MACHINE LEARNING

Corporación Universitaria Remington.
Facultad de ingeniería de sistemas
Seminario de machine learning en tiempo de datos

Estudiante:
BLANCA BELÉN BECERRA GÓMEZ

Tutor: Juan Carlos Briñez de León

Opción de Trabajo de grado Seminario-Diplomado.
2024.

Tabla de Contenidos

Contenido

Resumen

Según la OMS la tasa de niveles de obesidad entre jóvenes y adultos se ha ido incrementando con el paso del tiempo. por esto, se ha dado la tarea de la implementación de un algoritmo de machine Lear Ning de aprendizaje supervisado enfocado en el ámbito de la salud de los usuarios, este algoritmo se centra en desarrollar métodos que agilicen los diagnósticos médicos en pacientes que lo requieran, sin la necesidad de presentarse en un centro médico. Este sistema cuenta con la facilidad de entender los valores que se le brinden, buscando así predecir si un usuario cuenta con obesidad según su peso, altura, y su manejo de hábitos diarios. Estos datos obtenidos por los individuos serán tomados en cuenta para identificar en que nivel de obesidad se encuentra.

Este algoritmo implementado en este trabajo, cuenta con la información de 20758 datos de personas jóvenes, adultos- jóvenes y adultos con la finalidad de dar a conocer que la obesidad es una enfermedad crónica que se debe tener en cuenta y no dejarla pasar por desapercibido.

Palabras clave

Machine Lear Ning, clasificación de datos, riesgos de obesidad, predicciones, Salud.

Marco conceptual y contextual

El presente documento aborda los conceptos clave relacionados con el tema en estudio, así como el método utilizado para analizar los datos. Esto tiene como objetivo contextualizar los temas tratados y garantizar un desarrollo y comprensión efectiva del trabajo.

Obesidad.

Según las referencias encontradas La obesidad es una enfermedad crónica que afecta a la población a nivel global. Se caracteriza por la acumulación excesiva de grasa en diversas áreas del cuerpo, lo que puede conllevar riesgos significativos para la salud. Dependiendo de la distribución y el grado de acumulación de grasa, estos riesgos pueden convertirse en factores determinantes de la mortalidad a nivel mundial [1].

En este contexto, la Organización Mundial de la Salud (OMS) ha clasificado la obesidad en categorías específicas, las cuales se presentan en la tabla contenida en el documento *"Causas de la obesidad"* [1].

Clasificación	**IMC (kg/m2)**	**Riesgo Asociado a la salud**
Normo peso	18.5-24.9	Promedio
Exceso de peso	≥ 25	
Sobrepeso o Pre-Obeso	25 - 29.9	AUMENTADO
Obesidad Grado I o moderada	30 − 34.9	AUMENTADO MODERADO
Obesidad Grado II o severa	35 - 39.9	AUMENTO SEVERO
Obesidad Grado III o mórbida	≥ 40	AUMENTO MUY SEVERO

Tabla1: clasificación de obesidad según la OMS.

Alimentación saludable.

Según lo investigado, la alimentación saludable se define como un pilar fundamental para garantizar el funcionamiento óptimo del organismo humano. Esta práctica no solo contribuye al adecuado desempeño de las funciones corporales, sino que también ofrece múltiples beneficios a lo largo de las diferentes etapas de la vida.

Entre los principales beneficios de una alimentación equilibrada se encuentran la reducción significativa del riesgo de padecer enfermedades crónicas, como la obesidad, la diabetes tipo 2, y afecciones cardiovasculares. Además, una dieta adecuada ayuda a mantener, regular y mejorar la salud integral, fortaleciendo el sistema inmunológico y optimizando la energía necesaria para realizar actividades cotidianas.

En niños y jóvenes, la alimentación saludable desempeña un papel esencial en el crecimiento y desarrollo físico y cognitivo. Al proporcionar los nutrientes adecuados, fomenta un desarrollo óptimo del cerebro, fortalece los huesos y músculos, y favorece el rendimiento académico y deportivo. Asimismo, contribuye a instaurar hábitos alimenticios positivos que pueden perdurar en la vida adulta, reduciendo así los riesgos de enfermedades a largo plazo [2].

Actividad física.

Según las referencias consultadas, la actividad física puede clasificarse en dos conceptos clave, con el propósito de distinguir entre dos términos que suelen ser

utilizados de manera conjunta en la sociedad, aunque poseen diferencias significativas.

En primer lugar, el término **actividad física** se refiere a todas aquellas acciones realizadas por una persona en su vida cotidiana de forma espontánea y continua. Estas actividades implican un gasto energético derivado de los movimientos corporales, como caminar, subir escaleras o realizar tareas domésticas.

Por otro lado, el **ejercicio físico** se define como una forma específica e intencionada de actividad física. Este se lleva a cabo con regularidad y en algunos casos de manera diaria, con el objetivo principal de mantener o mejorar el estado físico del cuerpo. A diferencia de la actividad física general, el ejercicio físico se caracteriza por ser estructurado, planificado y orientado a alcanzar metas relacionadas con la salud, el rendimiento o la estética corporal [3].

Google Colaboratory.

Google Colaboratory, conocido como Colab, es una herramienta innovadora y altamente efectiva en el ámbito del aprendizaje automático. Este servicio, basado en la plataforma Jupyter Notebook, ofrece un entorno preconfigurado en lenguaje Python, eliminando la necesidad de instalar software adicional en el ordenador.

Su principal ventaja radica en su accesibilidad, ya que es completamente gratuito y puede utilizarse de manera sencilla a través de una cuenta de Google. Gracias a estas características, Colab se ha convertido en una solución ideal para quienes

desean explorar y desarrollar proyectos en áreas como la inteligencia artificial, el análisis de datos y la programación.

Además, su implementación ha tenido un impacto significativo en el campo de la docencia. Al facilitar el acceso a herramientas de alto rendimiento, Colab ha permitido a estudiantes y educadores abordar proyectos complejos de manera colaborativa y eficiente, fomentando el aprendizaje práctico y la experimentación en tiempo real [4,5].

Aprendizaje supervisado

El aprendizaje supervisado es un enfoque de aprendizaje automático en el que se proporciona información explícita sobre los datos presentados al modelo. Cada dato incluye una etiqueta que lo define, lo que permite al modelo identificar patrones y clasificar nuevos datos con etiquetas similares.

El objetivo principal del aprendizaje supervisado es analizar los datos de entrada, es decir, los datos iniciales proporcionados al sistema. A partir de este análisis, el modelo utiliza las características presentes en los datos para clasificarlos automáticamente. Finalmente, el proceso genera una salida precisa, correspondiente a las etiquetas asignadas a los datos procesados.

En este documento se desarrollará el concepto de aprendizaje supervisado con un enfoque particular en su funcionamiento y aplicaciones. La **Figura 1** ilustrará el proceso de clasificación, mostrando cómo se enseña al modelo (clasificador) a identificar y organizar los datos en función de las etiquetas suministradas [6].

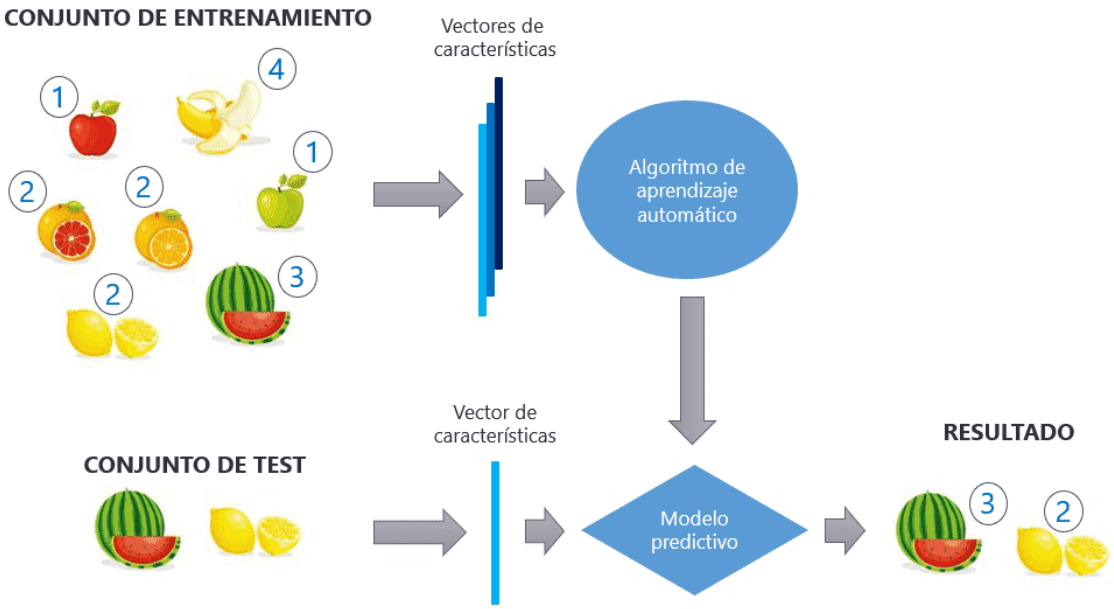


Figura 1. Aprendizaje supervisado.

El aprendizaje supervisado tiene un aspecto crucial que será central en el desarrollo del presente documento: el método de clasificación. Este enfoque permite que, una vez que el sistema ha sido entrenado con datos previamente etiquetados, pueda responder de manera inmediata y precisa al recibir un nuevo dato.

El funcionamiento de este proceso se detalla en la **Figura 1**, donde se ilustra cómo el sistema adquiere y aplica el conocimiento aprendido. Asimismo, en la **Figura 2** se muestra cómo el sistema genera respuestas automáticas al procesar nuevos datos, evidenciando su capacidad de clasificación eficiente basada en el aprendizaje supervisado [6].

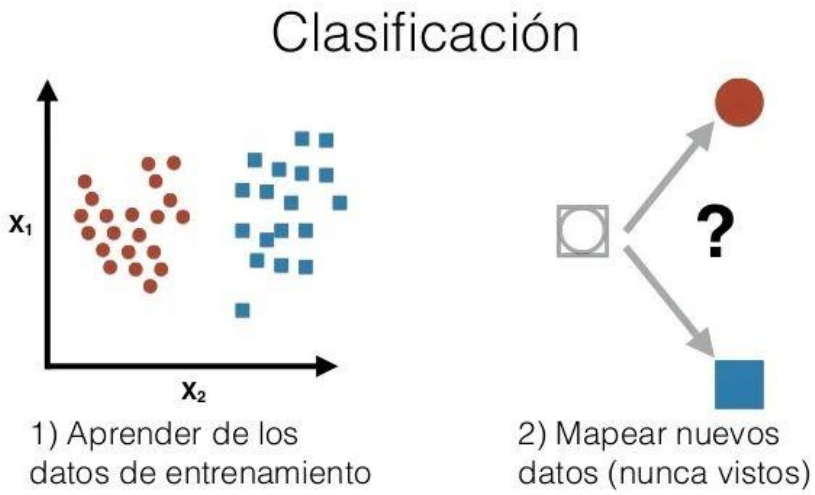


Figura 2. Aprendizaje supervisado, clasificación.

