Applied Data Science Capstone Project

By

Ramesh D Jadhav

Index

Executive Summary:

I have learn these following Course provided by Coursera:

1) Data Science:

2) Tools for Data Science:

3) Data Science methodology:

4) Python for Data Science, AI & Development:

5) Python Project for Data Science:

6) Database and SQL for Data Science with Python:

7) Data Analysis with Python:

8) Data Visualization with python:

9) Machine Learning with python:

10) Applied Data Science Capstone:

Introduction

- I understood the definition of Data Science and what is Data science with Fundamental of Data Sciences, with Additional the Many Paths to Data Science, Advice for new data scientists and The Sexiest Job in the 21st Century.
- Understand Data, and Types of Data, Important of Data And Data Visualizations with Python.
- And Learned what tools required, completely understand Data Science methodology, Role of phyton in development of AI, and important of SQL.
- Very well understand type of Machine Learning and Completed Various Assignment and Project.

Data collection and data wrangling methodology

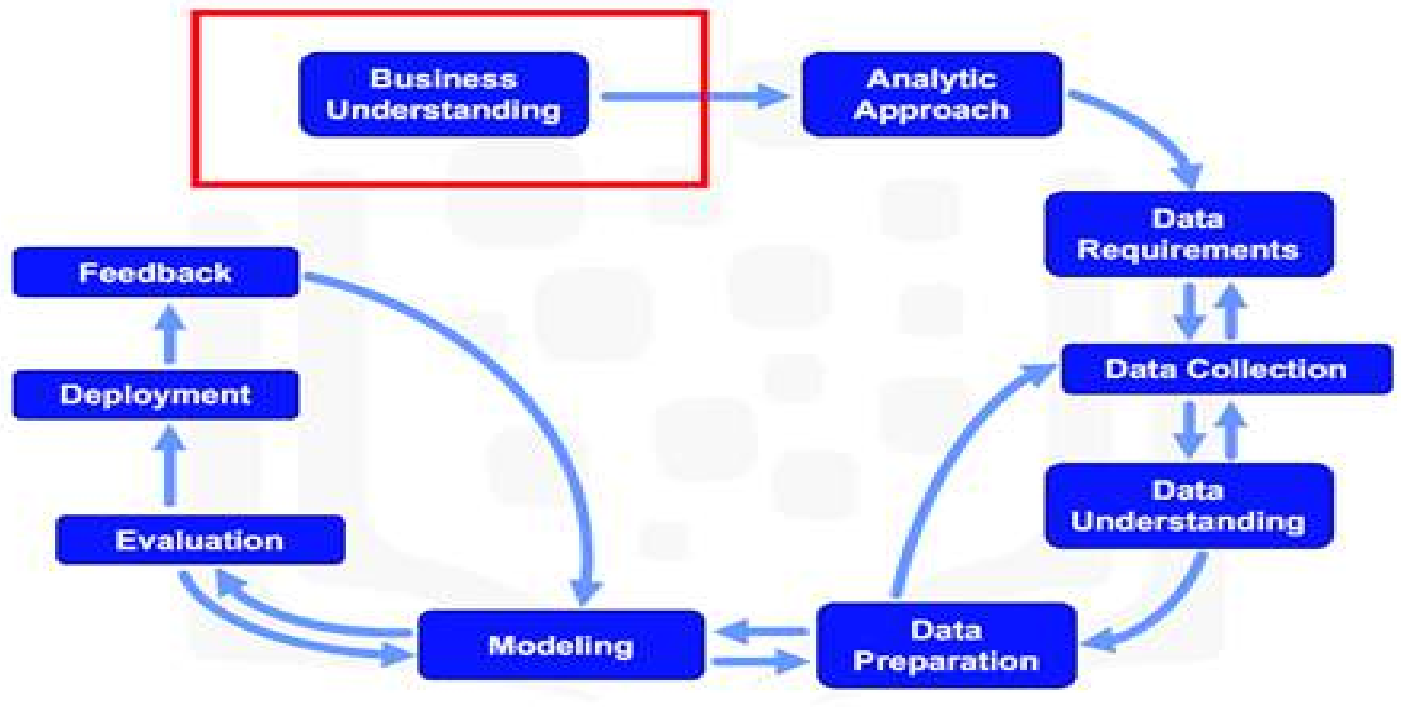
- Data wrangling, a core data analysis technique is not done in one fell swoop–it’s an iterative process that helps you get to the cleanest, most usable data possible prior to your analysis. Without data wrangling, the data set could be nearly impossible to sift through to find crucial insights. Each step in the data wrangling process exposes new potential ways that the data might be “re-wrangled,” all driving towards the ultimate goal of generating the most robust data for final analysis.

Steps of Data wrangling

1. Discovering – allows you to understand your data and how it’s useful for analytic exploration and analysis
2. Structuring – gives you the ability to format data of all shapes and sizes to work with traditional applications
3. Cleaning – lets you fix and standardize the data that might distort your analysis
4. Enriching – allows you to take advantage of the wrangling.
5. Validating – identifies and surfaces data quality and consistency issues
6. Publishing – provides you the ability to plan for and deliver data for downstream analysis

Data Science methodology

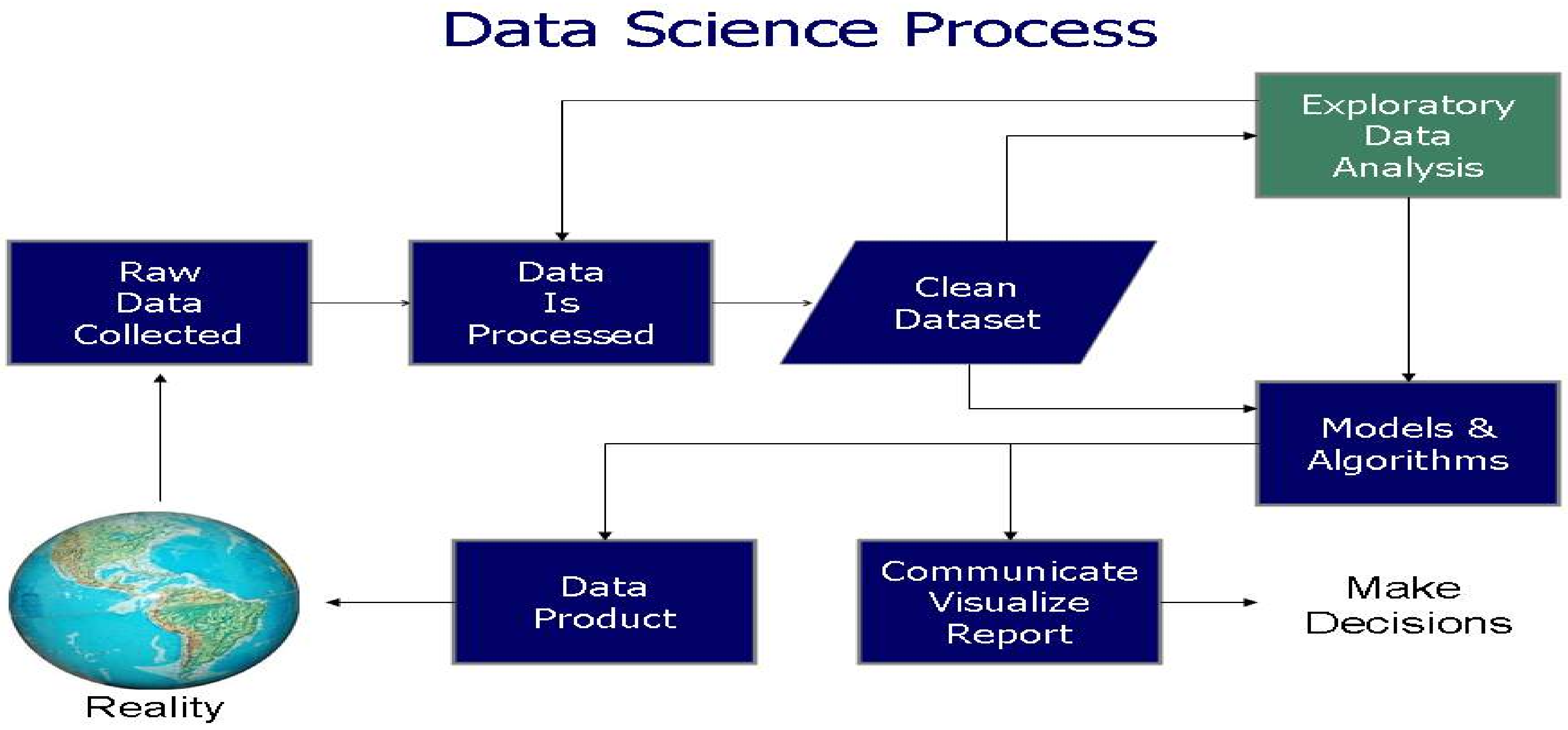
This is the **Data Science Methodology**, a flowchart that begins with business understanding.



EDA and interactive visual analytics methodology

- **Exploratory data analysis** is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

EDA and interactive visual analytics methodology



Tools for Data Science:

python™
R
SQL
Scala
Java™ POWERED
C++
julia
JS
php
GO
Visual Basic

Predictive analysis methodology

Predictive analytics is a branch of advanced analytics that makes predictions about future outcomes using historical data combined with statistical modeling, data mining techniques and machine learning. Companies employ predictive analytics to find patterns in this data to identify risks and opportunities.

Predictive analytics is often associated with big data and data science. Companies today are swimming in data that resides across transactional databases, equipment log files, images, video, sensors or other data sources. To gain insights from this data, data scientists use deep learning and machine learning algorithms to find patterns and make predictions about future events. These include linear and nonlinear regression, neural networks, support vector machines and decision trees. Learnings obtained through predictive analytics can then be used further within prescriptive analytics to drive actions based on predictive insights.

EDA using Data Visualization

EDA using Data Visualization. Exploratory data analysis is a way to better understand your data which helps in further Data preprocessing. And data visualization is key, making the exploratory data analysis process streamline and easily analyzing data using wonderful plots and charts.

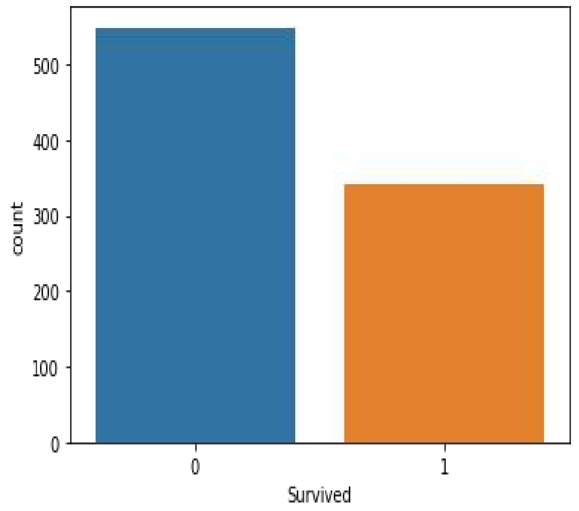
Data Visualization:
Data Visualization represents the text or numerical data in a visual format, which makes it easy to grasp the information the data express. We, humans, remember the pictures more easily than readable text, so Python provides us various libraries for data visualization like matplotlib, seaborn, plotly, etc. In this tutorial, we will use Matplotlib and seaborn for performing various techniques to explore data using various plots.

Exploratory Data Analysis:
Creating Hypotheses, testing various business assumptions while dealing with any Machine learning problem statement is very important and this is what EDA helps to accomplish. There are various tootle and techniques to understand your data, And the basic need is you should have the knowledge of Numpy for mathematical operations and Pandas for data manipulation.

