INTRODUCCIÓN A PYTHON

CLASE 12

Contenido

“

Este espacio curricular permite fortalecer el espíritu crítico y la actitud creativa del futuro graduado. Lo alienta a integrar los conocimientos previos, recreando la práctica en el campo real, al desarrollar un proyecto propio.

”

Programación II

Avanzamos con el uso de las librerías, en este apartado, vamos a completar los conocimientos de la librería pandas y complementarlo con MatPloylib para mostrar por medio de gráficos los datos obtenidos

Introducción

Pandas y Matplotlib se utilizan como introducción al análisis de datos en Python.

Un componente central de la toma de decisiones basada en la información es el análisis de datos. Python ofrece un potente conjunto de herramientas para explorar, visualizar y analizar datos mediante el uso de las bibliotecas Pandas y Matplotlib.

Los conjuntos de datos se pueden cargar, limpiar, transformar y se pueden realizar operaciones estadísticas utilizando la biblioteca de manipulación y análisis de datos Pandas. Por otro lado, Matplotlib es una biblioteca de visualización que nos permite hacer gráficos y visualizaciones llamativos y únicos.

Los datos de varias fuentes, como bases de datos o archivos CSV, se pueden cargar y explorar utilizando Pandas utilizando una variedad de funciones y técnicas. Los datos se pueden filtrar, ordenar y agrupar. Se pueden calcular estadísticas descriptivas y ciertas columnas se pueden someter a operaciones numéricas.

Después de preparar nuestros datos, podemos usar Matplotlib para producir gráficos y visualizaciones instructivas. Para comprender mejor nuestros datos y detectar patrones o tendencias, podemos crear gráficos de barras, diagramas de dispersión, histogramas y muchos otros tipos de gráficos.

Filtrado de Dataframe

Puede usar la función loc o el operador de indexación [] con una condición booleana para filtrar un DataFrame en Python usando la biblioteca Pandas. Aquí hay algunas ilustraciones:

1. Aplicar una condición para filtrar filas:

Supongamos que tiene un DataFrame llamado df y desea filtrar las filas donde la columna "edad" es mayor que 30.

```
df_filtrado = df.loc[df['edad'] > 30]
```

2. Las filas se pueden filtrar utilizando una variedad de criterios por:

Supongamos que desea mostrar solo las filas con columnas de "edad" mayores de 30 y "femenino" en la columna "sexo".

```
df_filtrado = df.loc[(df['edad'] > 30) &
(df['sexo'] == 'Femenino')]
```

3. Utilice una condición para filtrar un conjunto de columnas.

Imagina que solo quieres ver las columnas cuyo valor medio es superior a 0,5.

```
columnas_filtradas =
df.columns[df.mean() > 0.5]

df_filtrado = df[columnas_filtradas]
```

Estos son simplemente ejemplos simples, pero puede modificar las condiciones y las columnas para cumplir con sus requisitos. Para acceder a filas y columnas por etiquetas, use la función loc. Para indexar directamente por posición o nombre de columna, use el operador [].

Operaciones con DataFrame

Concatenar

La función concat en Pandas se puede usar para combinar dos DataFrames. Una ilustración fácil de cómo hacerlo se da a continuación:

Los pandas deben importarse como pd.

Tenga en cuenta que tiene dos DataFrames llamados df1 y df2. y desea concatenarlos verticalmente por filas.

Une los marcos de datos.

```
[[df1, df2]] = pd.
concat(df_concatenado).
```

mostrar el DataFrame resultante.

```
print (df_concatenado).
```

Los DataFrames df1 y df2 se combinan en este ejemplo usando la función concat de Pandas. Los proporcionamos a la función concat como una lista [df1, df2]. Por diseño, los DataFrames se apilan verticalmente y la concatenación se realiza por filas.

El parámetro eje, con valor 1, se puede utilizar para concatenar los DataFrames por columnas. Como ilustración, considere esto:

pitón.

Columna por columna únase a los DataFrames.

```
p.d. concat ([df1, df2], axis=1)
```

Los DataFrames se concatenarán horizontalmente en esta instancia, colocando las columnas una al lado de la otra.

Merge

La función de fusión en Pandas se puede usar para combinar o fusionar dos DataFrames. Aquí hay una ilustración simple de cómo hacerlo:

```
import pandas as pd

df=pd.merge(df1,df2, on='col_comun')

print(df)
```

En este ejemplo, los marcos de datos df1 y df2 se combinan mediante la función de combinación de Pandas. Usando el parámetro on, identificamos la columna común en la que ejecutar la combinación. Puede cambiar el nombre de la columna para adaptarlo a sus requisitos.

Existen numerosos tipos de combinación que se pueden usar, incluida la combinación interna, la combinación externa, la combinación izquierda y la combinación derecha. Con la ayuda del parámetro cómo, puede regular el tipo de combinación. Una combinación interna se ejecuta de forma predeterminada.

A continuación, se muestra una ilustración de una combinación izquierda:

```
df_combinado=pd.merge(df1,df2,
on=c_comun', how='left')
```

Tenga en cuenta que es crucial asegurarse de que las columnas o los índices que se unen sean del mismo tipo de datos y tengan valores idénticos antes de combinar DataFrames. Además, al usar una lista en el parámetro on, Pandas permite combinar DataFrames en función de varias columnas.