Pregunta problema

¿Cómo implementar un modelo de machine learning que desarrolle y optimice una base de datos, consiguiendo la clasificación de obesidad en una persona?

Acercamiento a los datos:

El conjunto de datos denominado *"Obesity Risk Dataset"* (Conjunto de datos sobre el riesgo de obesidad), creado por Jayesh Jain [7], ha sido seleccionado desde la plataforma Kaggle. Esta plataforma es ampliamente reconocida por ofrecer acceso a una gran variedad de conjuntos de datos sobre temas específicos, en formatos como CSV o TXT.

Kaggle se destaca por su facilidad de uso y por permitir a los usuarios interesados en análisis de datos acceder a información sin restricciones, lo que la convierte en una herramienta valiosa para investigadores y analistas.

En este caso, el conjunto de datos elegido proporciona información detallada sobre las características de las personas incluidas en el estudio del riesgo de obesidad. Este recurso resulta esencial para analizar patrones, identificar factores de riesgo y desarrollar modelos predictivos en el ámbito de la salud [7].

Descripción de variables.

El conjunto de datos está compuesto por 18 etiquetas, clasificadas en tres tipos principales: datos con números enteros, datos con números decimales y datos de tipo carácter [7] (**Figura 3**).

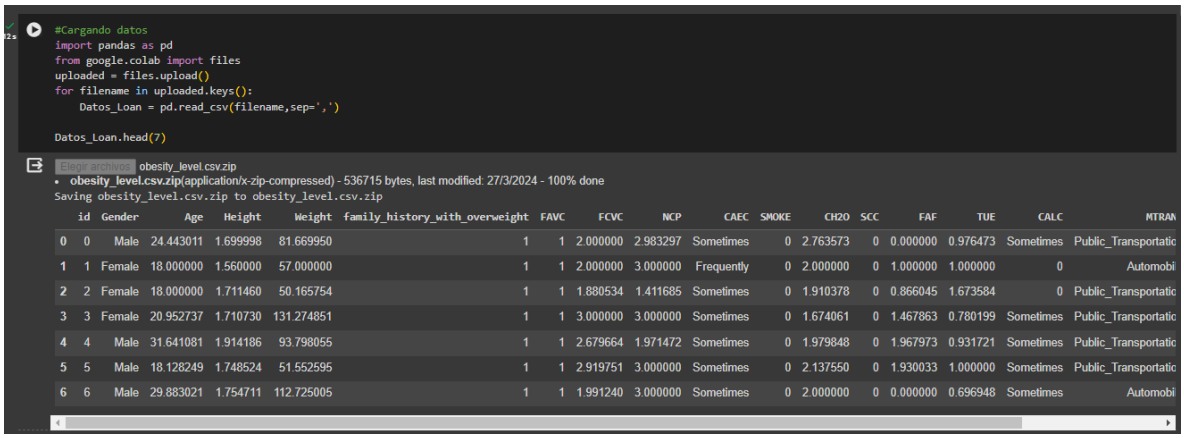
Para obtener una comprensión más detallada y verificar con precisión a qué tipo pertenece cada etiqueta, se importaron los datos y se realizó una inspección inicial de ellos utilizando la plataforma Google Colab. Este análisis permite visualizar las etiquetas y clasificarlas adecuadamente (**Figuras 3 y 4**).

Figura 3. Código para la importación de los datos en Colab.

Fuente: Elaboración propia.

Figura 4. Código para la visualización de los tipos de datos correspondientes a las etiquetas.

Fuente: Elaboración propia.



```
#Cargando datos
import pandas as pd
from google.colab import files
uploaded = files.upload()
for filename in uploaded.keys():
    Datos_Loan = pd.read_csv(filename,sep=',')

Datos_Loan.head(7)
```

Elegir archivos obesity_level.csv.zip
- **obesity_level.csv.zip**(application/x-zip-compressed) - 536715 bytes, last modified: 27/3/2024 - 100% done

Saving obesity_level.csv.zip to obesity_level.csv.zip

	id	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SCC	FAF	TUE	CALC	MTRAN
0	0	Male	24.443011	1.699998	81.669950	1	1	2.000000	2.983297	Sometimes	0	2.763573	0	0.000000	0.976473	Sometimes	Public_Transportatic
1	1	Female	18.000000	1.560000	57.000000	1	1	2.000000	3.000000	Frequently	0	2.000000	0	1.000000	1.000000	0	Automobi
2	2	Female	18.000000	1.711460	50.165754	1	1	1.880534	1.411685	Sometimes	0	1.910378	0	0.866045	1.673584	0	Public_Transportatic
3	3	Female	20.952737	1.710730	131.274851	1	1	3.000000	3.000000	Sometimes	0	1.674061	0	1.467863	0.780199	Sometimes	Public_Transportatic
4	4	Male	31.641081	1.914186	93.798055	1	1	2.679664	1.971472	Sometimes	0	1.979848	0	1.967973	0.931721	Sometimes	Public_Transportatic
5	5	Male	18.128249	1.748524	51.552595	1	1	2.919751	3.000000	Sometimes	0	2.137550	0	1.930033	1.000000	Sometimes	Public_Transportatic
6	6	Male	29.883021	1.754711	112.725005	1	1	1.991240	3.000000	Sometimes	0	2.000000	0	0.000000	0.696948	Sometimes	Automobi

Figura 3. Código de Importación de los datos en Colab. Fuente: Elaboración propia

```
Datos_Loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20758 entries, 0 to 20757
Data columns (total 18 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              20758 non-null  int64
 1   Gender                          20758 non-null  object
 2   Age                             20758 non-null  float64
 3   Height                          20758 non-null  float64
 4   Weight                          20758 non-null  float64
 5   family_history_with_overweight  20758 non-null  int64
 6   FAVC                            20758 non-null  int64
 7   FCVC                            20758 non-null  float64
 8   NCP                             20758 non-null  float64
 9   CAEC                            20758 non-null  object
 10  SMOKE                           20758 non-null  int64
 11  CH2O                            20758 non-null  float64
 12  SCC                             20758 non-null  int64
 13  FAF                             20758 non-null  float64
 14  TUE                             20758 non-null  float64
 15  CALC                            20758 non-null  object
 16  MTRANS                          20758 non-null  object
 17  0be1dad                         20758 non-null  object
dtypes: float64(8), int64(5), object(5)
memory usage: 2.9+ MB
```

figura 4. Código de visualización del tipo de dato al que corresponden las etiquetas. Fuente: Elaboración propia

Con base en la información presentada en la **Figura 4**, se ha clasificado el conjunto de datos en tres categorías principales según el tipo de datos que contienen.

1. **Datos de tipo entero:**

   Estos datos representan valores numéricos sin decimales y corresponden a las siguientes etiquetas:

   - **Id:** Identificador único.
   - **Historia familiar con sobrepeso:** Indica si existe antecedente familiar de sobrepeso.
   - **FAVC:** Consumo frecuente de alimentos ricos en calorías.
   - **SMOKE:** Indica si la persona fuma o no.
   - **SCC:** Consumo de bebidas calóricas.

2. **Datos de tipo decimal:**

   Estas etiquetas contienen valores numéricos con decimales, relacionados con mediciones o promedios:

   - **Edad:** Edad del participante.
   - **Altura:** Altura en metros.
   - **Peso:** Peso en kilogramos.
   - **FCVC:** Frecuencia de consumo de vegetales.
   - **NCP:** Número de comidas principales diarias.

- **CH2O:** Consumo diario de agua en litros.
- **FAF:** Frecuencia de actividad física.
- **TUE:** Tiempo de uso de dispositivos tecnológicos, medido en horas.

3. **Datos de tipo carácter:**

   Este grupo incluye etiquetas con valores de texto o categorías:

   - **Género:** Masculino o femenino.
   - **CAEC:** Consumo de alimentos entre comidas.
   - **CALC:** Consumo de alcohol.
   - **MTRANS:** Modo de transporte utilizado.
   - **0be1dad:** Variable objetivo que representa el nivel de obesidad.

Posibles aplicaciones.

En el presente análisis, centrado en los datos relacionados con los riesgos de obesidad en personas jóvenes y adultos jóvenes, se propone una clasificación simplificada que pueda tener aplicaciones prácticas en la sociedad, específicamente en el ámbito de la salud.

Por un lado, esta clasificación permitirá a los usuarios, especialmente aquellos con dudas sobre su estado físico, identificar de manera preliminar en qué categoría de obesidad podrían encontrarse, sin necesidad de acudir inicialmente a un centro médico. El objetivo principal es ofrecer información clara y sugerir la importancia de consultar a un profesional de la salud si su estado físico se encuentra en una categoría de riesgo.

Por otro lado, este enfoque también puede tener un impacto significativo en el campo de la salud desde una perspectiva técnica. La implementación de modelos de *machine learning* podría facilitar a médicos y especialistas la identificación del estado físico de una persona durante un chequeo médico, agilizando los tiempos de diagnóstico y proporcionando respuestas rápidas y precisas para los pacientes. Esto no solo optimiza los procesos clínicos, sino que también mejora la calidad del servicio y promueve la atención preventiva.

Aproximaciones con gráficos.

Considerando la variedad de información contenida en la base de datos relacionada con los riesgos de obesidad, se ha generado un gráfico para destacar los valores más relevantes. Este gráfico tiene como objetivo proporcionar una visión general de los datos que se analizarán, resaltando su importancia dentro del contexto del estudio.

En primer lugar, el código presentado en la **Figura 4** muestra los niveles de obesidad registrados en la columna **"0be1dad"**. Esta columna constituye la base central del análisis, ya que representa el tema principal del *dataframe* y es fundamental para la clasificación y comprensión de los riesgos asociados.

```python
#Cargando librerías
import seaborn as sns
import matplotlib.pyplot as plt

# Crear un gráfico de torta
frecuencias = Datos_Loan['0be1dad'].value_counts()
plt.figure(figsize=(8, 8))  # Tamaño del gráfico (opcional)
plt.pie(frecuencias, labels=frecuencias.index, autopct='%1.1f%%', startangle=140)
plt.title('Diagrama de Torta de Distribución de clasificacion de obesidad')
plt.show()
```

Figura 5. ***Código para la creación de un diagrama de torta que clasifica los niveles de obesidad. Fuente: Elaboración propia.***

.

