



MACHINE LEARNING

Fitting a Simple Linear Regression model in Python

PLAN OF THE SESSION - AGENDA

What is this session about?

- Get started with programing to create a Simple Linear Regression model
- Demonstrate a working example of fitting a model to our dataset
- Programing language - Python
- IDE - Spyder (installed through Anaconda)

What we would do in the Python code today?

- 1.Import the libraries
- 2.Import the dataset
- 3.Create Matrix of features and dependent var vector
- 4.Handle missing data using mean values
- 5.Split the dataset in to training and test subsets
- 6.Fitting the simple linear regression model to training and test sets
- 7.Visualise the results on a graph

At the end of this session, we should be able to create a Simple Linear Regression model in Python which could be used to predict the values of the dependent variable based on the values of the independent variable

STEP 1: IMPORT THE LIBRARIES

Pandas data analysis lib

- Used to import and manage dataset
- Open source library
- Provides high-performance and easy-to-use data analysis tools for Python
(Pandas.pydata.org, n.d.)

Matplotlib.pyplot

- Used to plot graph for visualisation
- Various states are preserved across function calls, so the plotting functions are directed to the current axes (part of fig.)
(Matplotlib.org, n.d.)

sklearn.preprocessing.Imputer

- The Imputer class provides basic strategies for handling the missing values – either using mean, median or most frequent value in the row or column.
(Scikit-learn.org, n.d.)

sklearn.cross_validation.train_test_split

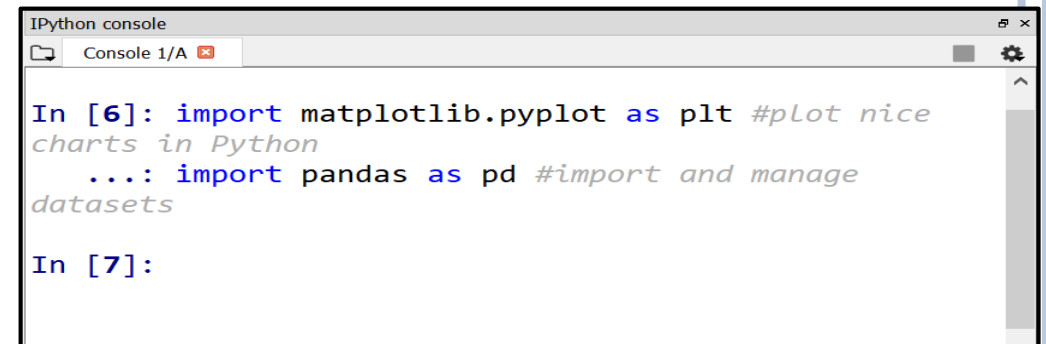
- Used to split arrays or matrices into random train and test subsets.
(Scikit-learn.org, n.d.)

sklearn.linear_model.LinearRegression

- Used to model a linear relationship between a dependent variable and one or more independent variables.

```
1#Importing the Libraries
2import matplotlib.pyplot as plt #plot nice charts in Python
3import pandas as pd #import and manage datasets
4
```

Code snippet



```
IPython console
Console 1/A

In [6]: import matplotlib.pyplot as plt #plot nice
charts in Python
...: import pandas as pd #import and manage
datasets

In [7]:
```

Console

STEP 2: IMPORT THE DATASET

```
5#importing the datasets
6dataset = pd.read_csv('Age_Salary_Data.csv')
7|
```

Code snippet

Alias for Pandas library

	A	B	C
1	Age	Salary	
2	44	72000	
3	27	48000	
4	30	54000	
5	38	61000	
6	40		
7	35	58000	
8		52000	
9	48	79000	
10	50	83000	
11	37	67000	
12			

Data in CSV file

The screenshot shows the Jupyter Notebook interface. The 'Variable explorer' on the right lists the variable 'dataset' as a 'DataFrame' with a size of '(10, 2)' and column names 'Age, Salary'. Below it, the 'dataset - DataFrame' window displays a table with 10 rows and 3 columns: 'Index', 'Age', and 'Salary'. The data is as follows:

Index	Age	Salary
0	44.00	72000.00
1	27.00	48000.00
2	30.00	54000.00
3	38.00	61000.00
4	40.00	nan
5	35.00	58000.00
6	nan	52000.00
7	48.00	79000.00
8	50.00	83000.00
9	37.00	67000.00

Red boxes highlight the 'nan' values in the 'Salary' column for index 4 and the 'Age' column for index 6.

