Actividad de Data Wrangling con Python y Pandas

En esta actividad, vamos a trabajar con el dataset de diabetes, que está disponible públicamente. El objetivo es practicar las funcionalidades de Pandas para detectar y resolver problemas comunes en los datos como valores en blanco, datos anómalos y realizar limpieza y procesamiento de datos.

Paso 1: Descargar el Dataset

Vamos a utilizar el dataset de diabetes disponible en la biblioteca de datasets de Scikit-Learn. Para esto, primero necesitamos descargarlo y convertirlo a un DataFrame de Pandas.

Código Python:

```
from sklearn.datasets import load_diabetes
import pandas as pd

# Cargar el dataset
diabetes = load_diabetes()

# Convertir a DataFrame de Pandas
df = pd.DataFrame(data=diabetes.data, columns=diabetes.feature_names)
df['target'] = diabetes.target

# Mostrar las primeras filas del DataFrame
print(df.head())
```

Paso 2: Detectar Valores en Blanco

Los valores en blanco pueden causar problemas en el análisis y modelado de datos, por lo que es importante identificarlos y manejarlos adecuadamente.

Código Python:

```
# Detectar valores en blanco
missing_values = df.isnull().sum()
print("Valores en blanco por columna:\n", missing_values)
```

Paso 3: Detectar Datos Anómalos

Los datos anómalos pueden ser valores extremos o fuera de rango que pueden distorsionar el análisis. Vamos a identificar estos valores.

Código Python:

```
import numpy as np

# Definir función para detectar valores anómalos
def detect_outliers(df, n, features):
    outlier_indices = []
    for col in features:
        # 1er cuartil (25%)
        Q1 = np.percentile(df[col], 25)
        # 3er cuartil (75%)
        Q3 = np.percentile(df[col], 75)
        # Rango intercuartil (IQR)
        IQR = Q3 - Q1
```

```
        # Rango límite
        outlier_step = 1.5 * IQR
        # Detectar índices de valores anómalos
        outlier_list_col = df[(df[col] < Q1 - outlier_step) | (df[col] > Q3 +
outlier_step)].index
        outlier_indices.extend(outlier_list_col)
    outlier_indices = Counter(outlier_indices)
    multiple_outliers = list(k for k, v in outlier_indices.items() if v > n)
    return multiple_outliers

# Detectar valores anómalos en todas las columnas
outliers = detect_outliers(df, 2, df.columns[:-1])
print("Índices de valores anómalos:\n", outliers)
```

Paso 4: Manejar Valores en Blanco

Si se encontraron valores en blanco, podemos optar por eliminarlos o imputarlos con valores como la media o la mediana.

Código Python:

```
# Imputar valores en blanco con la mediana de la columna
df.fillna(df.median(), inplace=True)

# Verificar que no hay valores en blanco
print("Valores en blanco después de imputación:\n", df.isnull().sum())
```

Paso 5: Manejar Datos Anómalos

Podemos decidir eliminar los datos anómalos o transformarlos dependiendo del contexto.

Código Python:

```
# Eliminar filas con valores anómalos
df_cleaned = df.drop(outliers, axis=0).reset_index(drop=True)

# Verificar la forma del DataFrame después de limpiar datos anómalos
print("Forma del DataFrame después de eliminar valores anómalos:",
df_cleaned.shape)
```

Paso 6: Verificar Tipos de Datos y Normalizar

Es importante asegurarnos de que todos los datos están en el formato correcto y, si es necesario, normalizarlos.

Código Python:

```
# Verificar tipos de datos
print("Tipos de datos:\n", df_cleaned.dtypes)

# Normalizar los datos (opcional, según sea necesario para el análisis o
modelado)
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df_cleaned[df_cleaned.columns[:-1]] =
scaler.fit_transform(df_cleaned[df_cleaned.columns[:-1]])
```

```
# Ver las primeras filas del DataFrame limpio y normalizado
print(df_cleaned.head())
```

Conclusión

En esta actividad, hemos descargado un dataset de diabetes, detectado y manejado valores en blanco y datos anómalos, y normalizado los datos. Estas son tareas comunes en el data wrangling que son fundamentales para preparar los datos para el análisis y modelado. Practicar estas habilidades es esencial para cualquier profesional de datos.