CSE-564 Visualization
Lab #1 Report

Piyush Soni ***SBUID:113273375***

Project Video link: here

- Video is also uploaded with the .zip file to the blackboard.

Dataset:

Data for this Lab was selected from Kaggle, which originates from the UCI Machine Learning Repository. It was collected from homes in suburbs of Boston, Massachusetts in 1978. This dataset was formerly intended for the prediction of *Median value of homes in Suburbs of Boston (**MEDV**)* and its dependency on the locality.

Attribute Information:

- **CRIM:** per capita crime rate by town
- **ZN:** proportion of residential land zoned for lots over 25,000 sq.ft.
- **INDUS:** proportion of non-retail business acres per town
- **CHAS:** Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- **NOX:** nitric oxides concentration (parts per 10 million)
- **RM:** average number of rooms per dwelling
- **AGE:** proportion of owner-occupied units built prior to 1940
- **DIS:** weighted distances to five Boston employment centres
- **RAD:** index of accessibility to radial highways
- **TAX:** full-value property-tax rate per $10,000
- **PTRATIO:** pupil-teacher ratio by town
- **B:** 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- **LSTAT:** % lower status of the population
- **MEDV:** Median value of owner-occupied homes in $1000's

Dataset before processing was pre-processed using Pandas and Numpy libraries of python and after storing in .csv visualized using ***d3.js***, ***jQuery***, ***JavaScript***, ***HTML*** and ***CSS.***

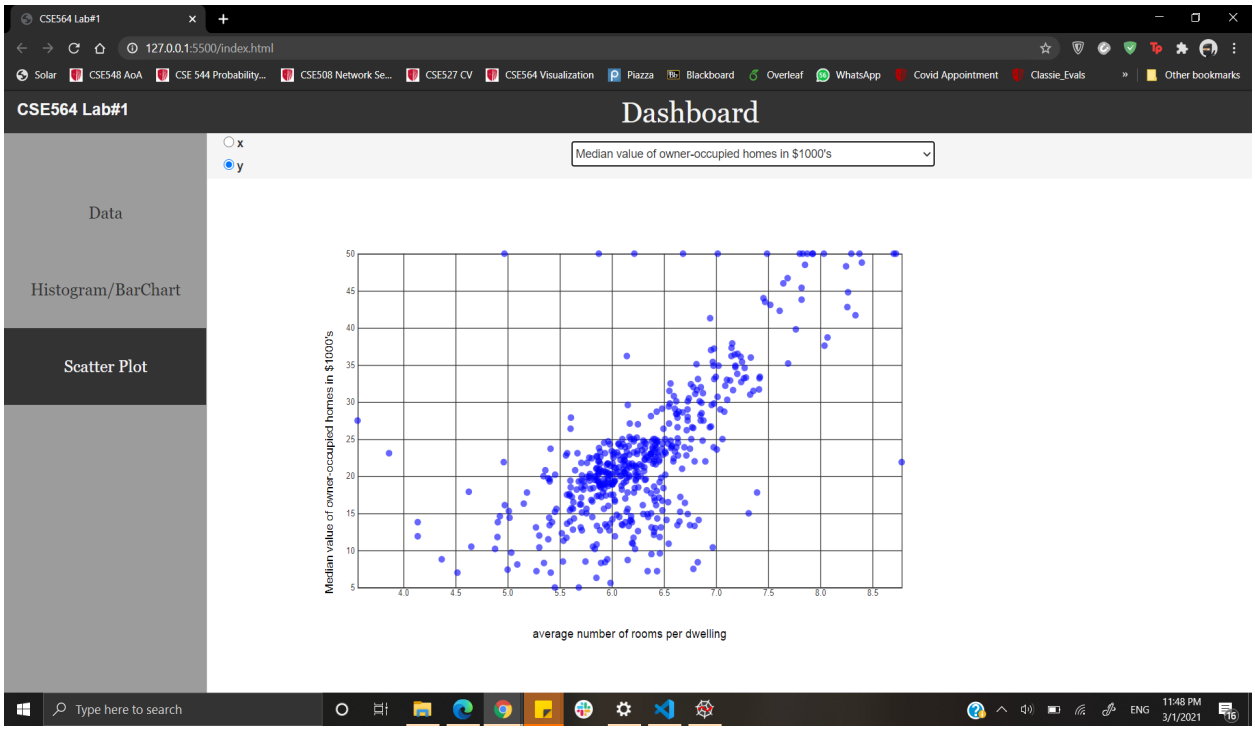
I find the hypothesis of researchers behind the dataset very interesting. They had a hypothesis that people were willing to pay more for clean air —hence the term “hedonic pricing” which in this case is used to describe the monetary value that people assign to factors not inherent to the property but to its surrounding area. In order to explore more about this hypothesis I selected this dataset for my project.