Imported dataset

Notice that the imported dataset has the value 'nan' in the cells that were blank in the datasheet. These cells would later be modified as part of handling the missing data.

STEP 3: CREATE MATRIX OF FEATURES AND DEPENDENT VAR VECTOR

```
8#create matrix of features and dependent var vector
9X = dataset.iloc[:, :-1].values #matrix of features
10y = dataset.iloc[:, -1:].values #dependent var vector
11
```

Matrix of features

- A term used in Machine Learning to describe the list of columns in the dataset that contain independent variables. (Albert Cyberhulk, n.d.)

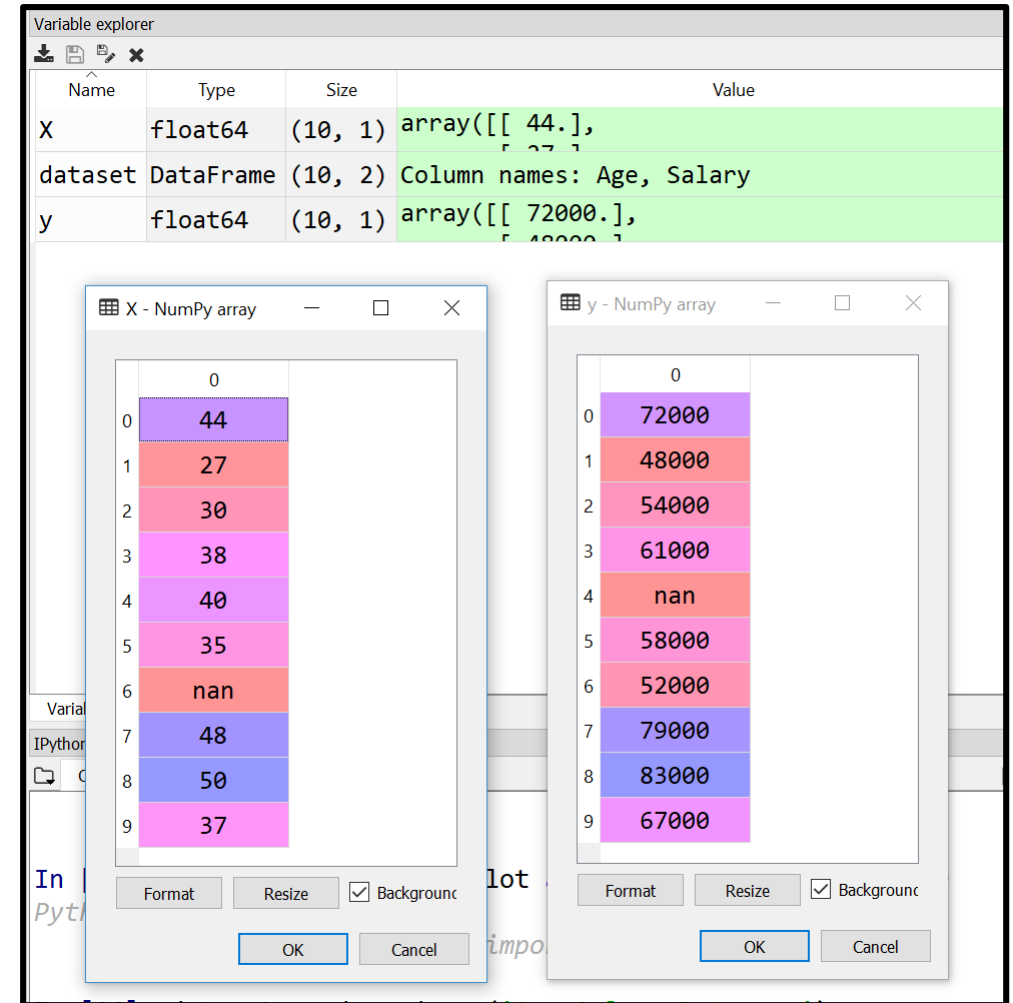
Dependent variable vector

- A term used in Machine Learning to define the list of dependent variables in the existing dataset. (Albert Cyberhulk, n.d.)

