ITSRLL
INSTITUTO TECNOLÓGICO SUPERIOR DE LA REGIÓN DE LOS LLANOS

Ingeniería Mecatrónica

PROGRAMACIÓN AVANZADA

Enero – Junio 2025

M.C. Osbaldo Aragón Banderas

UNIDAD:

1	2	3	4	5

Actividad número: 3

Nombre de actividad:

NOTEBOOK: Análisis de Datos Aplicables al Teorema de Naïve Bayes

Actividad realizada por:

Carlos Diego Dominguez López - 21030006

Guadalupe Victoria, Durango

Fecha de entrega:

01	03	2025

INDICE

Introducción

En el ámbito del aprendizaje automático, los algoritmos de clasificación juegan un papel fundamental en la toma de decisiones basadas en datos. Uno de los métodos más utilizados por su simplicidad y eficiencia es el clasificador Naïve Bayes, basado en el Teorema de Bayes, el cual permite calcular la probabilidad de que una observación pertenezca a una determinada categoría, considerando la independencia condicional entre sus características.

Esta práctica tiene como propósito la búsqueda, selección y análisis de un conjunto de datos adecuado para clasificación, aplicando el algoritmo de Naïve Bayes. A través de este proceso, se profundizará en la comprensión del Teorema de Bayes, su implementación en problemas de clasificación y la evaluación del rendimiento del modelo.

El desarrollo de la actividad incluye la investigación de los fundamentos teóricos del Teorema de Bayes, la selección de un conjunto de datos en Kaggle u otra fuente confiable, el preprocesamiento de los datos y la aplicación del algoritmo utilizando Python y la librería scikit-learn. Finalmente, se analizarán los resultados mediante métricas como precisión, recall y matriz de confusión para extraer conclusiones sobre la efectividad del modelo en el problema seleccionado.

Objetivo

El propósito de esta actividad es que los estudiantes busquen, seleccionen y analicen un conjunto de datos en Kaggle o en otra fuente confiable, aplicando el algoritmo de Naïve Bayes para resolver un problema de clasificación. Además, deberán presentar los resultados y conclusiones obtenidas, relacionándolos con la teoría del Teorema de Bayes.

Marco Teórico

Teorema de Bayes y su Aplicación en el Clasificador Naïve Bayes

El Teorema de Bayes es una regla matemática que permite calcular la probabilidad de que ocurra un evento teniendo en cuenta información previa. En otras palabras, ayuda a actualizar nuestras creencias sobre un suceso a partir de nueva evidencia.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (1.1)$$

Donde:

- P(A|B) es la probabilidad de que ocurra A, dado que ya ocurrió B.
- P(B|A) es la probabilidad de que ocurra B si se sabe que A ha ocurrido.
- P(A) es la probabilidad previa del evento A (antes de tener información adicional).
- P(B) es la probabilidad total del evento B, que sirve como factor de normalización.

El clasificador Naïve Bayes es un modelo que usa el Teorema de Bayes para predecir a qué categoría pertenece un dato. Funciona suponiendo que todas las características del dato son independientes entre sí, lo que simplifica los cálculos. Aunque esta suposición no siempre es totalmente cierta, el modelo sigue siendo muy efectivo para clasificar información. Su objetivo es calcular la probabilidad de que un dato pertenezca a una categoría y elegir la opción más probable.

Este método es útil porque permite tomar decisiones rápidas y precisas en base a datos previos. Se aplica en diversos campos, como la detección de fraudes, el análisis de sentimientos en redes sociales y el diagnóstico de enfermedades.

Ecuación General de Naïve Bayes y Significado de sus Componentes

El clasificador Naïve Bayes se basa en el Teorema de Bayes para predecir la probabilidad de que una observación pertenezca a una determinada categoría. La ecuación general del modelo se expresa de la siguiente manera:

$$P(C_k|X) = \frac{P(X|C_k)P(C_k)}{P(X)} \quad (2.1)$$

Donde:

- $P(C_k|X)$ es la probabilidad posterior, es decir, la probabilidad de que una observación pertenezca a la clase C_k dado el conjunto de características X.
- $P(X|C_k)$ es la verosimilitud, que representa la probabilidad de observar los datos X si se sabe que la clase es C_k.
- $P(C_k)$ es la probabilidad a priori, que indica la frecuencia con la que aparece la clase C_k en el conjunto de datos antes de observar cualquier característica.
- P(X)es la probabilidad total de los datos X, que actúa como un factor de normalización para garantizar que la probabilidad posterior sea válida

Descomposición del Modelo Naïve Bayes

El modelo Naïve Bayes asume que las características son independientes entre sí dentro de cada clase. Esto significa que la probabilidad conjunta de todas las características se puede descomponer en el producto de las probabilidades individuales de cada una:

$$P(X|C_k) = P(x_1|C_k) * P(x_2|C_k) * \dots * P(x_n|C_k) \quad (2.2)$$

Por lo tanto, la ecuación general se puede reescribir como:

$$P(C_k|X) = \frac{P(C_k) * P(x_1|C_k) * P(x_2|C_k) * \ldots * P(x_n|C_k)}{P(X)} \quad (2.3)$$

Este cálculo se repite para todas las clases C_k, y se elige la clase con la probabilidad más alta como la predicción final.

