COIT20277 Introduction to Artificial Intelligence

Week 3

- Supervised Learning: Regression
- Unsupervised Learning

Acknowledgement of Country

I respectfully acknowledge the Traditional Custodians of the land on which we live, work and learn. I pay my respects to the First Nations people and their Elders, past, present and future

Supervised Learning: Regressions

- The supervised learning algorithms explored in this tutorial are:
  - Linear Regression
  - Logistic Regression

- You can learn more about these algorithms implemented in scikit-learn by accessing https://scikit-learn.org/stable/supervised_learning.html and https://scikit-learn.org/stable/unsupervised_learning.html

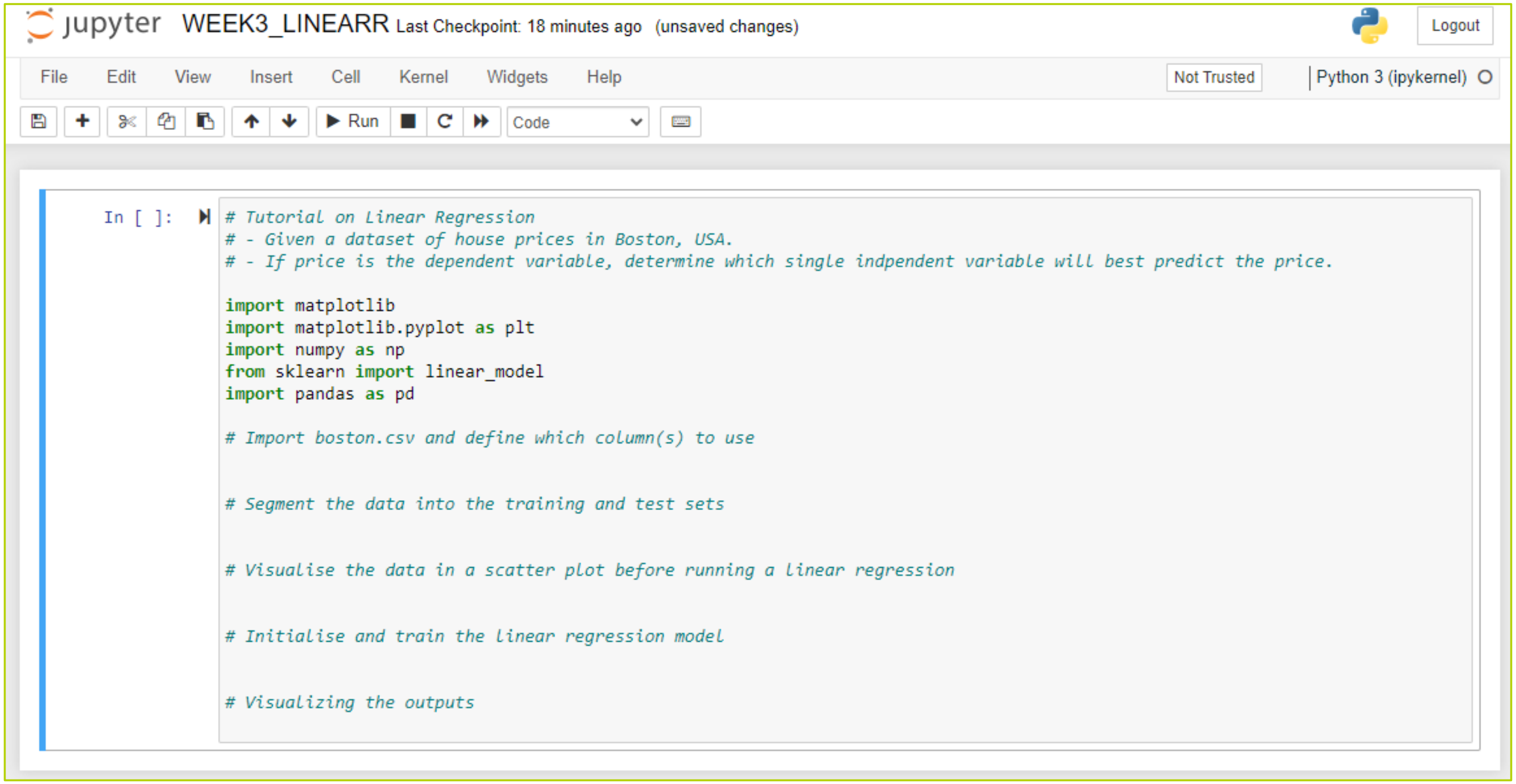
Unsupervised Learning

- The unsupervised learning algorithm explored in this tutorial is:
    - K-means clustering

- You can learn more about K-means and other unsupervised learning algorithms implemented in scikit-learn by accessing https://scikit-learn.org/stable/unsupervised_learning.html

Linear Regression

- Tutorial on Linear Regression
  - Given a dataset of house prices, `boston.csv`, in Boston, USA.
  - Price is the dependent variable, while other columns are independent variables.
  - Your job is to complete the Python program, `WEEK3_LINEARR.ipynb` to determine which independent variable will best predict the price.

