DHRUV GARG 16 BCE 1190

LINEAR REGRESSION

LAB 3

Linear regression theory:

Linear regression refers to predicting the value of 'y' given the value of 'x'. The model works well if the two variables are either positively correlated or negatively correlated. The higher the correlation value identified, the better our linear regression model would be. In this experiment, I implemented 3 linear regression models on both the datasets:

1. **Simple linear regression**- this method followed the gradient descent optimization. The function used here (similar to the cost function) was the sum of squared errors. The major task of the algorithm involves reducing the squared error, to the least possible value.

$$J(w) = \frac{1}{2}\sum_{i}(\text{target}^{(i)} - \text{output}^{(i)})^2, \quad \text{output}^{(i)}$$

2. **Scikit learn's LinearRegression** – this is imported from "linear_model" library.
3. **RANSACK regression** – this regression model identifies the outliers in the data, and does not take them into account, improving accuracy.

PART 1: BOSTON HOUSING Dataset

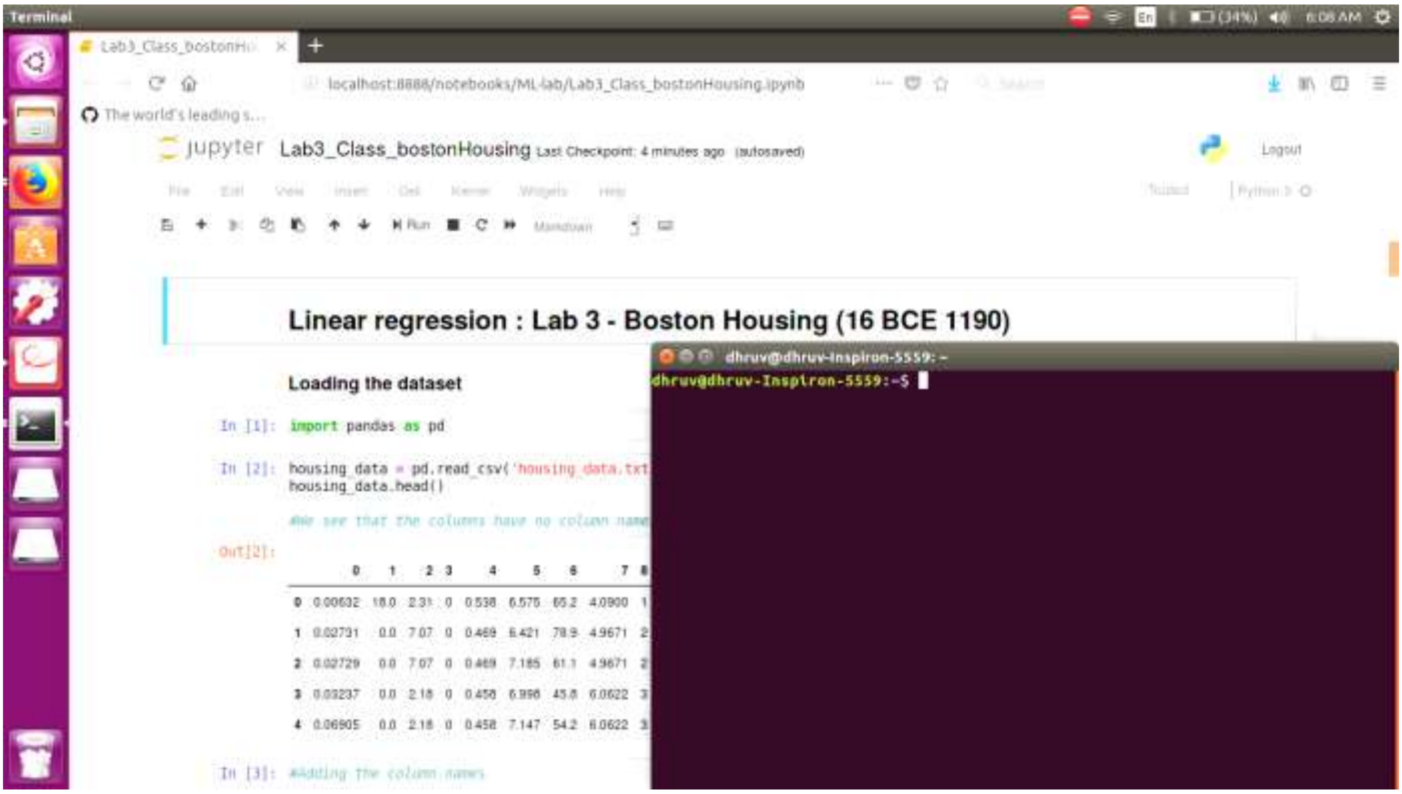
The value to be predicted was the price of the house. First, the text file was imported as a csv file using **separator = '\s+'**. We know that for a linear model, we must have a pair attributes that have good positive or negative correlation coefficient. To view various attributes and their linear dependency, **pairplots are plotted**. It was observed that RM and MDEV are are largely linearly dependent (~0.70). To **quantify this correlation**, a **heatmap** was plotted. Next, we implemented the **gradient descent** linear regression model. The model was optimized in **4 epochs**. The data was scaled using sklearn's **StandardScalar,** and a line with intercept (0,0) was obtained. Same line was also the output for the sklearn's linearRegression. It suggested that the number or rooms and the cost are directly proportional, but some of the house costs could not be justified by the same logic (**outliers**). The **most sophisticated** linear regression model used was the **RANSACK regression** model. It gave the **best regression line** for the data points, since its line ignored the outliers and was not pushed up by the same.

