

ADSO-GA04-Actividad de Aprendizaje 6-Análisis de Datos

Juan Sebastian Quiceno Cano – Iyer Smith Torres Jaramillo

SENA-CTMA

ADSO-Ciencia Datos

Luis Sánchez

04/04/2025

Índice

Actividad de Contextualización	3
Actividad 1 EDA.....	7
Paso 1: Instalación de bibliotecas necesarias	7
Paso 2: Importación de librerías y cargar datos	8
Paso 3: Descripción general del conjunto de datos.....	9
Paso 4: Identificación de valores atípicos	10
Paso 5: Visualización de la distribución de los datos (Histogramas).....	11
Paso 6: Relación entre las características (Gráficas de dispersión)	12
Paso 7: Correlación entre características	13
Paso 8: Identificación de patrones	14
Paso 9: Conclusiones del análisis exploratorio	15
Actividad 2 – Modelos.....	16
Paso 1: Instalación de bibliotecas necesarias.....	16
Paso 2: Importación de librerías y carga de datos.....	16
Paso 3: Preparación de los datos	17
Paso 4: Creación del modelo de regresión lineal	18
Paso 5: Hacer predicciones	18
Paso 6: Evaluación del modelo	18
Paso 7: Visualización de resultados	19

Paso 8: Conclusiones del modelo	20
Actividad 3 –Proyecto de Aplicación	21
Paso 1: Instalación de bibliotecas	22
Paso 2: Importación de librerías	22
Paso 3: Crear DataFrame	23
Paso 4: Sacar el promedio general	23
Paso 5: Variable objetivo	23
Paso 6: Separa las variables creadas	24
Paso 7: Dividir el conjunto de entrenamiento y prueba.....	24
Paso 8: Entrenar el model, realizar predicción y configurar la grafica.....	24
Paso 9: Resultados:	25
Actividad 4 Power BI.....	26
Conclusiones	26

Actividad de Contextualización

1. ¿Qué entiende por "Ciencia de Datos"? ¿Cómo se relaciona con otras disciplinas como la estadística y la informática?

R// La ciencia de datos es la disciplina que estudia grandes cantidades de datos, esto con el fin de la realización de informes que serán presentados a respectivas empresas para saber como están la ventas, cual es la tendencia, cuales son las probabilidades de tener aumento en ganancias en el próximo año, periodo, mes, y muchos otros tipos de informes más, Podemos relacionar la ciencia de datos con las medidas de dispersión, como la varianza y la desviación estándar, también con las medidas estadísticas como la media, mediana y moda.

2. ¿Qué impacto tiene el análisis de datos en la toma de decisiones empresariales y científicas?

R// El análisis de datos transforma grandes volúmenes de información en conocimientos útiles para la toma de decisiones.

En el ámbito empresarial:

- **Optimización de procesos:** Permite identificar inefficiencias y mejorar la productividad.
- **Estrategias de mercado:** Facilita la segmentación de clientes y personalización de productos o servicios.
- **Gestión de riesgos:** Ayuda a detectar fraudes, prever crisis y tomar decisiones financieras más acertadas.

En el ámbito científico:

- **Validación de hipótesis:** Facilita la interpretación de resultados con modelos estadísticos.
- **Optimización de experimentos:** Reduce costos y mejora la precisión de los estudios.

- **Descubrimiento de patrones:** Ayuda a identificar correlaciones y tendencias en grandes volúmenes de datos.

3. ¿Cómo cree que la Ciencia de Datos puede transformar diferentes industrias y sectores?

R// La Ciencia de Datos está revolucionando múltiples industrias al optimizar procesos, mejorar la toma de decisiones y generar nuevas oportunidades.

Transformación en diferentes sectores:

- **Salud:** Diagnóstico temprano con IA, desarrollo de tratamientos personalizados y optimización de recursos hospitalarios.
- **Finanzas:** Detección de fraudes, evaluación de riesgos y análisis predictivo para inversiones.
- **Retail y Marketing:** Personalización de ofertas, predicción de demanda y optimización de la cadena de suministro.
- **Manufactura:** Mantenimiento predictivo, automatización de procesos y reducción de desperdicios.
- **Transporte y Logística:** Optimización de rutas, gestión de flotas y predicción de demanda.
- **Educación:** Aprendizaje personalizado, detección de patrones de desempeño y mejora en metodologías de enseñanza.
- **Agricultura:** Uso de sensores e IA para monitoreo de cultivos y optimización del riego.

En todos los sectores, la Ciencia de Datos impulsa eficiencia, reduce costos y permite innovaciones disruptivas.

4. ¿Cuáles son algunos ejemplos de proyectos o investigaciones que utilizan Ciencia de Datos en el mundo actual?

R// Existen numerosos proyectos e investigaciones que utilizan Ciencia de Datos para resolver problemas complejos y generar impacto en diversas áreas. Algunos ejemplos son:

