Learn the tools of the trade: Pandas, NumPy, Matplotlib, and Seaborn

Predictive Data Analysis with Python (https://www.educative.io/courses/predictive-data-analysis-with-python)

Pandas import convention

```
import pandas as pd
```

Pandas is now accessible with the acronym `pd`. You can also install Pandas using the built-in Python tool pip and run the following command.

```
$ pip install pandas
```

Create and name a Series

Create one-dimensional array to hold any data type. Invoke the `pd.Series()` method and then pass a list of values. Pandas will default count index from 0.

```
series1 = pd.Series([1,2,3,4]), index=['a', 'b', 'c', 'd'])
```

Set the Series name

```
srs.name = "Insert name"
```

Set index name.

```
srs.index.name = "Index name"
```

Create a DataFrame

Create a two-dimensional data structure with columns. Create and print a `df`.

```
df = pd.DataFrame(
         {"a" : [1 ,2, 3],
          "b" : [7, 8, 9],
          "c" : [10, 11, 12]},        index = [1, 2, 3])
```

Specify values in DataFrame columns

Specify how you want to organize your DataFrame by columns.

```
df = pd.DataFrame(
     [[1, 2, 3],
     [4, 6, 8],
     [10, 11, 12]],
     index=[1, 2, 3],
     columns=['a', 'b', 'c'])
```

Read and Write to `csv` file

Open the CSV file, copy the data, paste it in our Notepad, and save it in the same directory that houses your Python scripts. Use `read_csv` function build into Pandas and index it the way we want.

```
import pandas as pd
data = pd.read_csv('file.csv')

data = pd.read_csv("data.csv", index_col=0)
```

Read and write to Excel (https://www.educative.io/blog/excel-python-data-analytics) file

Call the `read_excel` function to access an Excel file. Pass the name of the Excel file as an argument.

```
pd.read_excel('file.xlsx')

df.to_excel('dir/myDataFrame.xlsx',  sheet_name='Sheet2')
```

Read and write to SQL Query (https://www.educative.io/blog/what-is-database-query-sql-nosql)

```
from sqlalchemy import create_engine
engine = create_engine('sqlite:///:memory:')
pd.read_sql("SELECT * FROM my_table;", engine)
pd.read_sql_table('my_table', engine)
pd.read_sql_query("SELECT * FROM my_table;", engine)
```

`(read_sql()` is a convenience wrapper around `read_sql_table()` and `read_sql_query())`

```
df.to_sql('myDf', engine)
```

Get the first element of a Series

Since Pandas indexes at 0, call the first element with `ser[0]`.

```
import pandas as pd

df = pd.read_csv

df['Name'].head(10)

# get the first element
ser[0]
```

Get the first 5 elements of a Series

Use `ser[:n]` to get the first n elements of a Series.

```
import pandas as pd

df = pd.read_csv

df['Name'].head(10)

ser[:5]
```

Get the last 5 elements in a Series

Use `ser[-n:]` to get the last n elements of a Series.

```
import pandas as pd

df = pd.read_csv

df['Name'].head(10)

ser[-5:]
```

Select a single value position

```
df.iloc[[0],[0]] 'Name'
df.iat([0],[0]) 'Name'
```

Select a single value by label

```
df.loc[[0], ['Label']] 'Name'
df.at([0], ['Label']) 'Name'
```

Access a DataFrame with a boolean index

In boolean indexing, we filter data with a boolean vector.

```
import pandas as pd

# dictionary of lists
dict = {'name':["name1", "name2", "name3", "name4"],
        'degree': ["degree1", "degree2", "degree3", "degree4"],
        'score':[1, 2, 3, 4]}

df = pd.DataFrame(dict, index = [True, False, True, False])

print(df)
```

Drop values from rows

```
s.drop(['a', 'c'])
```

Drop values from columns

```
df.drop('Value', axis=1)
```

Create a new column in a DataFrame

```
df['New Column'] = 0
```


Rename columns in a DataFrame

```
df.columns = ['Column 1', 'Column 2', 'Column 3']
```

Sort Series by labels along an axis

Sort Series by index labels and returns a new Series sorted by the label if inplace argument is `False`, otherwise it updates the original series and returns `None`.

```
Series.sort_index(self, axis=0, level=None, ascending=True, inplace=
False, kind='quicksort', na_position='last', sort_remaining=True)
```


Sort values along an axis (ascending order)

```
df.sort_values(by='Values')

# descending order
df.sort_values(ascending = False)
```

Adding ranks to particular entries

Specify how you want to rank a column and add ranks.

```
df.rank()
```

Retrieve rows and columns description

```
df.shape
```

Describe columns of DataFrame

```
df.columns
```

Retrieve index description

```
df.index
```

Get information on DataFrame

```
df.info()
```

Retrieve number of non-`NA` values

```
df.count()
```

Get sum of values

```
df.sum()

# cumulative sum

df.cumsum()
```

Subtract/Add 2 from all values

```
s.sub(2)

s.add(2)
```

Multiply/Divide all values by 2

```
s.mul(2)

s.div(2)
```

Find min/max values of a DataFrame

```
df.min()

df.max()
```

Get min/max index values

```
df.idxmin()

df.idxmax()
```

Get median or mean of values

```
df.mean()

df.median()
```

Describe a summary of data statistics

```
df.describe()
```

Apply a function to a dataset

```
f = # write function here
df.apply(f)

# apply a function by an element

f = # write function here
df.applymap(f)
```

Merge two DataFrames

```
pd.merge(df1, df2, on='subject_id')
```

Combine DataFrames across columns or rows: concatenation

```
print(pd.concat([df1, df2]))
```

Wrapping up and resources

Now that you're armed with the common operations and commands in Python, you can put them into practice. After all, working with real datasets is the best way to master Python and become a data analyst! There's still a lot more to learn that we didn't cover today such as:

- Statistics
- NumPy
- Matplotlib
- Advanced data wrangling
- Visualizations for data (https://www.educative.io/blog/d3js-tutorial)
- Data scraping
- Seaborn
- Scikit-learn (https://www.educative.io/blog/scikit-learn-tutorial-linear-regression)
- and more

To get started with these essential tools of the trade, check out Educative's course **Predictive Data Analysis for Python** (https://www.educative.io/courses/predictive-data-analysis-with-

python). You'll get hands-on practice with industry-standard examples and become fluent in the data analysis.

Keep reading about Python and data science

- Data Analysis Made Simple: Python Pandas Tutorial (https://www.educative.io/blog/python-pandas-tutorial)
- Master Algorithms with Python for Coding Interviews (https://www.educative.io/blog/python-algorithms-coding-interview)
- Stop Using Excel for Data Analytics: Upgrade to Python (https://www.educative.io/blog/excel-python-data-analytics)

WRITTEN BY
Amanda Fawcett