Python For Data Analysis

Pandas

Imen Ouled Dlala

imen.ouled_dlala@devinci.fr

September 14, 2022

General Plan

1. Introduction
2. Numpy library
3. Pandas library
4. Data analysis and visualization
   - Seaborn, Matplotlib, Bokeh
5. Webscrapping
6. Machine learning and Datasets
   - Scikit-learn, tensorflow
7. API Django / Flask

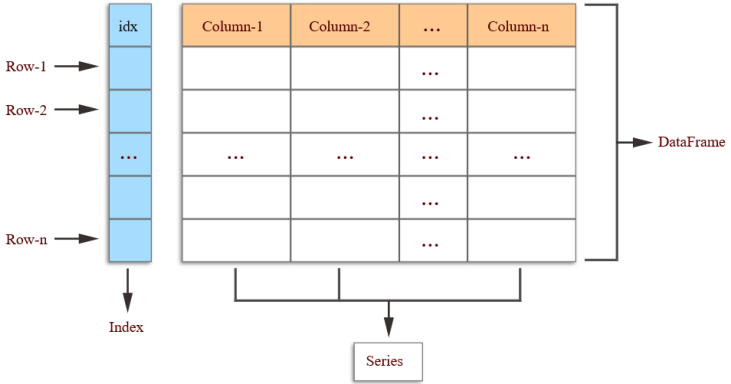
Overview

1. Pandas
   - Getting Started with Pandas
   - DataFrame and Series Basics - Selecting Rows and Columns
   - Indexes - How to Set, Reset, and Use Indexes
   - Filtering - Using Conditionals to Filter Rows and Columns
   - Updating Rows and Columns - Modifying Data Within DataFrames
   - Add/Remove Rows and Columns From DataFrames
   - Data cleaning
   - Pivot_table

Pandas

Pandas Panel Datas

pandas is a Python package providing fast, flexible, and expressive data structures.



Pandas

Getting Started with Pandas

Dataset: https://insights.stackoverflow.com/survey

Create DataFrame:

```
df= pd.read_csv("data/survey_results_public.csv")
df
```

	Respondent	MainBranch	Hobbyist	OpenSourcer	OpenSource	Employment
0	1	I am a student who is learning to code	Yes	Never	The quality of OSS and closed source software ...	Not employed, and not looking for work
1	2	I am a student who is learning to code	No	Less than once per year	The quality of OSS and closed source software ...	Not employed, but looking for work
2	3	I am not primarily a developer, but I write co...	Yes	Never	The quality of OSS and closed source software ...	Employed full-time

Pandas

Getting Started with Pandas

Pandas version:

print(pd.__version__)

Create a DataFrame from a dictionary

```
import pandas as pd

job = [ "journaliste", "capitaine"]
names = ["tintin", "hadock" ]
personage={"job": job, "names": names}
df= pd.DataFrame(personage)
df
```

	job	names
0	journaliste	tintin
1	capitaine	hadock

Pandas

Getting Started with Pandas

- DataFrame.tail(n=5) : Returns the last n rows
- DataFrame.head(n=5) : Returns the first n rows.
- DataFrame.info() : Prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.
- DataFrame.shape() : Returns a tuple representing the dimensionality of the DataFrame.
- DataFrame.describe() : returns the summary statistics for numerical columns

Pandas

DataFrame and Series Basics - Selecting Rows and Columns

DataFrame.columns	Returns the column labels of the DataFrame
DataFrame[col]	Returns column with label col as Series
DataFrame[[col1, col2]]	Returns columns as a new DataFrame
DataFrame.iloc[0]	Selection by position
DataFrame.loc['index_one']	Selection by index
DataFrame.iloc[0,:]	Returns the first row
DataFrame.iloc[0,0]	Returns the first element of first column

Pandas

Indexes - How to Set, Reset, and Use Indexes

DataFrame = pd.read_csv('data.csv',index_col="Column") : Column(s) to use as the row labels of the DataFrame, either given as string name or column index.
DataFrame.set_index(('column_one, inplace=False): Set the DataFrame index using existing columns (column_one).
DataFrame.reset_index(inplace=True) : Reset the index of the DataFrame, and use the default one instead.
DataFrame.sort_index(ascending=True, inplace=True) : Returns a new DataFrame sorted by label if inplace argument is False, otherwise updates the original DataFrame and returns None.

Pandas

Filtering - Using Conditionals to Filter Rows and Columns

The filters return a series of true/false values.