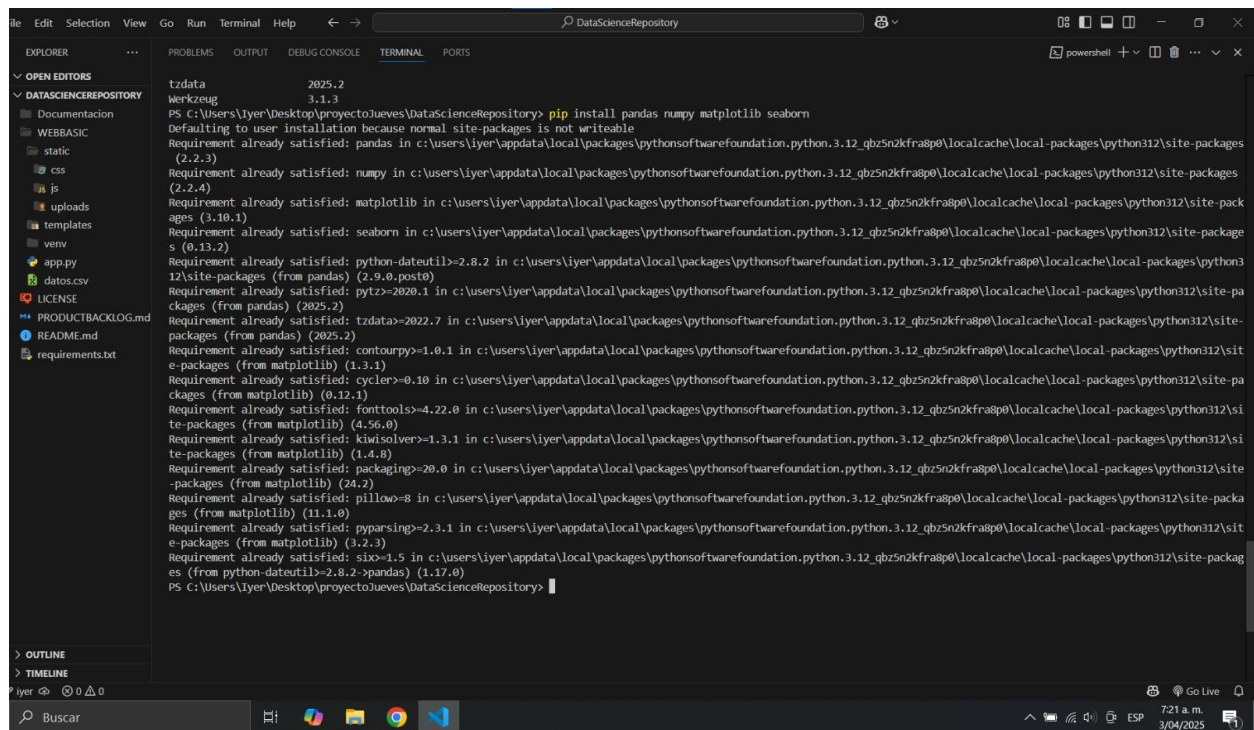
- **Predicción de enfermedades con IA:** Algoritmos de aprendizaje automático analizan imágenes médicas para detectar cáncer, enfermedades cardiovasculares y otras afecciones con alta precisión.
- **Modelos de cambio climático:** Investigaciones utilizan análisis de grandes volúmenes de datos para predecir patrones climáticos y evaluar el impacto del calentamiento global.
- **Sistemas de recomendación:** Empresas como Netflix, Amazon y Spotify emplean Ciencia de Datos para sugerir contenido personalizado basado en los hábitos de los usuarios.
- **Optimización del tráfico urbano:** Ciudades inteligentes utilizan modelos de predicción y análisis en tiempo real para mejorar la movilidad y reducir la congestión vehicular.
- **Detección de fraudes financieros:** Bancos y fintechs aplican algoritmos para identificar transacciones sospechosas y prevenir fraudes en tiempo real.
- **Análisis de redes sociales:** Estudios analizan tendencias, opiniones y comportamientos en plataformas digitales para entender dinámicas sociales y predecir fenómenos virales.

Estos proyectos demuestran cómo la Ciencia de Datos contribuye a mejorar la eficiencia, optimizar procesos y resolver desafíos en distintas áreas.

Actividad 1 EDA

Paso 1: Instalación de bibliotecas necesarias

Terminal: `pip install pandas matplotlib numpy seaborn`



```
tzdata 2025.2
werkzeug 3.1.3
PS C:\Users\iyer\Desktop\proyectoJueves\DataScienceRepository> pip install pandas numpy matplotlib seaborn
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (2.2.3)
Requirement already satisfied: numpy in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (2.2.4)
Requirement already satisfied: matplotlib in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (3.10.1)
Requirement already satisfied: seaborn in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (0.13.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (3.2.3)
Requirement already satisfied: six>=1.5 in c:\users\iyer\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
PS C:\Users\iyer\Desktop\proyectoJueves\DataScienceRepository>
```

Package	Version
blinker	1.9.0
click	8.1.8
colorama	0.4.6
contourpy	1.3.1
cycler	0.12.1
Flask	3.1.0
Flask-MySQLdb	2.0.0
fonttools	4.56.0
itsdangerous	2.2.0
Jinja2	3.1.6
kiwisolver	1.4.8
MarkupSafe	3.0.2
matplotlib	3.10.1
mysql-connector-python	9.2.0
mysqlclient	2.2.7
numpy	2.2.4
packaging	24.2
pandas	2.2.3
pillow	11.1.0
pip	25.0.1
pyparsing	3.2.3
python-dateutil	2.9.0.post0
pytz	2025.2
seaborn	0.13.2
six	1.17.0
tzdata	2025.2
Werkzeug	3.1.3
PS C:\Users\Iyer\Desktop\proyectoJueves\DataScienceRepository>	

Paso 2: Importacion de librerías y cargar datos

```
import numpy as np
import pandas as pd
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import seaborn as sns
```

Ahora cargaremos el conjunto de datos de Iris desde seaborn y los mostraremos en la terminal, con las siguientes líneas.


```
data = sns.load_dataset('iris')
print (data.head())
```

```
PS C:\Users\Iyer\Desktop\actividad1> python app.py run
   sepal_length  sepal_width  petal_length  petal_width  species
0           5.1           3.5           1.4           0.2   setosa
1           4.9           3.0           1.4           0.2   setosa
2           4.7           3.2           1.3           0.2   setosa
3           4.6           3.1           1.5           0.2   setosa
4           5.0           3.6           1.4           0.2   setosa
```

Paso 3: Descripción general del conjunto de datos

Usando el método *describe()* obtendremos estadísticas descriptivas del conjunto de datos, como la media, la desviación, el max y el min.

```
#descripcion general del conjunto de datos
print (data.describe())
```

```
count      sepal_length  sepal_width  petal_length  petal_width
count      150.000000    150.000000    150.000000    150.000000
mean         5.843333         3.057333         3.758000         1.199333
std          0.828066         0.435866         1.765298         0.762238
min          4.300000         2.000000         1.000000         0.100000
25%          5.100000         2.800000         1.600000         0.300000
50%          5.800000         3.000000         4.350000         1.300000
75%          6.400000         3.300000         5.100000         1.800000
max          7.900000         4.400000         6.900000         2.500000
```