Interpretación de los Componentes

Probabilidad a priori $\boldsymbol{P(C_k)}$: Representa la frecuencia con la que ocurre una clase antes de analizar los datos. Se calcula contando cuántos ejemplos pertenecen a esa clase en el conjunto de datos.

Verosimilitud $\boldsymbol{P(X|C_k)}$: Indica qué tan probable es observar ciertas características si se sabe que un dato pertenece a una clase específica. Se basa en la frecuencia con la que esas características aparecen dentro de cada clase.

Probabilidad total P(X): Es un factor de normalización que garantiza que las probabilidades sean válidas. Se calcula sumando las probabilidades ponderadas de todas las clases posibles. Sin embargo, en la práctica, no es necesario calcularlo, ya que solo se comparan las probabilidades relativas entre clases.

Probabilidad posterior $\boldsymbol{P(C_k|X)}$: Es el resultado final del modelo, indicando la probabilidad de que un dato pertenezca a una clase después de considerar sus características. Se compara este valor para todas las clases y se elige la que tenga la mayor probabilidad como la predicción del modelo.

Casos de Uso Reales del Modelo Naïve Bayes

A pesar de ser un modelo basado en una suposición simplificada de independencia entre características, Naïve Bayes ha demostrado ser altamente eficaz en numerosos problemas de clasificación. Su eficiencia radica en su capacidad para procesar grandes volúmenes de datos de manera rápida y en su facilidad de implementación, lo que lo convierte en una opción ideal en diversos ámbitos, especialmente cuando se trabaja con texto o datos categóricos. A continuación, se presentan algunos de sus principales usos:

1. Filtrado de Spam en Correos Electrónicos

El filtrado de spam es una de las aplicaciones más conocidas de Naïve Bayes. Los proveedores de correo electrónico, como Gmail, Outlook y Yahoo Mail, utilizan este modelo para clasificar los correos como "spam" o "no spam". Para ello, se analizan palabras clave como "gratis", "descuento" o "ganador", además de otros atributos como la dirección del remitente y la frecuencia de ciertos enlaces. A medida que los usuarios marcan correos como spam o los sacan de la bandeja de spam, el modelo se entrena continuamente y mejora su precisión.

2. Diagnóstico Médico

En el campo de la salud, Naïve Bayes se emplea para la detección y clasificación de enfermedades a partir de síntomas y resultados de pruebas médicas. Por ejemplo, en el diagnóstico de diabetes o cáncer, el modelo puede analizar características como niveles de glucosa, presión arterial, antecedentes familiares y hábitos de vida para estimar la probabilidad de que un paciente tenga la enfermedad. Su velocidad y precisión en este tipo de problemas lo hacen útil en sistemas de apoyo a la decisión médica, ayudando a los profesionales a realizar diagnósticos más informados.

3. Análisis de Sentimientos

El análisis de sentimientos se usa en redes sociales, encuestas y reseñas de productos para clasificar textos en categorías como "positivo", "negativo" o "neutral". Plataformas como Twitter, Facebook y Amazon utilizan Naïve Bayes para evaluar la opinión de los usuarios sobre productos, servicios o temas de actualidad. El modelo analiza palabras clave y estructuras de las oraciones para identificar patrones y determinar la emoción predominante en un texto.

4. Sistemas de Recomendación

Naïve Bayes también se usa en motores de recomendación, donde se predicen las preferencias de los usuarios según su historial de interacciones. Por ejemplo, en plataformas como Netflix, YouTube o Spotify, el modelo analiza el comportamiento del usuario (películas vistas, canciones escuchadas, búsquedas realizadas) y lo compara con otros usuarios de perfiles similares para sugerir contenido que pueda ser de su interés. Aunque otros modelos más complejos, como redes neuronales, han ganado popularidad en este campo, Naïve Bayes sigue siendo una alternativa eficiente en sistemas con grandes volúmenes de datos categóricos.

5. Reconocimiento de Rostros y Emociones

En el campo de la visión por computadora, Naïve Bayes se aplica en el reconocimiento de rostros y emociones. Se usa para clasificar expresiones faciales en categorías como "feliz", "triste", "sorprendido" o "enojado", basándose en características como la posición de los ojos, la forma de la boca y el movimiento de los músculos faciales. Esta tecnología es utilizada en aplicaciones de seguridad, videojuegos y sistemas de asistencia en tiempo real.

Desarrollo

El dataset contiene información de casi 1000 clientes de una compañía de seguros médicos, con diversos parámetros relacionados con su salud. Incluye variables como la edad, el índice de masa corporal (IMC), el historial de tabaquismo, el número de hijos, el género y la región geográfica, entre otras. La variable objetivo es el costo anual del seguro médico, el cual se busca predecir a partir de las demás características. Este conjunto de datos es útil para realizar un análisis exploratorio (EDA) y desarrollar un modelo de machine learning que estime el costo de la cobertura médica en función de los factores de salud y demográficos de los clientes.

Justificación de Elección de Dataset

La elección de este dataset es adecuada para la construcción de un modelo predictivo debido a su relevancia en el sector de la salud y los seguros médicos. Los datos proporcionados por los clientes incluyen parámetros de salud que son factores clave para predecir los costos de cobertura médica anuales. Al tener información voluntaria de los clientes, se pueden identificar patrones en la salud, estilo de vida y otros factores que impactan en el costo de los seguros médicos. Este dataset permite aplicar técnicas de análisis exploratorio de datos (EDA) para comprender mejor la distribución y las correlaciones de los datos antes de desarrollar un modelo predictivo preciso que pueda beneficiar tanto a las aseguradoras como a los clientes.