What is iloc?

- Purely integer based indexing method provided by Pandas.
- 0-based
- When slicing, start bounds is included while the upper bound is excluded.

(Pandas.pydata.org, n.d.)



The Variable explorer window displays three variables: X (float64, (10, 1)), dataset (DataFrame, (10, 2)), and y (float64, (10, 1)). Below the table, two NumPy array windows are shown: 'X - NumPy array' and 'y - NumPy array'. The 'X' array contains the following values: 44, 27, 30, 38, 40, 35, nan, 48, 50, 37. The 'y' array contains the following values: 72000, 48000, 54000, 61000, nan, 58000, 52000, 79000, 83000, 67000.

Name	Type	Size	Value
X	float64	(10, 1)	array([[44.], [27.], [30.], [38.], [40.], [35.], [nan], [48.], [50.], [37.]])
dataset	DataFrame	(10, 2)	Column names: Age, Salary
y	float64	(10, 1)	array([[72000.], [48000.], [54000.], [61000.], [nan], [58000.], [52000.], [79000.], [83000.], [67000.]])

	0
0	44
1	27
2	30
3	38
4	40
5	35
6	nan
7	48
8	50
9	37

	0
0	72000
1	48000
2	54000
3	61000
4	nan
5	58000
6	52000
7	79000
8	83000
9	67000

Variable explorer view

STEP 4: HANDLE MISSING DATA

```
12#Handle missing data
13from sklearn.preprocessing import Imputer #import Imputer class
14imputer = Imputer(missing_values = 'NaN', strategy = 'mean', axis=0) #create object, impute along column
15imputer = imputer.fit(X[: , 0:1]) #run calculations and save values such as mean, SD, etc
16X[: , 0:1] = imputer.transform(X[: , 0:1]) #transform the data using the values calculated above
17
18imputer = imputer.fit(y[: , 0:1]) #run calculations and save values such as mean, SD, etc
19y[: , 0:1] = imputer.transform(y[: , 0:1]) #transform the data using the values calculated above
20
```

- axis = 0: Impute along the column
- axis = 1: Impute along the row

Why handle missing data?

- Datasets with missing values are incompatible with scikit-learn estimators which assume that all values in an array are numerical, and that all have and hold meaning.

(Scikit-learn.org, n.d.)

Imputation strategy

- Strategy = “mean”: Replace the missing values using the mean along the axis.
- Strategy = “median”: Replace the missing values using the median along the axis.
- Strategy = “most_frequent”: Replace the missing values using the most frequent value along the axis.

(Scikit-learn.org, n.d.)

Fit vs Transform

- Fit: Finds the internal parameters of a model that will be used to transform data.
- Transform: Applies the parameters to the data.
- You may fit a model to one set of data, and then transform it on a completely different set.

(scikit-learn, n.d.)

Missing value is populated with the mean value

Name	Type	Size	Value
X	float64	(10, 1)	array([[44.], [27.], [30.], [38.], [40.], [35.], [38.7778], [48.], [50.], [37.]])
dataset	DataFrame	(10, 2)	Column names: Age, Salary
y	float64	(10, 1)	array([[72000.], [48000.], [54000.], [61000.], [63777.8], [58000.], [52000.], [79000.], [83000.], [67000.]])

Index	Age	Salary
0	44.00	72000.00
1	27.00	48000.00
2	30.00	54000.00
3	38.00	61000.00
4	40.00	63777.78
5	35.00	58000.00
6	38.78	52000.00
7	48.00	79000.00
8	50.00	83000.00
9	37.00	67000.00

```
(missing_values = 'NaN', strategy = 'mean', axis=0) #create object, impute along column
```

STEP 5: SPLIT THE DATASET IN TO TRAINING AND TEST SUBSETS

```
21#Split dataset in to training and test sets
22from sklearn.cross_validation import train_test_split
23X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
24
```

What are the subsets and their purpose?