Consideraciones

Para aplicar ciencia de datos utilizando la biblioteca Pandas en Python, puedes seguir los siguientes pasos:

1. Importar Pandas: Comienza importando la biblioteca Pandas en tu script de Python utilizando la siguiente línea de código:

```
import pandas as pd
```

2. Cargar los datos: Utiliza la función `read_csv()` de Pandas para cargar tus datos desde un archivo CSV u otros formatos compatibles. Por ejemplo:

```
data = pd.read_csv('archivo.csv')
```

3. Exploración de los datos: Utiliza diversas funciones y métodos de Pandas para explorar y comprender tus datos. Algunas operaciones comunes incluyen:

- `head()`: Muestra las primeras filas del DataFrame.
- `info()`: Proporciona información sobre el DataFrame, como el tipo de datos y el recuento de valores no nulos.
- `describe()`: Calcula estadísticas descriptivas para columnas numéricas.
- "value_counts()`: Cuenta los valores únicos en una columna.

4. Preprocesamiento de datos: Lleva a cabo tareas de limpieza y preparación de datos para garantizar que estén en un formato adecuado para su análisis. Esto puede incluir el manejo de valores faltantes, la eliminación de duplicados, la codificación de variables categóricas, la normalización de datos, etc.

5. Análisis y manipulación de datos: Utiliza las funcionalidades de Pandas para realizar diferentes operaciones en los datos, como filtrado, selección de columnas, agregación, ordenamiento, entre otros. Algunos métodos útiles incluyen:

- `loc[]` y `iloc[]`: Para acceder a filas y columnas específicas del DataFrame.
- Operadores de comparación (`<`, `>`, `==`, etc.): Para realizar filtros basados en condiciones.
- `groupby()`: Para agrupar y resumir datos según una columna o conjunto de columnas.
- `sort_values()`: Para ordenar el DataFrame por una o varias columnas.

6. Visualización de datos: Utiliza la biblioteca Matplotlib, junto con las funcionalidades de visualización integradas de Pandas, para crear gráficos y visualizaciones que ayuden a entender mejor los datos. Algunas funciones útiles son:

- `plot()`: Para crear gráficos básicos, como diagramas de líneas, barras o dispersión.
- `hist()`: Para generar histogramas.
- `boxplot()`: Para crear diagramas de caja y bigotes.
- `heatmap()`: Para crear mapas de calor.

7. Modelado y análisis: Si deseas realizar análisis más avanzados, puedes utilizar otras bibliotecas populares de ciencia de datos en Python, como scikit-learn, StatsModels o TensorFlow, para aplicar algoritmos de aprendizaje automático, modelos estadísticos u otras técnicas analíticas.

Datos abiertos

Los conjuntos de datos conocidos como "conjuntos de datos de datos abiertos" son aquellos que cualquiera puede usar, acceder y compartir sin costo alguno. Estos conjuntos de datos son producidos por una variedad de organizaciones, incluidas agencias gubernamentales, instituciones académicas, grupos sin fines de lucro, empresas con fines de lucro y comunidades de datos abiertos.

Los conjuntos de datos de datos abiertos se emplean para una variedad de funciones, como:

1. Los conjuntos de datos abiertos brindan a los ciudadanos acceso a la información pública y la capacidad de aprender cómo se administran los recursos y las políticas gubernamentales, lo que promueve la transparencia y la rendición de cuentas.

3. Desarrollo de aplicaciones y servicios: los desarrolladores pueden crear aplicaciones y servicios de vanguardia que agregan valor para los usuarios utilizando conjuntos de datos abiertos como fuente de datos.

4. Tomar decisiones bien informadas: los conjuntos de datos abiertos ofrecen datos y estadísticas que pueden ayudar a los tomadores de decisiones en una

variedad de áreas, incluida la formulación de políticas, la planificación urbana, el desarrollo económico y más.

5. Conocimiento y aprendizaje: los conceptos de análisis de datos, estadísticas y programación se pueden enseñar en entornos educativos utilizando conjuntos de datos abiertos.

Los procesos colaborativos, abiertos e innovadores son fomentados por la disponibilidad de conjuntos de datos abiertos. Se fomenta el desarrollo de soluciones innovadoras y el empoderamiento de la comunidad en general al otorgar acceso a datos pertinentes y actualizados.

Los conjuntos de datos de datos abiertos deben cumplir con una serie de estándares, que incluyen ser utilizables, accesibles, libres de restricciones de derechos de autor y respetar la privacidad y seguridad de los datos confidenciales.

Fuentes de datos

datahub.io
Conjuntos de datos, desde datos del mercado financiero y crecimiento de la población hasta precios de criptomonedas.

https://datahub.io/search

Gobierno de la ciudad
https://buenosaires.gob.ar/datosabiertos

World Health Organization
(http://www.who.int/research/en/)

Data.gov (http://data.gov)

European Union Open Data Portal (http://open-data.europa.eu/en/data/)

Amazon Web Service public datasets
(http://aws.amazon.com/datasets)
Facebook Graph
(http://developers.facebook.com/docs/graph-api)

Bibliografía

Python para Todos" de Raúl González Duque. (http://mundogeek.net/tutorial-python/).

Miller, Curtis. (2016). Hands-On: Data Analysis with Numpy and pandas. (1ed).Packt>. Birmingham

recursos

John Hunter, Darren Dale, Eric Firing, Michael Droettboom. (2012). https://matplotlib.org/3.1.0/gallery/style_sheets/style_sheets_reference.html. matplotlib.or,. consultado en 2022

Pandas.pydata.org. (2022). https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html. Consultado en 2022