2. Problem Statement In this assignment students will build the random forest model after normalizing the variable to house pricing from boston data set. Following the code to get data into the environment: import numpy as np import pandas as pd import matplotlib.pyplot as plt import seaborn as sns from sklearn.model\_selection import train\_test\_split from sklearn.preprocessing import StandardScaler from sklearn import datasets boston = datasets.load\_boston() features = pd.DataFrame(boston.data, columns=boston.feature\_names) targets = boston.target NOTE: The solution shared through Github should contain the source code used and the screenshot of the output.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

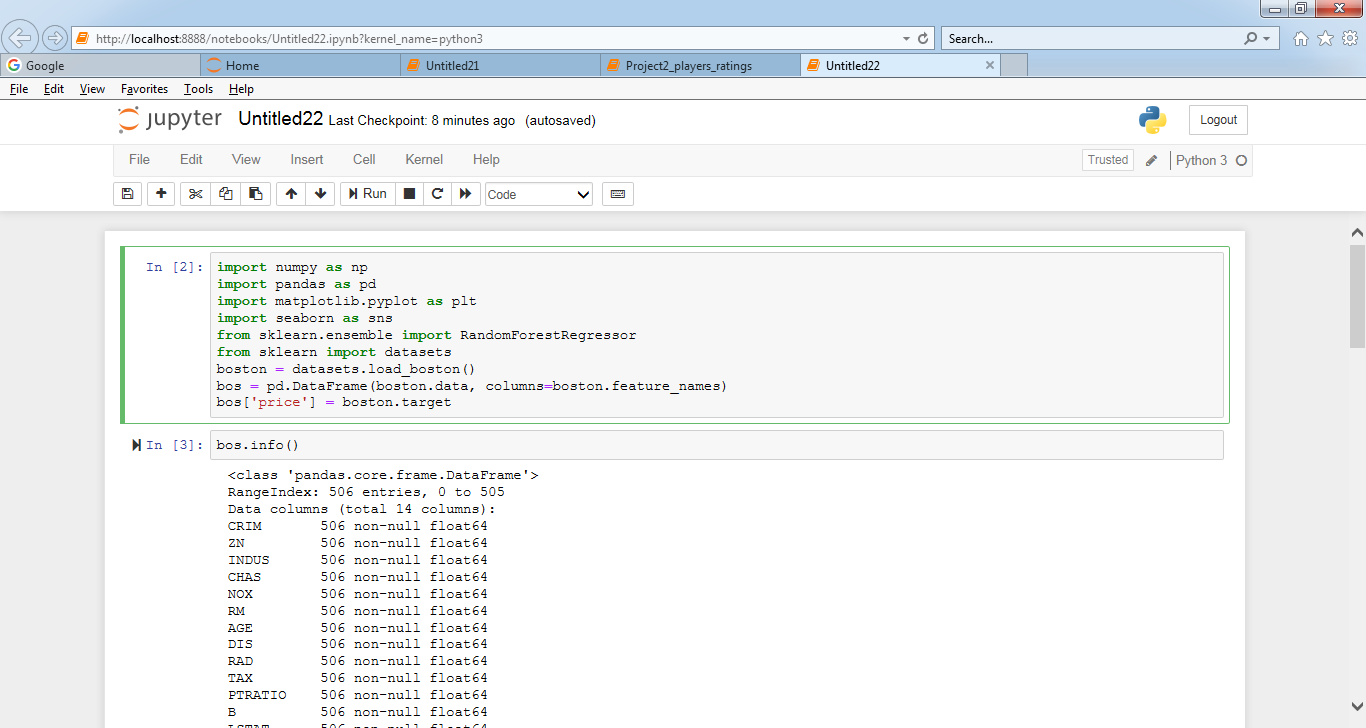
from sklearn.ensemble import RandomForestRegressor