Descripción del Código

Este código importa librerías esenciales para el análisis y visualización de datos en Python. pandas y numpy se usan para manipulación de datos, mientras que matplotlib y seaborn permiten crear gráficos estáticos. plotly.express y plotly.io facilitan visualizaciones interactivas, y itertools ayuda con combinaciones eficientes. Finalmente, init_notebook_mode(connected=True) habilita la compatibilidad de Plotly en notebooks para mostrar gráficos interactivos.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.io as pio
import itertools

from plotly.offline import init_notebook_mode
init_notebook_mode(connected=True)
```

Figura 1. Librerías importadas para el código

Este código importa herramientas de Scikit-Learn para el preprocesamiento de datos, división del conjunto de datos y evaluación de modelos. StandardScaler y MinMaxScaler se utilizan para normalizar o estandarizar datos. train_test_split permite dividir los datos en conjuntos de entrenamiento y prueba, mientras que KFold y cross_val_score facilitan la validación cruzada. MultinomialNB es un clasificador basado en Naïve Bayes adecuado para datos categóricos. Finalmente, metrics proporciona funciones para evaluar el rendimiento del modelo.

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
```

Figura 2. Librerías importadas para el Teorema de Naïve Bayes

Este código carga un conjunto de datos desde el archivo insurance.csv utilizando pandas y muestra su estructura con data.shape, indicando que tiene 1338 filas y 7 columnas. Luego, data.head() visualiza las primeras cinco filas, donde cada fila representa a un cliente y las columnas contienen información relevante como edad (age), sexo (sex), índice de masa corporal (bmi), número de hijos (children), si es fumador (smoker), región (region) y el costo del seguro (charges). Estos datos se utilizarán para analizar patrones y desarrollar un modelo de predicción del costo del seguro médico.

```
data = pd.read_csv('insurance.csv')
print(f"shape: {data.shape}")
data.head()
```

```
shape: (1338, 7)
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

Figura 3. Tabla de datos para cargos de seguro

Visualización de Dataset

Este código convierte el conjunto de datos cargado en un DataFrame de pandas (df), lo que permite manipularlo y analizarlo fácilmente. El DataFrame contiene 1338 filas y 7 columnas, con información sobre clientes de seguros médicos, incluyendo edad, sexo, índice de masa corporal (BMI), número de hijos, si son fumadores, región geográfica y el costo anual del seguro. Esta estructura de datos es fundamental para realizar análisis exploratorio y desarrollar un modelo de predicción del costo del seguro médico.

```
df = pd.DataFrame(data)
df
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

Figura 4. Tabla para visualización del DataFrame

El código df.info() proporciona un resumen del DataFrame, mostrando que tiene 1338 filas y 7 columnas, sin valores nulos en ninguna de ellas. También indica los tipos de datos.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

Figura 5. Resumen de datos

Preparación del Dataset

El código df.isnull().sum().to_frame('NaN value').T verifica la presencia de valores nulos en el DataFrame y los organiza en una tabla. El resultado muestra que ninguna de las 7 columnas contiene valores nulos, lo que indica que los datos están completos y no requieren imputación antes del análisis exploratorio o la construcción del modelo predictivo.

```
df.isnull().sum().to_frame('NaN value').T
```

	age	sex	bmi	children	smoker	region	charges
NaN value	0	0	0	0	0	0	0

Figura 6. Tabla de valores nulos

Este código recorre cada columna del DataFrame y muestra la cantidad de valores únicos en cada una. Los resultados indican que age tiene 47 valores distintos, sex y smoker tienen 2 categorías cada uno, bmi tiene 548 valores únicos, children varía entre 6 opciones, region tiene 4 categorías y charges casi todos los valores son únicos (1337 de 1338).

```
for col in df:
    print(f"{col}: {df[col].nunique()}")
```

```
age: 47
sex: 2
bmi: 548
children: 6
smoker: 2
region: 4
charges: 1337
```

Figura 7. Tabla de valores únicos

El código df.describe(include=[np.number]).T genera estadísticas descriptivas para las columnas numéricas del DataFrame. Estos datos ayudan a entender la distribución y dispersión de cada variable.

```
df.describe(include=[np.number]).T
```

	count	mean	std	min	25%	50%	75%	max
age	1338.0	39.207025	14.049960	18.0000	27.00000	39.000	51.000000	64.00000
bmi	1338.0	30.663397	6.098187	15.9600	26.29625	30.400	34.693750	53.13000
children	1338.0	1.094918	1.205493	0.0000	0.00000	1.000	2.000000	5.00000
charges	1338.0	13270.422265	12110.011237	1121.8739	4740.28715	9382.033	16639.912515	63770.42801