- The complete dataset is split in to subsets, one of which is used to train the model while the other is used to test it.
- Training subset: This subset contains the input values and the actual output values. The purpose is to train the model on the actual data.
- Test subset: This subset contains the input values only while the output values are predicted by the model.

(Towards Data Science, n.d.), (set?, n.d.)

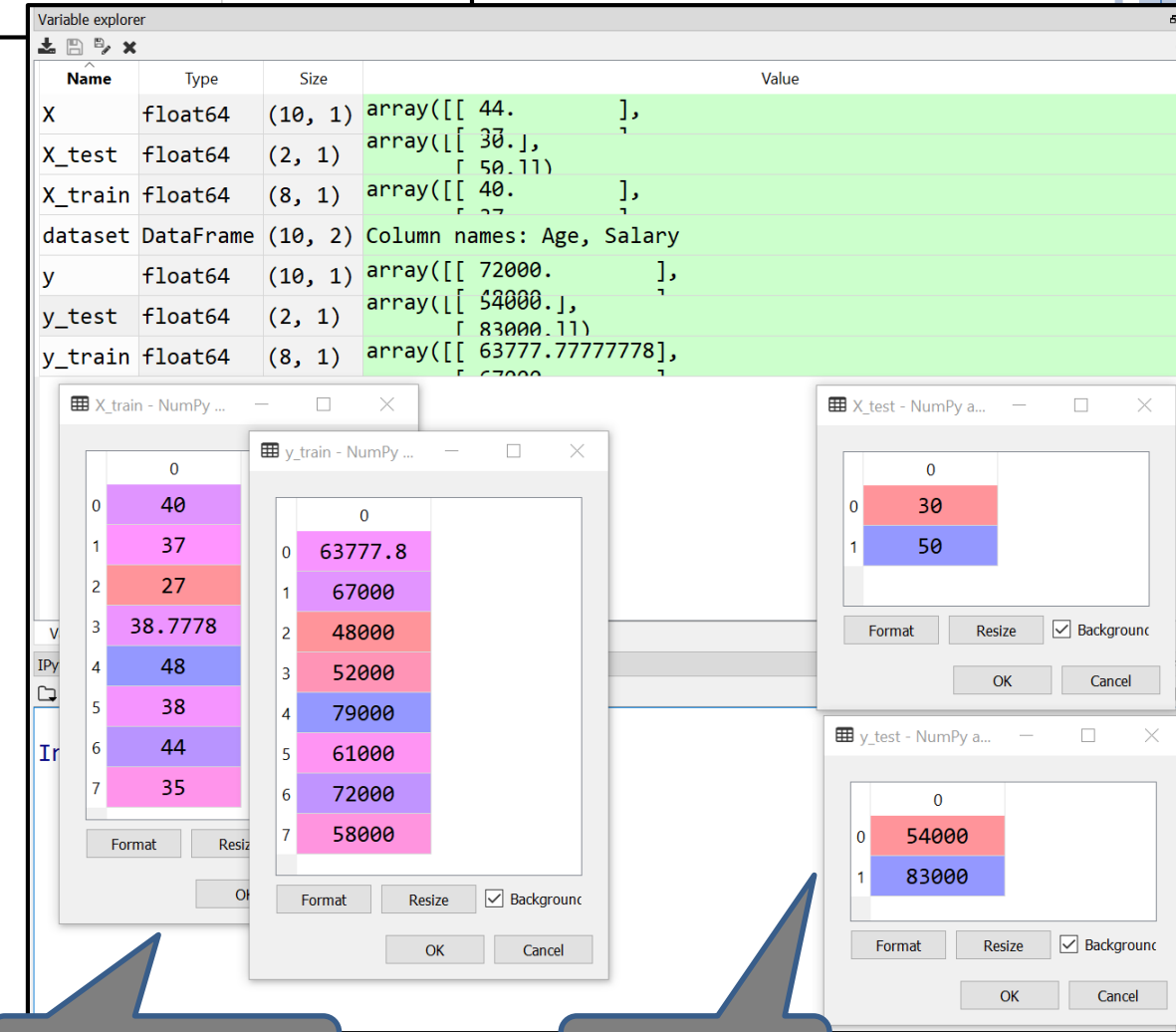
What is test_size (train_size)?