from sklearn import datasets

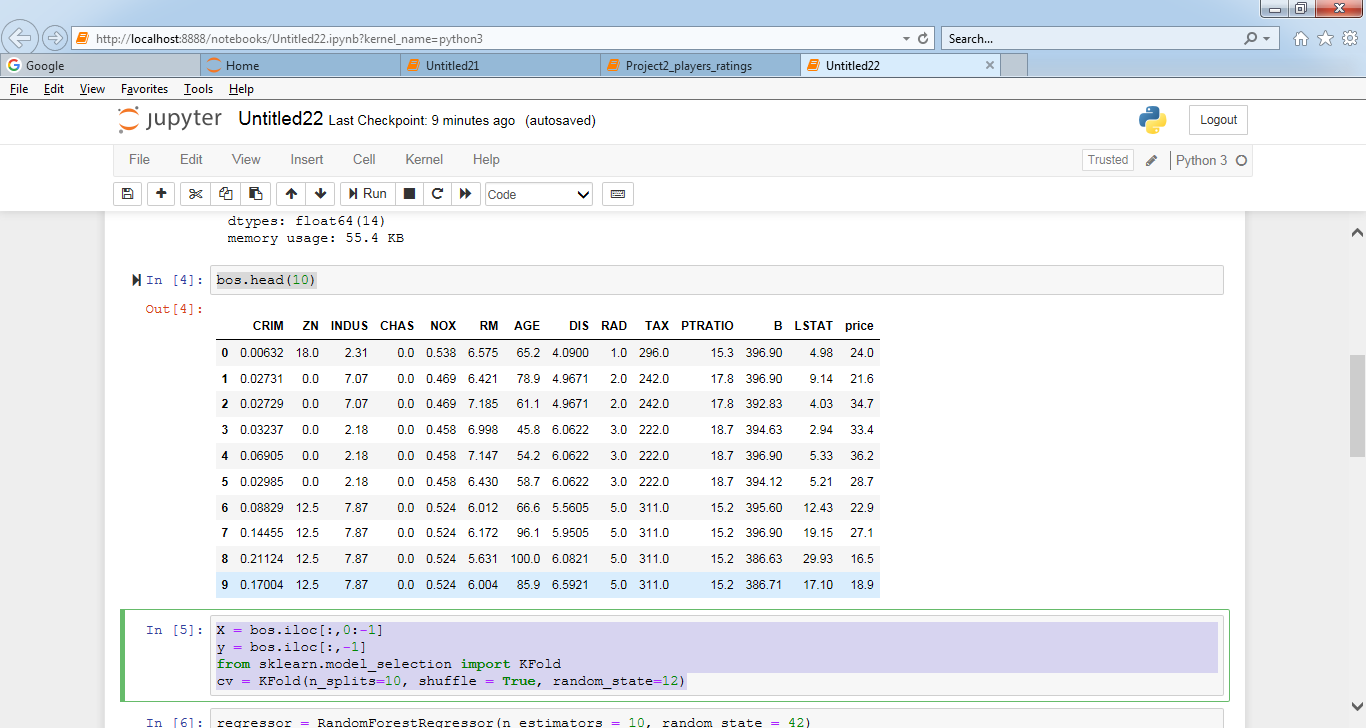
boston = datasets.load\_boston()

bos = pd.DataFrame(boston.data, columns=boston.feature\_names)

bos['price'] = boston.target



bos.head(10)



X = bos.iloc[:,0:-1]

y = bos.iloc[:,-1]

from sklearn.model\_selection import KFold

cv = KFold(n\_splits=10, shuffle = True, random\_state=12)

regressor = RandomForestRegressor(n\_estimators = 10, random\_state = 42)

for train\_fold, valid\_fold in cv.split(X):

train = X.loc[train\_fold] # Extract train data with cv indices

valid = X.loc[valid\_fold] # Extract valid data with cv indices

train\_y = y.loc[train\_fold]

valid\_y = y.loc[valid\_fold]

model = regressor.fit(X = train, y = train\_y)

model\_pred = model.predict(X = valid)