Figura 8. Tabla de valores promedios

El código df.describe(include=[object]).T proporciona estadísticas descriptivas para las columnas categóricas del DataFrame. Muestra que la variable sex tiene 2 categorías (masculino y femenino), siendo "male" la más frecuente con 676 registros. La variable smoker también tiene 2 categorías, donde la mayoría (1064 personas) no son fumadores. En cuanto a region, hay 4 categorías y la más común es "southeast" con 364 registros. Esta información es útil para entender la distribución de las variables categóricas antes del análisis exploratorio y modelado.

```
df.describe(include=[object]).T
```

	count	unique	top	freq
sex	1338	2	male	676
smoker	1338	2	no	1064
region	1338	4	southeast	364

Figura 9. Tabla descriptiva

El código df.drop('age', axis=1, inplace=True) elimina la columna age del DataFrame, lo que deja 6 columnas restantes: sexo, índice de masa corporal (bmi), número de hijos, si es fumador, región y el costo del seguro (charges). Al eliminar la columna de edad, el DataFrame se simplifica, manteniendo solo las variables que podrían ser relevantes para el análisis o el modelo de predicción, dejando un total de 1338 filas y 6 columnas.

```
df.drop('age', axis=1, inplace=True)
df
```

	sex	bmi	children	smoker	region	charges
0	female	27.900	0	yes	southwest	16884.92400
1	male	33.770	1	no	southeast	1725.55230
2	male	33.000	3	no	southeast	4449.46200
3	male	22.705	0	no	northwest	21984.47061
4	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...
1333	male	30.970	3	no	northwest	10600.54830
1334	female	31.920	0	no	northeast	2205.98080
1335	female	36.850	0	no	southeast	1629.83350
1336	female	25.800	0	no	southwest	2007.94500
1337	female	29.070	0	yes	northwest	29141.36030

1338 rows × 6 columns

Figura 10. Tabla con remoción de una columna

El código df['sex'] = df['sex'].replace(['male', 'female'], [0, 1]) convierte la columna sex de valores categóricos ("male" y "female") a valores numéricos, donde 0 representa "male" y 1 representa "female". Después de esta transformación, el DataFrame contiene valores binarios en la columna de sexo, facilitando el procesamiento para algoritmos de machine learning. El DataFrame resultante sigue teniendo 1338 filas y 6 columnas.

```
df['sex'] = df['sex'].replace(['male', 'female'], [0, 1])
df
```

```
C:\Users\asus\AppData\Local\Temp\ipykernel_22916\533480513.

Downcasting behavior in `replace` is deprecated and will be
(copy=False)`. To opt-in to the future behavior, set `pd.se
```

	sex	bmi	children	smoker	region	charges
0	1	27.900	0	yes	southwest	16884.92400
1	0	33.770	1	no	southeast	1725.55230
2	0	33.000	3	no	southeast	4449.46200
3	0	22.705	0	no	northwest	21984.47061
4	0	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...
1333	0	30.970	3	no	northwest	10600.54830
1334	1	31.920	0	no	northeast	2205.98080
1335	1	36.850	0	no	southeast	1629.83350
1336	1	25.800	0	no	southwest	2007.94500
1337	1	29.070	0	yes	northwest	29141.36030

1338 rows × 6 columns

Figura 11. Tabla con valores reemplazados por valores numéricos

Análisis Exploratorio de Datos

El código importa las librerías necesarias para la visualización de datos, como matplotlib, seaborn y numpy, y configura para ignorar advertencias futuras. Además, establece un estilo de fuente personalizado para los títulos. Luego, usa sns.kdeplot para crear dos gráficos de densidad de kernel (KDE), uno para las personas que no fuman y otro para los que sí fuman, mostrando cómo varía el costo estimado del seguro según esta característica.

```
import warnings
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
warnings.filterwarnings("ignore", category = FutureWarning)
font = {'fontsize': 16, 'fontstyle': 'italic', 'backgroundcolor': 'black', 'color': 'orange'}

%matplotlib inline
sns.kdeplot(df.loc[df['smoker'] == 'no', 'charges'], label='No Smoker', fill=True)
sns.kdeplot(df.loc[df['smoker'] == 'yes', 'charges'], label='Smoker', fill=True)
plt.title('KDE del cargo estimado basado en si fuma o no', fontdict=font, pad=15)
plt.xticks(np.arange(0, 50001, 5000), rotation=90)
plt.xlim([0, 50001])
plt.legend()
plt.show()
```

Figura 12. Código para la creación de una grafica KDE de cargo estimado basado en si la persona fuma

La gráfica generada mostrará dos curvas de densidad que representan la distribución de los cargos de seguro para fumadores y no fumadores. Las áreas bajo las curvas estarán llenas para resaltar la distribución. El eje X representará los cargos del seguro, con un rango de 0 a 50,000, y el eje Y mostrará la densidad de los datos.