- The value of this parameter represents the proportion of the dataset to be included in the test split (train split).
- If float, the value should be between 0.0 and 1.0. If int then the value represents the absolute number of test samples (train samples). If None then the value is set to the complement of the train size (test size).

(Scikit-learn.org, n.d.)

What is random_state

- A pseudo-random number generator state used for random sampling. (Scikit-learn.org, n.d.)



Training sets with 8 records each

Test sets with 2 records each

STEP 6: FITTING THE SIMPLE LINEAR REGRESSION MODEL TO DATA

```
25#Fitting simple linear regression to the training set
26from sklearn.linear_model import LinearRegression
27regressor = LinearRegression()
28regressor.fit(X_train, y_train)
29
30#Predicting the test set results
31y_pred_test = regressor.predict(X_test)
32y_pred_train = regressor.predict(X_train)
33
```

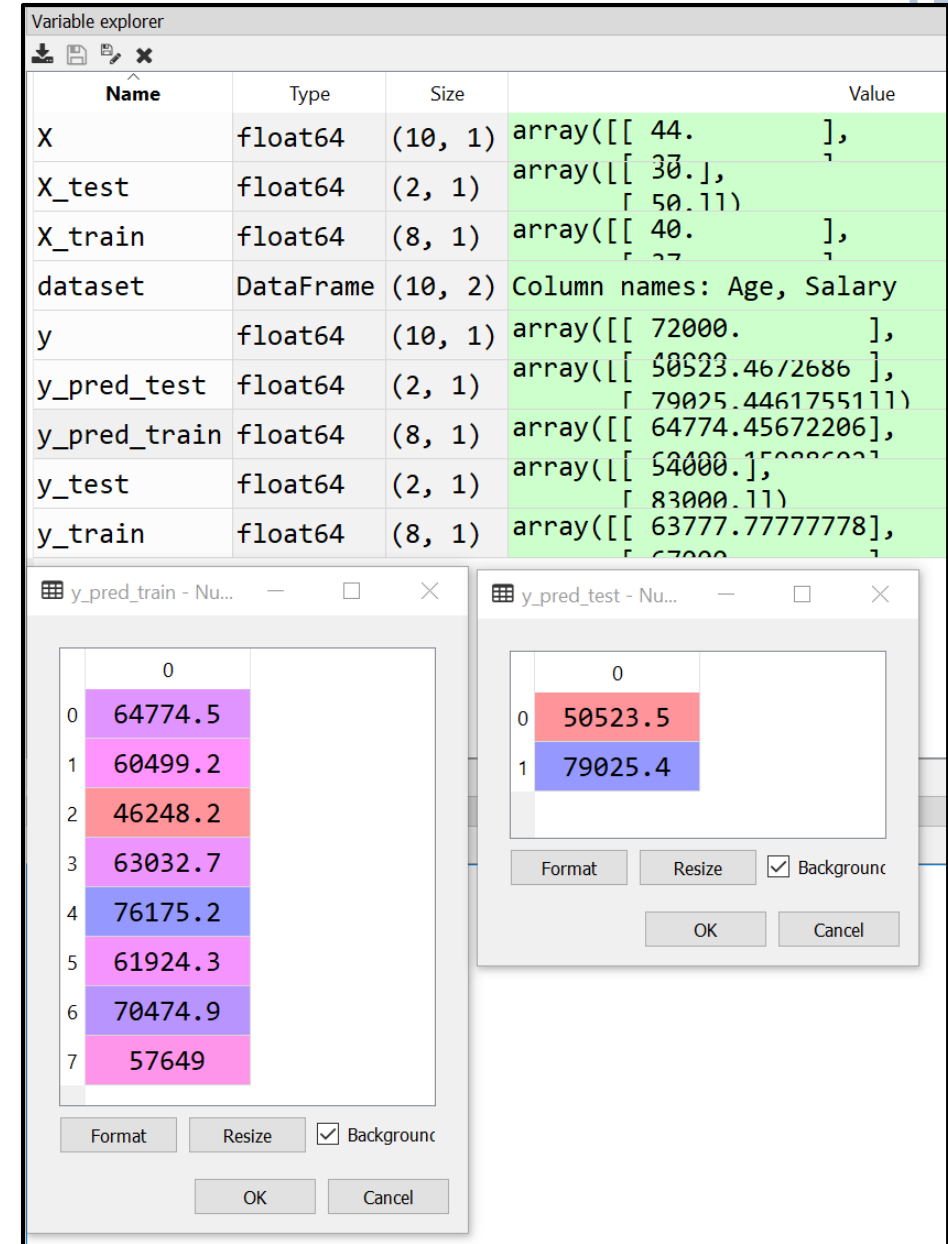
We are using the model to predict the outcome.