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people['last']=="Doe"
```

```
0    False
1     True
2     True
Name: last, dtype: bool
```

Pandas

Filtering - Using Conditionals to Filter Rows and Columns

We apply the filter to our DataFrame to get all the true values back and not the false

```
filt= df_people['last']=="Doe"
```

```
df_people[filt]
```

	email	first	last
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people.loc[filt]
```

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

Pandas

Filtering - Using Conditionals to Filter Rows and Columns

Otherwise:

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people[df_people['last']=="Schafer"]
```

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer

Pandas

Filtering - Using Conditionals to Filter Rows and Columns

Create a filter to get all the rows where the last name is Doe and the first name is John

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
filt=(df_people['last']=="Doe")& (df_people["first"]=="John")
```

```
df_people[filt]
```

	email	first	last
2	JohnDoe@email.com	John	Doe

Pandas

Filtering - Using Conditionals to Filter Rows and Columns

Display the email(s) column(s) of all the rows where the last name is Doe and the first name is John

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
?]: df_people.loc[filt, 'email']
```

```
2    JohnDoe@email.com
Name: email, dtype: object
```

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating columns: rename all columns

	email	first	last
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people.columns=['email', 'first_name', 'last_name']
```

	email	first_name	last_name
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating columns: uppercase all of the column names

	email	first_name	last_name
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people.columns=[x.upper() for x in df_people.columns]
```

	EMAIL	FIRST_NAME	LAST_NAME
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating columns: replace the underscore with space

	EMAIL	FIRST_NAME	LAST_NAME
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

```
df_people.columns= df_people.columns.str.replace('_', ' ')
```

	EMAIL	FIRST NAME	LAST NAME
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnDoe@email.com	John	Doe

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating some columns: rename some columns

```
df_people.rename(columns={'FIRST_NAME': 'first_name', 'LAST_NAME':'last_name' }, inplace=True)
```

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating rows: updating last name and email

```
df_people.loc[2]=['JohnSmith@email.com', 'John', 'Smith']
```

	EMAIL	first_name	last_name
0	CoreyMSchafer@gmail.com	Corey	Schafer
1	JaneDoe@email.com	Jane	Doe
2	JohnSmith@email.com	John	Smith

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating rows: updating last name and email

```
df_people.loc[2,['EMAIL','last_name']]= ['JohnDoe@email.com', 'Doe']
```

Pandas

Updating Rows and Columns - Modifying Data Within DataFrames

Updating rows: another way to update single values

```
df_people.at[2,'last_name']='Doe'
```

Pandas

Add/Remove Rows and Columns From DataFrames

Adding columns: add a new column called full name

```
df_people['full name']= df_people['last_name']+' '+ df_people['first_name']
```

	EMAIL	first_name	last_name	full name
0	CoreyMSchafer@gmail.com	Corey	Schafer	Schafer Corey
1	JaneDoe@email.com	Jane	Doe	Doe Jane
2	JohnDoe@email.com	John	Smith	Smith John

Pandas

Add/Remove Rows and Columns From DataFrames

Removing columns: Remove first_name and last_name columns

```
df_people.drop(columns=['first_name','last_name'], inplace=True)
```

Pandas

Add/Remove Rows and Columns From DataFrames

Adding rows:

```
df_people.append({"first":"imen"}, ignore_index=True)
```

	EMAIL	full name	first	last
0	CoreyMSchafer@gmail.com	Schafer Corey	Schafer	Corey
1	JaneDoe@email.com	Doe Jane	Doe	Jane
2	JohnDoe@email.com	Smith John	Smith	John
3	NaN	NaN	imen	NaN

Pandas

Add/Remove Rows and Columns From DataFrames

Adding rows: we can add a DataFrame to another DataFrame

```
df_people.append(df_2,ignore_index=True)
```

	EMAIL	full name	first	last
0	CoreyMSchafer@gmail.com	Schafer Corey	Schafer	Corey
1	JaneDoe@email.com	Doe Jane	Doe	Jane
2	JohnDoe@email.com	Smith John	Smith	John
3	ImenDlala@gmail.com	NaN	Imen	Sarah
4	SarahTana@email.com	NaN	Dlala	Tana

Pandas

Add/Remove Rows and Columns From DataFrames

Removing rows:

```
df_people.drop(index=4, inplace=True)
```

```
df_people.drop(index=df_people[df_people['last']=='Jane'].index)
```

Pandas

Data cleaning

pd.isnull()	Checks for null Values, Returns Boolean Arrray
pd.notnull()	Opposite of pd.isnull()
df.dropna()	Drop all rows that contain null values
df.dropna(axis=1)	Drop all columns that contain null values
df.dropna(axis=1,thresh=n)	Drop all rows that have less than n non null values
df.fillna(x)	Replace all null values with

Pandas

Pivot_table

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.pivot_table.html

End

Good Lecture!