Paso 4: Identificación de valores atípicos

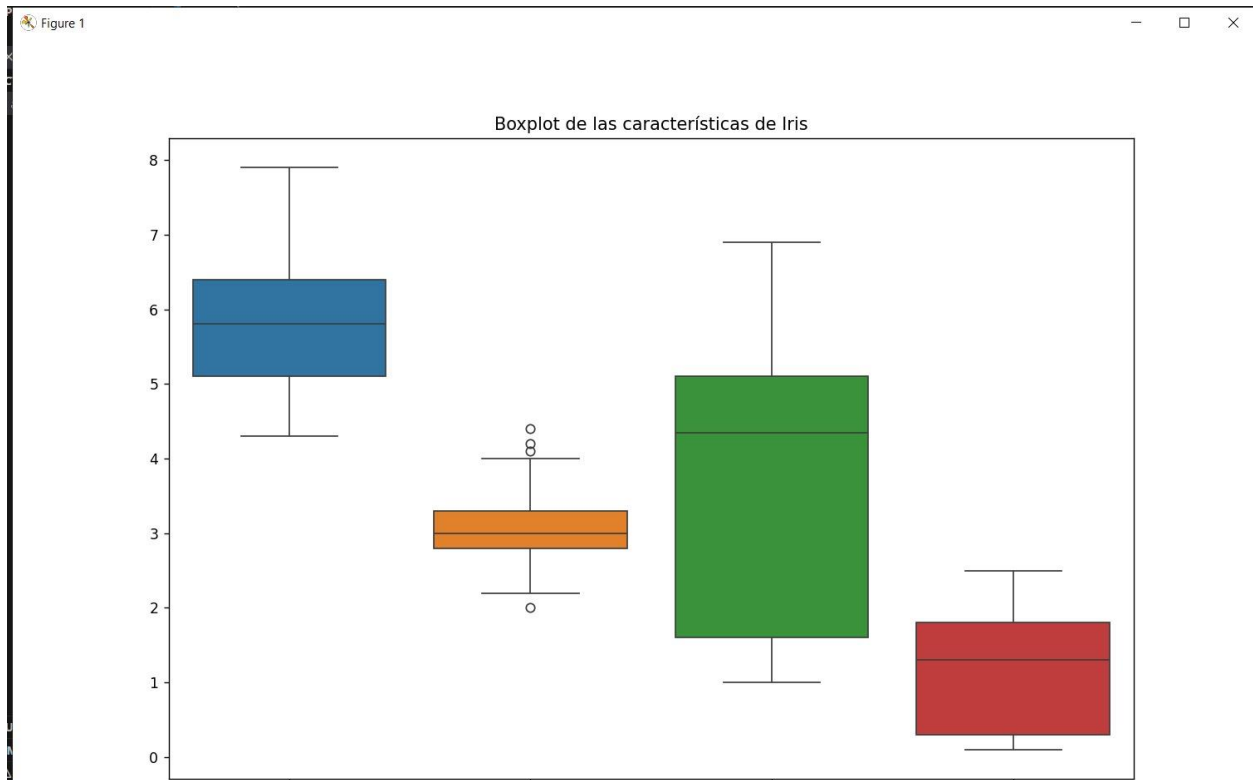
Los valores atípicos(outliers) pueden ser identificado con un diagrama de caja (boxplot) o mediante el uso de la desviación estándar.

```
import matplotlib
matplotlib.use('TkAgg')
```

Primero usamos la instrucción 'TkAgg', es una instrucción de los backends interactivos de matplotlib que utiliza la biblioteca TKinter para mostrar gráficos en ventanas independientes.

```
# generar boxplots cada una de las características del conjunto de datos
plt.figure(figsize=(12, 8))

#boxplot de todas las columnas numericas
sns.boxplot(data=data[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']])
plt.title("Boxplot de las características de Iris")
plt.show()
```

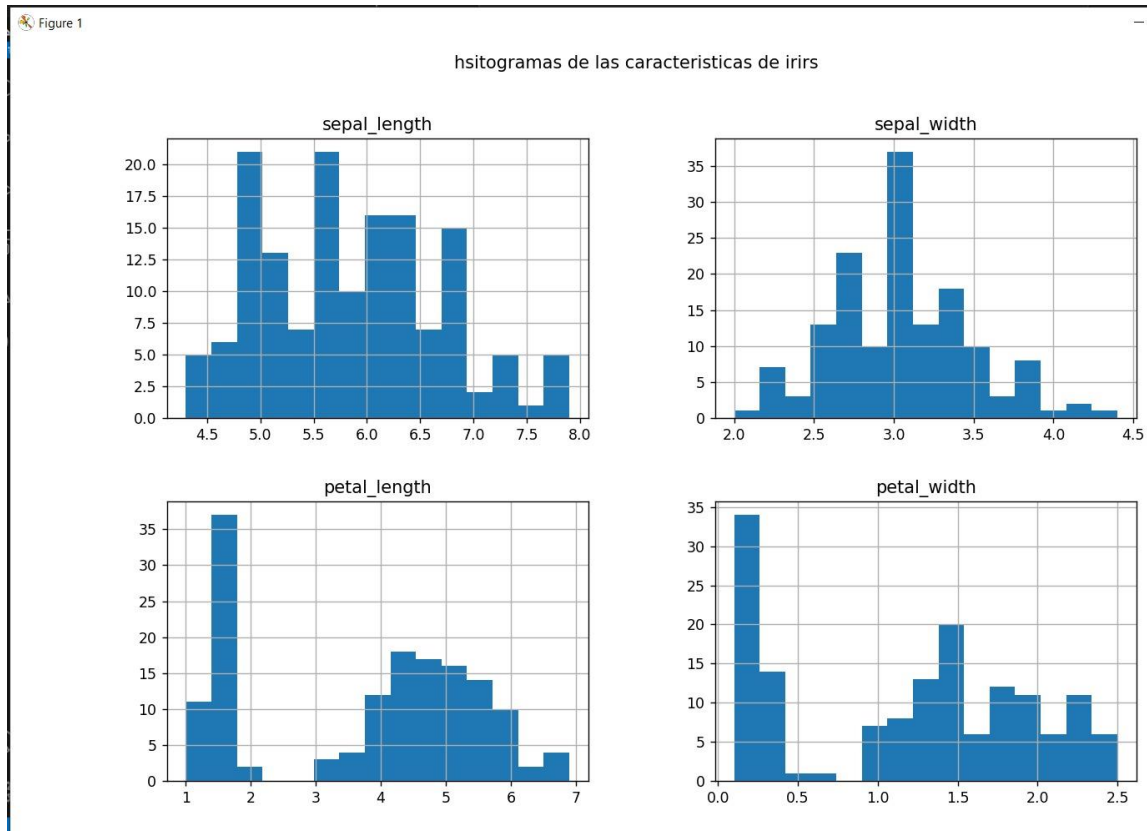


Paso 5: Visualización de la distribución de los datos (Histogramas)

Usaremos histogramas para ver cómo se distribuyen las variables numéricas. Esto es útil para ver la forma de las distribuciones (simétrica, sesgada, etc.).

```
#histograma de las características numericas
data[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']].hist (bins=15, figsize=(12, 8))
plt.suptitle ("hsitogramas de las características de irirs")
plt.show()
```

Con esta línea de código le daremos el título y generaremos la imagen para ser mostrada.