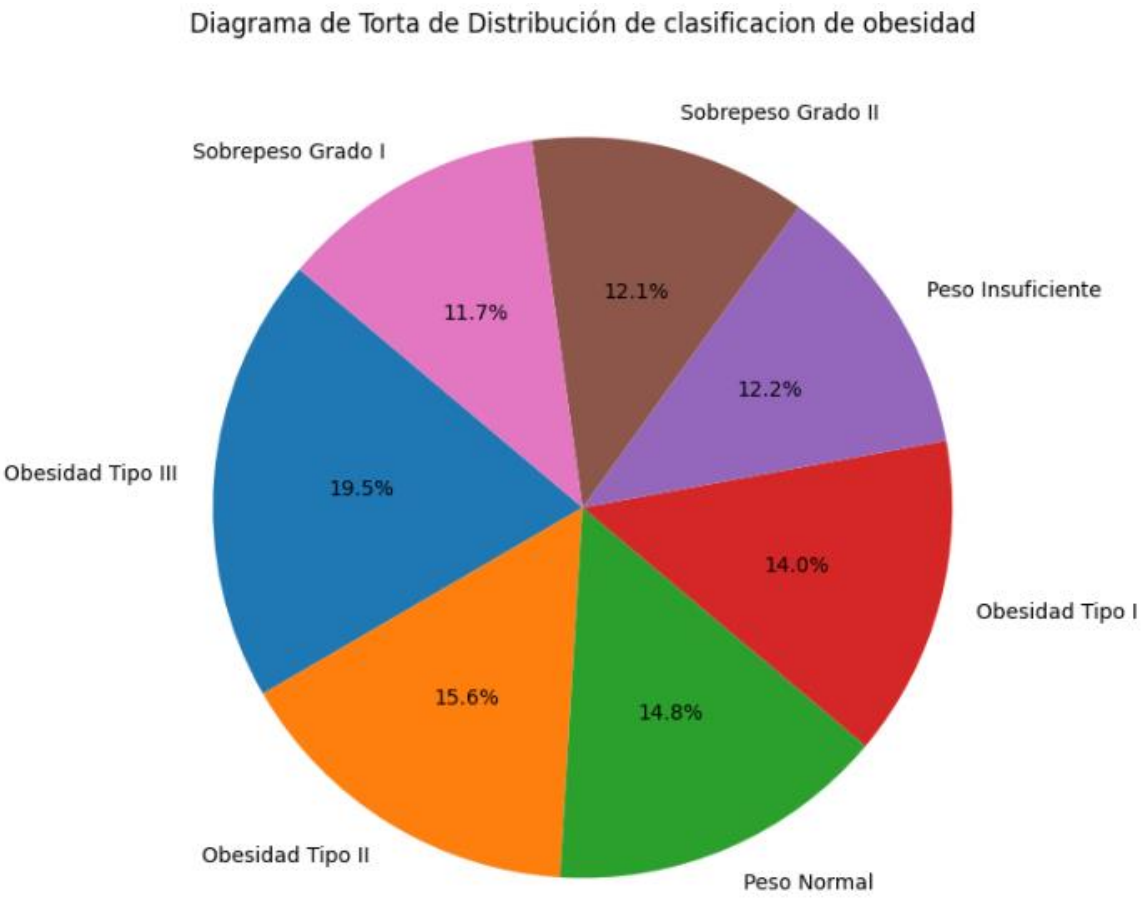


Figura 6. Diagrama de torta de distribución de clasificación de obesidad. Fuente: Elaboración propia.

- Con base en los resultados presentados en la **Figura 5**, se observa que el 19.5% de las personas registradas en la base de datos presentan obesidad tipo III, lo que representa el mayor porcentaje en comparación con las demás clasificaciones. Por otro lado, las categorías de obesidad tipo I, obesidad tipo II y peso normal muestran diferencias menos significativas entre sí. De manera similar, las clasificaciones de peso insuficiente, sobrepeso grado I y sobrepeso grado II presentan proporciones relativamente menores dentro del conjunto de datos.

- A continuación, la **Figura 7** muestra el código utilizado para importar los datos de la columna **"Gender"** del *dataframe*. Este análisis permite visualizar la distribución de los valores según género, representando en el eje X las categorías de género y en el eje Y la frecuencia correspondiente de cada una.

```
#Cargando librerías
import seaborn as sns
import matplotlib.pyplot as plt

# Crear un gráfico de barras con Seaborn
sns.countplot(data=Datos_Loan, x='Gender')
plt.title('Distribución del género')
plt.xlabel('Genero')
plt.ylabel('Frecuencia')
plt.show()
```

Figura 7. Código de grafico de barras de distribución de género. Fuente: Elaboración propia.

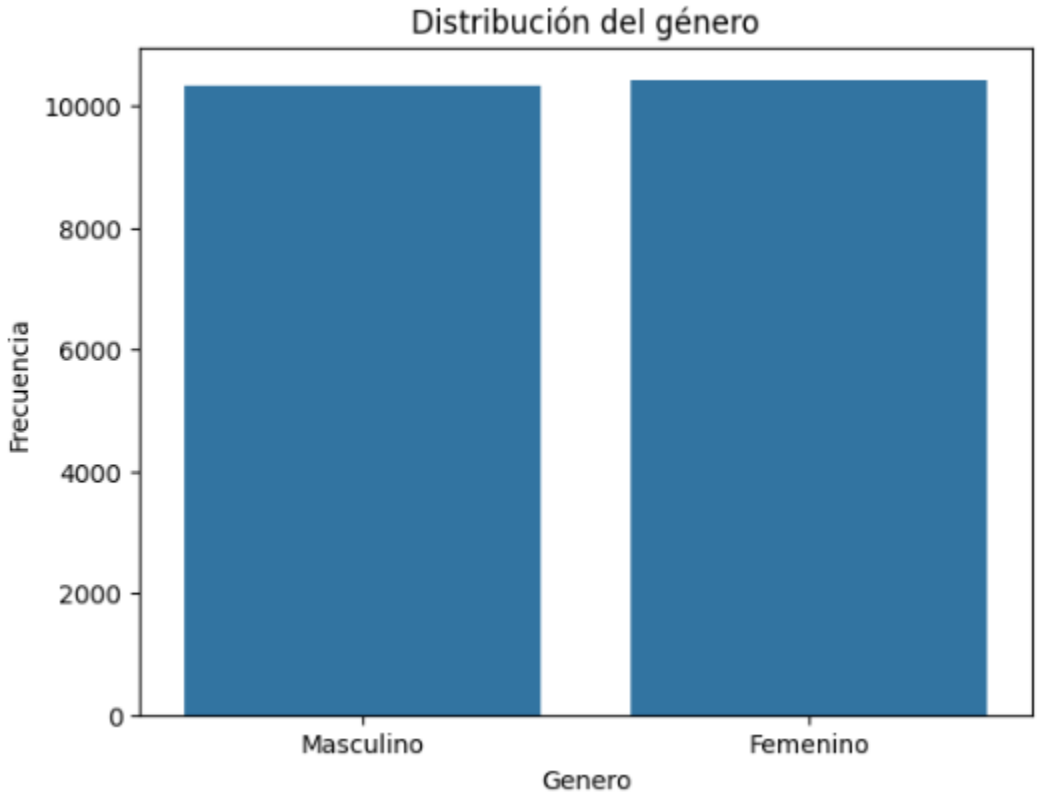


Figura 8. Grafico de barras de la distribución del género. Fuente: Elaboración propia.

De manera similar, en la **Figura 8** se han considerado los datos correspondientes para implementar la **Figura 9**, ya que los datos previos presentaban una disparidad mínima que no era fácilmente perceptible de forma visual. Por esta razón, en la **Figura 9** se muestra una representación más detallada de los porcentajes mediante un gráfico de torta, lo que permite visualizar claramente las áreas correspondientes y las diferencias

porcentuales entre las categorías, como se puede evidenciar a continuación.

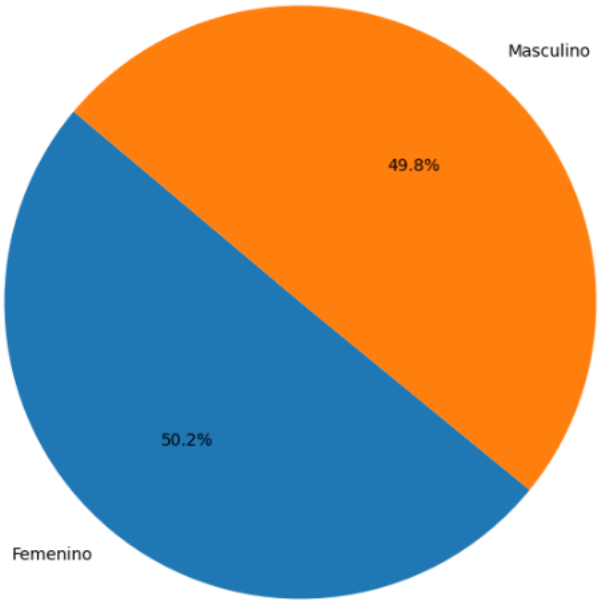


Figura 9. Diagrama de torta de distribución de clasificación de género.

- En el diagrama de torta se puede observar que, a pesar de la mínima disparidad entre los datos de género, el índice más alto corresponde a las personas de sexo femenino, con un 50.2%, en comparación con el sexo masculino, que alcanza un 49.8%.
- Por otro lado, se presentarán figuras que proporcionarán información relevante sobre la posible causa principal que comparten las personas con obesidad, sobrepeso o peso normal.
- Una de las principales causas del sobrepeso en las personas es la falta de actividad física. En este caso, las **Figuras 10 y 11** presentan un mapa de calor que analiza los diferentes medios de transporte utilizados por las personas que integran este *dataframe*. Se ha importado la información de las columnas **"0be1dad"**, que

corresponde a la clasificación de obesidad, y **"MTRANS"**, que indica el medio de transporte utilizado por los individuos en el conjunto de datos.

```
[14] from matplotlib import pyplot as plt
     import seaborn as sns
     import pandas as pd
     plt.subplots(figsize=(8, 8))
     df_2dhist = pd.DataFrame({
         x_label: grp['0be1dad'].value_counts()
         for x_label, grp in _df_23.groupby('MTRANS')
     })
     sns.heatmap(df_2dhist, cmap='viridis')
     plt.xlabel('MTRANS')
     _ = plt.ylabel('0be1dad')
```

Figura 10. Código de mapa de calor de obesidad por medio de transporte. Fuente: Elaboración propia.