Preprocessing:

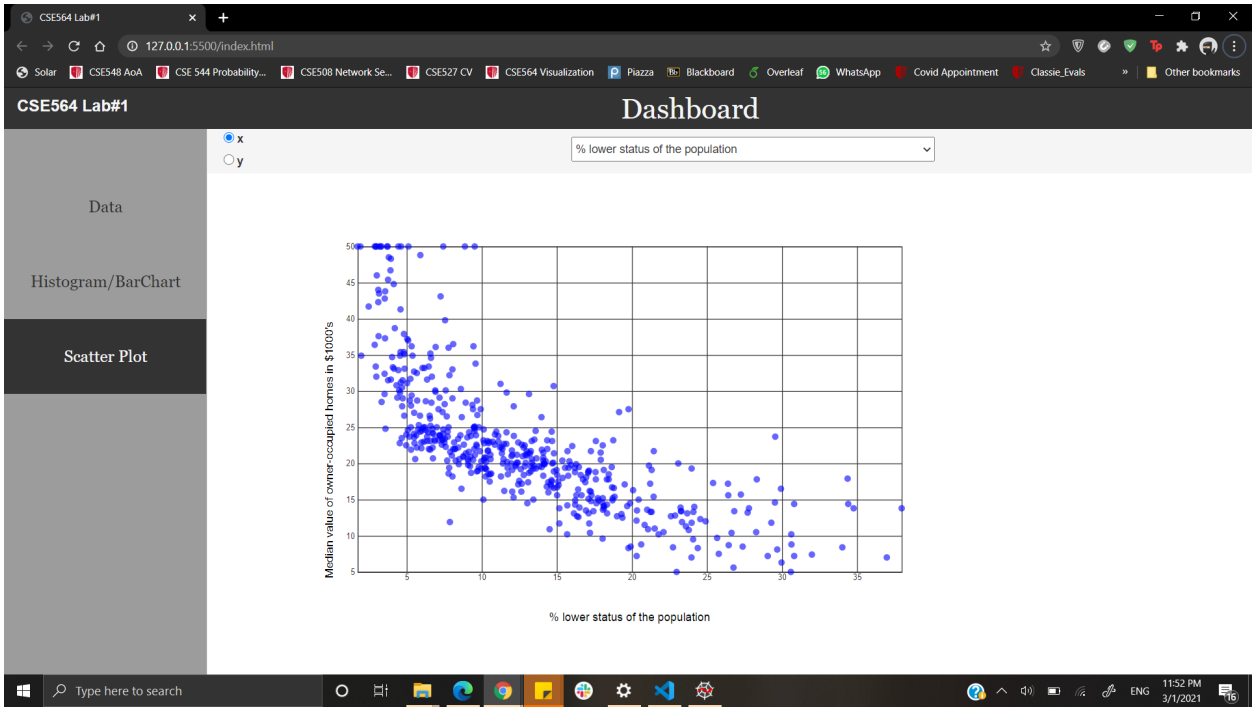
During pre-processing of data missing values of some of continuous variables were replaced by its median value.