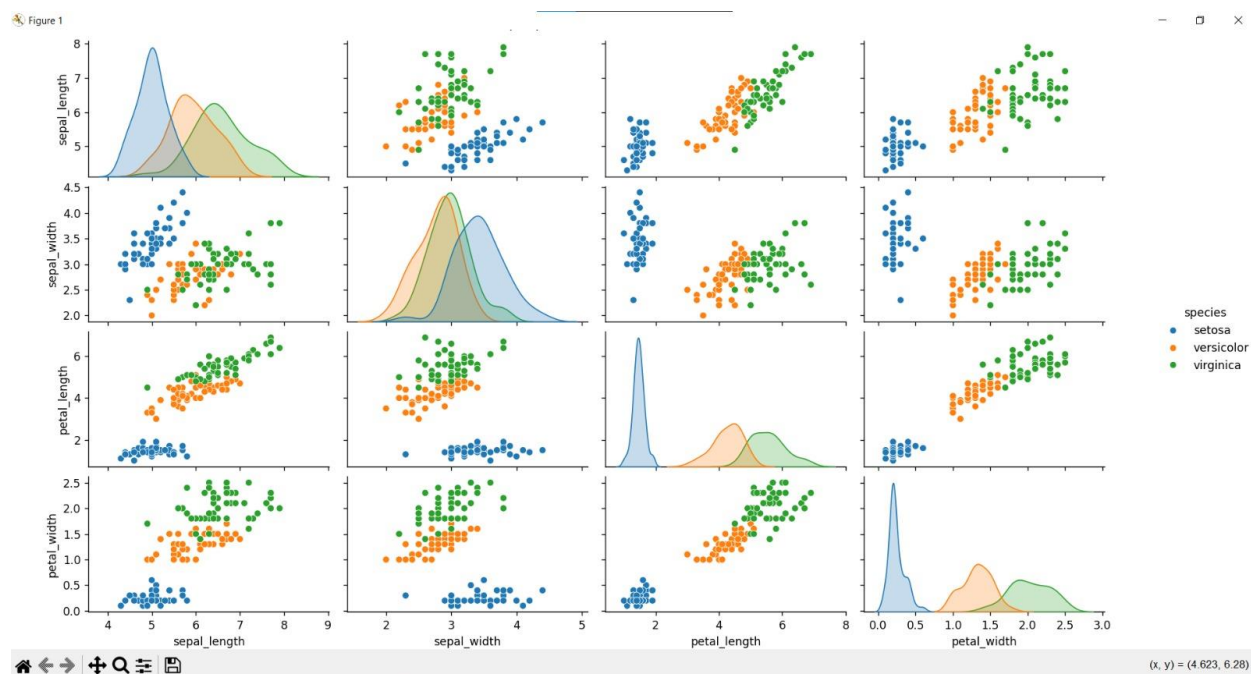


Paso 6: Relación entre las características (Graficas de dispersión)

Para ver cómo se relacionan entre sí las diferentes características del conjunto de datos, podemos utilizar un par de gráficos de dispersión (scatter plots).

```
#pairplot para ver las relacion entre las caracteristicas numericas

sns.pairplot(data, hue='species')
plt.suptitle("pairplot de las caracteristicas de irirs", y=1.02 )
plt.show()
```



Paso 7: Correlación entre características

Es útil ver cómo se correlacionan las características entre sí. Usamos un mapa de calor (heatmap) para visualizar la matriz de correlación.

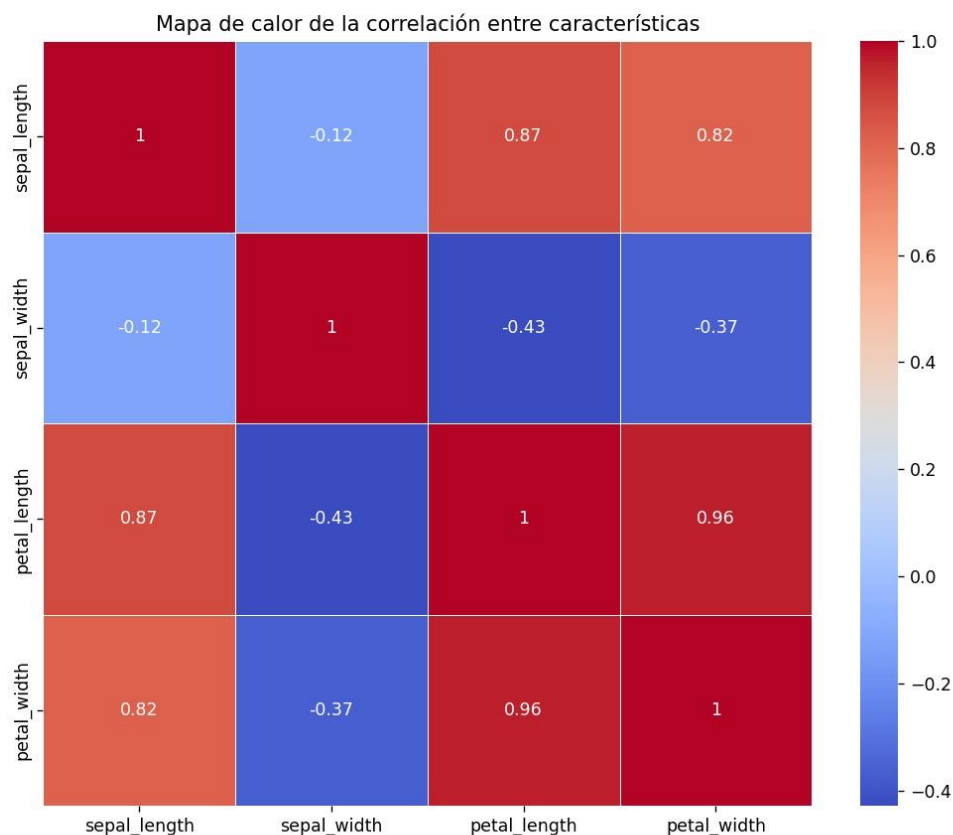
```
# Seleccionar solo las columnas numéricas
numeric_data = data.select_dtypes(include=['number'])

# Calcular la matriz de correlación
correlation_matrix = numeric_data.corr()

# Mostrar el heatmap
plt.figure(figsize=(10,8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Mapa de calor de la correlación entre características")
plt.show()
```