1) CountPlot

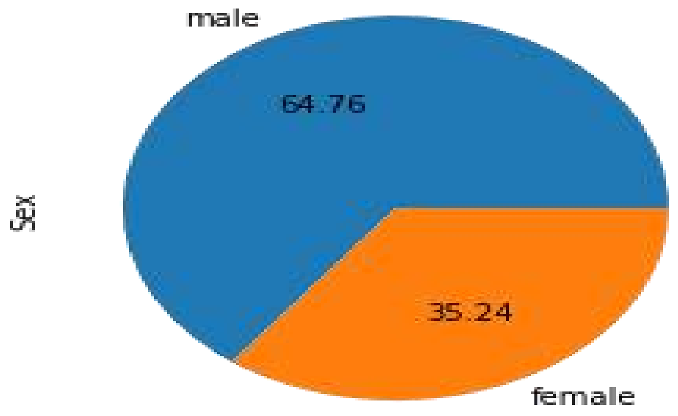
- Countplot is basically a count of frequency plot in form of a bar graph. It plots the count of each category in a separate bar. When we use the pandas’ value counts function on any column, It is the same visual form of the value counts function. In our data-target variable is survived and it is categorical so let us plot a countplot of this.

e.g
```
sns.countplot(data['Survived’])
plt.show()
```



Pie Chart

- The pie chart is also the same as the countplot, only gives you additional information about the percentage presence of each category in data means which category is getting how much weightage in data. let us check about the Sex column, what is a percentage of Male and Female members traveling.
- e.g data['Sex'].value_counts().plot(kind="pie", autopct="%.2f")
  plt.show()



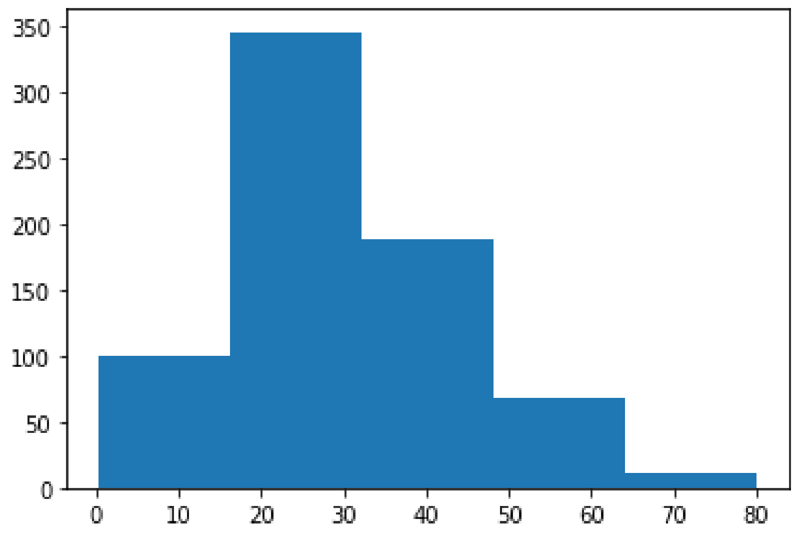
Numerical Data

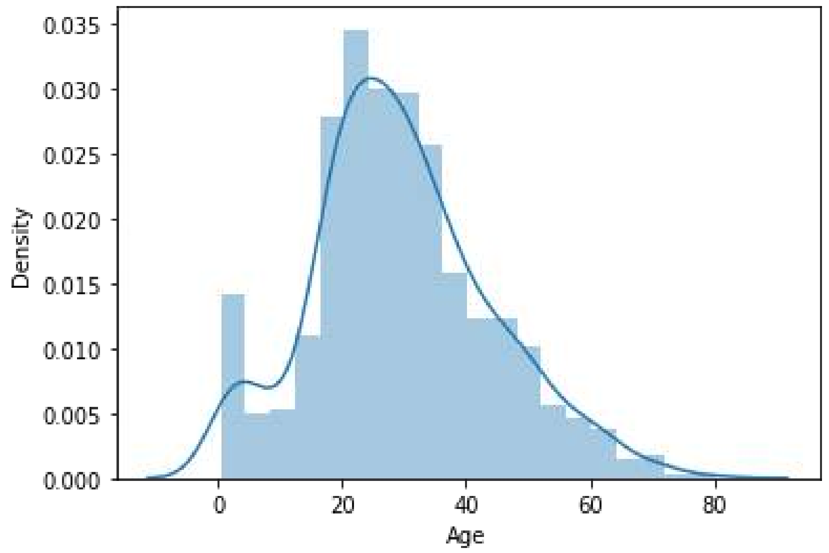
- Analyzing Numerical data is important because understanding the distribution of variables helps to further process the data. Most of the time you will find much inconsistency with numerical data so do explore numerical variables.
- **1) Histogram:**
- A histogram is a value distribution plot of numerical columns. It basically creates bins in various ranges in values and plots it where we can visualize how values are distributed. We can have a look where more values lie like in positive, negative, or at the center(mean). Let’s have a look at the Age column.
- **2) Distplot:**
- Distplot is also known as the second Histogram because it is a slight improvement version of the Histogram. Distplot gives us a KDE(Kernel Density Estimation) over histogram which explains PDF(Probability Density Function) which means what is the probability of each value occurring in this column. If you have study statistics before then definitely you should know about PDF function.

Histogram and Distplot

e.g. plt.hist(data['Age'], bins=5 plt.show()

e.g sns.distplot(data['Age']) plt.show()





Boxplot

Boxplot is a very interesting plot that basically plots a 5 number summary. to get 5 number summary some terms we need to describe.
•Median – Middle value in series after sorting
•Percentile – Gives any number which is number of values present before this percentile like for example 50 under 25th percentile so it explains total of 50 values are there below 25th percentile
•Minimum and Maximum – These are not minimum and maximum values, rather they describe the lower and upper boundary of standard deviation which is calculated using Interquartile range(IQR).

IQR = Q3 - Q1
Lower_boundary = Q1 - 1.5 * IQR
Upper_bounday = Q3 + 1.5 * IQR

Here Q1 and Q3 is 1st quantile(25th percentile) and 3rd Quantile(75th percentile)

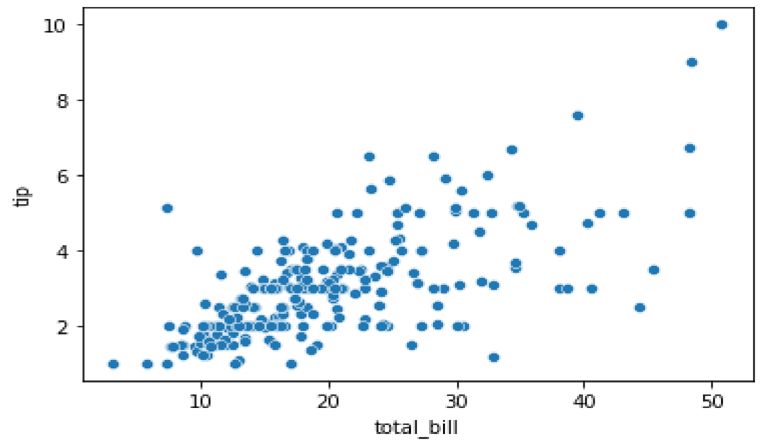
Bivariate/ Multivariate Analysis

We have study about various plots to explore single categorical and numerical data. Bivariate Analysis is used when we have to explore the relationship between 2 different variables and we have to do this because, in the end, our main task is to explore the relationship between variables to build a powerful model. And when we analyze more than 2 variables together then it is known as Multivariate Analysis. we will work on different plots for Bivariate as well on Multivariate Analysis.

Numerical and Numerical
First, let's explore the plots when both the variable is numerical.

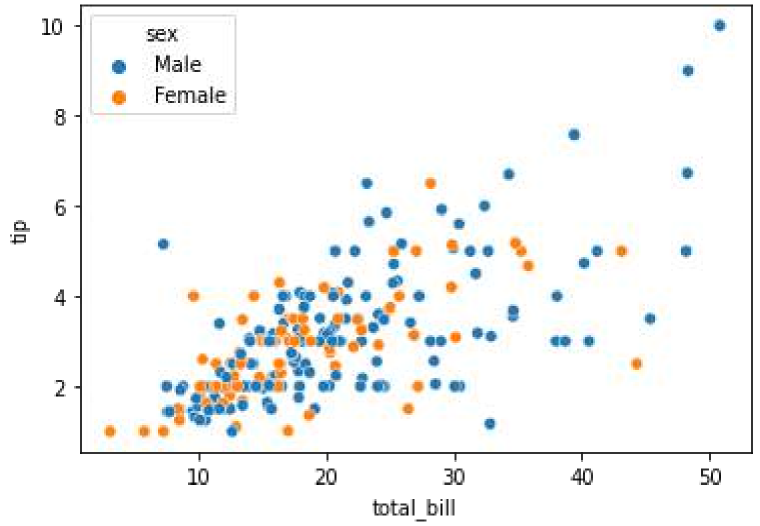
1) Scatter Plot
To plot the relationship between two numerical variables scatter plot is a simple plot to do. Let us see the relationship between the total bill and tip provided using a scatter plot.

e.g. sns.scatterplot(tips["total_bill"], tips["tip"])



Multivariate analysis with scatter plot

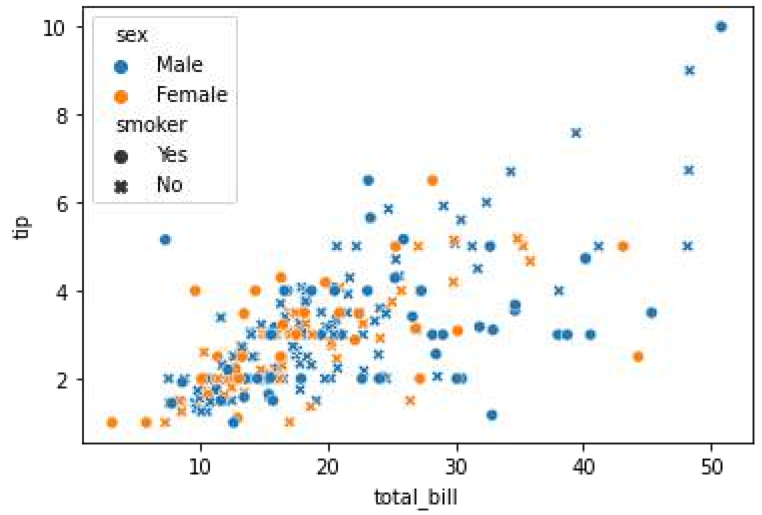
- we can also plot 3 variable or 4 variable relationships with scatter plot. suppose we want to find the separate ratio of male and female with total bill and tip provided.
- e.g. sns.scatterplot(tips["total_bill"], tips["tip"], hue=tips["sex"]) plt.show()



Multivariate analysis with scatter plot

- We can also see 4 variable multivariate analyses with scatter plots using style argument. Suppose now along with gender I also want to know whether the customer was a smoker or not so we can do this.

```
sns.scatterplot(tips["total_bill"], tips["tip"], hue=tips["sex"], style=tips['smoker']) plt.show()
```



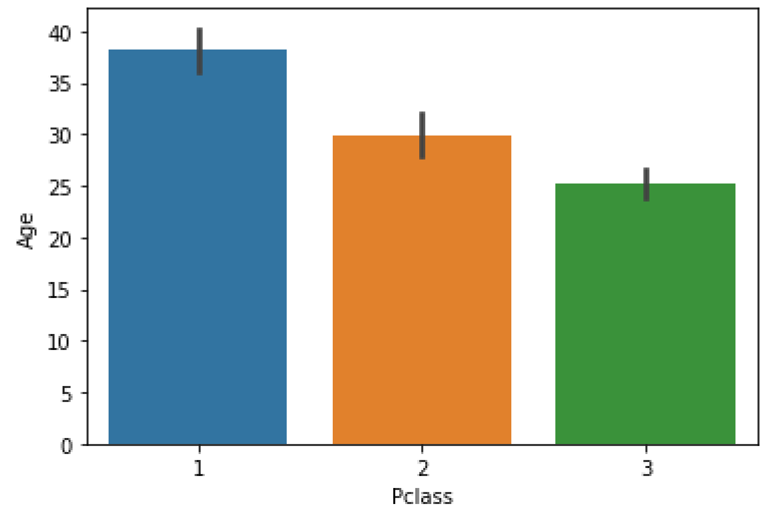
Numerical and Categorical

If one variable is numerical and one is categorical then there are various plots that we can use for Bivariate and Multivariate analysis.

1) Bar Plot

Bar plot is a simple plot which we can use to plot categorical variable on the x-axis and numerical variable on y-axis and explore the relationship between both variables. The blacktip on top of each bar shows the confidence Interval. let us explore P-Class with age.

