Taller limpieza de Datos

El proceso de limpieza de datos de este conjunto de datos se puede dividir en varios pasos clave, que te explico a continuación. Estos pasos son comunes para la preparación de datos en un proyecto de Machine Learning:

Descargar el archivo turistas_cartagena_data.csv

```
#leer archivo csv
import pandas as pd
import numpy as np
#omitir futere
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

df=pd.read_csv('turistas_cartagena_data.csv')
print(df.info())
print(df.head())
# Guarda una copia del dataset original antes de comenzar la limpieza.
df_original = df.copy()
```

1. Identificación de Valores Nulos

Primero, hay que identificar los valores nulos o faltantes en las columnas del dataset. Se pueden usar varias estrategias para manejar estos valores:

- **Eliminar filas con valores nulos** (si son pocos y no afectan mucho el análisis).
- **Imputar valores** mediante la media, la mediana, o algún valor específico que tenga sentido en el contexto del dato.
- imputar valores significa **rellenar o reemplazar los valores faltantes** (nulos) en un conjunto de datos con un valor adecuado, en lugar de dejarlos vacíos o eliminarlos por completo. Existen diferentes métodos para imputar estos valores, dependiendo de la naturaleza del dato y del contexto.

Código:

```
#imprimir registros con valores nulos
print(df[df.isnull().any(axis=1)])
```

```
# Identificar valores nulos
df.isnull().sum()
```

```
# Eliminar filas con valores nulos
df_cleaned = df.dropna(how="all")
print(df_cleaned)
```

```
#reemplazar los valores faltantes (nulos) en un conjunto de datos con
mean
df['Número de Visitantes'].fillna(df['Número de Visitantes'].mean(),
inplace=True)
```

2. Eliminación de Filas Duplicadas

Los datos duplicados pueden distorsionar los resultados del modelo. Para eliminarlos:

Código:

```
# Identificar filas duplicadas
duplicados = df[df.duplicated()]
#imprimir data set sin datos duplicados
print(duplicados)
```

```
# Eliminar duplicados
df_cleaned = df.drop_duplicates()
print(df_cleaned)
```

3. Detección de Valores Inconsistentes

En este paso, se deben buscar valores que estén fuera de los rangos aceptables. Por ejemplo, una precipitación negativa no tiene sentido, así que hay que corregir estos valores:

Código:

```
# Identificar valores inconsistentes (precipitación negativa)
df_inconsistentes = df[df['Precipitación (mm)'] < 0]

# Corregir valores inconsistentes (estableciendo a 0 o la media, por
ejemplo)
df.loc[df['Precipitación (mm)'] < 0, 'Precipitación (mm)'] =
df['Precipitación (mm)'].mean()
```

4. Normalización de Datos

La normalización es importante para que los valores de las distintas columnas estén en rangos comparables, especialmente cuando se utilizan modelos de Machine Learning que se ven

afectados por escalas de variables diferentes (por ejemplo, redes neuronales o métodos de distancia).

Dos técnicas comunes de normalización son:

- **Min-Max Scaling**: Escala los datos a un rango entre 0 y 1.
- **Estandarización**: Centra los datos en la media con una desviación estándar de 1.

Código para Min-Max Scaling:

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
df[['Número de Visitantes', 'Temperatura Media (°C)', 'Precipitación (mm)', 'Costo Promedio de Alojamiento (COP)', 'Tasa de Cambio (USD/COP)', 'Cantidad de Vuelos y Cruceros Disponibles', 'Ocupación Hotelera (%)']] = scaler.fit_transform(df[['Número de Visitantes', 'Temperatura Media (°C)', 'Precipitación (mm)', 'Costo Promedio de Alojamiento (COP)', 'Tasa de Cambio (USD/COP)', 'Cantidad de Vuelos y Cruceros Disponibles', 'Ocupación Hotelera (%)']])
```

Código para Estandarización:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df[['Número de Visitantes', 'Temperatura Media (°C)', 'Precipitación (mm)', 'Costo Promedio de Alojamiento (COP)', 'Tasa de Cambio (USD/COP)', 'Cantidad de Vuelos y Cruceros Disponibles', 'Ocupación Hotelera (%)']] = scaler.fit_transform(df[['Número de Visitantes', 'Temperatura Media (°C)', 'Precipitación (mm)', 'Costo Promedio de Alojamiento (COP)', 'Tasa de Cambio (USD/COP)', 'Cantidad de Vuelos y Cruceros Disponibles', 'Ocupación Hotelera (%)']])
```

5. Verificación de los Datos Limpiados

Después de realizar las transformaciones y limpieza, es importante revisar el dataset final para asegurarse de que todos los problemas se hayan resuelto correctamente.

Código:

```
# Revisar la limpieza final
df_cleaned.info()
df_cleaned.describe()
```

Resumen del Proceso:

1. **Identificar y manejar valores nulos**: rellenar o eliminar.
2. **Eliminar duplicados**: verificar filas repetidas y eliminarlas.
3. **Corregir valores inconsistentes**: manejar valores fuera de rango.
4. **Normalizar los datos**: ajustar las escalas de las variables.
5. **Verificar la limpieza**: revisar la estructura final de los datos.

Comparación entre el dataset original y el limpiado

Puedes comparar los dos DataFrames (el original y el limpio) usando varias técnicas. Algunas ideas incluyen:

a) Comparar las dimensiones

Comparar el número de filas y columnas de ambos datasets.

```
print(f"Dimensiones originales: {df_original.shape}")
print(f"Dimensiones después de la limpieza: {df_cleaned.shape}")
```

b) Comparar filas duplicadas eliminadas

Si eliminaste filas duplicadas, puedes ver cuántas filas fueron eliminadas.

```
duplicados_eliminados = df_original.shape[0] - df_cleaned.shape[0]
print(f"Número de duplicados eliminados: {duplicados_eliminados}")
```

c) Comparar valores nulos antes y después

Puedes ver cuántos valores nulos había en el dataset original y cuántos hay después de la limpieza.

```
print("Valores nulos antes de la limpieza:")
print(df_original.isnull().sum())

print("\nValores nulos después de la limpieza:")
print(df_cleaned.isnull().sum())
```

d) Resumen estadístico antes y después de la limpieza

Puedes comparar estadísticas descriptivas (media, mediana, mínimo, máximo, etc.) antes y después de la limpieza para ver cómo cambian los datos.

```
pd.set_option('display.max_columns', None)  # Mostrar todas las
columnas
pd.set_option('display.width', 1000)        # Ajustar el ancho de la
salida
```

```
print("Estadísticas del dataset original:")
print(df_original.describe())

print("\nEstadísticas del dataset limpio:")
print(df_cleaned.describe())
```