Observations:

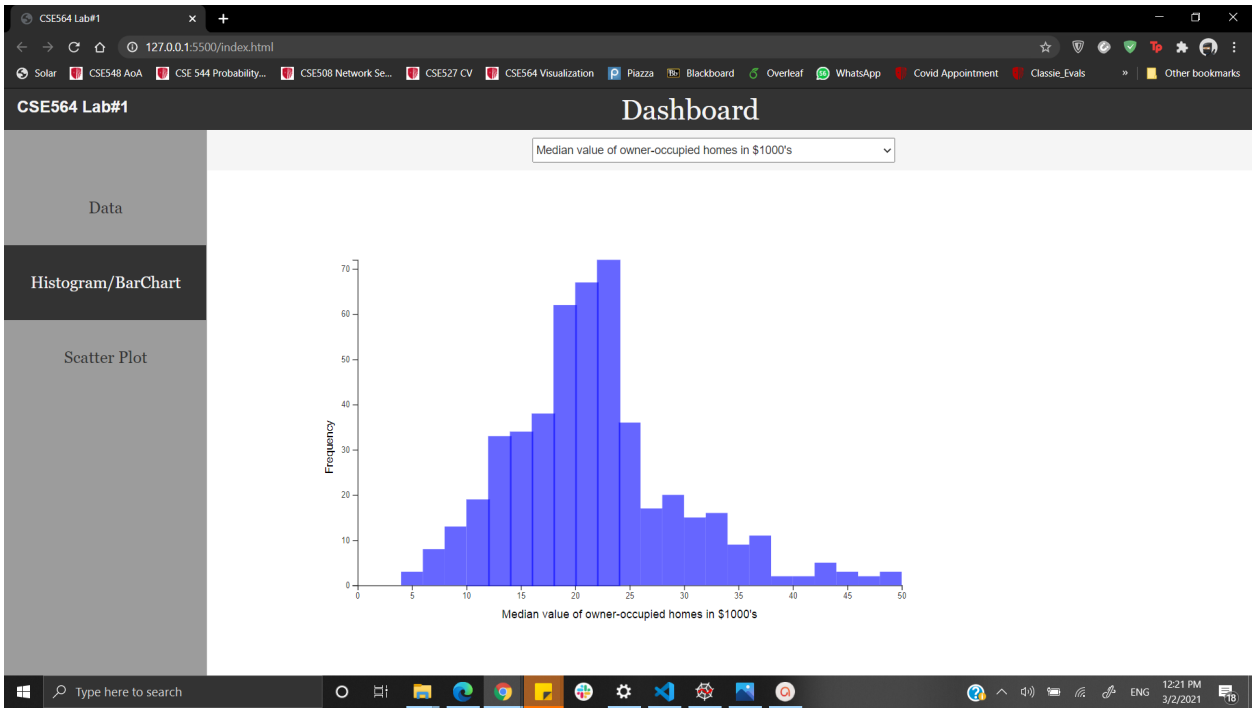
- As observed from the Dashboard, for a high average number of rooms per dwelling, Median value of homes is high.



- Median Value of homes with a higher percentage of lower class population is low.



- MEDV looks to have a normal distribution (the predictions).



Implementation (Code Structure):

This project was implemented in several steps:

- ***data_preprocessing.py*:** Preprocessing for this project was done in python using Pandas and Numpy

- ***Index.html:*** This is the main Dashboard page which deals with all HTML and jQuery functions. It contains 3 Sidebar menu items:
    - **Data:** It contains all the info about the dataset pre-uploaded in the Dashboard.

    - **Histogram/ BarChart:** It deals with a single dropdown menu to select attributes. Based on the attribute, whether it is categorical or numerical it plots Histogram or BarChart. Both Histogram and BarChart supports mouse hover feature in which bins will be highlighted with its value on the top of it. Whereas Histogram also supports a dragging feature in which bin size will decrease if dragged to the left and will increase otherwise.

    - **Scatter Plot:** This page features a single attribute dropdown to select variables and two radio buttons to select on which axis one would like to update (x or y). Then, it will plot a scatter plot on selected variables. It also provides a mouse hover feature in which it will highlight the selected point with Red color, and also increases the size of the point when hovered.

- ***global_vars.js:*** This file deals with all the global variables that are being used in the whole project. From here we can change any properties of the backend like size of plots, transition-time, initial values etc.
- ***homeVis.js:*** This is important to run the project as it updates the backend based on the items selected on UI.
- ***histogram.js, scatter.js, bar.js:*** These files maintain the creation and updation of the respective plots.
- For smoothly transition between the update and during plot change, I have used

```
data().join(
        enter=>
        update =>
        exit =>
        )
```

  method.
- For css, styling I have used a bootstrap library.