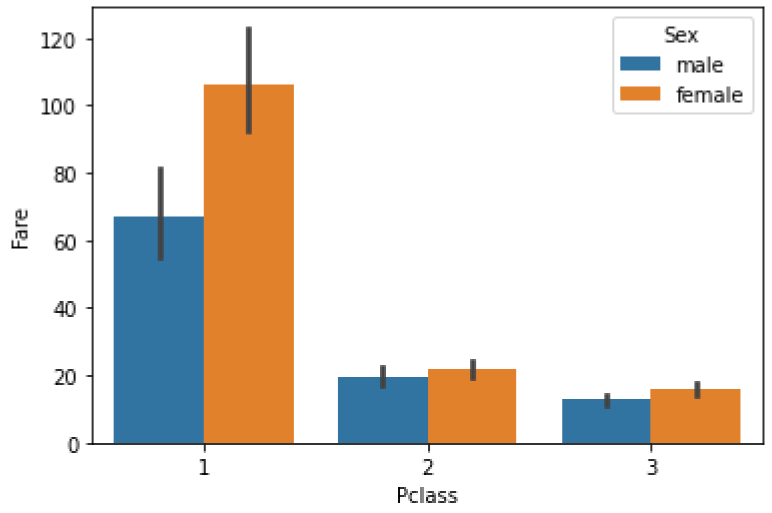
```
e.g. sns.barplot(data['Pclass'], data['Age’])
plt.show()
```



Multivariate analysis using Bar plot

- Hue’s argument is very useful which helps to analyze more than 2 variables. Now along with the above relationship we want to see with gender.

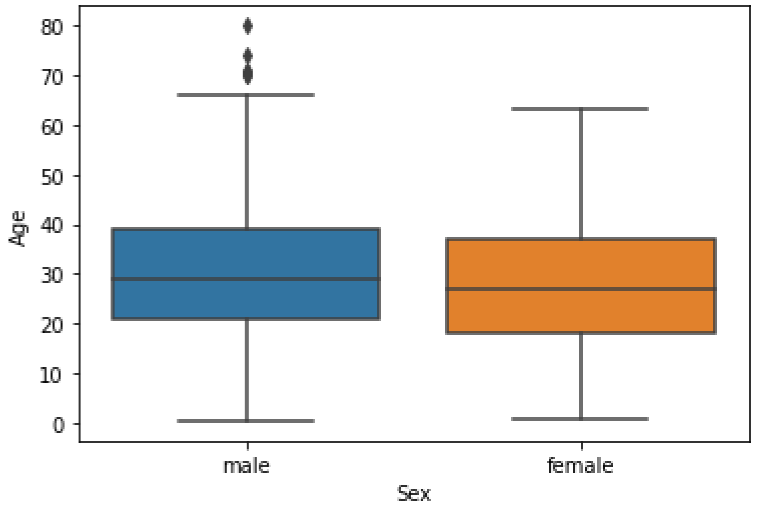
e.g. sns.barplot(data['Pclass'], data['Fare'], hue = data["Sex"]) plt.show()



Boxplot

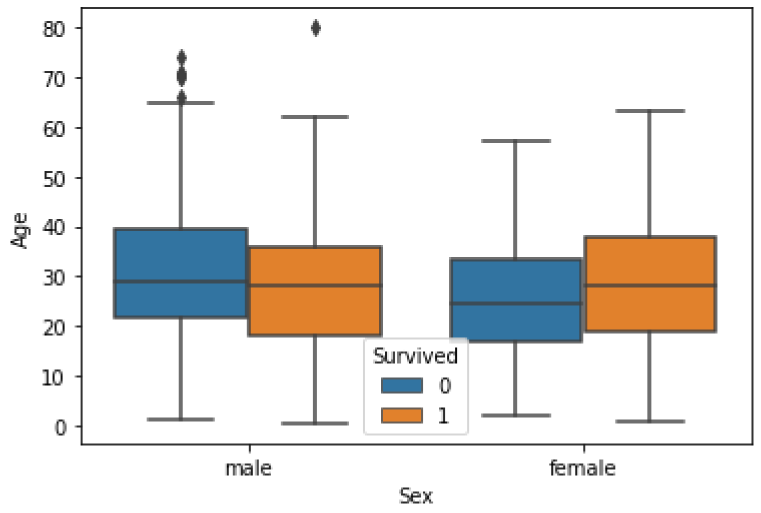
- We have already study about boxplots in the Univariate analysis above. we can draw a separate boxplot for both the variable. let us explore gender with age using a boxplot.

e.g. sns.boxplot(data['Sex'], data["Age"])



Multivariate analysis with boxplot

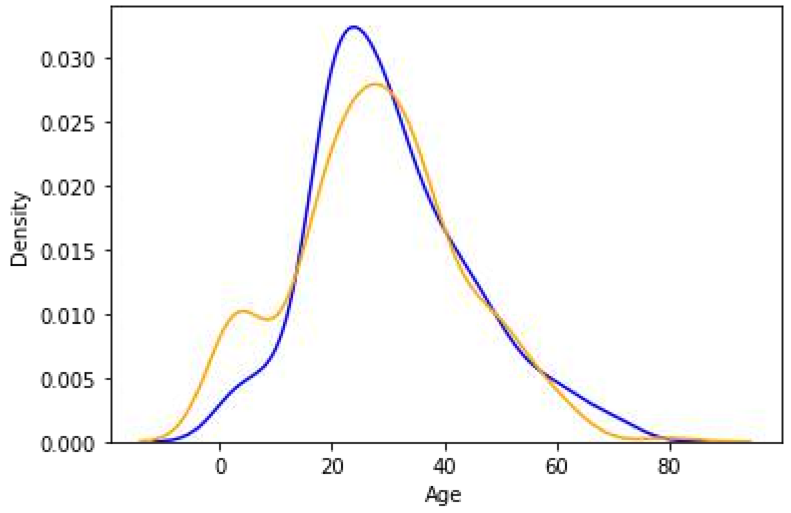
- Along with age and gender let’s see who has survived and who has not.
- e.g. sns.boxplot(data['Sex'], data["Age"], data["Survived"]) plt.show()



3) Distplot

- Distplot explains the PDF function using kernel density estimation. Distplot does not have a hue parameter but we can create it. suppose we want to see the probability of people with an age range that of survival probability and find out whose survival probability is high to the age range of death ratio.

e.g. sns.distplot(data[data['Survived'] == 0]['Age'], hist=False, color="blue")
sns.distplot(data[data['Survived'] == 1]['Age'], hist=False, color="orange")
plt.show()



As we can see the graph is really very interesting. the blue one shows the probability of dying and the orange plot shows the survival probability. If we observe it we can see that children’s survival probability is higher than death and which is the opposite in the case of aged peoples. This small analysis tells sometimes some big things about data and It helps while preparing data stories.

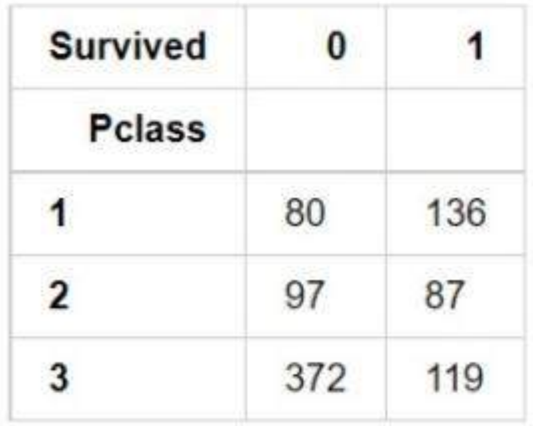
Categorical and Categorical

Now we will work on categorical and categorical columns.

1) Heatmap

If you have ever used a crosstab function of pandas then Heatmap is a similar visual representation of that only. It basically shows that how much presence of one category concerning another category is present in the dataset. let me show first with crosstab and then with heatmap.

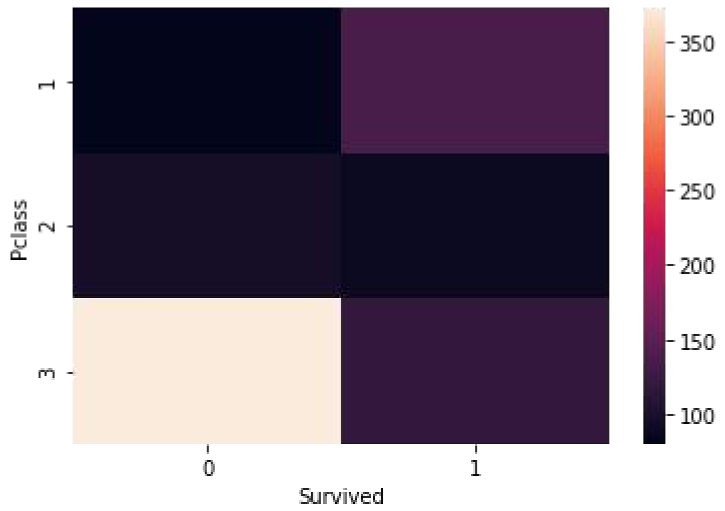
e.g. pd.crosstab(data['Pclass'], data['Survived'])



Survived	0	1
Pclass		
1	80	136
2	97	87
3	372	119

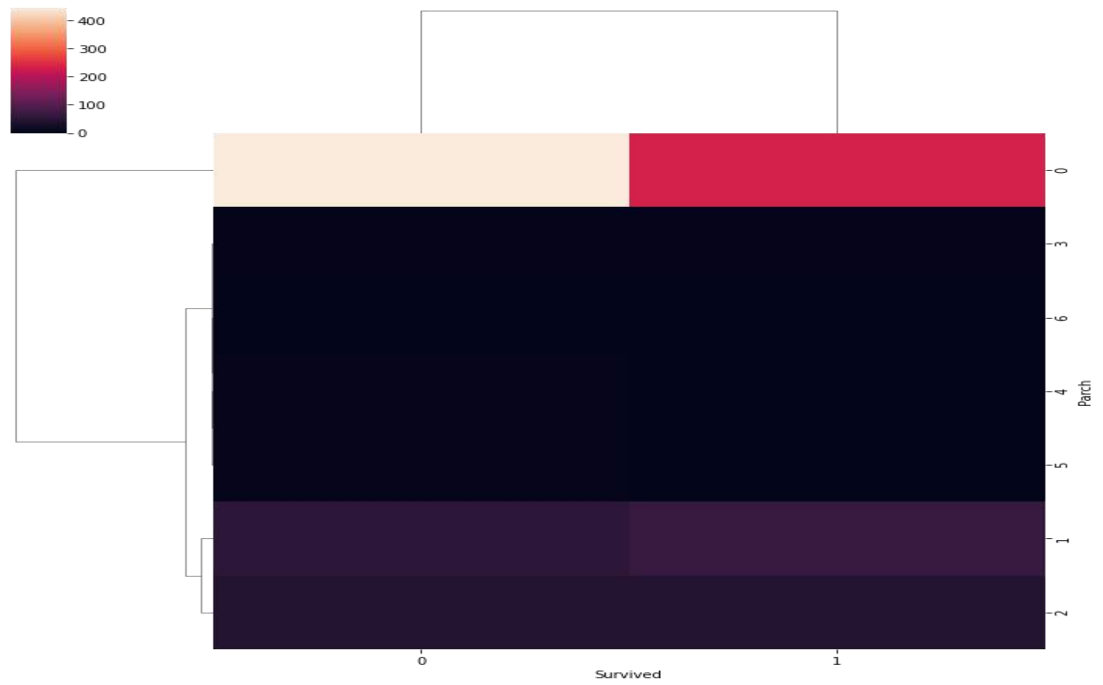
Now with heatmap, we have to find how many people survived and died.

e.g. sns.heatmap(pd.crosstab(data['Pclass'], data['Survived']))