“data.select_dtypes(include=['number'])” selecciona solo las columnas numéricas, ignorando las categóricas como "setosa".

Figure 1

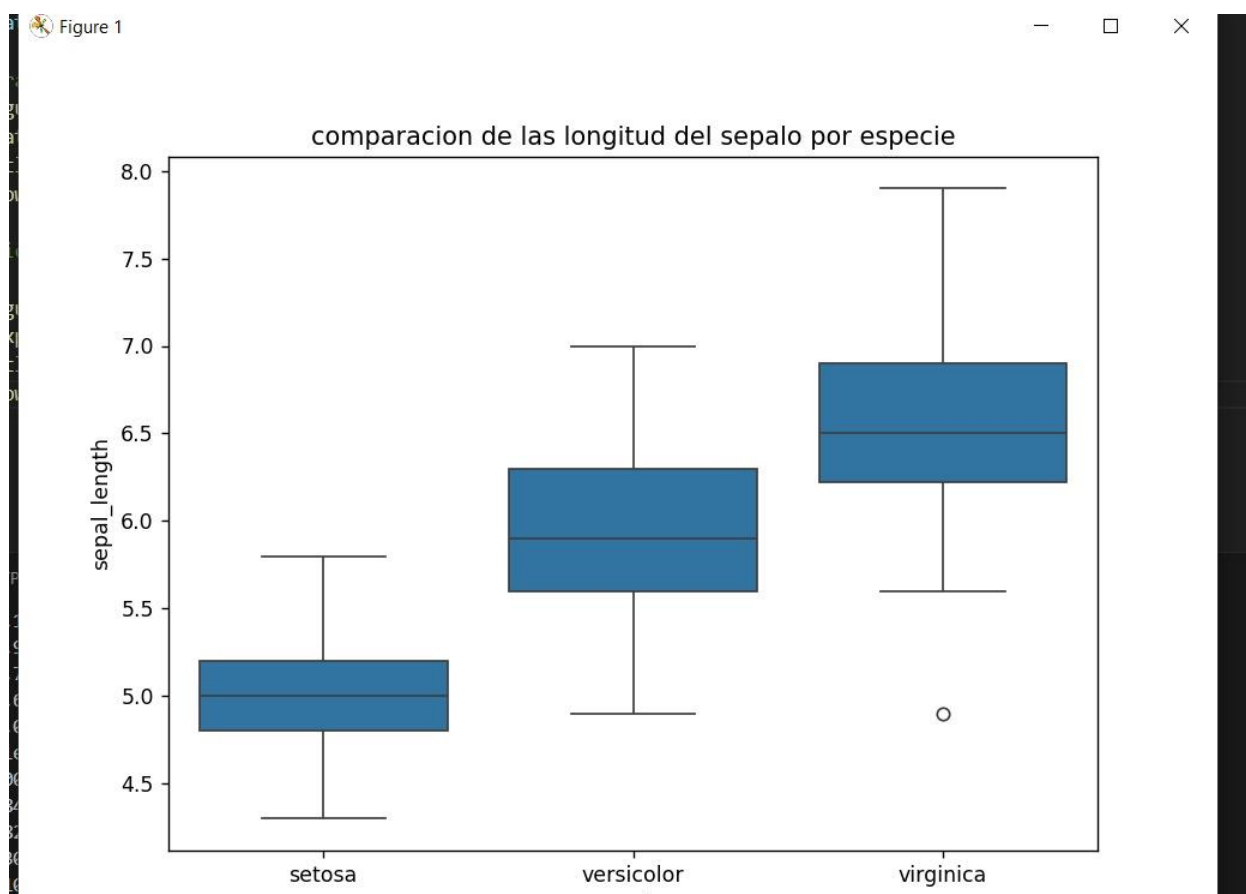


Paso 8: Identificación de patrones

Los patrones en los datos pueden ser observados al examinar cómo las características se distribuyen según las especies. Por ejemplo, podemos crear un gráfico de cajas para comparar la longitud del sépalo entre las diferentes especies de iris (si está usando su propio conjunto de datos deberá observar qué patrones pueden identificarse).

```
# garfico de caja para comparar la longitud del sepalo entre las especies

plt.figure(figsize=(8,6))
sns.boxplot(x='species', y='sepal_length', data=data)
plt.title("comparacion de las longitud del sepalo por especie")
plt.show()
```



Paso 9: Conclusiones del análisis exploratorio

Después de realizar estas visualizaciones, los aprendices pueden realizar las siguientes conclusiones:

- En el punto cuatro podemos observar varios puntos que están por fuera de los bigotes, esos son conocidos como los valores atípicos, mas que todo en la segunda caja
- En el punto cuatro el tercer grupo (longitud del pétalo) muestra una gran dispersión, mientras que el cuarto grupo (ancho del pétalo) tiene valores más concentrados.
- El mapa de calor muestra que **petal_length** y **petal_width** tienen la correlación más fuerte (**0.96**), indicando que pétalos más largos suelen ser más anchos. **Sepal_length** también se relaciona positivamente con ambas características del pétalo (**0.87** y **0.82**). En contraste, **sepal_width** tiene correlaciones negativas con las demás características, lo que sugiere una relación inversa.
- El gráfico muestra que **Setosa** está bien separada de **Versicolor** y **Virginica**, especialmente en dimensiones de los pétalos. **Versicolor** y **Virginica** se solapan, pero pueden distinguirse con **petal_length** y **petal_width**. **Sepal_width** es menos útil para **diferenciar especies**.