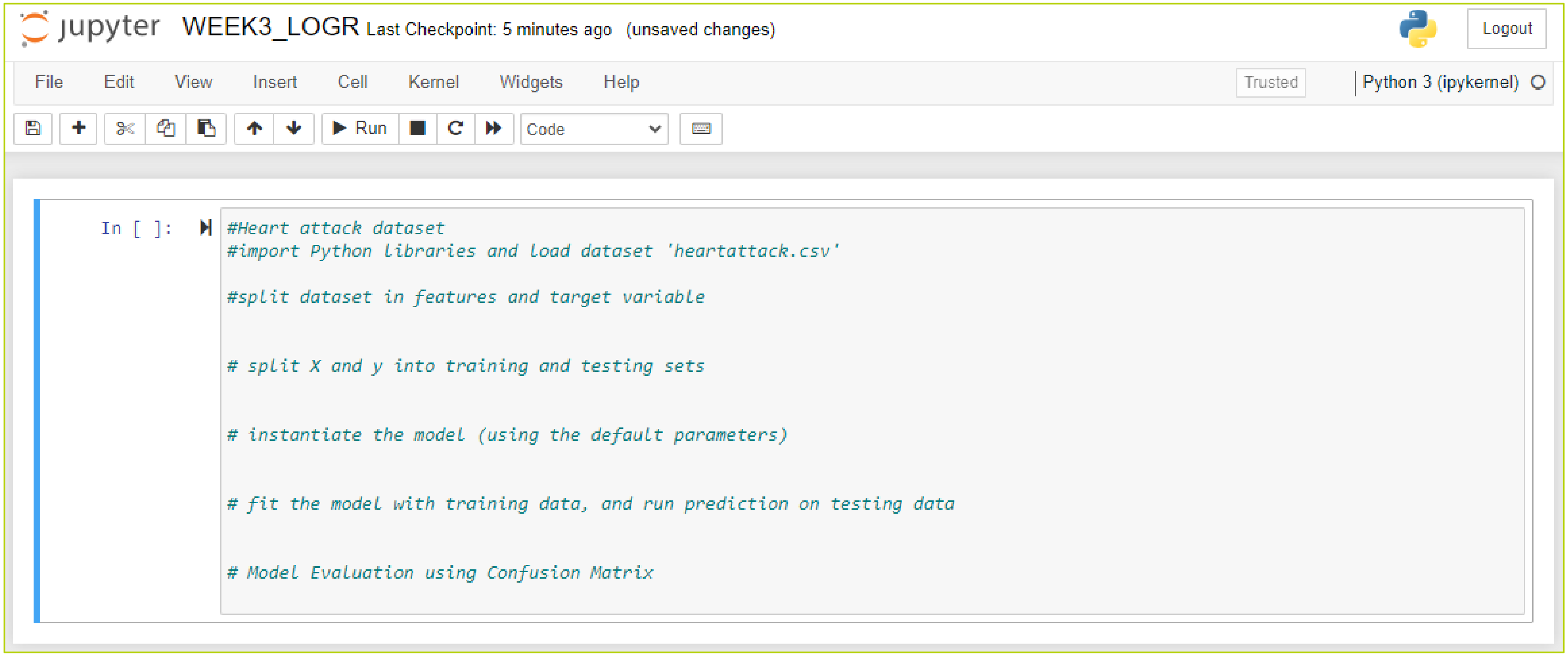


jupyter WEEK3_LINEARR Last Checkpoint: 18 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel) ○
Run
Code
In [ ]:

```
# Tutorial on Linear Regression
# - Given a dataset of house prices in Boston, USA.
# - If price is the dependent variable, determine which single indpendent variable will best predict the price.

import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from sklearn import linear_model
import pandas as pd

# Import boston.csv and define which column(s) to use


# Segment the data into the training and test sets


# Visualise the data in a scatter plot before running a linear regression


# Initialise and train the linear regression model


# Visualizing the outputs
```

Logistic Regression

- Tutorial on Logistic Regression
  - Given a dataset on persons who had or hadn’t had a heart attack before, `heartattack.csv`.
  - The last column is a binary [0,1] meaning a heart attack had occurred or not.
  - Your task is to complete the Python program, `WEEK3_LOGR.ipynb` to train a logistic regressor to predict if a person will have a heart attack or not.