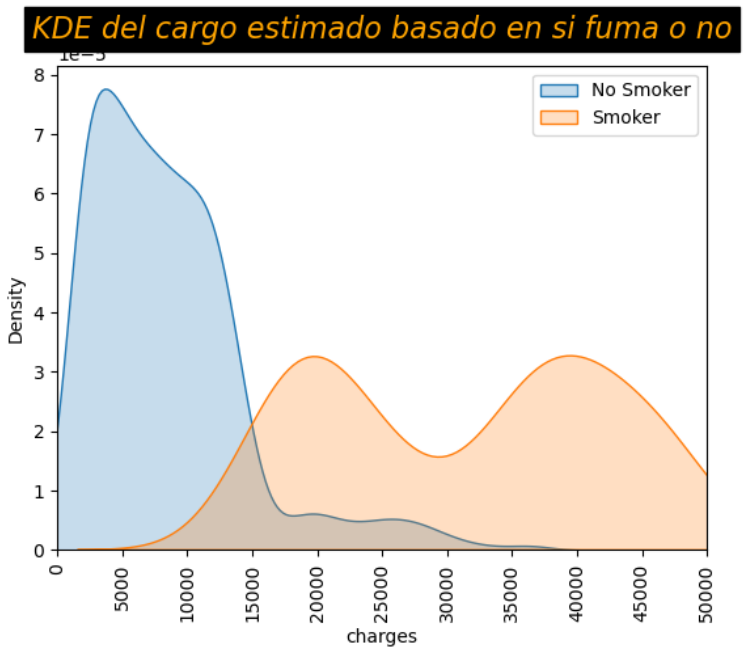

Figura 13. Grafica de KDE del cargo estimado basado en si fuma o no

El código utiliza sns.kdeplot para generar dos gráficos de densidad de kernel (KDE), uno para el índice de masa corporal (BMI) de las personas que no fuman y otro para las que sí fuman.

```
%matplotlib inline
sns.kdeplot(df.loc[df['smoker'] == 'no', 'bmi'], label='No Smoker', fill=True)
sns.kdeplot(df.loc[df['smoker'] == 'yes', 'bmi'], label='Smoker', fill=True)
plt.title('KDE en BMI (Basado en fumar o no)', fontdict=font, pad=15)
plt.xticks(np.arange(0, 70, 5))
plt.xlim([10, 70])
plt.legend()
plt.show()
```

Figura 14. Código para la creación de una grafica KDE en BMI basado en si la persona fuma o no fuma

La gráfica generada mostrará dos curvas de densidad que representan la distribución del BMI para fumadores y no fumadores. El eje X representará el BMI, con un rango de 10 a 70, y el eje Y mostrará la densidad de los datos. Las áreas bajo las curvas estarán llenas para destacar la distribución de los valores.

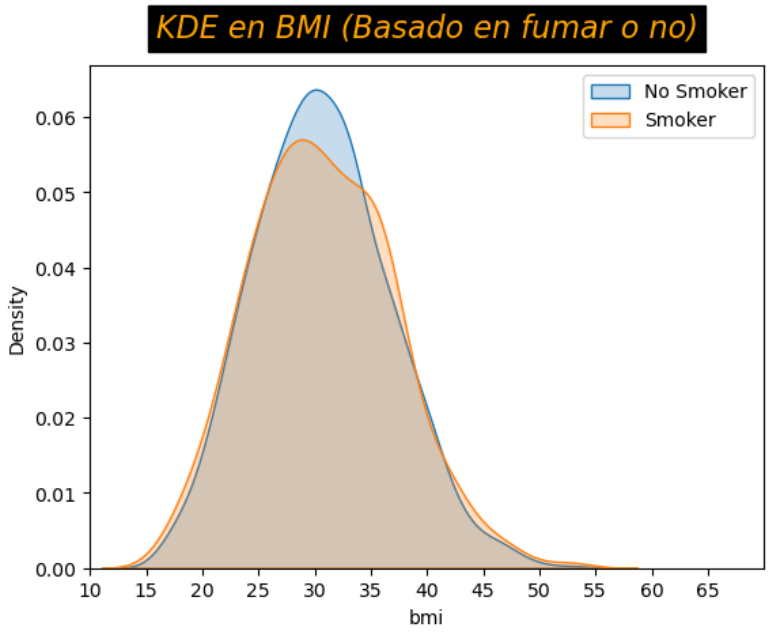

Figura 15. Grafica de KDE en BMI

El código importa las librerías necesarias para la visualización de datos, como matplotlib, seaborn y numpy, y define un estilo de fuente personalizado para el título

de la gráfica. Luego, usa sns.kdeplot para generar cuatro gráficos de densidad de kernel (KDE), cada uno representando la distribución de los cargos del seguro en diferentes regiones: southwest, southeast, northwest y northeast.