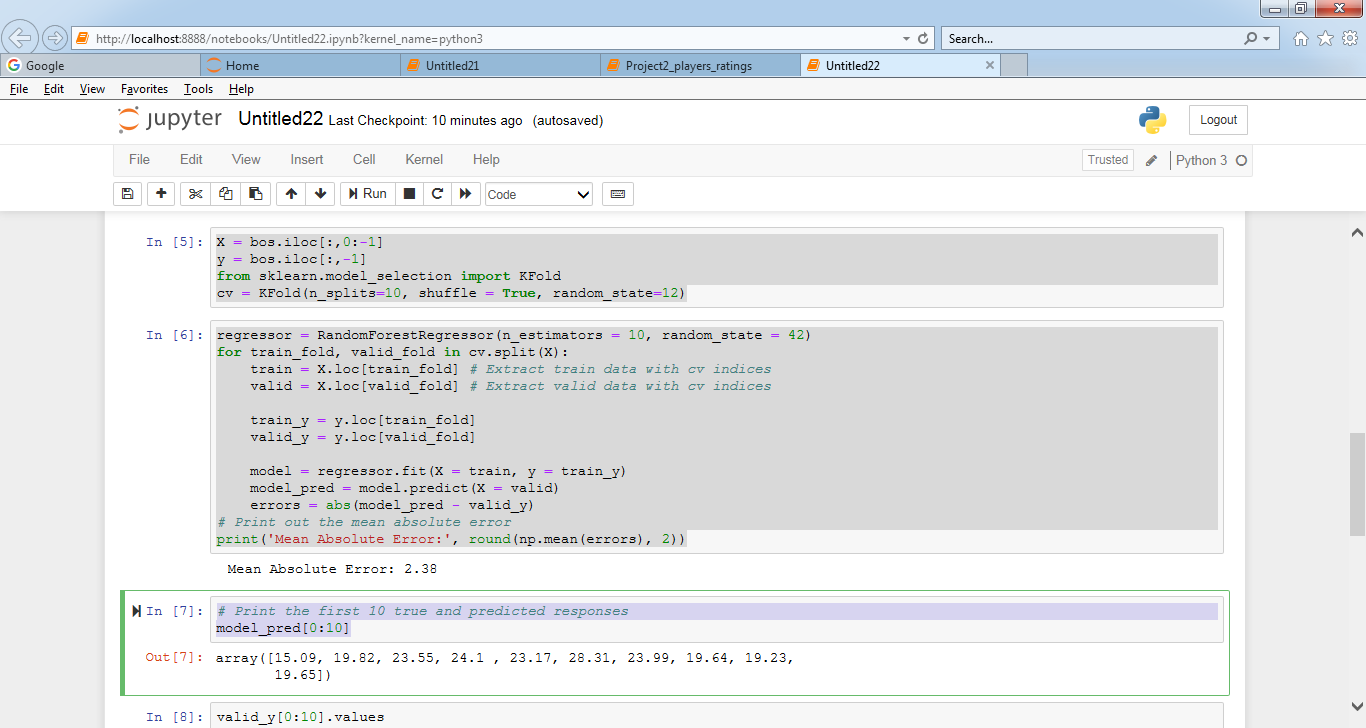
errors = abs(model\_pred - valid\_y)

# Print out the mean absolute error

print('Mean Absolute Error:', round(np.mean(errors), 2))

# Print the first 10 true and predicted responses

model\_pred[0:10]



valid\_y[0:10].values

# Splitting the dataset into Training set and Test set

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state = 42)

regressor = RandomForestRegressor(n\_estimators = 10, random\_state = 42)

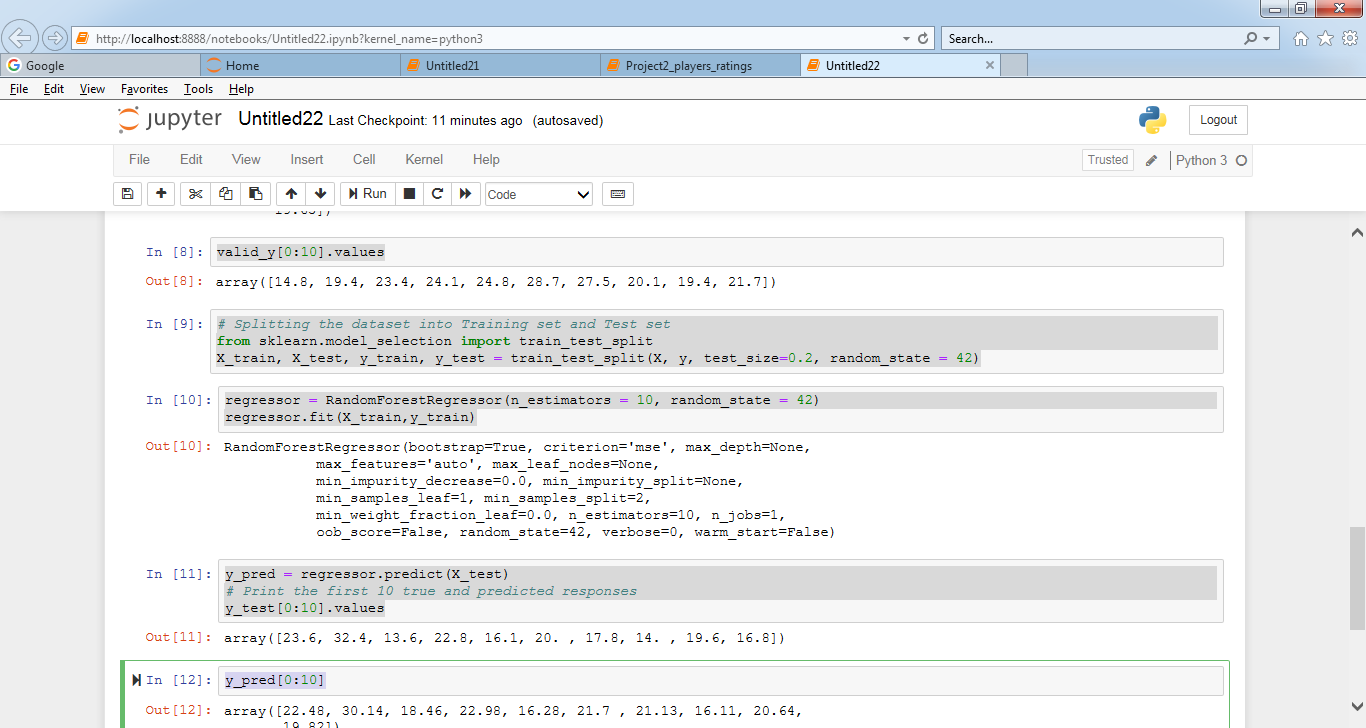
regressor.fit(X\_train,y\_train)