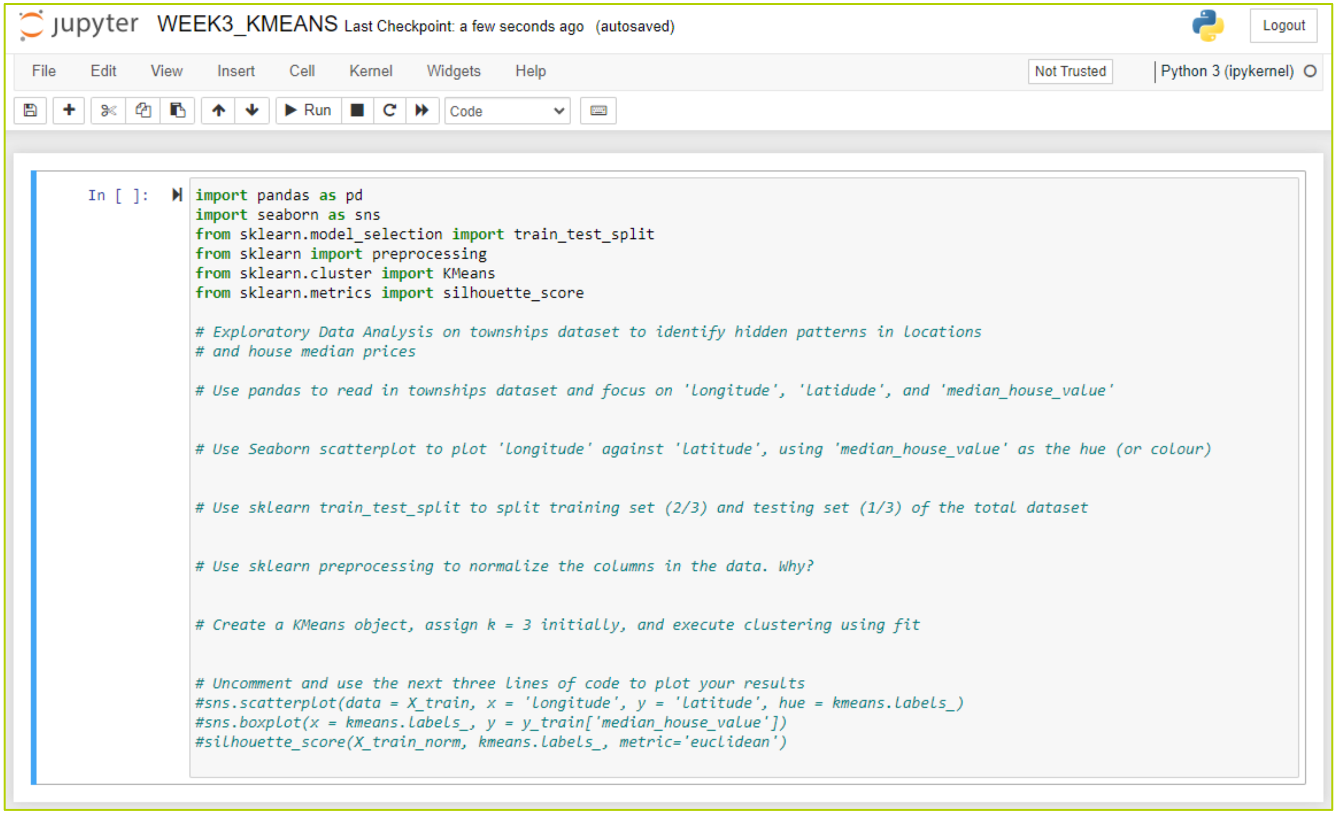
```
#Heart attack dataset
#import Python libraries and load dataset 'heartattack.csv'

#split dataset in features and target variable


# split X and y into training and testing sets


# instantiate the model (using the default parameters)


# fit the model with training data, and run prediction on testing data


# Model Evaluation using Confusion Matrix
```

CQUniversity AUSTRALIA

BE WHAT YOU WANT TO BE
cqu.edu.au

K-means Clustering

- Tutorial on K-means Clustering
  - Given a dataset on townships in Australia with longitude, latitude, population, median house prices, etc. in `townships.csv`.
  - Your task is to implement K-means Clustering in a Python program similar as below, `WEEK3_KMEANS.ipynb`, to find the hidden clusters.
  - You can experiment with different values for K, the number of clusters.



```
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

# Exploratory Data Analysis on townships dataset to identify hidden patterns in locations
# and house median prices

# Use pandas to read in townships dataset and focus on 'longitude', 'latidude', and 'median_house_value'


# Use Seaborn scatterplot to plot 'longitude' against 'latitude', using 'median_house_value' as the hue (or colour)


# Use sklearn train_test_split to split training set (2/3) and testing set (1/3) of the total dataset


# Use sklearn preprocessing to normalize the columns in the data. Why?


# Create a KMeans object, assign k = 3 initially, and execute clustering using fit


# Uncomment and use the next three lines of code to plot your results
#sns.scatterplot(data = X_train, x = 'longitude', y = 'latitude', hue = kmeans.labels_)
#sns.boxplot(x = kmeans.labels_, y = y_train['median_house_value'])
#silhouette_score(X_train_norm, kmeans.labels_, metric='euclidean')
```

THANK YOU

TIME FOR DISCUSSION & QUESTIONS