Cluster map

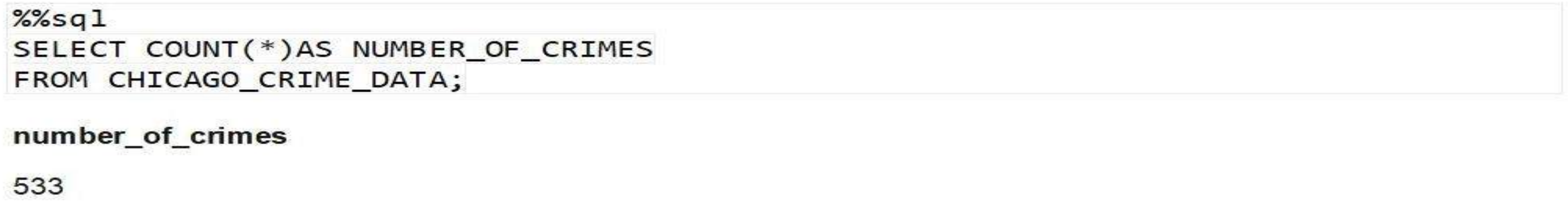
- we can also use a cluster map to understand the relationship between two categorical variables. A cluster map basically plots a dendrogram that shows the categories of similar behavior together.
- e.g. sns.clustermap(pd.crosstab(data['Parch'], data['Survived']))
  plt.show()



Cluster map

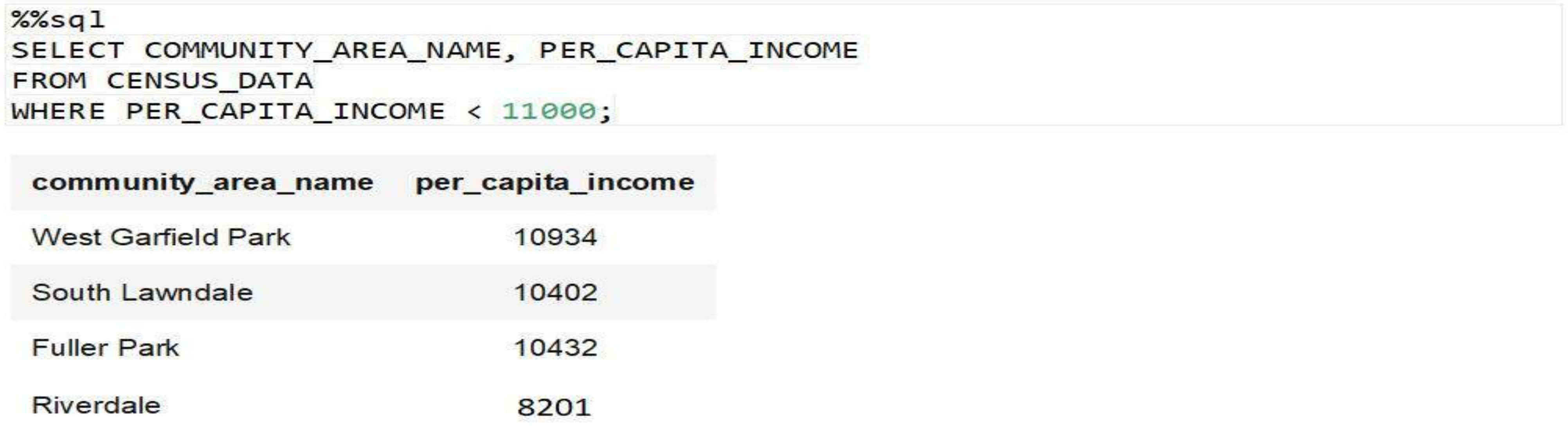
- If you know about clustering algorithms mainly about DBSCAN, then you should know about dendrogram. So, these are all the plots that are mostly used while doing exploratory analysis. There are some more plots that you can draw like violent plot, line plot, a joint plot which are not mostly used.

EDA with SQL :



```
%%sql
SELECT COUNT(*)AS NUMBER_OF_CRIMES
FROM CHICAGO_CRIME_DATA;
```

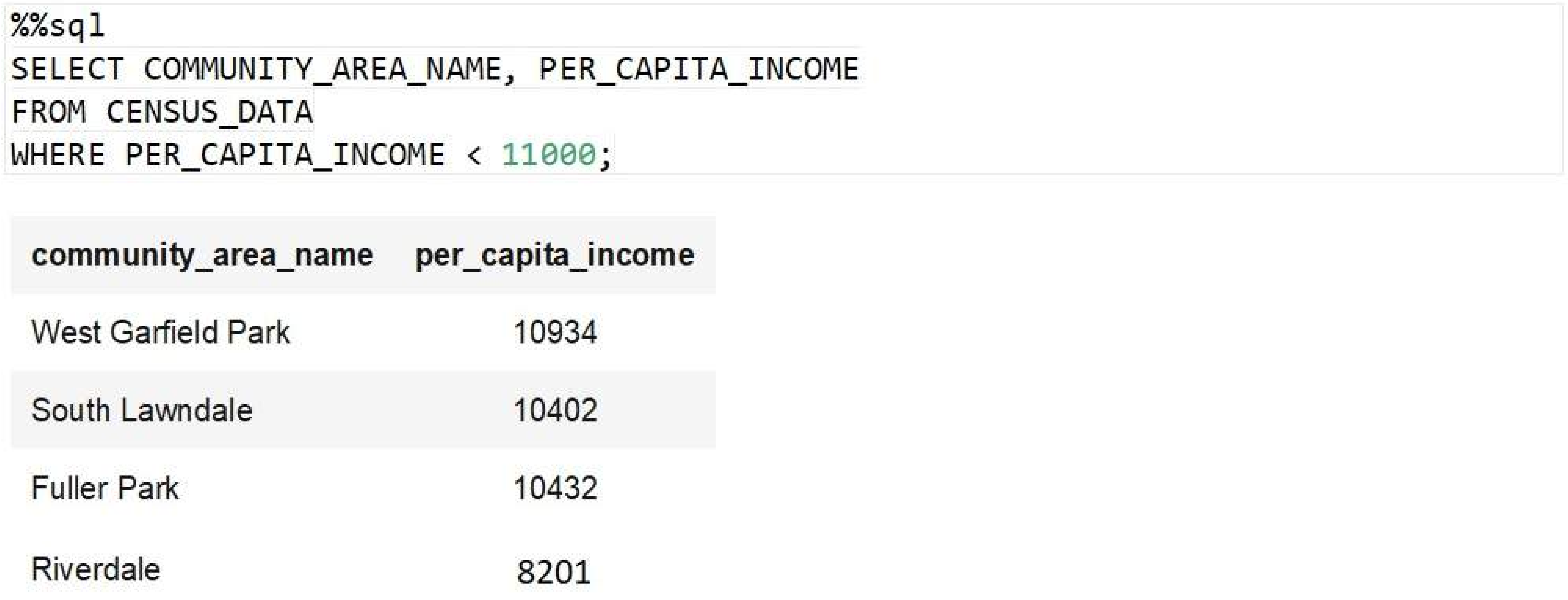
number_of_crimes
533



```
%%sql
SELECT COMMUNITY_AREA_NAME, PER_CAPITA_INCOME
FROM CENSUS_DATA
WHERE PER_CAPITA_INCOME < 11000;
```

community_area_name	per_capita_income
West Garfield Park	10934
South Lawndale	10402
Fuller Park	10432
Riverdale	8201

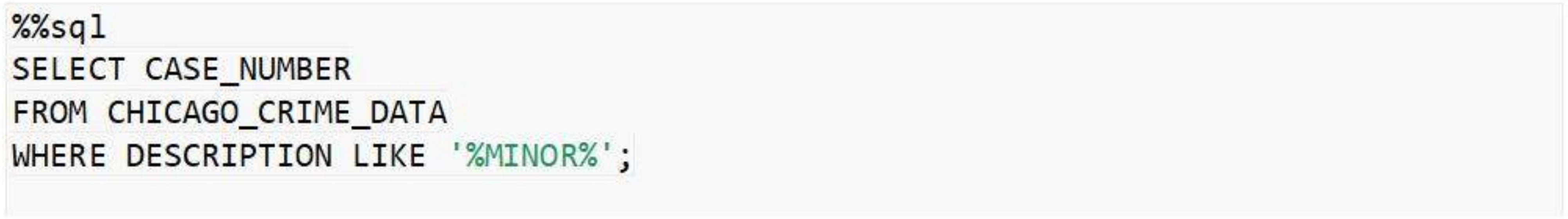
EDA with SQL :



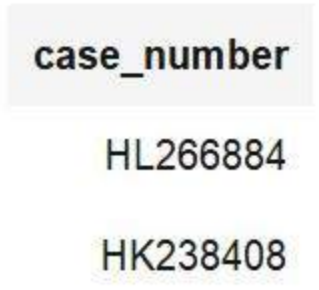
```
%%sql
SELECT COMMUNITY_AREA_NAME, PER_CAPITA_INCOME
FROM CENSUS_DATA
WHERE PER_CAPITA_INCOME < 11000;
```

community_area_name	per_capita_income
West Garfield Park	10934
South Lawndale	10402
Fuller Park	10432
Riverdale	8201

EDA with SQL :

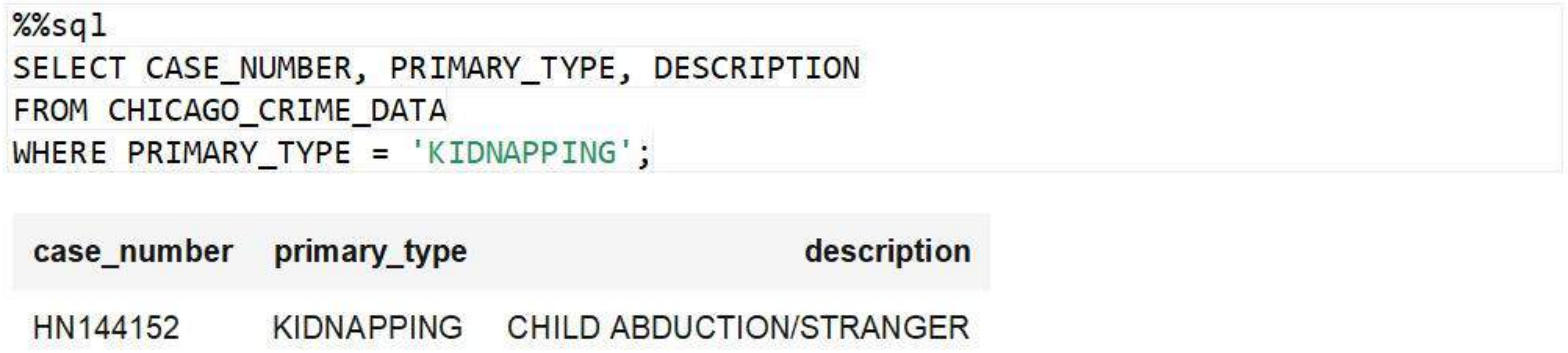


```
%%sql
SELECT CASE_NUMBER
FROM CHICAGO_CRIME_DATA
WHERE DESCRIPTION LIKE '%MINOR%';
```