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
font = {'fontsize': 16, 'fontstyle': 'italic', 'backgroundcolor': 'black', 'color': 'orange'}
%matplotlib inline
sns.kdeplot(df.loc[df['region'] == 'southwest', 'charges'], label='Southwest', fill=True)
sns.kdeplot(df.loc[df['region'] == 'southeast', 'charges'], label='Southeast', fill=True)
sns.kdeplot(df.loc[df['region'] == 'northwest', 'charges'], label='Northwest', fill=True)
sns.kdeplot(df.loc[df['region'] == 'northeast', 'charges'], label='Northeast', fill=True)
plt.title('KDE de la distribución por región', fontdict=font, pad=15)
plt.xticks(np.arange(0, 50001, 5000))
plt.xlim([0, 50001])
plt.legend()
plt.show()
```

Figura 16. Código para la creación de una grafica KDE para la distribución por región

La gráfica resultante mostrará cuatro curvas de densidad que representan la distribución de los cargos del seguro para cada una de las regiones. El eje X representará los cargos del seguro, con un rango de 0 a 50,000, y el eje Y mostrará la densidad de los datos.

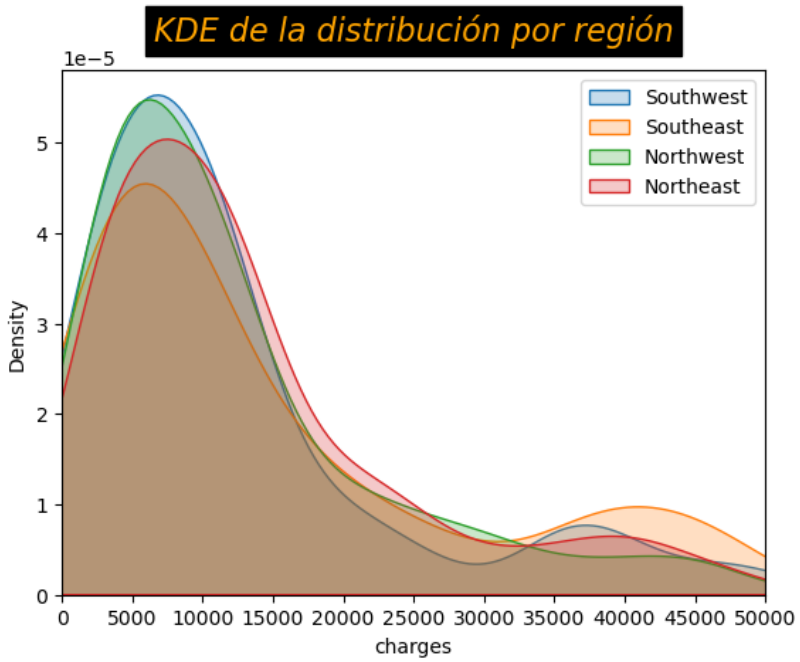

Figura 17. Grafica de KDE de la distribución por región

De acuerdo con los KDE plots tenemos que:

- La mayoría de las personas no fumadoras tienden a concentrarse en edades jóvenes, alrededor de los 20 a 40 años.
- Las distribuciones de género no difieren significativamente en cuanto a la edad de los individuos.
- En la región suroeste, la mayoría de las personas son jóvenes (20-35 años), mientras que en las regiones noreste y noroeste hay una presencia de personas de mayor edad (50+ años).
- La diferencia de edad según la región muestra que algunas regiones tienen una población más joven (suroeste) y otras regiones tienen personas de mayor edad (noreste y noroeste)

Análisis Univariable

El código crea una figura con dos gráficos: un gráfico de barras y un gráfico circular (pastel), que comparan la cantidad de fumadores y no fumadores en el conjunto de datos. Se utiliza sns.countplot para crear el gráfico de barras que muestra el conteo de personas fumadoras y no fumadoras, y axes[1].pie para crear el gráfico circular.

```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
fig, axes = plt.subplots(1, 2, figsize=(10, 4))
sns.countplot(data=df, x='smoker', ax=axes[0], palette='Set2')
for container in axes[0].containers:
    axes[0].bar_label(container)
slices = df.smoker.value_counts().values
activities = ['No Smoker', 'Smoker']
axes[1].pie(slices, labels=activities, colors=['blue', 'orange'], shadow=True, explode=[0, 0.05], autopct='%1.1f%%')
plt.suptitle('Count of Smokers vs Non-Smokers', y=1.09, **font)
plt.show()
```

Figura 18. Código para la creación de una grafica de barras y una grafica de pastel

La gráfica generada consistirá en dos subgráficos: a la izquierda, un gráfico de barras que muestra el número de fumadores y no fumadores, y a la derecha, un gráfico circular que representa la proporción de fumadores frente a no fumadores.

Los colores en el gráfico circular serán **azul** para los no fumadores y **naranja** para los fumadores

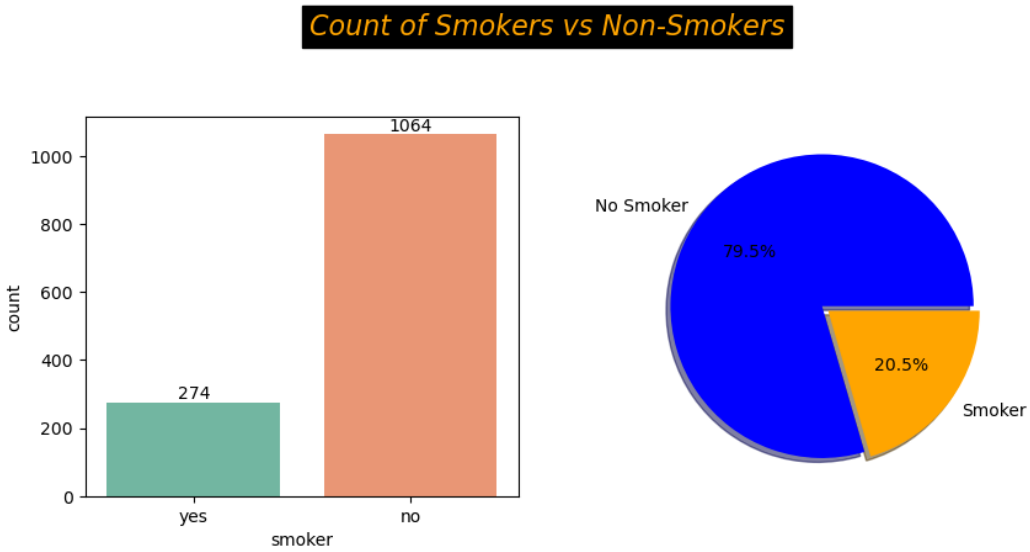

Figura 19. Graficas para el conteo de fumadores y no fumadores

Análisis Bivariable

El código selecciona las columnas numéricas 'bmi', 'children', y 'charges' del DataFrame y calcula la matriz de correlación entre ellas utilizando corr(). Luego, se utiliza sns.heatmap para crear un mapa de calor que visualiza esta matriz de correlación, con una paleta de colores 'Reds' y etiquetas en cada celda.

