Sesión 4: Diccionarios y Dataframes

Diccionarios

Los diccionarios son una estructura de datos en Python que permite almacenar pares clave-valor. Son útiles cuando se necesita acceder a valores a través de una clave única.

Creación y acceso a elementos

Los diccionarios se crean utilizando llaves `{}` y separando las claves y los valores con dos puntos `:`. Para acceder a los elementos de un diccionario, se utiliza la clave entre corchetes `[]`.

```
person = {
    "name": "John",
    "age": 30,
    "city": "New York"
}

print(person["name"])  # John
print(person["age"])  # 30
```

Métodos y operaciones básicas

Los diccionarios en Python tienen varios métodos y operaciones para facilitar su manejo:

```
person = {
    "name": "John",
    "age": 30,
    "city": "New York"
}

# Añadir un nuevo par clave-valor
person["country"] = "USA"

# Modificar el valor asociado a una clave
person["age"] = 31

# Eliminar un par clave-valor
del person["city"]

# Verificar si una clave está en el diccionario
print("name" in person)  # True

# Obtener todas las claves del diccionario
keys = person.keys()
print(keys)

# Obtener todos los valores del diccionario
values = person.values()
print(values)

# Obtener todos los pares clave-valor del diccionario
```

```
items = person.items()
print(items)
```

Comprensión de diccionarios

La comprensión de diccionarios es una forma concisa de crear diccionarios a partir de iteraciones.

```
squares = {x: x**2 for x in range(1, 6)}
print(squares)  # {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

Introducción a Dataframes con Pandas

Pandas es una biblioteca de Python que proporciona estructuras de datos y herramientas de análisis de datos. Es especialmente útil para manipular y analizar datos tabulares, como hojas de cálculo o bases de datos.

Instalación y uso de la biblioteca Pandas

Para instalar Pandas, se puede utilizar el siguiente comando:

```
pip install pandas
```

Una vez instalado, se puede importar la biblioteca y comenzar a utilizarla.

```
import pandas as pd
```

Creación y manipulación de Dataframes

Un DataFrame es una estructura de datos bidimensional en Pandas que puede contener múltiples columnas y filas. Se pueden crear DataFrames a partir de diferentes fuentes de datos, como listas, diccionarios o archivos CSV.

```
import pandas as pd

# Creación de un DataFrame a partir de un diccionario
data = {
    "name": ["Alice", "Bob", "Charlie"],
    "age": [25, 30, 35],
    "city": ["New York", "San Francisco", "Los Angeles"]
}

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

Operaciones básicas: selección, filtrado y agregación

Pandas proporciona varias funciones y métodos para realizar operaciones básicas en DataFrames, como seleccionar columnas y filas, filtrar datos y agregar información.

```
import pandas as pd
```

```python

data = {
    "name": ["Alice", "Bob", "Charlie"],
    "age": [25, 30, 35],
    "city": ["New York", "San Francisco", "Los Angeles"]
}

df = pd.DataFrame(data)

# Seleccionar una columna
ages = df["age"]
print(ages)

# Seleccionar varias columnas
columns = df[["name", "city"]]
print(columns)

# Seleccionar filas por índice
row = df.loc[0]  # Primera fila
print(row)

# Filtrar filas por condición
older_than_25 = df[df["age"] > 25]
print(older_than_25)

# Ordenar filas por columna
sorted_by_age = df.sort_values("age")
print(sorted_by_age)

# Agregar una nueva columna
df["country"] = ["USA", "USA", "USA"]
print(df)

# Calcular estadísticas básicas
mean_age = df["age"].mean()
print(mean_age)

sum_age = df["age"].sum()
print(sum_age)

# Agrupar filas por columna y calcular agregaciones
grouped_by_city = df.groupby("city")["age"].mean()
print(grouped_by_city)
```

Ejemplo de análisis exploratorio de datos

¡Por supuesto! Aquí tienes un tutorial paso a paso para realizar un Análisis Exploratorio de Datos (EDA) en el conjunto de datos del Titanic de Kaggle utilizando pandas.