Actividad 2 – Modelos

Paso 1: Instalación de bibliotecas necesarias

Si aún no tienes instaladas las bibliotecas necesarias, puedes instalar las siguientes:

```
pip install pandas numpy matplotlib seaborn scikit-learn
```

Paso 2: Importación de librerías y carga de datos

Comenzamos por importar las bibliotecas necesarias y cargar el conjunto de datos Iris desde la librería `seaborn`.


```

1  import pandas as pd
2  import numpy as np
3  import matplotlib.pyplot as plt
4  import seaborn as sns
5  from sklearn.ensemble import RandomForestClassifier
6  from sklearn.metrics import mean_squared_error, r2_score
7  from sklearn.model_selection import train_test_split
8  from sklearn.linear_model import LogisticRegression
9  from sklearn.datasets import load_iris
10 from sklearn.metrics import accuracy_score
11 from sklearn.neighbors import KNeighborsClassifier
12 from sklearn.neural_network import MLPClassifier
13 from sklearn.tree import DecisionTreeClassifier
14 from sklearn.svm import SVC
15
16

```

```

#Cargar el conjunto de datos Iris desde seaborn
data = sns.load_dataset('iris')
#Ver las primeras filas del conjunto de datos
print(data.head())
|

```

Paso 3: Preparación de los datos

```

5
6
7  # Seleccionamos las características (features) y la variable objetivo (target)
8
9
10 x = data[['sepal_length', 'sepal_width', 'petal_width']] # Variables
11
12
13 y = data['petal_length'] # Variable objetivo
14 # Dividir el conjunto de datos en entrenamiento y prueba (80%entrenamiento, 20% prueba)
15
16
17 X_train, X_test, y_train, y_test = train_test_split(x, y,
18 test_size=0.2, random_state=42)
19 print("Tamaño del conjunto de entrenamiento:", x_train.shape)
20 print("Tamaño del conjunto de prueba:", x_test.shape)
21
22

```

Paso 4: Creación del modelo de regresión lineal

```
#4

# Crear el modelo de regresión lineal
model = LinearRegression()
# Entrenar el modelo
model.fit(x_train, y_train)
# Ver los coeficientes y la intersección (intercepto) del modelo
print(f"Coeficientes: {model.coef_}")
print(f"Intersección (intercepto): {model.intercept_}")

#5
```

Paso 5: Hacer predicciones

```
#5

# Realizar predicciones sobre el conjunto de prueba
y_pred = model.predict(x_test)
# Mostrar las predicciones y los valores reales
predictions_df = pd.DataFrame({'Real': y_test, 'Predicción':
y_pred})
print(predictions_df.head())
```

Paso 6: Evaluación del modelo

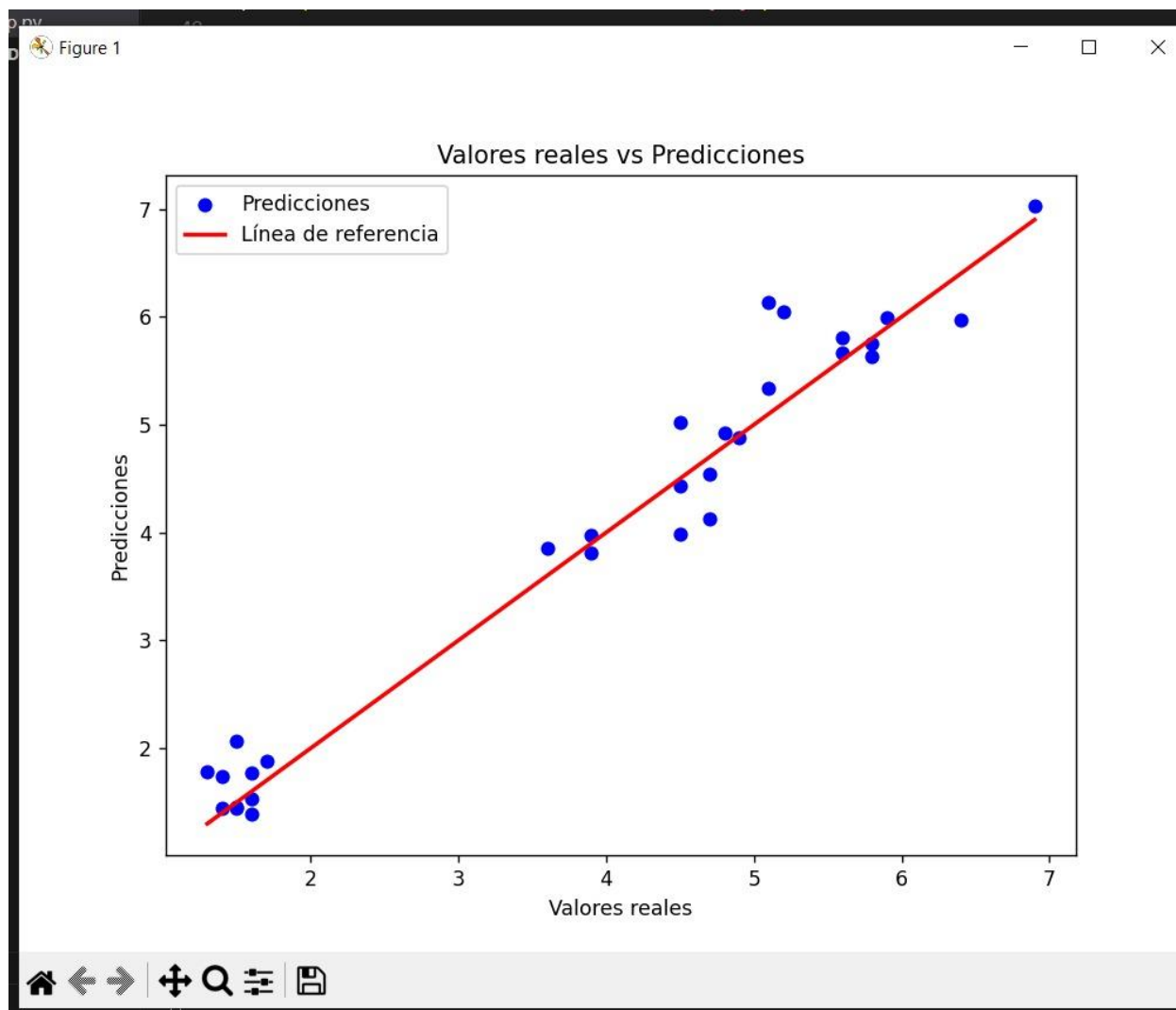
```
#6

# Calcular el error cuadrático medio (MSE) y  $R^2$ 
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Error cuadrático medio (MSE): {mse}")
print(f"Coeficiente de determinación  $R^2$ : {r2}")
```