What is linear regression?

- In statistics, linear regression is a linear approach for modeling the relationship between a scalar dependent variable y and one or more independent variables denoted X . (En.wikipedia.org, n.d.)
- The case of one independent variable is called simple linear regression. For more than one independent variable, the process is called multiple linear regression. (En.wikipedia.org, n.d.)

What is meant by fitting a model?

- It is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points. (En.wikipedia.org, n.d.)
- By fitting a model, you're making your algorithm learn the relationship between the independent and dependent variables so that it could be used to predict the outcome (dependent variable value). (quora.com, n.d.)



The image shows a 'Variable explorer' window at the top and two prediction windows below it. The 'Variable explorer' window lists variables: X, X_test, X_train, dataset, y, y_pred_test, y_pred_train, y_test, and y_train. The 'y_pred_train' window shows a list of predicted values for the training set, and the 'y_pred_test' window shows predicted values for the test set.

Name	Type	Size	Value
X	float64	(10, 1)	array([[44.], [30.], [50.11], [40.], [37.], [45.], [48.], [49.], [50.], [51.]])
X_test	float64	(2, 1)	array([[30.], [50.11]])
X_train	float64	(8, 1)	array([[40.], [37.], [45.], [48.], [49.], [50.], [51.], [52.]])
dataset	DataFrame	(10, 2)	Column names: Age, Salary
y	float64	(10, 1)	array([[72000.], [50523.46/2686], [79025.4461755111], [64774.45672206], [54000.], [63000.], [70000.], [83000.11], [63777.77777778], [67000.]])
y_pred_test	float64	(2, 1)	array([[50523.46/2686], [79025.4461755111]])
y_pred_train	float64	(8, 1)	array([[64774.45672206], [54000.], [63000.], [70000.], [83000.11], [63777.77777778], [67000.], [72000.]])
y_test	float64	(2, 1)	array([[54000.], [83000.11]])
y_train	float64	(8, 1)	array([[63777.77777778], [67000.], [72000.], [50523.46/2686], [79025.4461755111], [64774.45672206], [54000.], [63000.]])

	0
0	64774.5
1	60499.2
2	46248.2
3	63032.7
4	76175.2
5	61924.3
6	70474.9
7	57649

	0
0	50523.5
1	79025.4

STEP 7: VISUALISE THE RESULT ON A GRAPH

```
34#Visualise the Training test results
35#observation points
36plt.scatter(X_train, y_train, color = 'red', label = 'Actual values / Observation points')
37#regression line with predictions
38plt.plot(X_train, y_pred_train, color = 'blue', label = 'Regression line with predicted values')
39plt.title('Age vs Salary (Training set)')
40plt.xlabel('Age')
41plt.ylabel('Salary')
42plt.legend(loc = 'upper left')
43plt.show()
```

'plt' is the alias for matplotlib.pyplot library that we imported in the beginning

What are we trying to do here?