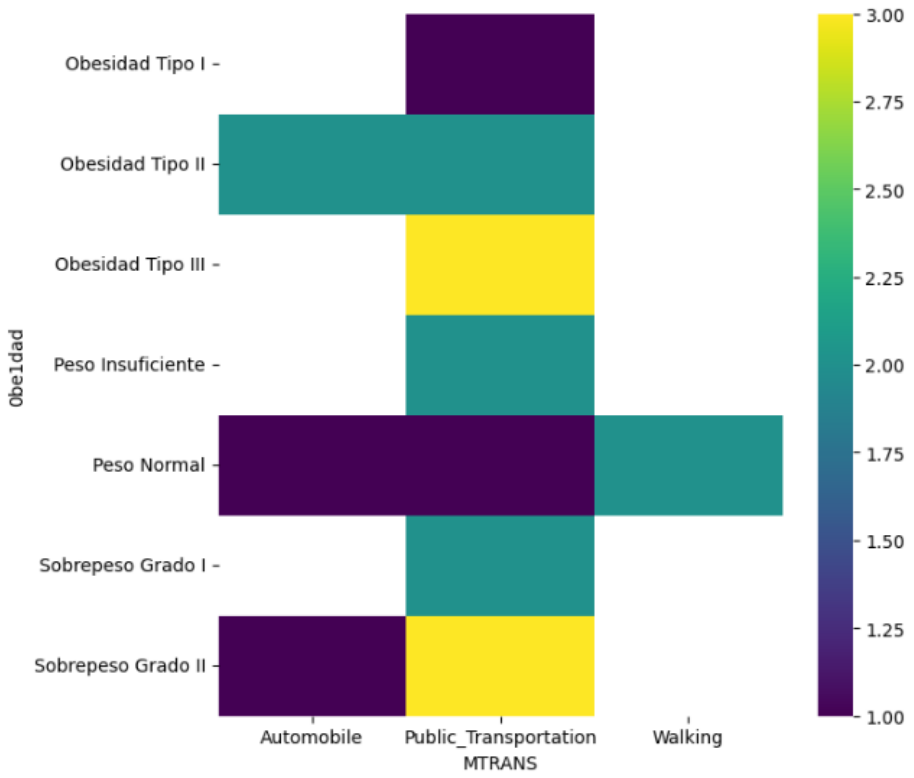


Figura 11. Mapa de calor, obesidad por medio de transporte. Fuente: Elaboración propia.

- Como se puede observar en la **Figura 11**, los colores representan la cantidad de usuarios en cada categoría, utilizando una variación cromática. Los tonos violetas corresponden a las clasificaciones menos seleccionadas, mientras que los tonos amarillos indican las clasificaciones más frecuentes. A partir de este análisis, se puede interpretar que, en general, las personas en todas las categorías de obesidad tienden a utilizar el transporte público como su medio de transporte principal. Además, se observa que los usuarios con sobrepeso grado II y obesidad tipo II son los que más frecuentemente emplean este medio de transporte.

Objetivos:

Objetivo general.

Implementar un algoritmo computacional de aprendizaje supervisado (clasificación) de machine learning. Utilizando datos de riesgos de obesidad para identificar patrones, obtener análisis y generar toma de decisiones.

Objetivos específicos.

- Determinar y ejecutar la base de datos para conocer su información, y clasificar los datos más relevantes.
- Diseñar e Implementar la arquitectura de un algoritmo de aprendizaje supervisado, Junto con la información brindada del dataframe, para entrenar el algoritmo de machine learning.
- Verificar el desempeño del algoritmo de aprendizaje supervisado, para la obtención y precisión de la toma de decisiones.
- Validar el funcionamiento del algoritmo de toma de decisiones a partir de datos nuevos.

Desarrollo e implementación del aprendizaje

El proyecto desarrollado se enmarca en un entorno orientado a la implementación de un algoritmo de **machine learning**, específicamente dentro del ámbito del aprendizaje supervisado mediante el **método de clasificación**. Con base en esto, se integró un *dataframe* que contiene información sobre el riesgo de obesidad, de modo que dicho conjunto de datos pueda ser procesado por el algoritmo para aprender de ellos y generar

predicciones, o en este caso, diagnósticos. Para lograr mejores resultados, se utilizaron una serie de modelos previamente entrenados, los cuales permitieron obtener predicciones precisas. Se seleccionaron cuatro modelos para este propósito, a saber:

1. **KNN (K-Nearest Neighbor)**: Este clasificador es ampliamente utilizado en ejercicios de clasificación debido a su capacidad para ofrecer predicciones rápidas. Su técnica se basa en comparar un nuevo dato con los datos previamente entrenados, seleccionando aquellos más cercanos en términos de similitud, lo que le permite proporcionar un resultado fundamentado en esta comparación [8].
2. **Bayes**: Este método de aprendizaje se centra en calcular la probabilidad y la similitud de un nuevo dato en relación con las clases ya entrenadas. A partir de estas probabilidades, el algoritmo decide en qué categoría encaja mejor el nuevo dato, tomando en cuenta sus características o similitudes más destacadas [9].
3. **LDA (Linear Discriminant Analysis)**: Este modelo busca encontrar las dimensiones más relevantes en las que los datos se presentan, realizando una combinación lineal de las características que se asemejan entre sí. De esta forma, cuando se ingresa un nuevo dato, se optimiza su clasificación al agruparlo en el grupo más adecuado [10].
4. **SVM (Support Vector Machine)**: Según las referencias consultadas [11], este modelo se utiliza para dividir los datos en dos grupos a través de una función lineal. Su objetivo es encontrar la línea que mejor separe ambos grupos, maximizando las características relevantes para lograr predicciones precisas.

Gracias a estos modelos, se diseñará un conjunto de preguntas clave, que reflejan las principales causas registradas en el *dataframe* y que son comunes entre los usuarios que proporcionaron esta información. Estas preguntas constituirán el proceso mediante el cual los usuarios ingresarán sus datos al algoritmo para finalmente obtener su diagnóstico.

Procesamiento de los datos

Importar datos 1.1.

Como primer paso, se importan los datos al algoritmo para su visualización. Se carga el archivo en formato CSV previamente descargado y almacenado en el equipo. Una vez cargados, los datos se mostrarán en una tabla organizada por columnas (Figura 12), donde se podrán observar las variables que lo componen. Estas variables incluyen las posibles causas de obesidad, así como el género, la edad y los niveles de obesidad de los individuos.

```
#Cargando datos
import pandas as pd
from google.colab import files
uploaded = files.upload()
for filename in uploaded.keys():
    Datos_Loan = pd.read_csv(filename,sep=',')

Datos_Loan.head(7)
```

```
Elegir archivos  obesity_level.csv.zip
• obesity_level.csv.zip(application/x-zip-compressed) - 536715 bytes, last modified: 27/3/2024 - 100% done
Saving obesity_level.csv.zip to obesity_level.csv.zip
```

	id	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	
0	0	Male	24.443011	1.699998	81.669950	1	1	2.000000	2.983297	Sometimes	0	2.76
1	1	Female	18.000000	1.560000	57.000000	1	1	2.000000	3.000000	Frequently	0	2.00
2	2	Female	18.000000	1.711460	50.165754	1	1	1.880534	1.411685	Sometimes	0	1.91
3	3	Female	20.952737	1.710730	131.274851	1	1	3.000000	3.000000	Sometimes	0	1.67
4	4	Male	31.641081	1.914186	93.798055	1	1	2.679664	1.971472	Sometimes	0	1.97
5	5	Male	18.128249	1.748524	51.552595	1	1	2.919751	3.000000	Sometimes	0	2.13
6	6	Male	29.883021	1.754711	112.725005	1	1	1.991240	3.000000	Sometimes	0	2.00

figura 12. Importación de los datos. Fuente: Elaboración propia.

Conocer los Datos 2.1

Como segundo paso, se utiliza la función Datos_Loan.info() (Figura 13), que proporciona una visión más clara de los datos que se están manejando. Esta función permite obtener información detallada sobre la cantidad de datos y las características más precisas de los mismos. Además, facilita la verificación del tipo de cada variable, ya sea de tipo carácter, tipo float o tipo entero.

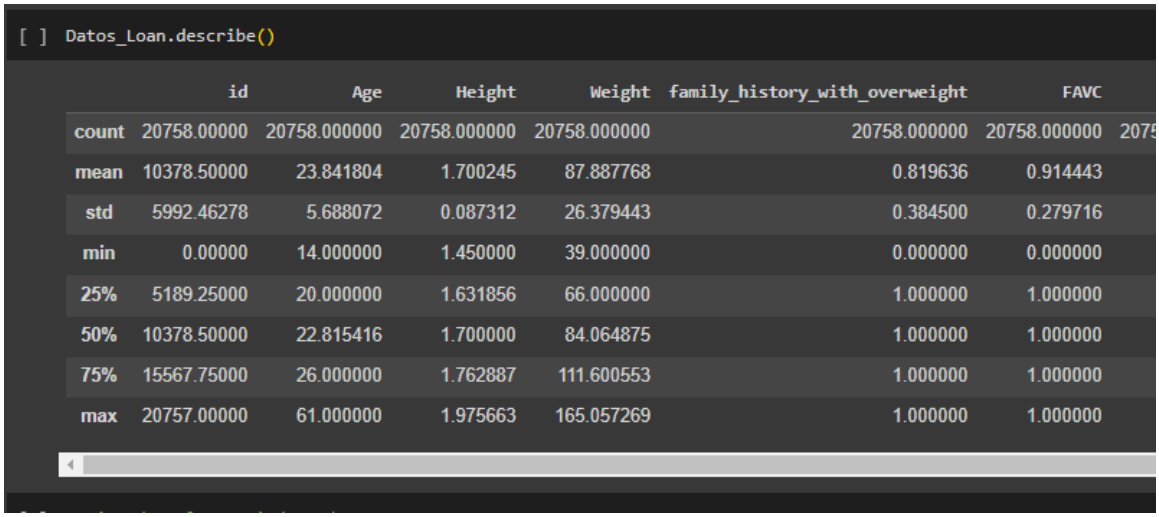
```
Datos_Loan.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20758 entries, 0 to 20757
Data columns (total 18 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              20758 non-null  int64
 1   Gender                          20758 non-null  object
 2   Age                             20758 non-null  float64
 3   Height                          20758 non-null  float64
 4   Weight                          20758 non-null  float64
 5   family_history_with_overweight  20758 non-null  int64
 6   FAVC                            20758 non-null  int64
 7   FCVC                            20758 non-null  float64
 8   NCP                             20758 non-null  float64
 9   CAEC                            20758 non-null  object
 10  SMOKE                           20758 non-null  int64
 11  CH2O                            20758 non-null  float64
 12  SCC                             20758 non-null  int64
 13  FAF                             20758 non-null  float64
 14  TUE                             20758 non-null  float64
 15  CALC                            20758 non-null  object
 16  MTRANS                          20758 non-null  object
 17  0be1dad                         20758 non-null  object
dtypes: float64(8), int64(5), object(5)
memory usage: 2.9+ MB

✓ Conectado a
```

Figura 13. Información de los datos. Fuente: Elaboración propia.