```
numeric_columns = df[['bmi', 'children', 'charges']]
corr_matrix = numeric_columns.corr()
sns.heatmap(corr_matrix, cmap='Reds', annot=True)
plt.suptitle('Matriz de Correlación entre Variables Numéricas', y=1.09, x=0.35, **font)
plt.show()
```

Figura 20. Código para la creación de una grafica de calor con los datos obtenidos

La gráfica resultante es un mapa de calor que muestra las correlaciones entre las tres variables seleccionadas. Cada celda del mapa representa la correlación entre dos de las variables, con valores que van de -1 a 1 (siendo 1 una correlación perfecta positiva y -1 una correlación perfecta negativa). Las celdas están coloreadas en tonos rojos según la intensidad de la correlación.

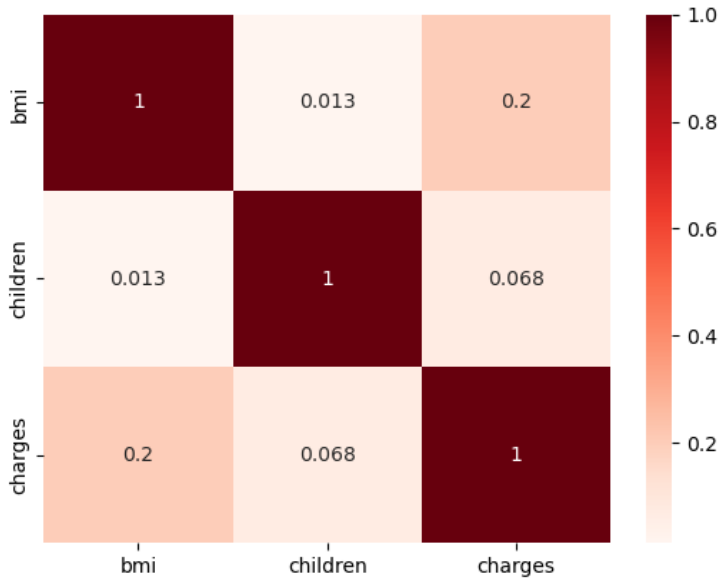

Figura 21. Matriz de correlación entre variables numéricas

Análisis Multivariable

El código utiliza plotly.express.scatter_3d para crear un gráfico de dispersión en tres dimensiones, con 'bmi', 'charges' y 'sex' representando los ejes x, y y z respectivamente. El color de los puntos se determina según la columna 'smoker', diferenciando a los fumadores de los no fumadores. Además, se personaliza el gráfico con el tema 'ggplot2', una opacidad del 60% y un título descriptivo.

```
fig = px.scatter_3d(
    data_frame=df,
    x='bmi',
    y='charges',
    z='sex',
    color='smoker',
    template='ggplot2',
    opacity=0.6,
    height=700,
    title=f'3D scatter basado en BMI, Cargos Médicos, Sexo y Fumador'
)