Paso 7: Visualización de resultados

```
#7

# Graficar los valores reales vs. las predicciones
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue', label='Predicciones')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()],
color='red', linewidth=2, label="Línea de referencia")
plt.xlabel("Valores reales")
plt.ylabel("Predicciones")
plt.title("Valores reales vs Predicciones")
plt.legend()
plt.show()
```



Paso 8: Conclusiones del modelo

1. **Impacto de las características:** Los coeficientes de la regresión muestran la influencia de cada característica en la predicción de la longitud del pétalo, permitiendo identificar cuáles son más relevantes.
2. **Precisión del modelo:** El error cuadrático medio (MSE) indica el nivel de error en las predicciones, lo que ayuda a evaluar si el modelo necesita ajustes o mejoras.

3. **Ajuste del modelo:** Si el coeficiente de determinación (R^2) es cercano a 1, significa que el modelo explica bien la variabilidad de la longitud del pétalo; si es bajo, se podría mejorar con más características o modelos más complejos.
 4. **Visualización de desempeño:** La comparación entre valores reales y predicciones a través de un gráfico de dispersión permite verificar si el modelo sigue una tendencia lineal o si hay patrones que no está capturando bien.
-

Actividad 3 –Proyecto de Aplicación

Introducción: Ya para finalizar con estos puntos y herramientas en Python, implementaremos en el proyecto personal de nuestro equipo de la rueda de la vida una herramienta que nos permitirá clasificar usuarios en diferentes niveles de bienestar, esto por medio de la importación de Clustering (K-Means).

El clustering con **K-Means** agrupa a los usuarios según sus respuestas en las diferentes áreas (salud, finanzas, familia, deporte, etc.). Esto nos permite **segmentar** a los usuarios en distintos niveles de bienestar.

¿Qué mostraría?

- Se identificarían **grupos** de personas con patrones similares en sus respuestas.
- Podríamos clasificar a los usuarios en **categorías como "Bajo bienestar", "Equilibrado" y "Alto bienestar"**.

- Se podría visualizar en un **gráfico de dispersión** o en una **tabla con la asignación de cada usuario a un grupo**.

¿Cómo lo implementaremos?

1. **Preprocesar los datos** (normalizar valores entre 0 y 1).
2. **Aplicar el algoritmo K-Means** para agrupar a los usuarios.
3. **Asignar cada usuario a un cluster** y visualizar los resultados.
4. **Integrar en Flask** para mostrar los grupos en una interfaz web.

Paso 1: Instalación de bibliotecas

```
pip install pandas numpy scikit-learn flask matplotlib seaborn
```

Paso 2: Importación de librerías

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
matplotlib.use('Agg')
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

Las otras librerías nos permiten realizar una grafica con mas detalles, pero en caso de nuestro proyecto no es necesario, ya que las respuestas solo darán una fila por usuario o una respuesta por área (físico, economía, etc).

Paso 3: Crear DataFrame

```
Promedio = {
    'ocio': [ocio],
    'personal': [personal],
    'dinero': [dinero],
    'trabajo': [trabajo],
    'fisica': [fisica],
    'familiar': [familiar],
    'social': [social],
    'espiritual': [espiritual]
}
df = pd.DataFrame(Promedio)
```

En nuestro caso ya había hecho un data frame anteriormente para otro tipo de trabajo, ese mismo utilizaremos para el trabajo de hoy.

Paso 4: Sacar el promedio general

```
df['promedio'] = np.mean(df[['ocio', 'personal', 'dinero', 'trabajo', 'fisica', 'familiar', 'social', 'espiritual']])
print(df)
```

Usando la importación de numpy lograremos sacar el promedio general de las respuestas por todas las áreas del usuario.

Paso 5: Variable objetivo

```
# Crear la variable objetivo (estado general: 1 = Bien, 0 = Mal)
df['estado_general'] = (df['promedio'] >= 6).astype(int)
```

En este caso creamos una variable llamada “estado_general”, básicamente, esta línea clasifica los datos en dos categorías:

"Bueno" (1) si el promedio es 6 o más.

"Malo" (0) si el promedio es menor a 6.

Paso 6: Separa las variables creadas

```
# Separar variables predictoras y objetivo
X = df[['promedio']]
y = df['estado_general']
```

You, hace 19 minutos • Uncommitted changes

Paso 7: Dividir el conjunto de entrenamiento y prueba

```
You, hace 19 minutos • Uncommitted changes
# Dividir en conjunto de entrenamiento y prueba
X_train = pd.DataFrame({'promedio': np.random.randint(1, 11, size=50)})
y_train = (X_train['promedio'] >= 6).astype(int)
```