case_number
HL266884
HK238408

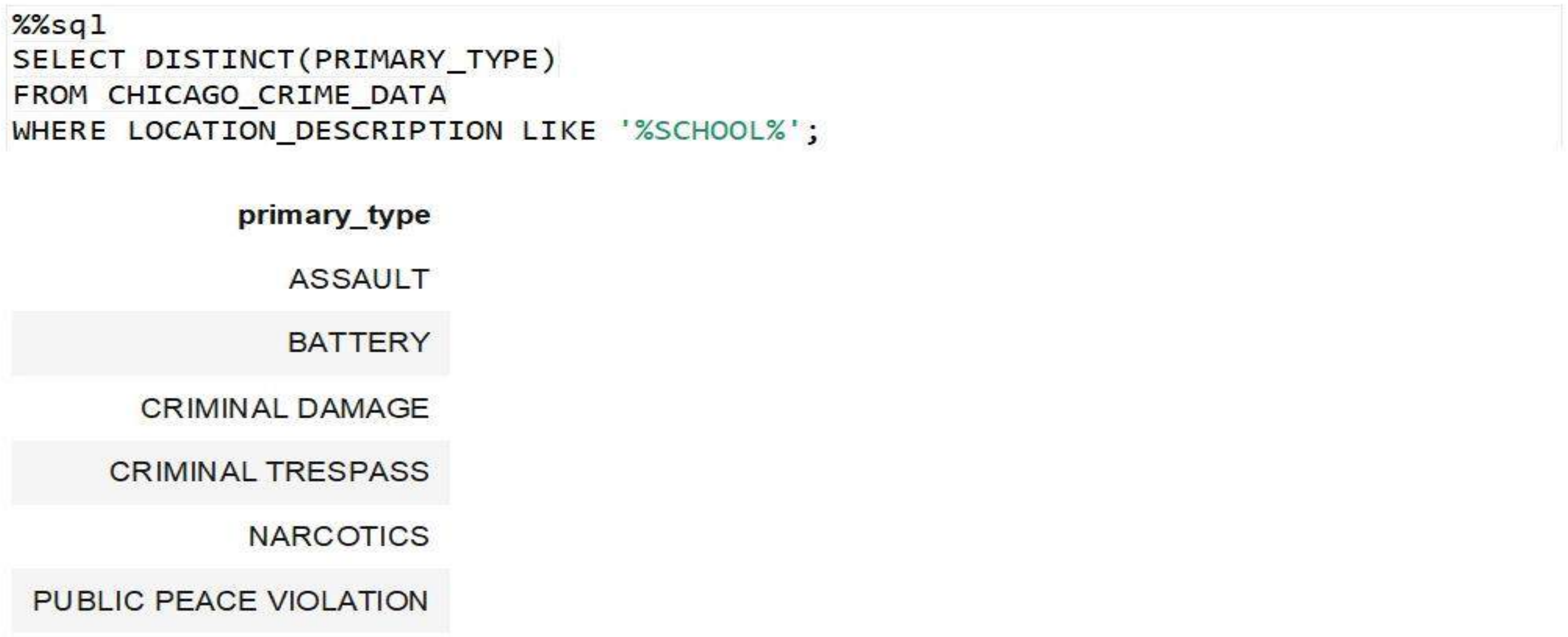
EDA with SQL :



```
%%sql
SELECT CASE_NUMBER, PRIMARY_TYPE, DESCRIPTION
FROM CHICAGO_CRIME_DATA
WHERE PRIMARY_TYPE = 'KIDNAPPING';
```

case_number	primary_type	description
HN144152	KIDNAPPING	CHILD ABDUCTION/STRANGER

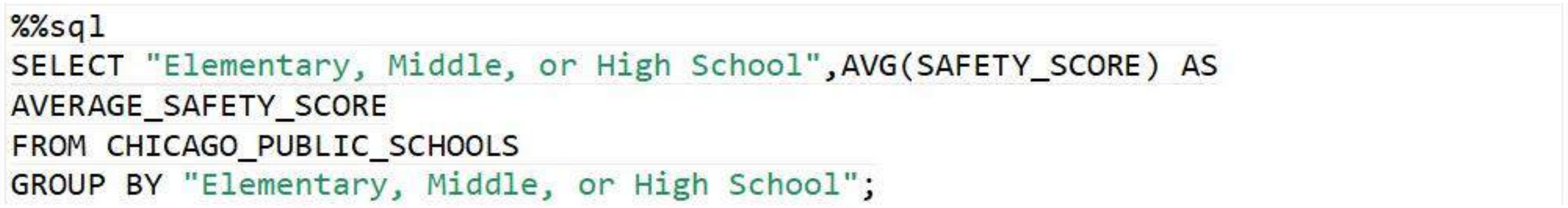
EDA with SQL :



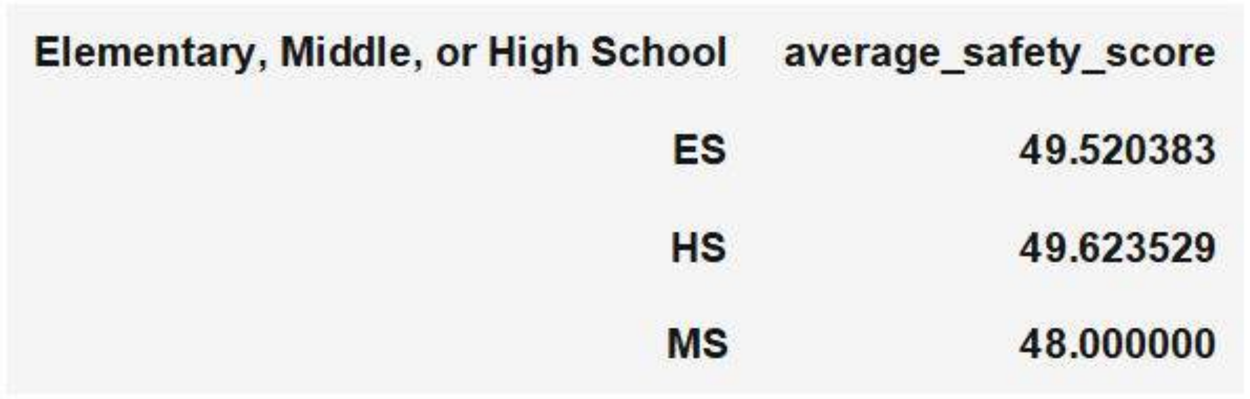
```
%%sql
SELECT DISTINCT(PRIMARY_TYPE)
FROM CHICAGO_CRIME_DATA
WHERE LOCATION_DESCRIPTION LIKE '%SCHOOL%';
```

primary_type
ASSAULT
BATTERY
CRIMINAL DAMAGE
CRIMINAL TRESPASS
NARCOTICS
PUBLIC PEACE VIOLATION

EDA with SQL :

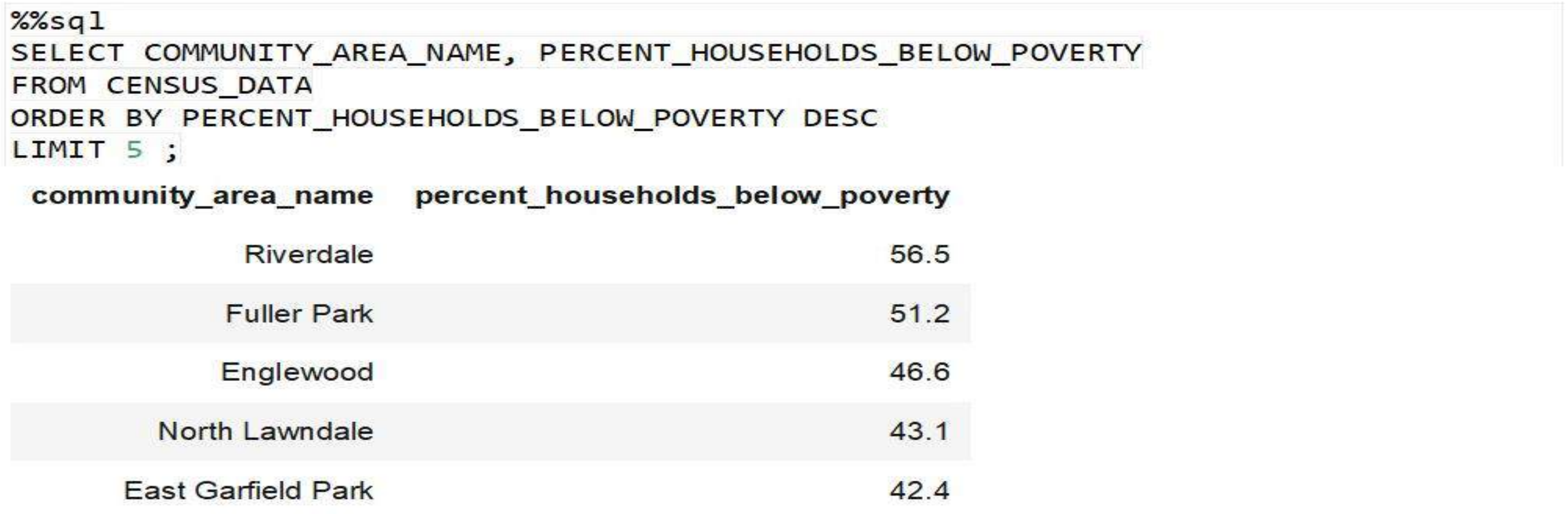


```
%%sql
SELECT "Elementary, Middle, or High School",AVG(SAFETY_SCORE) AS
AVERAGE_SAFETY_SCORE
FROM CHICAGO_PUBLIC_SCHOOLS
GROUP BY "Elementary, Middle, or High School";
```



Elementary, Middle, or High School	average_safety_score
ES	49.520383
HS	49.623529
MS	48.000000

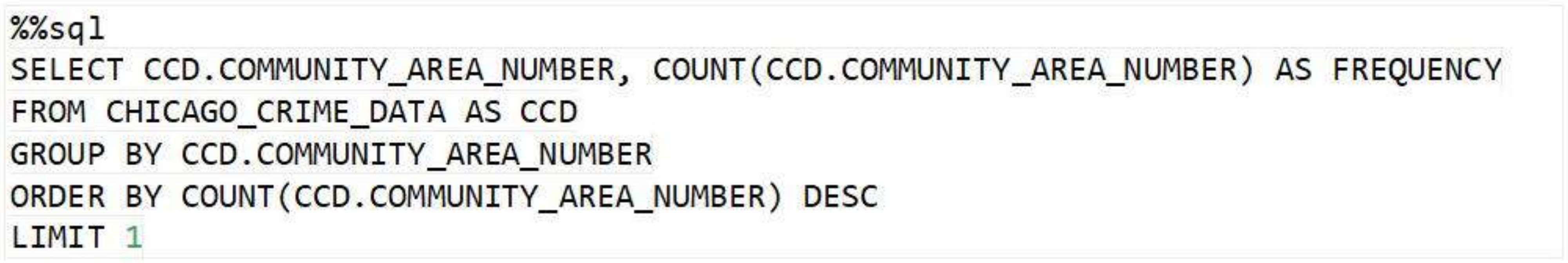
EDA with SQL :



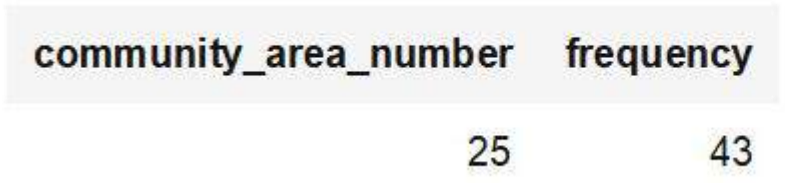
```
%%sql
SELECT COMMUNITY_AREA_NAME, PERCENT_HOUSEHOLDS_BELOW_POVERTY
FROM CENSUS_DATA
ORDER BY PERCENT_HOUSEHOLDS_BELOW_POVERTY DESC
LIMIT 5 ;
```

community_area_name	percent_households_below_poverty
Riverdale	56.5
Fuller Park	51.2
Englewood	46.6
North Lawndale	43.1
East Garfield Park	42.4

EDA with SQL :