En la (Figura 13) podemos observar de manera resumida que el conjunto de datos está compuesto por 17 etiquetas, cada una con un total de 20,758 registros. De estas etiquetas, 8 son de tipo float, 5 de tipo int y 5 de tipo object.

Descripción de los datos 3.1

A continuación, importamos la función **Datos_Loan.describe()** (Figura 14), que proporciona un resumen estadístico de los datos. Esta función muestra información clave como los valores mínimos, máximos, la media, los percentiles y la desviación estándar. La tabla resultante nos permite verificar si los datos están correctamente estructurados y sin errores. Además, esta herramienta es útil para realizar un análisis más numérico o estadístico del algoritmo, permitiendo una visión detallada de la distribución de los datos.



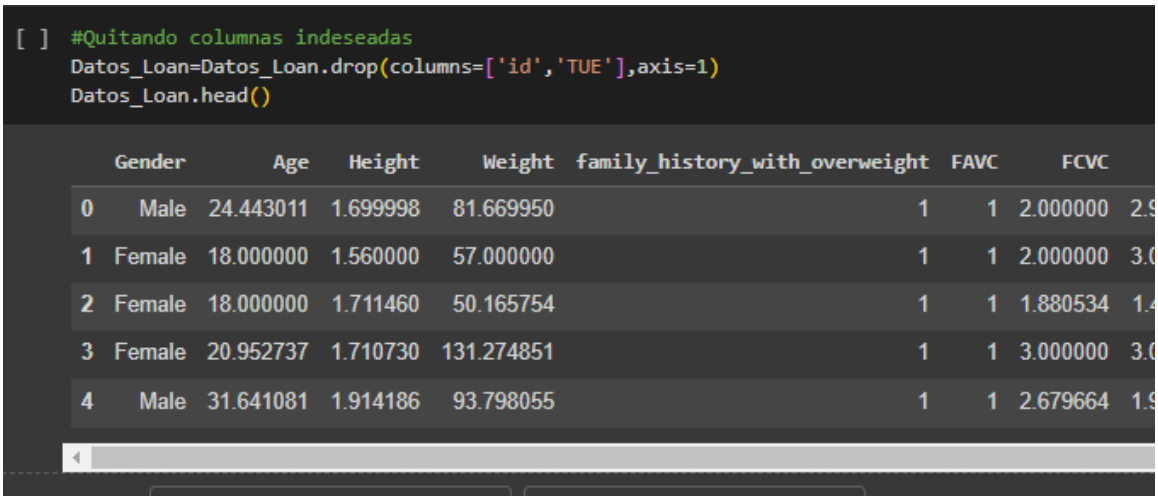
```
[ ] Datos_Loan.describe()
```

	id	Age	Height	Weight	family_history_with_overweight	FAVC	
count	20758.00000	20758.000000	20758.000000	20758.000000	20758.000000	20758.000000	2075
mean	10378.50000	23.841804	1.700245	87.887768	0.819636	0.914443	
std	5992.46278	5.688072	0.087312	26.379443	0.384500	0.279716	
min	0.00000	14.000000	1.450000	39.000000	0.000000	0.000000	
25%	5189.25000	20.000000	1.631856	66.000000	1.000000	1.000000	
50%	10378.50000	22.815416	1.700000	84.064875	1.000000	1.000000	
75%	15567.75000	26.000000	1.762887	111.600553	1.000000	1.000000	
max	20757.00000	61.000000	1.975663	165.057269	1.000000	1.000000	

Figura 14. Descripción estadística de los datos. Fuente: elaboración propia.

Eliminación de datos no deseadas 4.1

El siguiente paso consiste en eliminar las columnas no deseadas (Figura 15). Esta acción se realiza cuando se identifica que ciertos datos no son relevantes o no aportan al objetivo principal del análisis. En este caso, hemos decidido eliminar dos columnas: **id** y **TUE** (Tiempo de uso de dispositivos tecnológicos), ya que no contribuyen directamente al enfoque de nuestro análisis, que se centra en determinar los niveles de obesidad de una persona a partir de los datos que ingrese.



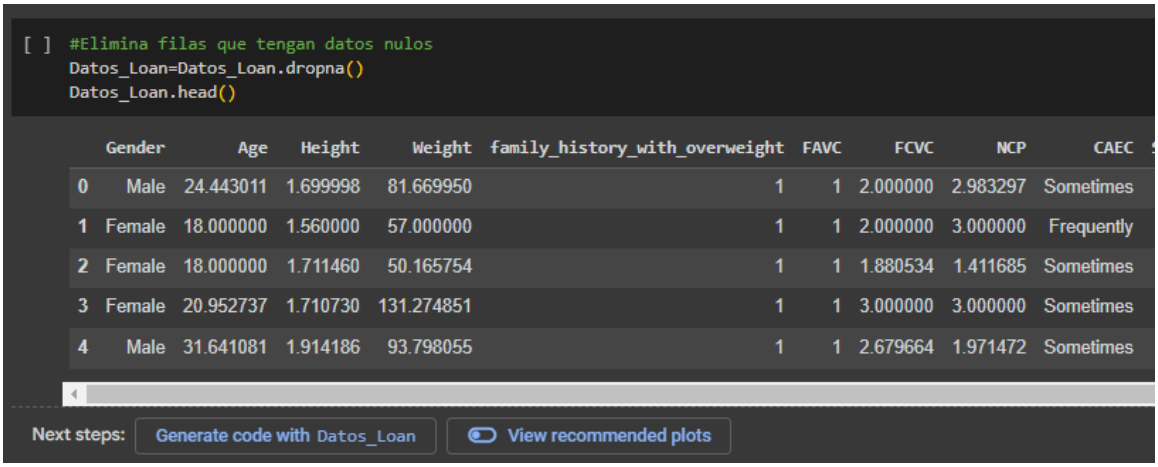
```
[ ]  #Quitando columnas indeseadas
     Datos_Loan=Datos_Loan.drop(columns=['id','TUE'],axis=1)
     Datos_Loan.head()
```

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	
0	Male	24.443011	1.699998	81.669950	1	1	2.000000	2.9
1	Female	18.000000	1.560000	57.000000	1	1	2.000000	3.0
2	Female	18.000000	1.711460	50.165754	1	1	1.880534	1.4
3	Female	20.952737	1.710730	131.274851	1	1	3.000000	3.0
4	Male	31.641081	1.914186	93.798055	1	1	2.679664	1.9

Figura 15. Eliminación de columnas indeseadas. Fuente: Elaboración propia.

Eliminación de datos nulos 5.1

Al analizar los datos, es fundamental asegurarse de que no existan valores nulos. Dado que se empleará aprendizaje automatizado, los modelos de clasificación utilizarán la información proporcionada para hacer sus predicciones. Es crucial mantener valores completos y correctos, ya que la presencia de datos faltantes podría afectar la precisión de las decisiones que el modelo tome, resultando en predicciones erróneas (Figura 16).



```
[ ]  #Elimina filas que tengan datos nulos
     Datos_Loan=Datos_Loan.dropna()
     Datos_Loan.head()
```

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC
0	Male	24.443011	1.699998	81.669950	1	1	2.000000	2.983297	Sometimes
1	Female	18.000000	1.560000	57.000000	1	1	2.000000	3.000000	Frequently
2	Female	18.000000	1.711460	50.165754	1	1	1.880534	1.411685	Sometimes
3	Female	20.952737	1.710730	131.274851	1	1	3.000000	3.000000	Sometimes
4	Male	31.641081	1.914186	93.798055	1	1	2.679664	1.971472	Sometimes

Next steps: Generate code with Datos_Loan View recommended plots

Figura 16. Eliminación de filas con valores nulos. Fuente: Elaboración Propia.

Análisis de niveles de estudio 6.1

Aquí tienes el texto mejorado:

Este paso es esencial para el desarrollo de este análisis. En primer lugar, se realiza la declaración de los datos que se desean analizar. Cada columna contiene varios datos, como se mencionó en secciones anteriores, y estos datos han sido proporcionados por diferentes usuarios (Figuras 17, 18, 19, 20). Para contextualizar, tomemos como ejemplo la columna "Género" (Figura 17). Esta columna ha sido seleccionada para su análisis y, por tanto, proporcionará la información correspondiente. En este caso, solo tenemos dos posibles valores: "Female" (femenino) y "Male" (masculino). De manera similar, los análisis posteriores arrojarán datos para cada categoría. En el caso de los datos numéricos, se busca establecer sus límites, ya que los modelos de aprendizaje automático se entrenarán con estos valores. Si posteriormente se encuentran valores fuera de esos límites, podrían surgir dificultades en las predicciones.