Paso 1: Configuración del entorno

Para empezar, necesitarás tener Python instalado en tu máquina. También necesitarás las bibliotecas pandas, numpy, matplotlib y seaborn. Si no las tienes instaladas, puedes hacerlo utilizando pip:

```
pip install pandas numpy matplotlib seaborn
```

Paso 2: Descarga del conjunto de datos

El conjunto de datos del Titanic está disponible en Kaggle. Para descargarlo, sigue estos pasos:

1. Ve a la página del conjunto de datos del Titanic en Kaggle: https://www.kaggle.com/c/titanic/data
2. Si no tienes una cuenta de Kaggle, tendrás que crear una. Es gratis y solo toma unos minutos.
3. Una vez que hayas iniciado sesión, encontrarás la opción para descargar el conjunto de datos. Haz clic en "Download" (descargar) y guárdalo en un lugar en el que puedas encontrarlo fácilmente.

Paso 3: Importar las bibliotecas necesarias

Antes de empezar, necesitamos importar las bibliotecas que vamos a utilizar. Abre tu editor de código favorito, crea un nuevo archivo Python y escribe el siguiente código:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Paso 4: Cargar el conjunto de datos

Ahora, cargaremos el conjunto de datos del Titanic que descargaste en el Paso 2. Asegúrate de tener la ruta correcta del archivo.

```
df = pd.read_csv('ruta/a/tu/archivo/train.csv')
```

Paso 5: Iniciar el Análisis Exploratorio de Datos

¡Ahora la diversión comienza! Vamos a explorar nuestro conjunto de datos.

Resumen de los datos

Primero, vamos a obtener una descripción general de nuestros datos utilizando los métodos `info()` y `describe()`.

```
df.info()
df.describe()
```

Verificar valores nulos

Es importante saber si nuestro conjunto de datos tiene algún valor nulo. Para eso, podemos usar el método `isnull()` combinado con `sum()`.

```
df.isnull().sum()
```

Análisis de variables categóricas

A continuación, vamos a examinar cómo se distribuyen nuestras variables categóricas. Usaremos seaborn para crear un gráfico de conteo.

```
sns.countplot(x='Sex', data=df)
```

Análisis de la tasa de supervivencia

Después, vamos a explorar la tasa de supervivencia en función de diferentes categorías. Para esto, crearemos un gráfico de barras.

```
sns.barplot(x='Sex', y='Survived', data=df)
```

Correlación entre las variables

Finalmente, vamos a examinar la relación entre nuestras diferentes variables. Para esto, crearemos un mapa de calor de correlación.

```
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
```

¡Y eso es todo! Has completado un análisis exploratorio básico del conjunto de datos del Titanic. Recuerda que este es solo el comienzo. Dependiendo de los resultados de estas exploraciones, puedes profundizar más en ciertas áreas o incluso hacer ingeniería de características para

crear nuevas variables que podrían ser útiles para predecir la supervivencia.

La ciencia de datos es un proceso iterativo y exploratorio. Te invito a que experimentes con estos datos, pruebes nuevos enfoques y veas qué puedes descubrir. ¡Feliz análisis de datos!

Ejercicios prácticos de diccionarios y Dataframes

1. Crear un diccionario que represente a una persona con nombre, edad, ciudad y país, y luego acceder e imprimir cada uno de los valores utilizando las claves del diccionario.
2. Crear un DataFrame a partir de una lista de diccionarios que representen personas, donde cada diccionario contenga información sobre el nombre, la edad y la ciudad de cada persona.
3. Filtrar el DataFrame creado en el ejercicio 2 para mostrar solo las personas mayores de 25 años y que vivan en una ciudad específica.
4. Agregar una columna al DataFrame del ejercicio 2 que represente el salario de cada persona y luego calcular el salario promedio de todas las personas.
5. Leer un archivo CSV utilizando Pandas y realizar análisis básicos en el DataFrame resultante, como seleccionar columnas, filtrar filas y calcular agregaciones.