Paso 8: Entrenar el modelo, realizar predicción y configurar la grafica

```
# Entrenar el modelo
model = LogisticRegression()
model.fit(X_train, y_train)

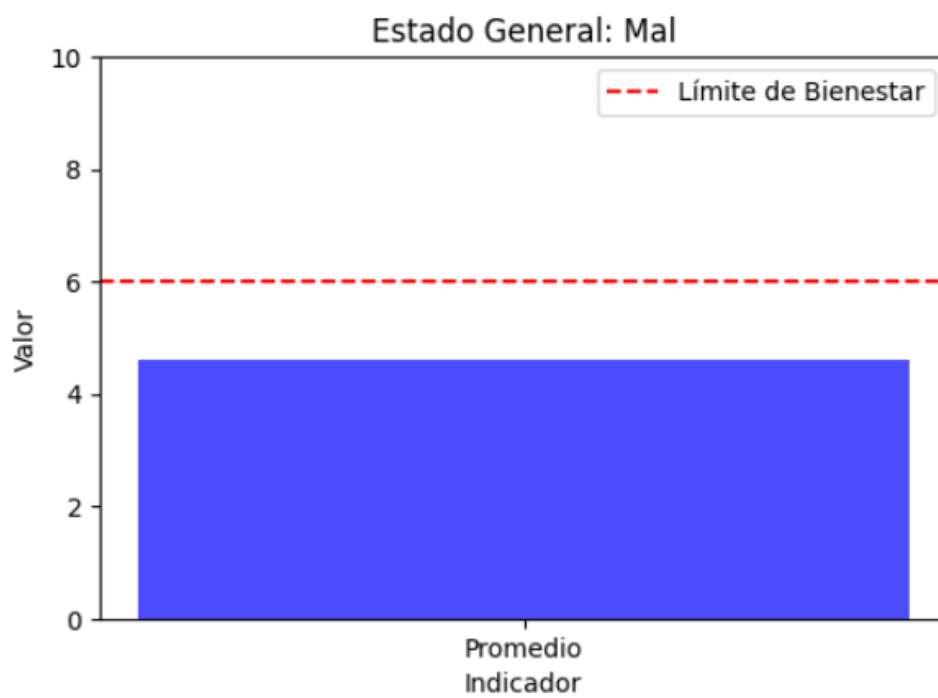
# Hacer predicción con la única muestra real
estado_predicho = model.predict(df[['promedio']])
```

You, hace 20 minutos • Uncommitted changes

```
plt.figure(figsize=(6, 4))
plt.bar(['Promedio'], [df['promedio'].values[0]], color='blue', alpha=0.7)
plt.axhline(y=6, color='red', linestyle='--', label='Límite de Bienestar')
plt.xlabel("Indicador")
plt.ylabel("Valor")
plt.ylim(0, 10)
plt.title(f"Estado General: {'Bien' if estado_predicho == 1 else 'Mal'}")
plt.legend()
plt.savefig('static/uploads/regresion.png') # Guarda en la carpeta 'static'
plt.close() # Cierra la figura para liberar memoria
```

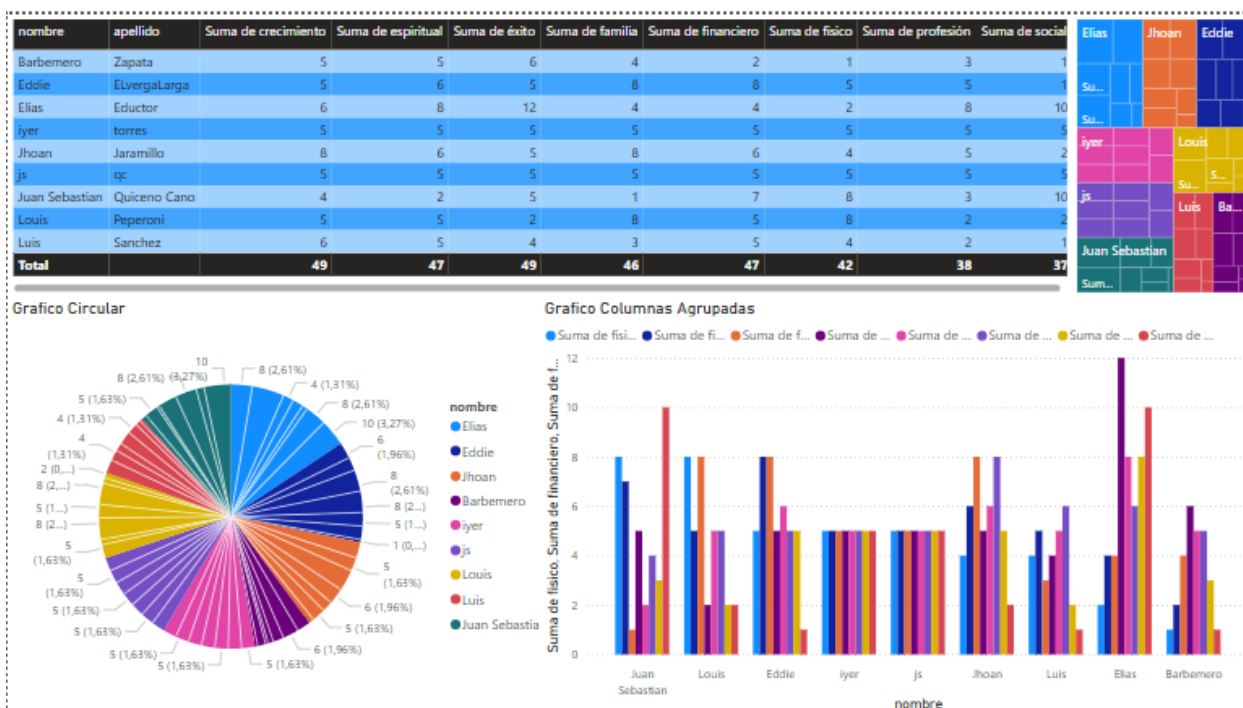

Paso 9: Resultados:

Regresion Logistica



[Ir a la Encuesta](#)

Actividad 4 Power BI



Ya para finalizar, hemos creado un DashBoard donde mostramos por medio de graficas de barras agrupadas, circular, treemap, etc. Las respuestas de cada usuario, sus sumas totales y en que ámbitos se destacan más.

Anexare el enlace al drive y al repositorio, para que puedan verlo más detalladamente.

<https://github.com/DataScience523/DataScienceRepository>

<https://drive.google.com/drive/>

Conclusiones

Podemos concluir que los encuestados en el proyecto rueda de la vida, llevan vida volátiles algunos, como otros tantos que sienten que su vida es normal en todos esos ámbitos,

con herramientas como power bi y csv, xlsx o Bases de datos podemos recopilar múltiples datos que generan información, con power BI graficar esa información hace que la comprensión y la toma de decisiones respecto a un tema específico sea más fácil de comprender, podemos saber hacia donde va la tendencia, el mercado, la salud de las personas y proponer movimientos, campañas y muchas más estrategias para alcanzar mejores metas en futuros lejanos y no tan lejanos.