PART 2: ABSENTEEISM FROM WORK Dataset

In this dataset, the **absenteeism time** was thought to be the prediction target attribute. However, from the correlation plots drawn, absenteeism time was not very correlated with other attributes. Multiple correlation plots were plotted to identify the highest correlated. Age and service time were chosen, as these two had a **positive correlation of 0.67**, which was used to predict the age from the service time. In the first model, **3 epochs** were used to arrive at the linear regression line. The data was scaled using **sklearn's standardScalar**, before being sent to the gradient descent algorithm. Same line was also the output for the sklearn's linearRegression. Lastly, the **RANSACK regression line was used**, which gave the **best output line**-owing to its ability to identify and filter out the outliers.

SCREENSHOTS

Boston housing dataset



Terminal
Lab3_Class_bostonHo
localhost:8888/notebooks/ML-lab/Lab3_Class_bostonHousing.ipynb
The world's leading s...
jupyter
Lab3_Class_bostonHousing Last Checkpoint: 4 minutes ago (autosaved)
Logout
6:08 AM

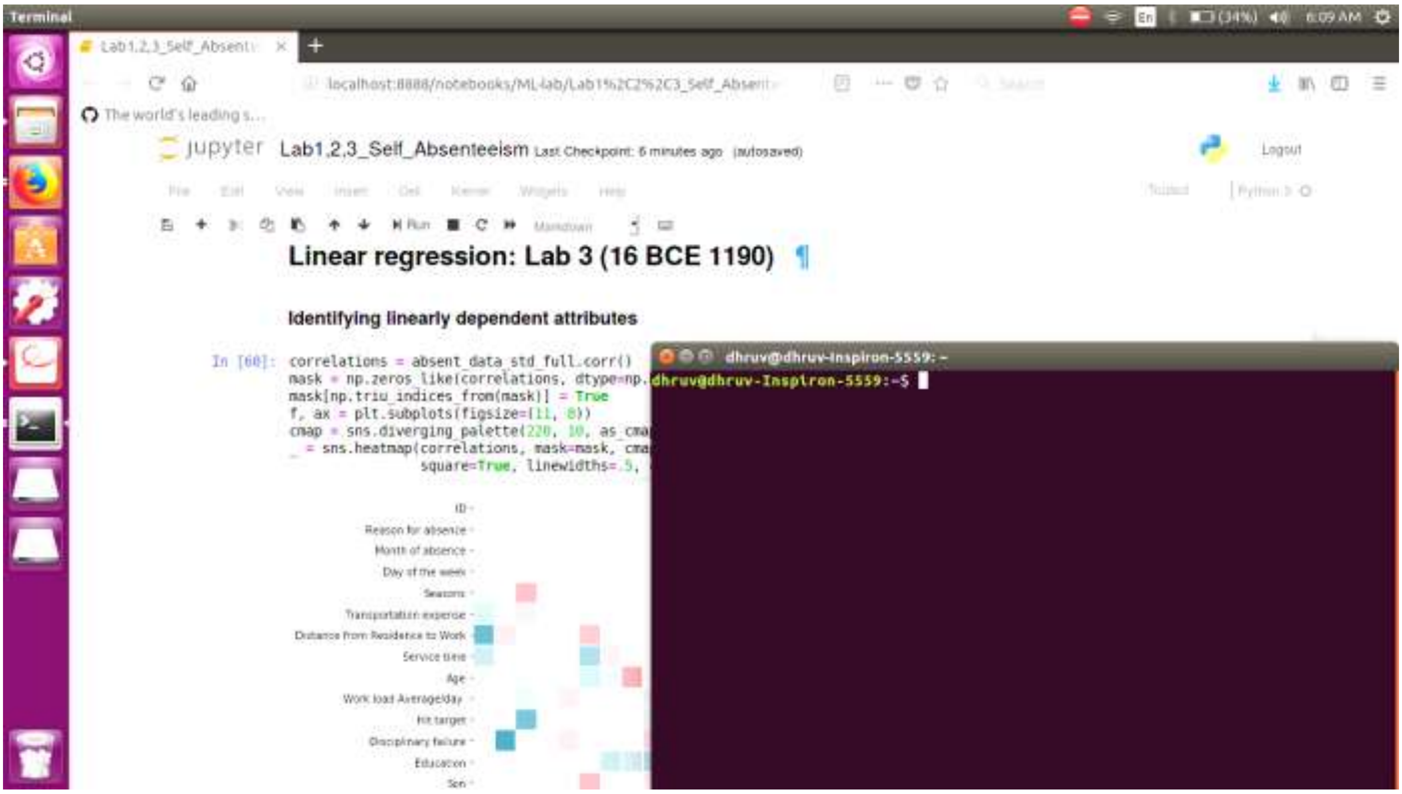
Linear regression : Lab 3 - Boston Housing (16 BCE 1190)