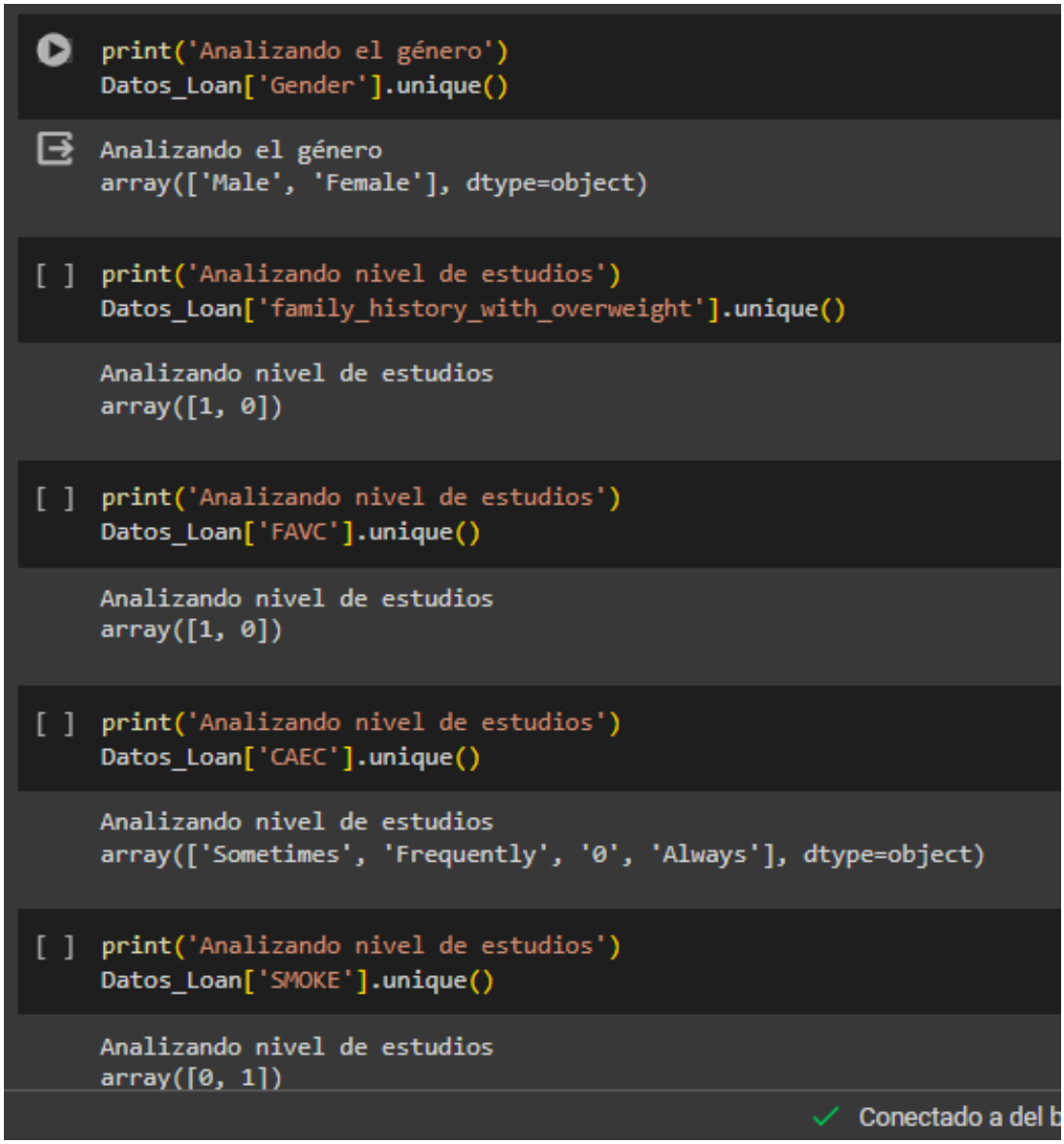


Figura 17. Análisis de niveles de estudio 1. Fuente: Elaboración propia.

```
print('Analizando nivel de estudios')
Datos_Loan['SCC'].unique()

Analizando nivel de estudios
array([0, 1])

[ ] print('Analizando nivel de estudios')
    Datos_Loan['FAF'].unique()

    Analizando nivel de estudios
    array([0.      , 1.      , 0.866045, ..., 0.540397, 0.271174, 0.988668])

[ ] print('Analizando nivel de estudios')
    Datos_Loan['CALC'].unique()

    Analizando nivel de estudios
    array(['Sometimes', '0', 'Frequently'], dtype=object)

[ ] print('Analizando nivel de estudios')
    Datos_Loan['MTRANS'].unique()

    Analizando nivel de estudios
    array(['Public_Transportation', 'Automobile', 'Walking', 'Motorbike',
           'Bike'], dtype=object)
```

Figura 18. Análisis de niveles de estudio 2. Fuente: Elaboración propia.

```
[ ] print('Analizando nivel de estudios')
    Datos_Loan['0be1dad'].unique()

    Analizando nivel de estudios
    array(['Overweight_Level_II', '0rmal_Weight', 'Insufficient_Weight',
           'Obesity_Type_III', 'Obesity_Type_II', 'Overweight_Level_I',
           'Obesity_Type_I'], dtype=object)

print('Analizando nivel de estudios')
Datos_Loan['FCVC'].unique()

Analizando nivel de estudios
array([2.        , 1.880534  , 3.        , 2.679664  , 2.919751  ,
       1.99124   , 1.397468  , 2.636719  , 1.        , 1.392665  ,
       2.203962  , 2.971588  , 2.668949  , 1.98989905, 2.417635  ,
       2.219186  , 2.919526  , 2.263245  , 2.649406  , 1.754401  ,
       2.303656  , 2.020785  , 2.068834  , 2.689929  , 2.979383  ,
       2.225731  , 2.843456  , 2.312528  , 2.962415  , 2.945967
```

Figura 19. Análisis de niveles de estudio 3. Fuente: Elaboración propia.

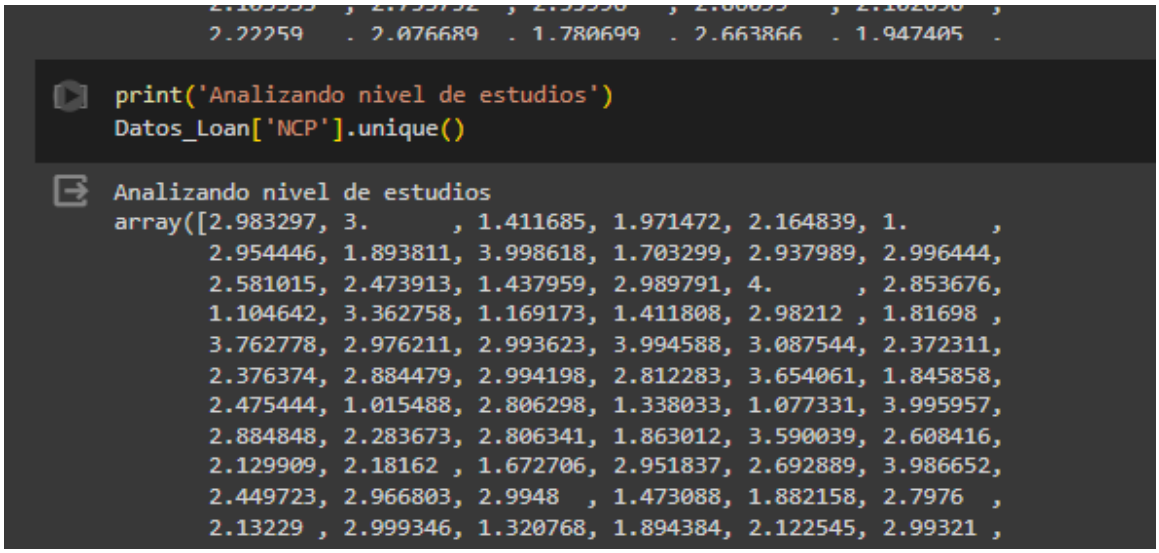


Figura 20. Análisis de niveles de estudio 4. Fuente: Elaboración propia.

Conversión de datos a números 7.1

Cuando trabajamos con predicciones, es fundamental contar con valores numéricos. Por esta razón, se han seleccionado ciertos datos que originalmente eran de tipo "objeto" y se han convertido a formato "entero". De este modo, se han realizado cinco modificaciones clave: género, obesidad, consumo de alcohol y medio de transporte (Figura 21). Estos datos han sido transformados en números, cuyas asignaciones han sido decididas de forma deliberada. Este proceso facilita tanto la realización de las predicciones finales como la toma de decisiones dentro de cada modelo, optimizando el análisis y mejorando

la precisión de los resultados.

```
#Convierte datos a números
Reemplazo_1={'Male':1,'Female':2}
Datos_Loan['Gender']=Datos_Loan['Gender'].map(Reemplazo_1)

Reemplazo_2= {'Obesity_Type_I': 3,'Obesity_Type_II': 4,'Obesity_Type_III': 5,'Overweight_Level_I': 1 ,'Overweight_Level_II': 2,'0rmal_Weight': 0,'Insufficient_Weight': 6}
Datos_Loan['0be1dad']=Datos_Loan['0be1dad'].map(Reemplazo_2)

Reemplazo_3={'Sometimes':1 , '0':0, 'Frequently':2}
Datos_Loan['CALC']=Datos_Loan['CALC'].map(Reemplazo_3)

Reemplazo_4={'Sometimes':1, 'Frequently':2, '0':0, 'Always':3}
Datos_Loan['CAEC']=Datos_Loan['CAEC'].map(Reemplazo_4)

Reemplazo_5={'Public_Transportation':1, 'Automobile':2, 'Walking':3, 'Motorbike':4,'Bike':5}
Datos_Loan['MTRANS']=Datos_Loan['MTRANS'].map(Reemplazo_5)

Datos_Loan.head(15)
```

Figura 21. conversión de datos a números. Fuente: Elaboración propia.

En comparación con la tabla presentada inicialmente, se puede observar en la Figura 22 cómo ha quedado la tabla después de haber realizado todos los cambios previamente mencionados.

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SCC	FAF	CALC	MTRANS	0be1dad
0	1	24.443011	1.699998	81.669950	1	1	2.000000	2.983297	1	0	2.763573	0	0.000000	1	1	2
1	2	18.000000	1.560000	57.000000	1	1	2.000000	3.000000	2	0	2.000000	0	1.000000	0	2	0
2	2	18.000000	1.711460	50.165754	1	1	1.880534	1.411685	1	0	1.910378	0	0.866045	0	1	6
3	2	20.952737	1.710730	131.274851	1	1	3.000000	3.000000	1	0	1.674061	0	1.467863	1	1	5
4	1	31.641081	1.914186	93.798055	1	1	2.679664	1.971472	1	0	1.979848	0	1.967973	1	1	2
5	1	18.128249	1.748524	51.552595	1	1	2.919751	3.000000	1	0	2.137550	0	1.930033	1	1	6
6	1	29.883021	1.754711	112.725005	1	1	1.991240	3.000000	1	0	2.000000	0	0.000000	1	2	4
7	1	29.891473	1.750150	118.206565	1	1	1.397468	3.000000	1	0	2.000000	0	0.598655	1	2	4
8	1	17.000000	1.700000	70.000000	0	1	2.000000	3.000000	1	0	3.000000	1	1.000000	0	1	1
9	2	26.000000	1.638836	111.275646	1	1	3.000000	3.000000	1	0	2.632253	0	0.000000	1	1	5
10	2	20.000000	1.650000	65.000000	1	1	3.000000	3.000000	1	0	3.000000	0	1.000000	1	1	1
11	1	22.000000	1.700000	70.000000	1	0	2.000000	3.000000	0	0	2.000000	0	2.000000	0	3	0
12	1	18.000000	1.811189	108.251044	1	1	2.000000	2.164839	1	0	2.530157	0	1.000000	0	1	3
13	2	21.412538	1.729045	131.529267	1	1	3.000000	3.000000	1	0	1.959531	0	1.425712	1	1	5
14	2	20.000000	1.570000	49.000000	0	0	2.000000	1.000000	1	0	1.000000	0	3.000000	0	3	0

Figura 21. Tabla final. Fuente: Elaboración propia.