y\_pred = regressor.predict(X\_test)

# Print the first 10 true and predicted responses

y\_test[0:10].values

y\_pred[0:10]

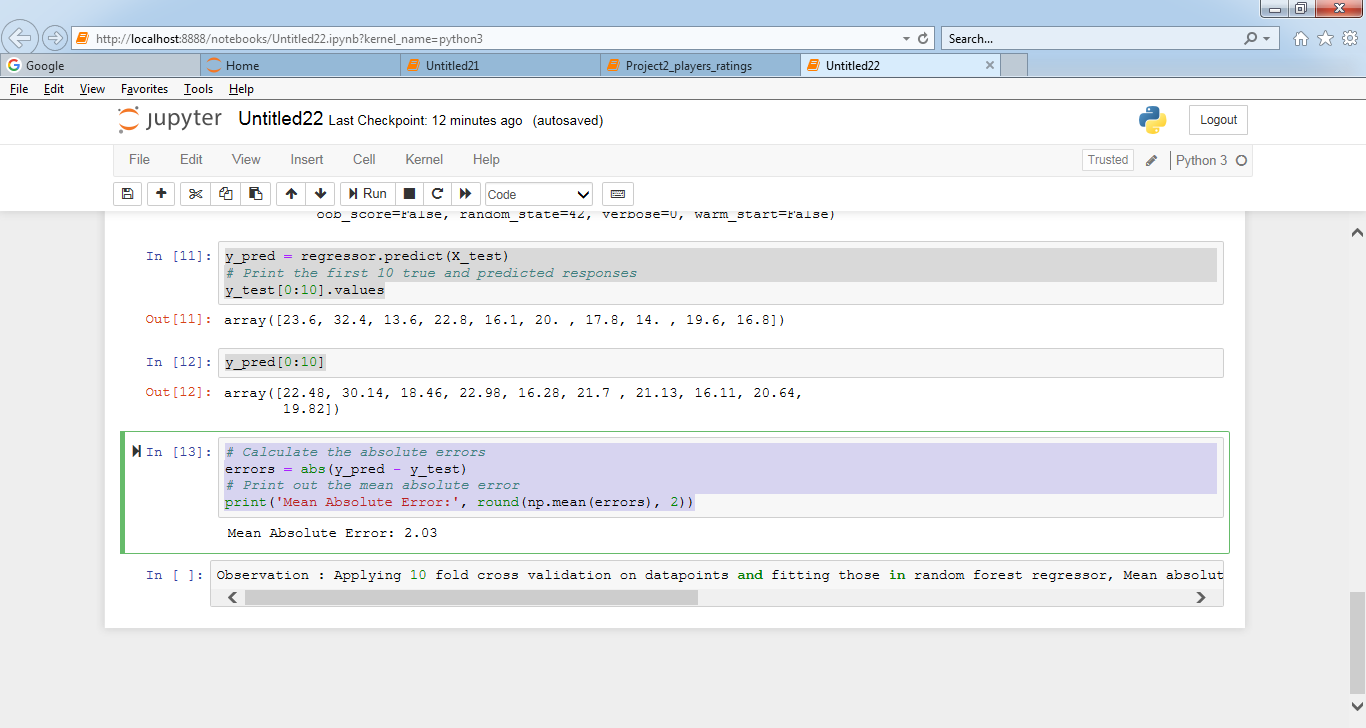


# Calculate the absolute errors

errors = abs(y\_pred - y\_test)

# Print out the mean absolute error

print('Mean Absolute Error:', round(np.mean(errors), 2))

Observation : Applying 10 fold cross validation on datapoints and fitting those in random forest regressor, Mean absolute error value is 2.38 as opposed to Mean absolute error value of 2.03 while applying train test split in random forest regressor.