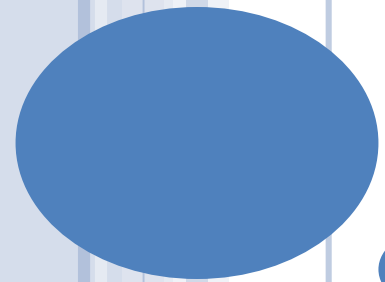
- On a graph, we have plotted the observation points and the simple linear regression line. (Eremenko and de Ponteves, n.d.)
- Purpose is to see the linear dependency and how the predictions of this simple linear model can be close to the real observations. (Eremenko and de Ponteves, n.d.)
- Similar prediction and visualization could be performed for the test subset as well to see how good is our model is at predicting the outcome.



BIBLIOGRAPHY

- Pandas.pydata.org. (n.d.). *Python Data Analysis Library — pandas: Python Data Analysis Library*. [online] Available at: <https://pandas.pydata.org/> [Accessed 5 Dec. 2017].
- Matplotlib.org. (n.d.). Pyplot tutorial — Matplotlib 2.0.2 documentation. [online] Available at: https://matplotlib.org/users/pytest_tutorial.html [Accessed 5 Dec. 2017].
- Scikit-learn.org. (n.d.). 4.3. Preprocessing data — scikit-learn 0.19.1 documentation. [online] Available at: <http://scikit-learn.org/stable/modules/preprocessing.html#preprocessing> [Accessed 6 Dec. 2017].
- Scikit-learn.org. (n.d.). sklearn.cross_validation.train_test_split — scikit-learn 0.16.1 documentation. [online] Available at: http://scikit-learn.org/0.16/modules/generated/sklearn.cross_validation.train_test_split.html [Accessed 6 Dec. 2017].
- Pandas.pydata.org. (n.d.). Indexing and Selecting Data — pandas 0.21.0 documentation. [online] Available at: <https://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-integer> [Accessed 6 Dec. 2017].
- Scikit-learn.org. (n.d.). 4.3. *Preprocessing data — scikit-learn 0.19.1 documentation*. [online] Available at: <http://scikit-learn.org/stable/modules/preprocessing.html#imputation> [Accessed 7 Dec. 2017].
- Scikit-learn.org. (n.d.). *sklearn.preprocessing.Imputer — scikit-learn 0.19.1 documentation*. [online] Available at: <http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Imputer.html> [Accessed 7 Dec. 2017].
- scikit-learn, F. (n.d.). *Fitting data vs. transforming data in scikit-learn*. [online] Stackoverflow.com. Available at: <https://stackoverflow.com/questions/31572487/fitting-data-vs-transforming-data-in-scikit-learn> [Accessed 7 Dec. 2017].
- Towards Data Science. (n.d.). *Train/Test Split and Cross Validation in Python – Towards Data Science*. [online] Available at: <https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6> [Accessed 7 Dec. 2017].
- set?, W. (n.d.). *What is the difference between test set and validation set?*. [online] Stats.stackexchange.com. Available at: <https://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set> [Accessed 7 Dec. 2017].
- En.wikipedia.org. (n.d.). *Curve fitting*. [online] Available at: https://en.wikipedia.org/wiki/Curve_fitting [Accessed 7 Dec. 2017].
- quora.com. (n.d.). *What does fitting a model mean in data science?*. [online] Available at: <https://www.quora.com/What-does-fitting-a-model-mean-in-data-science> [Accessed 7 Dec. 2017].
- En.wikipedia.org. (n.d.). *Linear regression*. [online] Available at: https://en.wikipedia.org/wiki/Linear_regression [Accessed 7 Dec. 2017].
- Eremenko, K. and de Ponteves, H. (n.d.). *Simple Linear Regression in Python - Step 4*. [video] Available at: <https://www.udemy.com/machinelearning/learn/v4/t/lecture/5768342?start=0> [Accessed 7 Dec. 2017].





THANK YOU FOR YOUR TIME!!