PRE-SESSION WK1

DAWSON, C.

Python for Data Science

- Python Versions

```
python --version
```

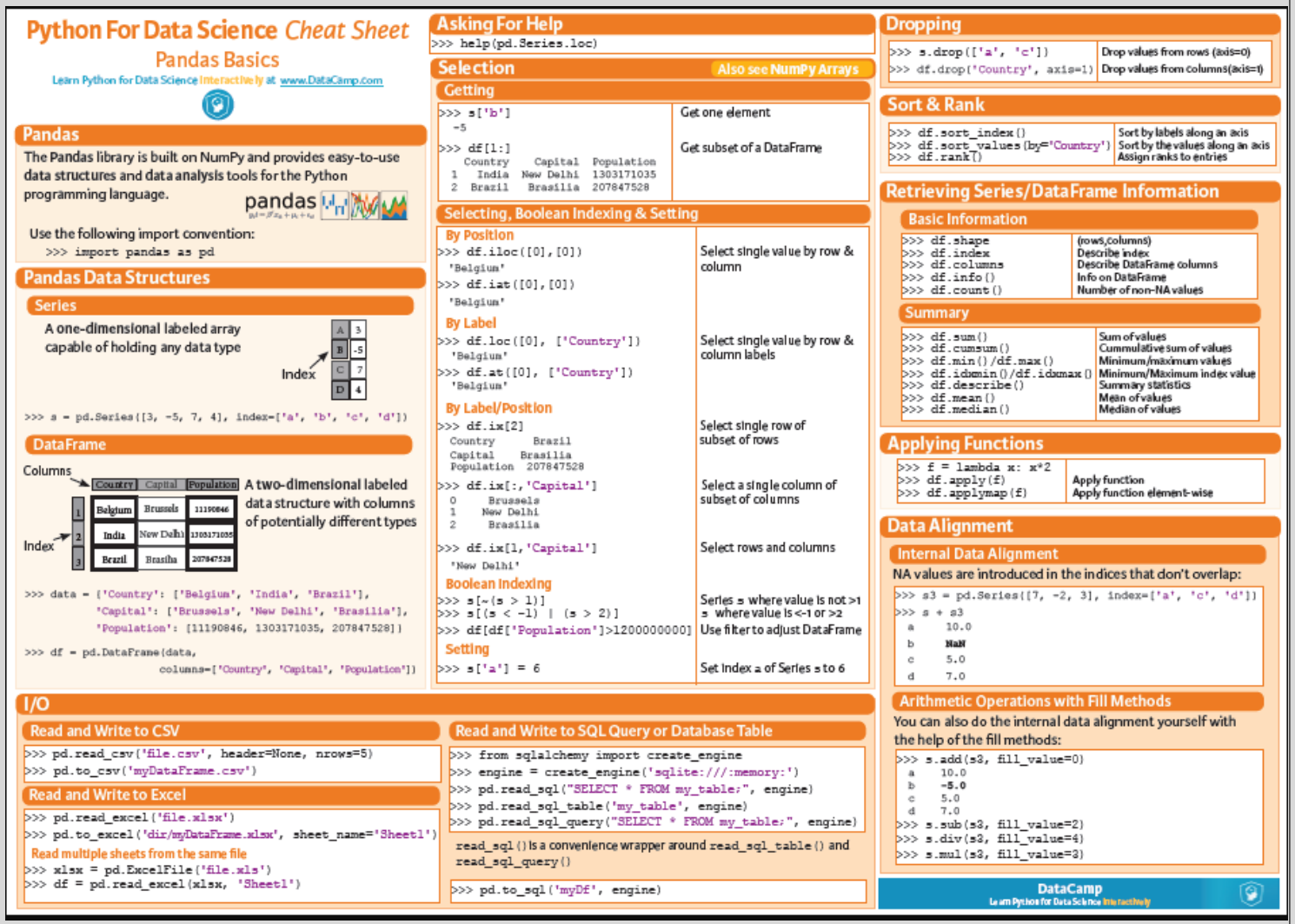
```
sofija@sofija-VirtualBox:~$ python --version
Python 3.7.3
```

- Data Types

The following list shows the seven most common Python data types:

- `integer` : These are whole numbers, positive or negative (including 0). Example: `100`.
- `float` : Floating-point numbers are real numbers, rational or irrational. In most cases, this means numbers with decimal fractions. Example: `123.45`.
- `string` : Strings are sequences of characters, or text, enclosed in quotes. Example: `"any text"`.
- `boolean` : Can be one of two values, true or false. Example: `True` or `False`.
- `list` : An ordered sequence of elements. With a list, the order of the elements can be changed. Example: [*value1* , *value2* , ...].
- `tuple` : An unchangeable ordered sequence of elements. Example: (*value1* , *value2* , ...).
- `dictionary` : This is Python's mapping data type. Dictionaries map keys to values as a means of storing information. Example: { *key1* :*value1*, *key2* :*value2*, ...}.

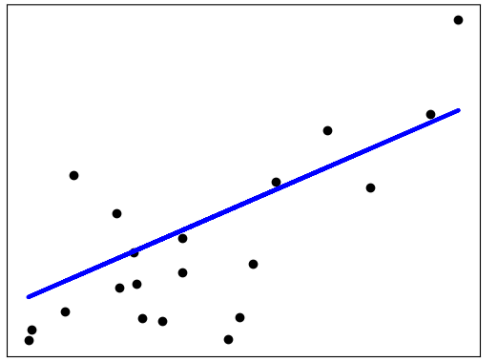
- Pandas



Python for Data Science: Linear Regression

```
print(__doc__)

# License: BSD 3 clause

import matplotlib.pyplot as plt

import numpy as np

from sklearn import datasets, linear_model

from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset

diabetes_X, diabetes_y =
datasets.load_diabetes(return_X_y=True)

# Use only one feature

diabetes_X = diabetes_X[:, np.newaxis, 2]

# Split the data into training/testing sets

diabetes_X_train = diabetes_X[:-20]

diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets

diabetes_y_train = diabetes_y[:-20]

diabetes_y_test = diabetes_y[-20:]

# Create linear regression object

regr = linear_model.LinearRegression()

# Train the model using the training sets

regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set

diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients

print('Coefficients: \n', regr.coef_)

# The mean squared error

print('Mean squared error: %.2f'

    % mean_squared_error(diabetes_y_test,
diabetes_y_pred))

# The coefficient of determination: 1 is perfect prediction

print('Coefficient of determination: %.2f'

    % r2_score(diabetes_y_test, diabetes_y_pred))

# Plot outputs

plt.scatter(diabetes_X_test, diabetes_y_test,  color='black')

plt.plot(diabetes_X_test, diabetes_y_pred, color='blue',
linewidth=3)

plt.xticks(())

plt.yticks(())

plt.show()
```

The coefficients, residual sum of squares and the coefficient of determination are also calculated.



Out:

```
Coefficients:
 [938.23786125]
Mean squared error: 2548.07
Coefficient of determination: 0.47
```