Modelo de toma de decisiones

En los siguientes pasos, se detallarán los procesos que deben tenerse en cuenta para entrenar los modelos, con el objetivo de lograr una toma de decisiones precisa y adecuada.

División de entradas y salidas 1.1

En este paso, es fundamental comprender los rangos o longitudes de los datos con los que se trabajará. Por esta razón, se ha decidido que, para los valores de entrada, se utilizarán 15 datos, los cuales serán proporcionados por el usuario para obtener su diagnóstico. En cuanto a los datos de salida, se ha seleccionado únicamente 1 dato, ya que el modelo generará una única respuesta basada en su predicción a partir de los datos ingresados (Figura 23).

```
import numpy as np
Datos_matriz=np.array(Datos_Loan)

X = Datos_matriz[:,0:15] #datos de entrada (Todas las variables del cliente)
Y = Datos_matriz[:,-1] #Datos de salida (La decisiòn del nivel de obesidad)
```

Figura 23. División en entradas y salidas. Fuente: Elaboración propia.

División de datos de entrenamiento y validación 2.1

La división de datos se realiza de la manera indicada en la Figura 24, con el propósito de minimizar cualquier inconveniente que pueda surgir durante el entrenamiento del algoritmo o en el momento de la toma de decisiones. Esta función cumple un papel crucial, ya que asigna de manera precisa los datos correspondientes al algoritmo, asegurando que este aprenda y procese la información de forma adecuada.

```
import sklearn
from sklearn.model_selection import train_test_split
X_train, X_test,Y_train, Y_test= train_test_split(X,Y,test_size=0.1,random_state=751)
```

Figura 24. División de datos de entrenamiento y validación. Fuente: Elaboración propia.

```
#Para mejorar la escala de los datos se hace normalization (Ignorar)
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figura 25. Normalización de datos. Fuente: Elaboración propia

Además, se realiza un proceso de normalización de los datos, cuyo objetivo es ajustar y corregir valores nulos o duplicados, como se muestra en la Figura 25. Este paso es esencial para garantizar que los modelos de predicción trabajen con características claras y precisas, permitiendo así un mejor desempeño en el desarrollo de sus predicciones.

Evaluación de casos mediante todos los modelos de predicciones 3.1

De esta forma se entrenan los modelos seleccionados (Figura 26), los cuales permitirán tomar decisiones según su capacidad para clasificar los datos de manera precisa. Para iniciar, en la parte superior del código se realiza el llamado de los cuatro modelos previamente definidos: KNN, Bayes, LDA y SVM. Estos modelos han sido configurados para analizar y clasificar la información, siendo los responsables de proporcionar un diagnóstico adecuado cuando una nueva persona ingrese sus datos en el

algoritmo.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score,precision_score

Modelo_0 = KNeighborsClassifier(5)
Modelo_0.fit(X_train, Y_train)
Y_pred_0 =Modelo_0.predict (X_test)
print("Accuracy KNN",accuracy_score(Y_test, Y_pred_0))

Modelo_1 = GaussianNB()
Modelo_1.fit(X_train, Y_train)
Y_pred =Modelo_1.predict (X_test)
print("Accuracy Bayes",accuracy_score(Y_test, Y_pred))

Modelo_2 = LinearDiscriminantAnalysis()
Modelo_2.fit(X_train, Y_train)
Y_pred_2 =Modelo_2.predict (X_test)
print("Accuracy LDA",accuracy_score(Y_test, Y_pred_2))

Modelo_3 = SVC()
Modelo_3.fit(X_train, Y_train)
Y_pred_3 =Modelo_3.predict (X_test)
print("Accuracy SVM",accuracy_score(Y_test, Y_pred_3))
```

Figura 26. Evaluando casos mediante todos los clasificadores. Fuente: Elaboración propia.

```
Accuracy KNN 0.7288053949903661
Accuracy Bayes 0.6339113680154143
Accuracy LDA 0.8053949903660886
Accuracy SVM 0.8516377649325626
```

Figura 27. Resultados de los clasificadores. Fuente: Elaboración propia.

En la Figura 27 se observa cómo los clasificadores presentan sus resultados en términos de rangos de probabilidad. Estos valores, que fluctúan entre 0 y 1, reflejan el nivel de confianza que los usuarios pueden depositar en los modelos.

Al realizar un análisis preliminar, se puede concluir que los modelos presentan un alto grado de confiabilidad, ya que sus valores están muy cercanos a 1, lo que evidencia su precisión en las predicciones realizadas.

Probando los modelos entrenados 4.1

Para probar los modelos entrenados, es esencial implementar un conjunto de preguntas relevantes que permitan llevar a cabo el análisis y obtener predicciones precisas (Figura 28). En este caso, se plantearon 15 preguntas enfocadas en identificar si los hábitos del usuario que ingresa sus datos son similares a los registrados en el *dataframe*. Esto permite que el modelo realice una predicción precisa y determine el nivel de obesidad en el que se encuentra el usuario.

Los datos ingresados son sometidos a un proceso de normalización (Figura 29), con el objetivo de interpretarlos de manera más efectiva, optimizar el procesamiento y garantizar la obtención de respuestas precisas y confiables.

```
#Probando el modelo entrenado sobre un nuevo sujeto
Target=np.zeros((1,15))
Target[0,0]=float(input('Ingrese género, 1 para Masculino y 2 para Femenino: '))
Target[0,1]=float(input('Ingrese su edad: '))
Target[0,2]=float(input('Ingrese su atura: '))
Target[0,3]=float(input('Ingrese su peso: '))
Target[0,4]=float(input('¿tiene familiares con obesidad?, 1 para si y 0 para no: '))
Target[0,5]=float(input('¿Consumo frecuente de alimentos ricos en calorías?, 1 para si y 0 para no: '))
Target[0,6]=float(input('Consumos frecuentes de vegetales, entre 1 y 3: '))
Target[0,7]=float(input('Cuantos alimentos principales consume, entre 1 y 3: '))
Target[0,8]=float(input('¿consume alimentos entre comidad?, 0 para no, 1 para aveces, 2 frecuentemente y 3 siempre: '))
Target[0,9]=float(input('¿Fuma?, 0 para no, 1 para si: '))
Target[0,10]=float(input('¿cuantos litros toma al dia?, entre 1 y 3:  '))
Target[0,11]=float(input('¿Usted toma bebidas caloricas?, 0 para no, 1 si: '))
Target[0,12]=float(input('¿hace actividad fisica con frecuencia?, 0 para no, 1 para si: '))
Target[0,13]=float(input('¿consume bebidad alcoholicas?, 0 para no, 1 para aveces, 2 frecuentemente: '))
Target[0,14]=float(input('¿cual es su medio de traspoerte?, 1 para trasporte publico, 2 auto, 3 camiando, 4 moto, 5 cicla: '))
```

Figura 28. Probando los modelos con nuevos datos. Fuente: elaboración propia.

```
Target = scaler.transform(Target) #Normalizar los datos

Prediction_0 =Modelo_0.predict (Target)
Prediction_1 =Modelo_1.predict (Target)
Prediction_2 =Modelo_2.predict (Target)
Prediction_3 =Modelo_3.predict (Target)
```

Figura 29. Normalización de los datos ingresados. Fuente: elaboración propia.

Imprimir las predicciones 5.1

Como paso final, se imprimen las predicciones generadas por cada modelo, permitiendo que estos identifiquen las similitudes en los datos y realicen sus respectivas predicciones. Este proceso se lleva a cabo de manera individual para cada modelo, mostrando los

niveles de obesidad identificados y facilitando la toma de decisiones. (Figuras 30, 31, 32, 33).

```
print(" ")

if Prediction_0==0:
  print("Según KNN, se encuentra en peso normal")
elif Prediction_0==1:
  print("Según KNN, se encuentra en Sobrepeso nivel I")
elif Prediction_0==2:
  print("Según KNN, se encuentra en Sobrepeso nivel II")
elif Prediction_0==3:
  print("Según KNN, se encuentra en Obesidad tipo I")
elif Prediction_0==4:
  print("Según KNN, se encuentra en Obesidad tipo II")
elif Prediction_0==5:
 print("Según KNN, se encuentra en Obesidad tipo III")
else:
  print("Según KNN, es un Peso insuficiente")

print(" ")
```

Figura 30. Predicción, modelo KNN. Fuente: Elaboración propia

```
if Prediction_1==0:
  print("Según bayes, se encuentra en peso normal")
elif Prediction_1==1:
  print("Según bayes, se encuentra en Sobrepeso nivel I")
elif Prediction_1==2:
  print("Según bayes, se encuentra en Sobrepeso nivel II")
elif Prediction_1==3:
  print("Según bayes, se encuentra en Obesidad tipo I")
elif Prediction_1==4:
  print("Según bayes, se encuentra en Obesidad tipo II")
elif Prediction_1==5:
  print("Según bayes, se encuentra en Obesidad tipo III")
else:
  print("Según bayes, es un Peso insuficiente")

print(" ")
```

Figura 31. Predicción, modelo bayes. Fuente: Elaboración propia

```
if Prediction_2==0:
  print("Según LDA, se encuentra en peso normal")
elif Prediction_2==1:
  print("Según LDA, se encuentra en Sobrepeso nivel I")
elif Prediction_2==2:
  print("Según LDA, se encuentra en Sobrepeso nivel II")
elif Prediction_2==3:
  print("Según LDA, se encuentra en Obesidad tipo I")
elif Prediction_2==4:
  print("Según LDA, se encuentra en Obesidad tipo II")
elif Prediction_2==5:
  print("Según LDA, se encuentra en Obesidad tipo III")
else:
  print("Según LDA, es un Peso insuficiente")

print(" ")
```

Figura 32. Predicción, modelo LDA. Fuente: Elaboración propia

```
if Prediction_5==0:
  print("Según SVM, se encuentra en peso normal")
elif Prediction_5==1:
  print("Según SVM, se encuentra en Sobrepeso nivel I")
elif Prediction_5==2:
  print("Según SVM, se encuentra en Sobrepeso nivel II")
elif Prediction_5==3:
  print("Según SVM, se encuentra en Obesidad tipo I")
elif Prediction_5==4:
  print("Según SVM, se encuentra en Obesidad tipo II")
elif Prediction_5==5:
  print("Según SVM, se encuentra en Obesidad tipo III")
else:
  print("Según SVM, es un Peso insuficiente")

print(" ")
```

Figura 33. Predicción, modelo SVM. Fuente: Elaboración propia

Resultado de las predicciones 6.1

Finalmente, se pueden analizar los datos ingresados por dos usuarios en cada una de las preguntas, junto con las predicciones generadas para ellos. En este caso, uno de los usuarios presenta sobrepeso, mientras que el otro se encuentra dentro de un rango de peso normal. (Figuras 34 y 35).

