Managing CSV Files with Pandas

INTERNSHIPSTUDIO

What is a CSV File?

- A CSV file is the most common, simple, and easiest method to store tabular data.
- This particular format arranges tables by following a specific structure divided into rows and columns. It is these rows and columns that contain your data.
- The core purpose of the CSV format is to help you present the tabular data compactly and concisely.
- A new line terminates each row to start the next row. Similarly, a comma, also known as the delimiter, separates columns within each row.

Example-

City, State, Capital, Population
New York, New York, No, 8.623 Million
Austin, Texas, Yes, 0.95 Million
Miami, Florida, No, 0.463 Million

Reading CSV Files

Once you go through the installation, you can use the read_csv() function to read a CSV file. We will try to read the "nba.csv" file, which we have uploaded earlier.

- First we have to import the Pandas library:

```
import pandas as pd
# reading the csv
data = pd.read_csv("nba.csv")
```

- import pandas as pd

  When we execute this code, it will read the CSV file "nba.csv" from the current directory. You can see from the script above that to read a CSV file, you have to pass the file path to the read_csv() method of the Pandas library. The read_csv() method then returns a Pandas DataFrame that contains the data of the CSV file.

  You can display the first five rows of the CSV file via the head() method of the Pandas DataFrame, as shown below:

```
data.head()
```

Pandas Functions- Data I/O

INTERNSHIPSTUDIO

Function	Usage	Comments
pd.read_csv()	Read CSV file	.csv, .tsv or .txt
pd.read_excel()	Read spreadsheet	.xls or .xlsx
pd.read_json()	Read JSON document	.json
pd.read_sql()	Read directly from DB query	Needs 3rd party library to support (e.g. sqlalchemy)
df.to_csv()	Write to CSV file	.csv, .tsv or .txt
df.to_excel()	Write to Excel file	.xls or .xlsx
df.to_json()	Write to JSON document	.json
df.to_sql()	Write to DB table	Needs 3rd party library to support (e.g. sqlalchemy)

Pandas Functions- Data Preview

Name	Usage	Comments
df.head()	Preview the first n (default=5)	Can define "n" by df.head(n)
df.tail()	Preview the last n (default=5)	Can define "n" by df.tail(n)
df.sort_values()	Sort the data frame on a specific column	Can sort with multiple columns, either ascending or descending
df.columns	Display all column names	Can also set column names by assign a list of column names
df.dtypes	Display data types of the columns	Return a list of types
df.shape	Display the shape of the data frame	Return a tuple: (row_count, col_count)
df.describe()	Show basic stats of each column	Will show different stats for different types of column (numerical, datetime and etc.)
s.value_counts()	Count occurrences of each value	Use df['...'] to get a column of data frame as series

Pandas Functions- Data Cleansing

Name	Usage	Comments
s.isna()	Whether there are null values existing	Boolean
df.dropna()	Delete missing values	Can be applied on a series or a data frame
df.fillna()	Fill missing values with a certain value	Can be applied on a series or a data frame
df.drop_duplicates()	Delete all duplicated values	Can be applied on a series or a data frame
df.drop()	Delete columns or rows	Need to specify axis
df.rename()	Rename columns or rows	Need to specify axis
df.reset_index()	Convert index column to a data column	A new index will be automatically added using incremental integers

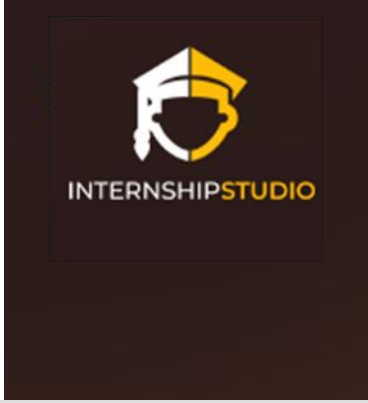
Functions- Data Transformation

INTERNSHIPSTUDIO

Name	Usage	Comments
pd.to_datetime()	Convert string column to datetime	Support customised format
s.astype()	Convert column data type	Need to be careful to NaN values when convert between float/double and integer
s.apply()	Invoke function on values of Series	Very flexible function that you must practice a lot
df.apply()	Apply a function along an axis of the DataFrame	When you need to apply a function on multiple columns
df.explode()	Transform each element of a list-like to a row, replicating the index values	When you used df.apply to return multiple column in a list, then translate them back to columns

Pandas

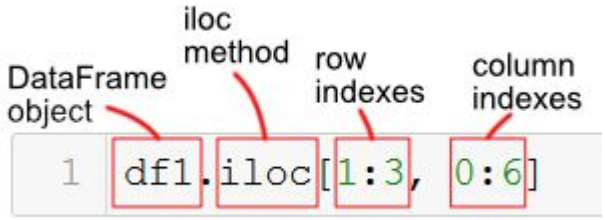
Functions- Data Aggregation

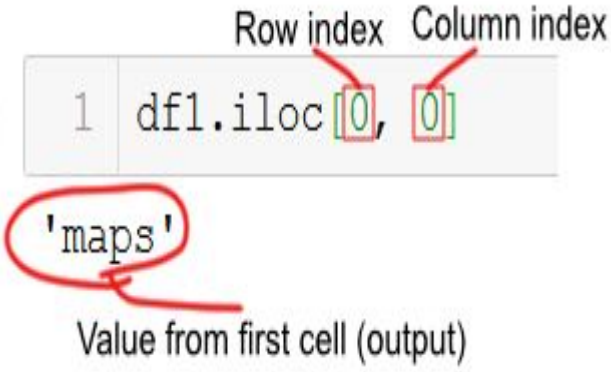


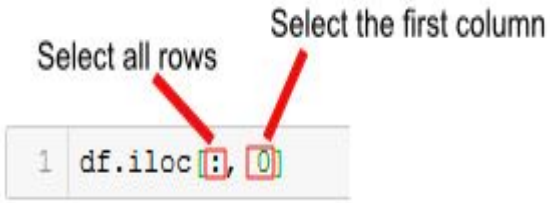
Name	Usage	Comments
pd.concat()	Concatenate two dataframe	Can be horizontal or vertical
pd.merge()	Joining two data frames	Just like joining tables in DB, need to be careful joining methods (left/right/inner)
df.groupby()	Grouping column(s) and then apply more aggregating functions	Very commonly used with agg() function
df.groupby().agg()	Aggregating records for each group	Need to be used along with groupby() function
pd.pivot_table()	Convert data frame to pivot table	May have multi-level indices and columns

Pandas Functions- Slicing

	0	1	2	3	4	5	6	7	8	9
	Study	Film	ext	neur	imp	soc	lie	traitanx	state1	EA1
0	maps	3	18.0	9.0	7.0	10.0	3.0	24.0	22.0	24.0
1	maps	3	16.0	12.0	5.0	8.0	1.0	41.0	40.0	9.0
2	maps	3	6.0	5.0	3.0	1.0	2.0	37.0	44.0	1.0
3	maps	3	12.0	15.0	4.0	6.0	3.0	54.0	40.0	5.0
4	maps	3	14.0	2.0	5.0	6.0	3.0	39.0	67.0	12.0







https://www.sharpsightlabs.com/blog/pandas-iloc/

Pandas

Practice session

- We are going explore many datasets in this last session

Pandas

INTERNSHIPSTUDIO

- What are csv files?
- Write basic code for reading csv files in pandas?
- Write the program for reading titanic.csv file and show only 10 rows?
- How to add only specific column while reading a csv file ?
- Write code for Read csv file to Dataframe with custom delimiter(,) ?

Thank you