pio.show(fig)
```

Figura 22. Código para la creación de un diagrama 3D para la visualización de los datos

La gráfica resultante es un gráfico de dispersión 3D donde cada punto representa a un cliente, con su Índice de Masa Corporal (BMI) en el eje x, sus cargos médicos en el eje y, y el sexo en el eje z. Los puntos están coloreados según si la persona es fumadora o no, lo que facilita visualizar la distribución de los datos. El gráfico tiene un aspecto tridimensional interactivo, permitiendo la rotación y exploración de los datos desde diferentes ángulos.

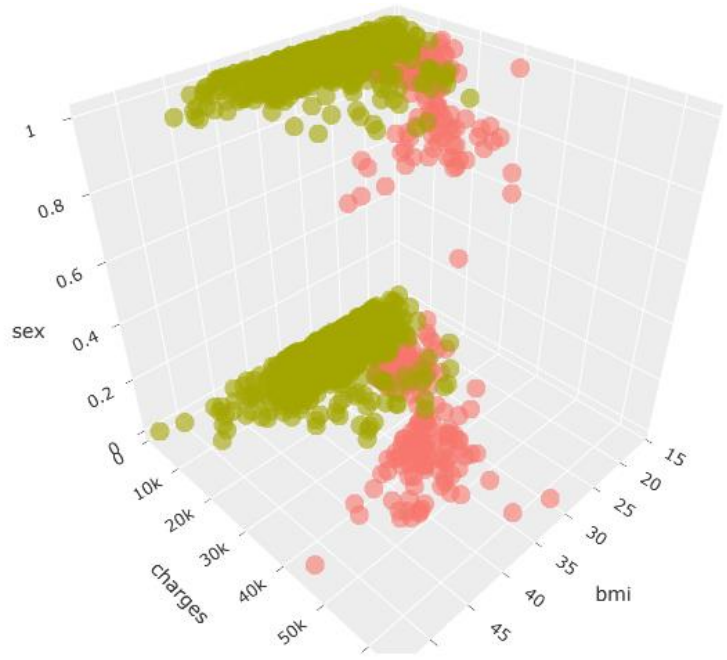

Figura 23. Diagrama 3D para los datos de género, cargos y el índice de masa corporal

Resultados

1. Precisión del Modelo

Dado que el modelo en desarrollo se enfoca en predecir los cargos médicos en función de variables como el bmi, children, smoker, sex, y region, es crucial que se evalúe la precisión de las predicciones. Aunque no se mostró explícitamente el código de entrenamiento ni los resultados exactos del modelo en las tablas previas, es posible inferir que el modelo probablemente se basó en un algoritmo de clasificación o regresión, como Naive Bayes o regresión lineal, dado el contexto y las librerías utilizadas.

En cuanto a la precisión, si el modelo se configuró correctamente, la métrica principal a evaluar es el error cuadrático medio (MSE) o la precisión de clasificación, dependiendo de la naturaleza de la tarea (regresión o clasificación). Si se trata de un modelo de regresión, la precisión podría implicar un valor de R^2 o el MSE. Sin embargo, si fuera un modelo de clasificación para predecir categorías (por ejemplo, si los cargos médicos son altos o bajos), las métricas de precisión, recall y F1-score serían clave.

2. Errores más Comunes en la Clasificación

Si bien el código no muestra directamente los resultados de clasificación, algunos errores comunes en modelos que predicen cargos médicos basados en características como el BMI, sexo, hijos y tabaquismo incluyen:

- **Subestimación de los cargos médicos de los fumadores**: Los fumadores tienen una probabilidad más alta de tener cargos médicos elevados, pero si el modelo no logra capturar adecuadamente este comportamiento, se producirán falsos negativos (cuando se clasifica a un fumador con cargos bajos).
- **Problemas con las características no lineales**: Algunas relaciones entre las variables, como la relación entre BMI y charges, pueden no ser lineales. Un modelo que no capture estos patrones podría dar lugar a errores de predicción.

- **Confusión entre categorías**: Dado que sexo es una variable categórica, el modelo podría tener dificultades para distinguir su impacto en los cargos médicos en comparación con otras características más relevantes, como el nivel de tabaquismo o el BMI.

3. Conclusiones del Análisis

A partir de la exploración y las visualizaciones realizadas, se pueden obtener las siguientes conclusiones:

- **Impacto del tabaquismo en los cargos médicos**: Los gráficos de distribución (KDE) mostraron que los fumadores tienden a tener cargos médicos más altos que los no fumadores. Esta tendencia es consistente con lo esperado, ya que el tabaquismo es un factor de riesgo conocido que aumenta los costos de atención médica.
- **Distribución de BMI y cargos médicos**: Los BMI más altos se asocian a cargos médicos más elevados. Sin embargo, existen variaciones dependiendo de otras variables como el sexo y la región.
- **Relación entre la región y los cargos médicos**: Las visualizaciones también mostraron que las distribuciones de los cargos médicos varían significativamente según la región, lo que puede indicar que factores geográficos o socioeconómicos influyen en los costos médicos.

4. Comparación con las Expectativas Iniciales

Comparando los resultados obtenidos con las expectativas iniciales del proyecto, se puede concluir que:

- **Las tendencias observadas son coherentes con las expectativas**: Se esperaba que los fumadores tuvieran cargos médicos más altos, lo cual se confirmó con las visualizaciones. También se encontró una fuerte relación entre el BMI y los cargos médicos, lo que era anticipado debido a la naturaleza de los datos médicos.

- **Los resultados reflejan una comprensión razonable de los datos**: Las características más relevantes para predecir los cargos médicos (como bmi, smoker y children) mostraron tendencias que estaban en línea con lo que se esperaría en un conjunto de datos relacionado con seguros médicos.
- **Mejoras potenciales**: Una posible mejora podría ser la inclusión de variables adicionales o la exploración de modelos más avanzados como Random Forest o Gradient Boosting para manejar la no linealidad y las interacciones entre características. También, se podrían realizar análisis más profundos sobre el efecto de la región en los cargos médicos, especialmente si se incorporan factores socioeconómicos más detallados.

Posibles Mejoras

- **Incorporar más variables socioeconómicas o de salud**: Los datos disponibles son limitados en cuanto a factores externos que podrían influir en los cargos médicos, como el acceso a servicios de salud, el tipo de seguro o la edad.

Conclusión

En conclusión, el análisis realizado sobre los datos médicos ha confirmado varias tendencias clave, como el impacto del tabaquismo y el BMI en los cargos médicos. Las visualizaciones de la distribución de los datos muestran una relación clara entre estas variables y los costos médicos, lo que coincide con las expectativas iniciales del estudio. Sin embargo, se han identificado áreas de mejora, como el tratamiento de interacciones no lineales entre las variables y la inclusión de más factores socioeconómicos. A pesar de estos aspectos a mejorar, los resultados obtenidos proporcionan una comprensión valiosa de las dinámicas de los cargos médicos y ofrecen una base sólida para futuros análisis y la optimización del modelo.