Loading the dataset

```
In [1]: import pandas as pd

In [2]: housing_data = pd.read_csv('housing_data.txt
        housing_data.head()

        #We see that the columns have no column name

Out[2]:
           0     1     2  3      4      5     6       7  8
   0 0.00632  18.0  2.31  0  0.538  6.575  65.2  4.0900  1
   1 0.02731   0.0  7.07  0  0.469  6.421  78.9  4.9671  2
   2 0.02729   0.0  7.07  0  0.469  7.185  61.1  4.9671  2
   3 0.03237   0.0  2.18  0  0.458  6.998  45.8  6.0622  3
   4 0.06905   0.0  2.18  0  0.458  7.147  54.2  6.0622  3

In [3]: #Adding the column names
```

```
dhruv@dhruv-Inspiron-5559: ~
dhruv@dhruv-Inspiron-5559:~$
```

Chosen dataset – Absenteeism at work



Linear regression: Lab 3 (16 BCE 1190) ¶

Identifying linearly dependent attributes

```
In [60]: correlations = absent_data_std_full.corr()
         mask = np.zeros_like(correlations, dtype=np.
         mask[np.triu_indices_from(mask)] = True
         f, ax = plt.subplots(figsize=(11, 8))
         cmap = sns.diverging_palette(220, 10, as_cma
         _ = sns.heatmap(correlations, mask=mask, cma
                         square=True, linewidths=.5,
```

```
dhruv@dhruv-Inspiron-5559: ~
dhruv@dhruv-Inspiron-5559:~$
```