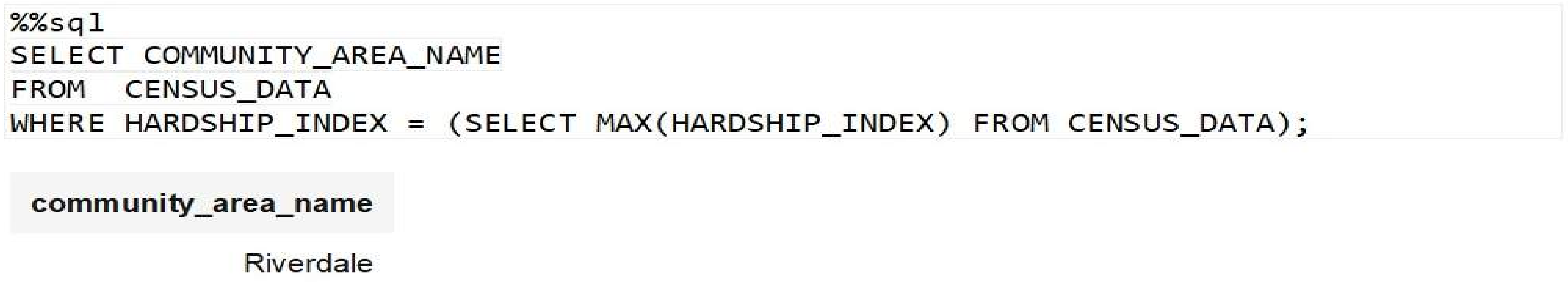


```
%%sql
SELECT CCD.COMMUNITY_AREA_NUMBER, COUNT(CCD.COMMUNITY_AREA_NUMBER) AS FREQUENCY
FROM CHICAGO_CRIME_DATA AS CCD
GROUP BY CCD.COMMUNITY_AREA_NUMBER
ORDER BY COUNT(CCD.COMMUNITY_AREA_NUMBER) DESC
LIMIT 1
```



community_area_number	frequency
25	43

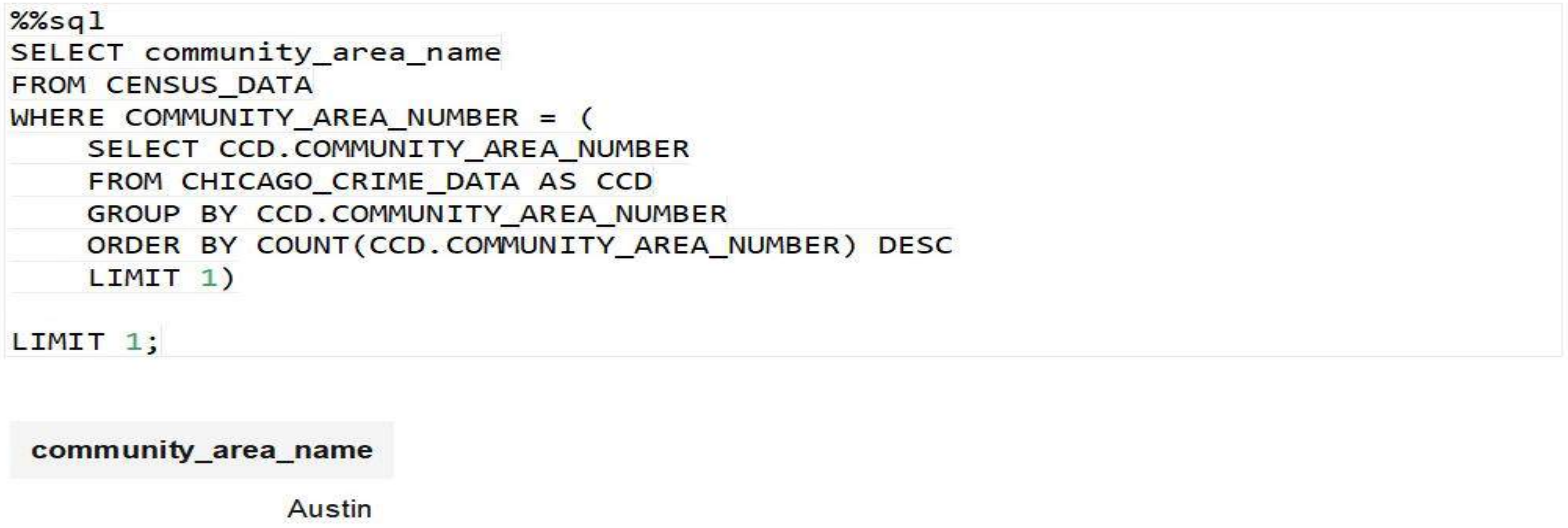
EDA with SQL :



```
%%sql
SELECT COMMUNITY_AREA_NAME
FROM  CENSUS_DATA
WHERE HARDSHIP_INDEX = (SELECT MAX(HARDSHIP_INDEX) FROM CENSUS_DATA);
```

community_area_name
Riverdale

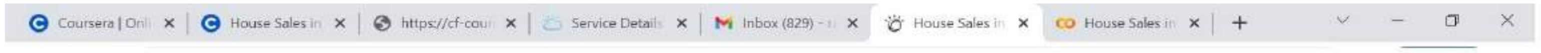
EDA with SQL :

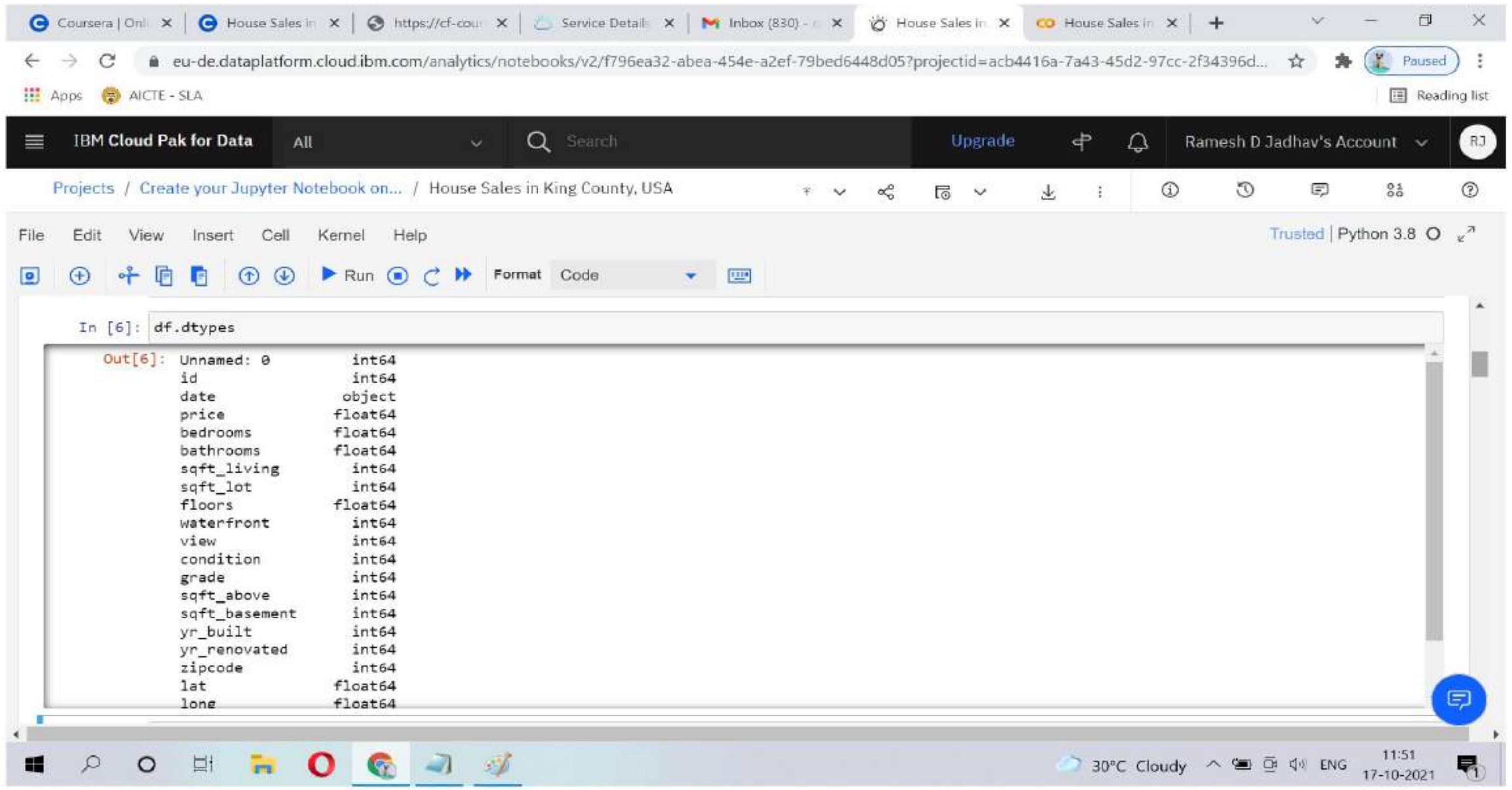


```
%%sql
SELECT community_area_name
FROM CENSUS_DATA
WHERE COMMUNITY_AREA_NUMBER = (
    SELECT CCD.COMMUNITY_AREA_NUMBER
    FROM CHICAGO_CRIME_DATA AS CCD
    GROUP BY CCD.COMMUNITY_AREA_NUMBER
    ORDER BY COUNT(CCD.COMMUNITY_AREA_NUMBER) DESC
    LIMIT 1)

LIMIT 1;
```