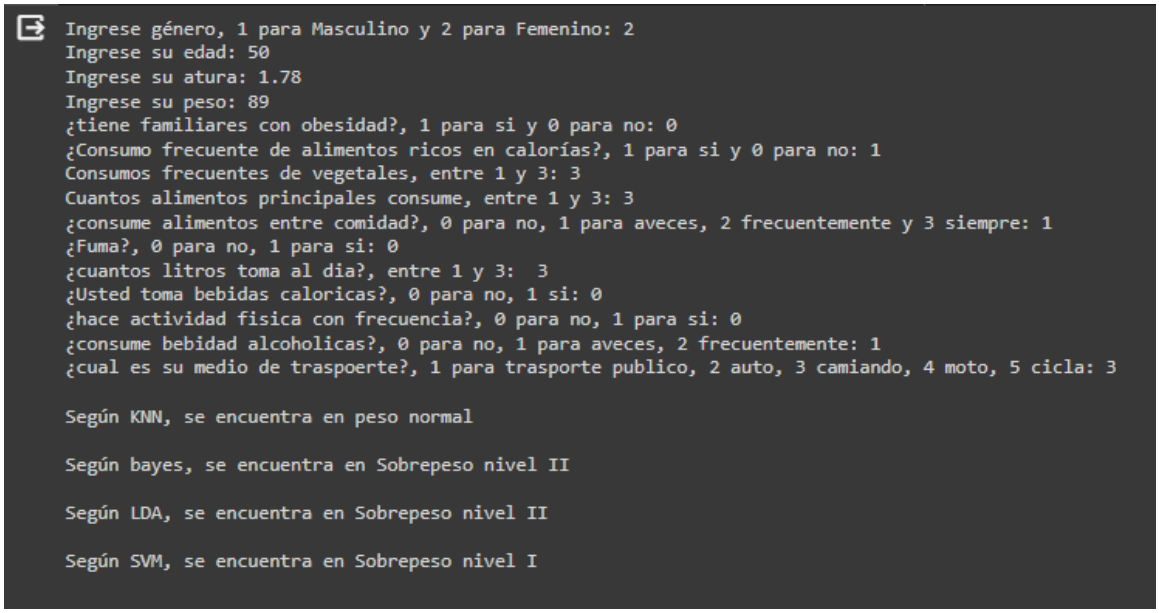


Figura 34. Resultados de las predicciones 1. Fuente: elaboración propia.

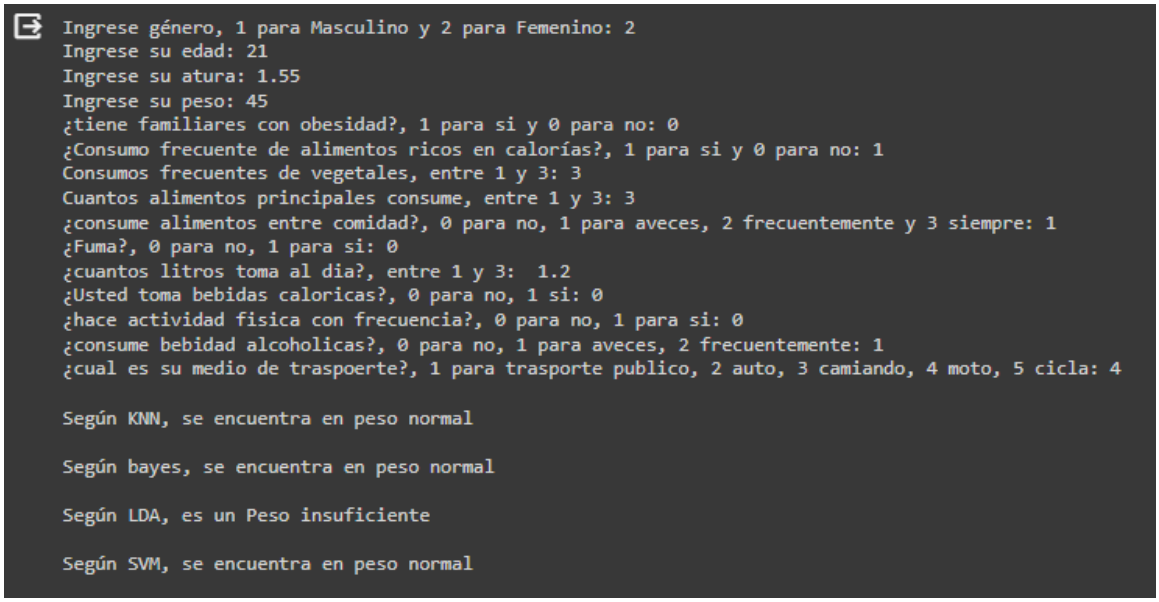


Figura 35. Resultados de las predicciones 2. Fuente: elaboración propia.

Implementación en contextos reales

En el ámbito tecnológico, podemos encontrar algoritmos similares al que hemos desarrollado en esta investigación. Por ejemplo, destaca un estudio sobre un algoritmo de clasificación denominado *Predicción de casos de obesidad infantil* [12], enfocado en realizar predicciones tempranas de obesidad en niños de apenas 3 años.

Otro caso relevante es el de un algoritmo aplicado a la detección de obesidad en adolescentes, titulado *Algoritmos de clasificación para la detección de obesidad en adolescentes: Un estudio comparativo entre KNN y árboles de decisión* [13]. Este estudio se centra en identificar casos de obesidad temprana en jóvenes de entre 15 y 19 años, comparando la eficacia de diferentes modelos de clasificación.

Resultados adicionales

Entre las aplicaciones que permiten realizar predicciones diagnósticas, destaca la posibilidad de implementar un sistema de alertas para pacientes interesados en conocer su estado físico sin necesidad de acudir a un centro médico. Para llevar a cabo este sistema, es fundamental recopilar información de personas que participen en el estudio, proporcionando datos reales sobre su estado físico. Estos datos serán esenciales para generar alertas personalizadas que ayuden a futuros usuarios a identificar momentos clave en los que deberían preocuparse por su salud y tomar medidas preventivas.

Conclusiones

Se concluye que los modelos de predicción utilizados en el diseño e implementación del algoritmo de aprendizaje supervisado (KNN, Bayes, LDA y SVM) demostraron ser altamente efectivos, proporcionando resultados óptimos y precisos en relación con los datos ingresados sobre riesgos de obesidad. Las pruebas realizadas validaron la precisión de estos modelos, destacando la importancia de un entrenamiento adecuado y la normalización de los datos.

El procesamiento de datos se identifica como un factor crítico, ya que la presencia de valores nulos o incorrectos podría comprometer la calidad de las predicciones. El análisis de los resultados permitió identificar diferentes niveles de obesidad basados en el índice de masa corporal (IMC). Estos datos, procesados y utilizados de manera adecuada, garantizaron predicciones específicas y confiables, contribuyendo al desarrollo de análisis futuros junto con los valores registrados por cada usuario en la base de datos.

En términos generales, este análisis subraya la capacidad de la inteligencia artificial para aplicarse en diversos campos. En este caso particular, la implementación de un algoritmo como el desarrollado podría ser de gran utilidad para los profesionales de la salud, ya que agilizaría la detección y evaluación de riesgos de obesidad, beneficiando tanto a personas en etapas iniciales de sobrepeso como a aquellas en estados más avanzados.

Referencias

Referencias

[1] Martinez, J. A., Moreno-Aliaga, M. J., Marques-Lopes, I., & Marti, A. (2002). Causas de obesidad.

[2] Basulto, J., Manera, M., Baladia, E., Miserachs, M., Pérez, R., Ferrando, C., ... & Revenga, J. (2013). Definición y características de una alimentación saludable. Monografia a Internet].

[3] Bascon, M. A. P. (1994). Actividad física y salud. Obtenido de https://archivos. csif. es/archivos/andalucia/ensenanza/revistas/csicsif/revista/pdf/Numero_42/MIGUEL_ANGEL_PRIETO_BASCON_01. pdf.

[4] Google Colab. (s. f.). Recuperado 29 de marzo de 2024, de https://research.google.com/colaboratory/intl/es/faq.html#whats-colaboratory.

[5] Aransay, J., Casado-García, Á., Domínguez, C., García-Domínguez, M., Heras, J., Inés, A., ... & Pérez, B. (2022). GitHub y Google Colaboratory para el desarrollo, comunicación y gestión de prácticas en los laboratorios de informática.

[6] García, A., Martínez, G., Nuñez, E., & Guzmán, A. (1998). Clasificación supervisada, inducción de arboles de decisión, algoritmo kd. Proc. Simp. Int. de Comp. CIC, 98, 602-614.

[7] Obesity Risk Dataset. (2024, 11 marzo). Kaggle. https://www.kaggle.com/datasets/jpkochar/obesity-risk-dataset

[8] Madariaga Fernández, C. J., Lao León, Y. O., Curra Sosa, D. A., & Lorenzo Martín, R. (2022). Empleo de algoritmos KNN en metodología multicriterio para la clasificación de clientes, como sustento de la planeación agregada. Retos de la Dirección, 16(1), 178-198.

[9] Fuster Coma, N. (2023). Métodos de Clasificación en Python: Aplicaciones a la Empresa (Doctoral dissertation, Universitat Politècnica de València).

[10] Balakrishnama, S. y Ganapathiraju, A. (1998). Análisis discriminante lineal: un breve tutorial. Instituto de Procesamiento de Señales e Información , 18 (1998), 1-8.

[11] Qisthiano, MR, Ruswita, I. y Prayesy, PA (2023). Método de implementación de SVM en el análisis de sentimientos que se utilizan con el uso de Python 3. Tecnología: Jurnal Ilmiah Sistem Informasi , 13 (1), 1-7.

[12] Suca, C., Córdova, A., Condori, A., Cayra, J., & Sulla, J. (2016). Comparación de algoritmos de clasificación para la predicción de casos de obesidad infantil. Perú: Universidad Nacional de San Agustín.

[13] Díaz, S. E. L., Sibaja, J. A. P., Martínez, A. F., & Vázquez, S. J. (2023). Algoritmos de clasificación para la detección de obesidad en adolescentes: Un estudio comparativo entre KNN y árboles de decisión. Revista de Investigación en Tecnologías de la Información, 11(23), 70-81.