community_area_name
Austin

Interactive map with Folium

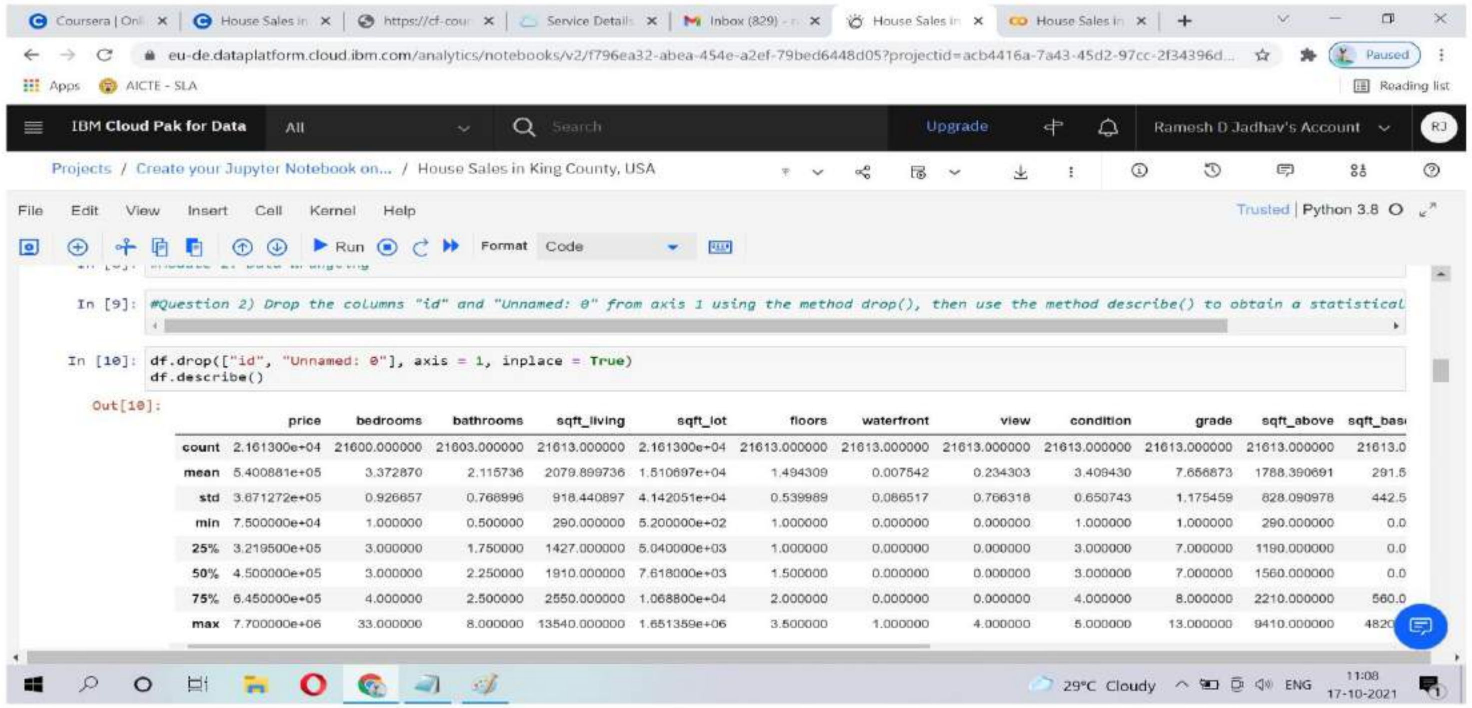




```
In [6]: df.dtypes
```

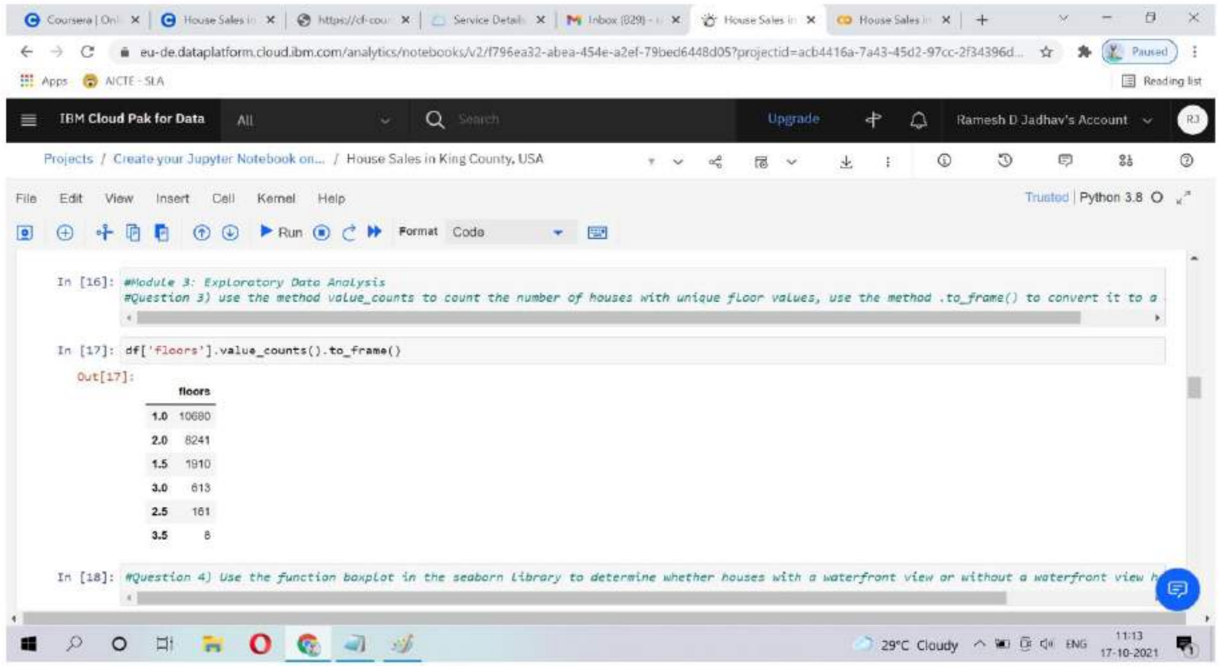
```
Out[6]: Unnamed: 0          int64
        id                  int64
        date               object
        price             float64
        bedrooms          float64
        bathrooms         float64
        sqft_living         int64
        sqft_lot            int64
        floors            float64
        waterfront          int64
        view                int64
        condition           int64
        grade               int64
        sqft_above          int64
        sqft_basement       int64
        yr_built            int64
        yr_renovated        int64
        zipcode             int64
        lat               float64
        long              float64
```

Interactive map with Folium


eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05?projectid=acb4416a-7a43-45d2-97cc-2f34396d...
Apps
AICTE - SLA
Reading list
IBM Cloud Pak for Data
All
Search
Upgrade
Ramesh D Jadhav's Account
RJ
Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
File Edit View Insert Cell Kernel Help
Trusted | Python 3.8
Run
Format Code
29°C Cloudy
ENG
11:08
17-10-2021

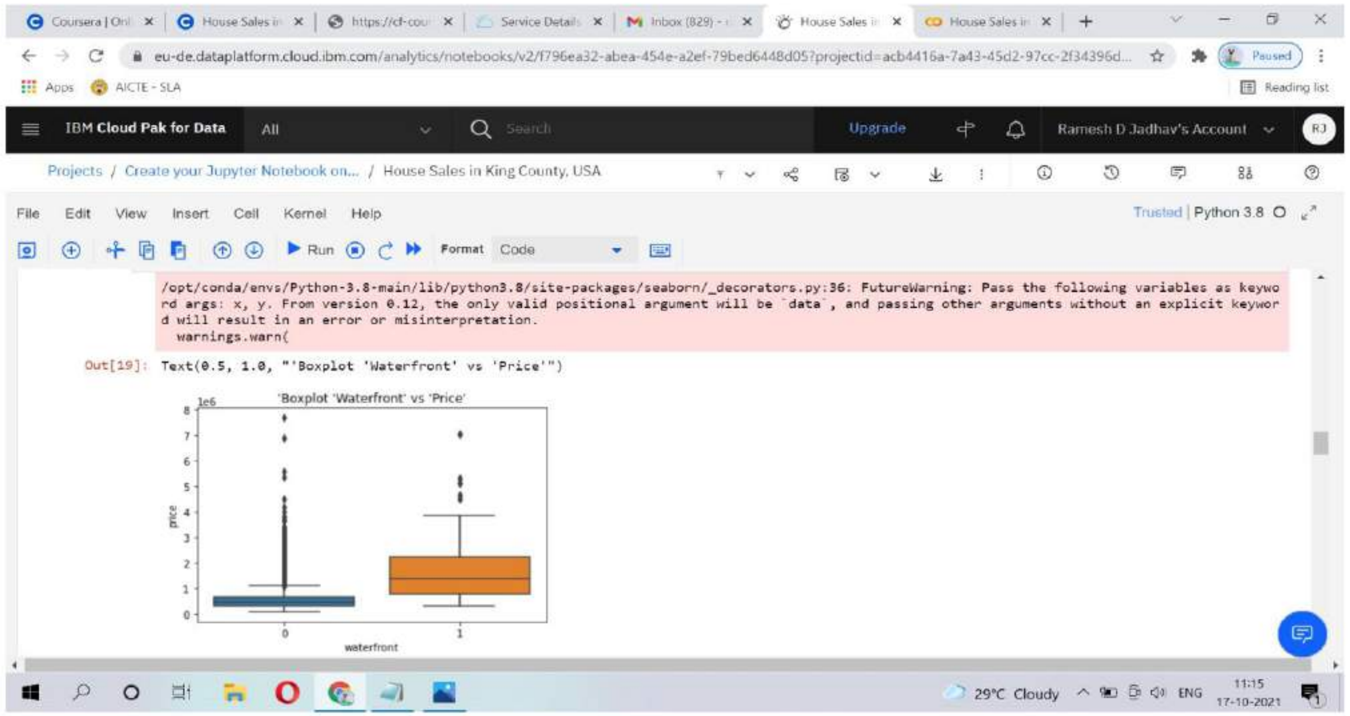
```
In [9]:  #Question 2) Drop the columns "id" and "Unnamed: 0" from axis 1 using the method drop(), then use the method describe() to obtain a statistical

In [10]: df.drop(["id", "Unnamed: 0"], axis = 1, inplace = True)
         df.describe()

Out[10]:
              price      bedrooms     bathrooms   sqft_living      sqft_lot        floors    waterfront          view     condition         grade    sqft_above  sqft_bas
count  2.161300e+04  21600.000000  21603.000000  21613.000000  2.161300e+04  21613.000000  21613.000000  21613.000000  21613.000000  21613.000000  21613.000000   21613.0
 mean  5.400881e+05      3.372870      2.115736   2079.899736  1.510697e+04      1.494309      0.007542      0.234303      3.409430      7.656873   1788.390691     291.5
  std  3.671272e+05      0.926657      0.768996    918.440897  4.142051e+04      0.539989      0.086517      0.766318      0.650743      1.175459    828.090978     442.5
  min  7.500000e+04      1.000000      0.500000    290.000000  5.200000e+02      1.000000      0.000000      0.000000      1.000000      1.000000    290.000000       0.0
  25%  3.219500e+05      3.000000      1.750000   1427.000000  5.040000e+03      1.000000      0.000000      0.000000      3.000000      7.000000   1190.000000       0.0
  50%  4.500000e+05      3.000000      2.250000   1910.000000  7.618000e+03      1.500000      0.000000      0.000000      3.000000      7.000000   1560.000000       0.0
  75%  6.450000e+05      4.000000      2.500000   2550.000000  1.068800e+04      2.000000      0.000000      0.000000      4.000000      8.000000   2210.000000     560.0
  max  7.700000e+06     33.000000      8.000000  13540.000000  1.651359e+06      3.500000      1.000000      4.000000      5.000000     13.000000   9410.000000      4820
```

Interactive map with Folium


eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05?projectid=acb4416a-7a43-45d2-97cc-2f34396d...
IBM Cloud Pak for Data
Ramesh D Jadhav's Account
Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
File Edit View Insert Cell Kernel Help
Trusted | Python 3.8
Run
Format Code

```
In [16]: #Module 3: Exploratory Data Analysis
         #Question 3) use the method value_counts to count the number of houses with unique floor values, use the method .to_frame() to convert it to a

In [17]: df['floors'].value_counts().to_frame()
Out[17]:
          floors
    1.0   10680
    2.0    8241
    1.5    1910
    3.0     613
    2.5     161
    3.5       8

In [18]: #Question 4) Use the function boxplot in the seaborn library to determine whether houses with a waterfront view or without a waterfront view h
```

Interactive map with Folium

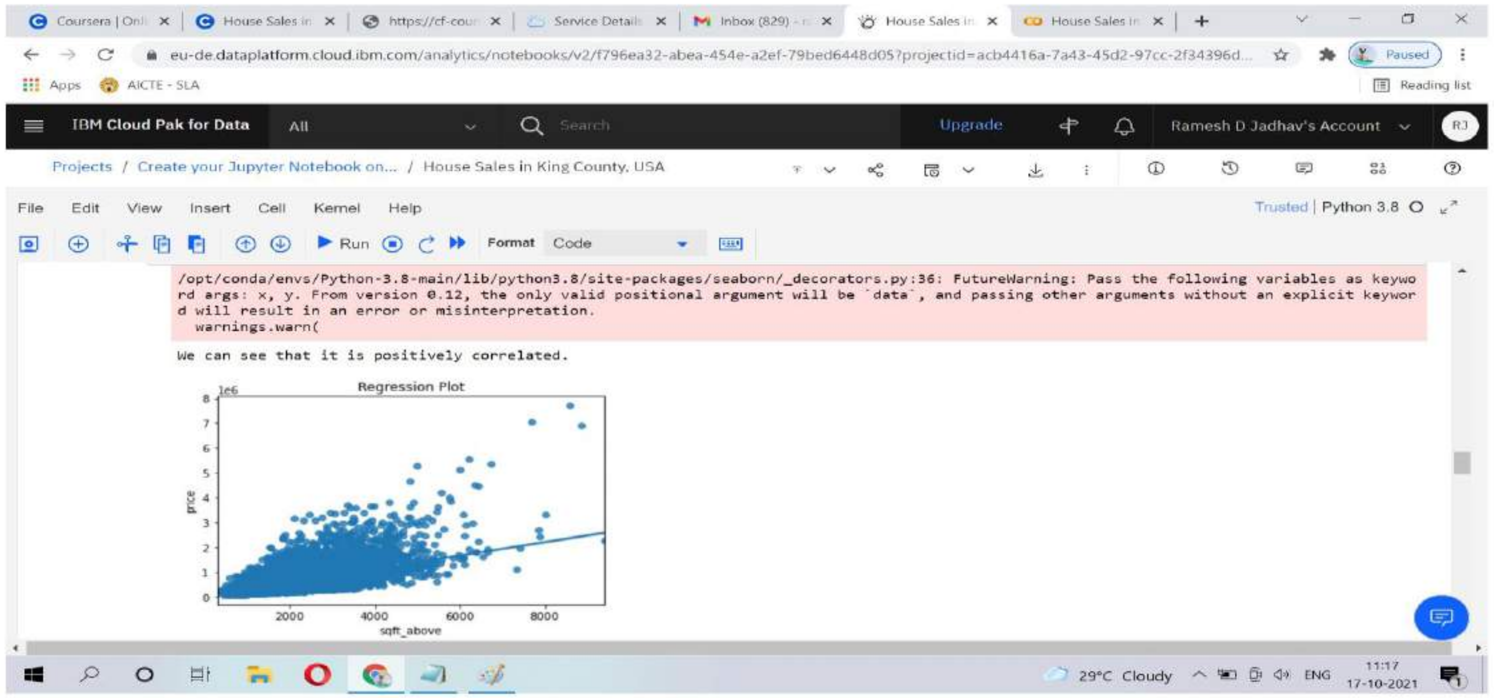


```
/opt/conda/envs/Python-3.8-main/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables as keywo
rd args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  warnings.warn(
```

```
Out[19]: Text(0.5, 1.0, "'Boxplot 'Waterfront' vs 'Price'")
```

Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
'Boxplot 'Waterfront' vs 'Price'
1e6
price
waterfront

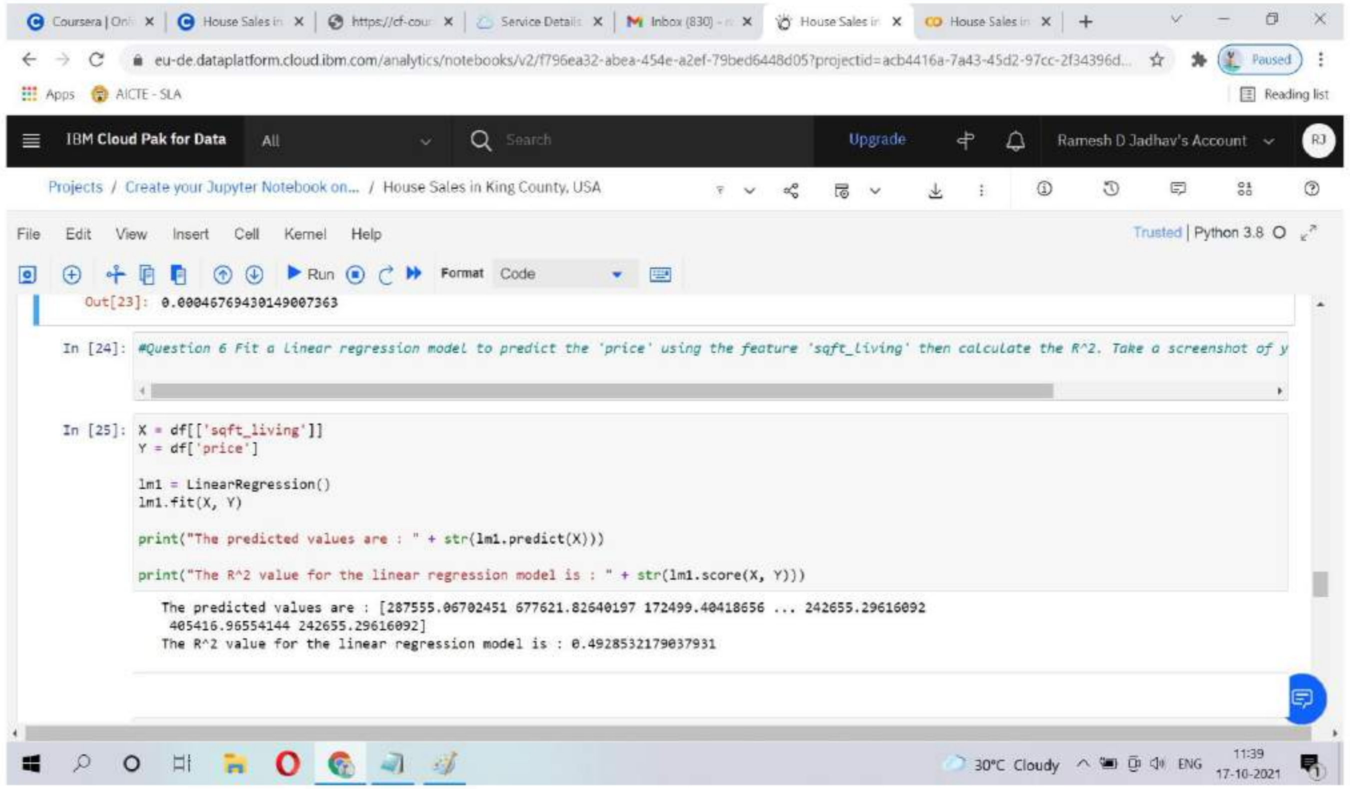
Interactive map with Folium



```
/opt/conda/envs/Python-3.8-main/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables as keywo
rd args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  warnings.warn(
```

We can see that it is positively correlated.

Interactive map with Folium


eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05?projectid=acb4416a-7a43-45d2-97cc-2f34396d...
IBM Cloud Pak for Data
Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
File Edit View Insert Cell Kernel Help
Trusted | Python 3.8
Run
Format Code

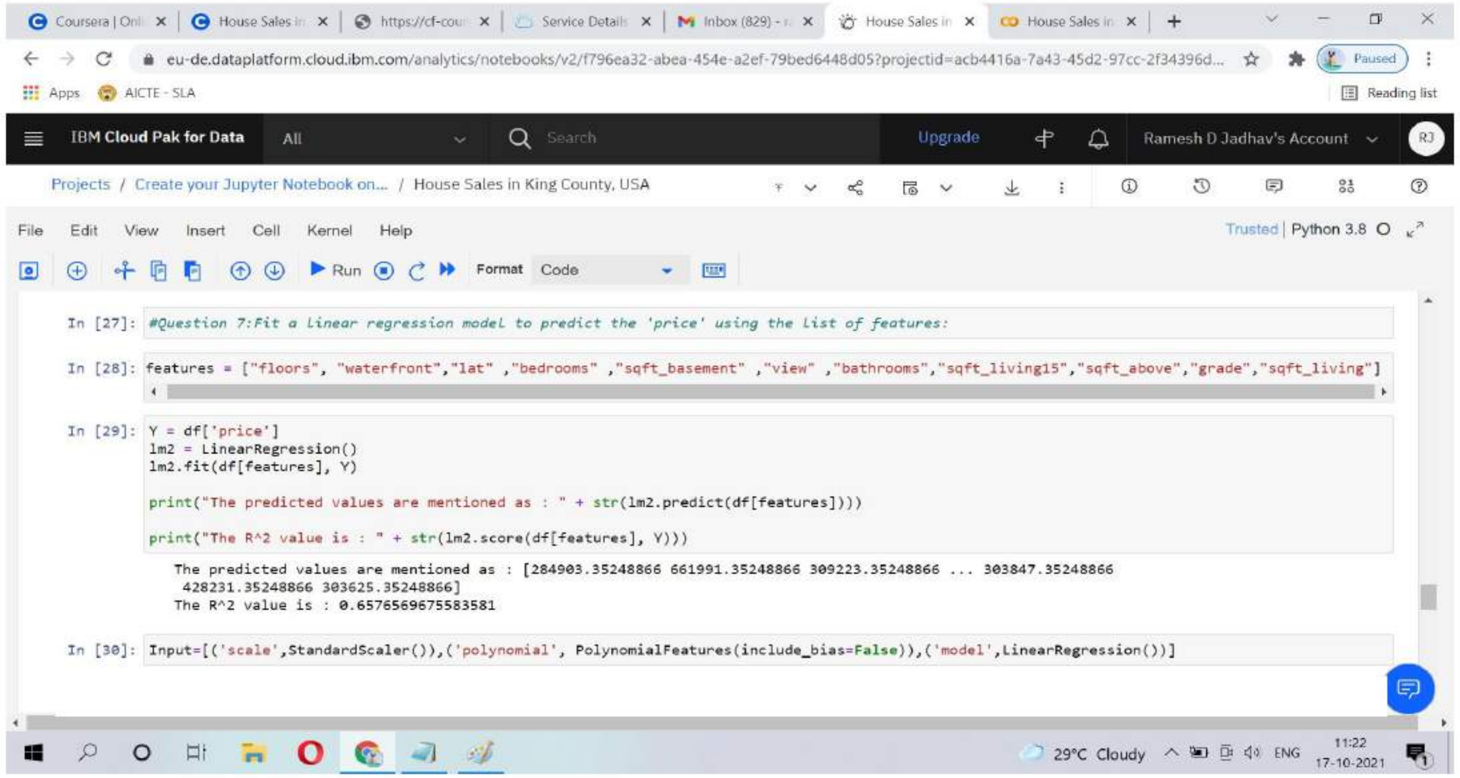
```
Out[23]: 0.00046769430149007363
```

```
In [24]: #Question 6 Fit a linear regression model to predict the 'price' using the feature 'sqft_living' then calculate the R^2. Take a screenshot of y
```

```
In [25]: X = df[['sqft_living']]
         Y = df['price']

         lm1 = LinearRegression()
         lm1.fit(X, Y)

         print("The predicted values are : " + str(lm1.predict(X)))

         print("The R^2 value for the linear regression model is : " + str(lm1.score(X, Y)))
```

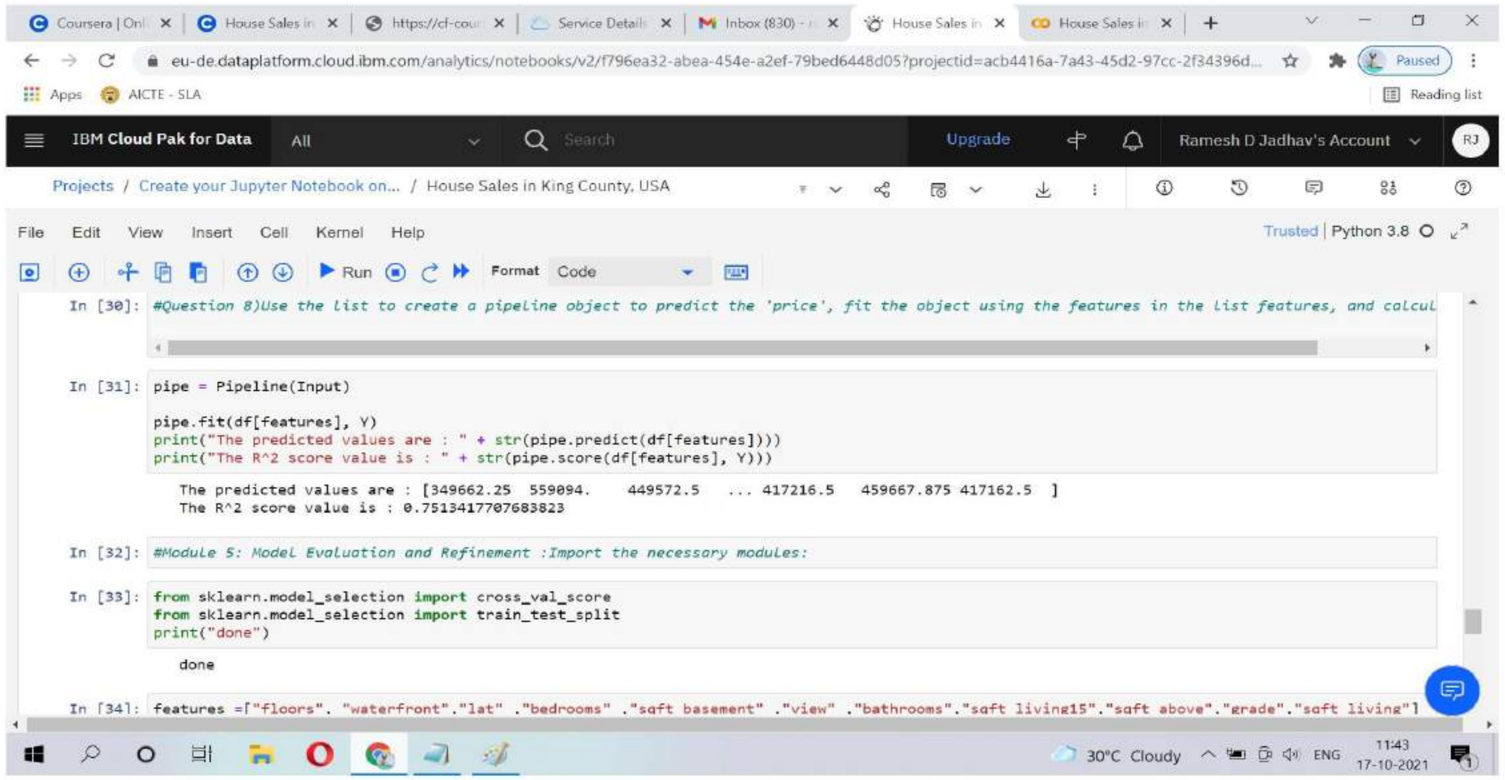
```
The predicted values are : [287555.06702451 677621.82640197 172499.40418656 ... 242655.29616092
 405416.96554144 242655.29616092]
The R^2 value for the linear regression model is : 0.4928532179037931
```

Interactive map with Folium



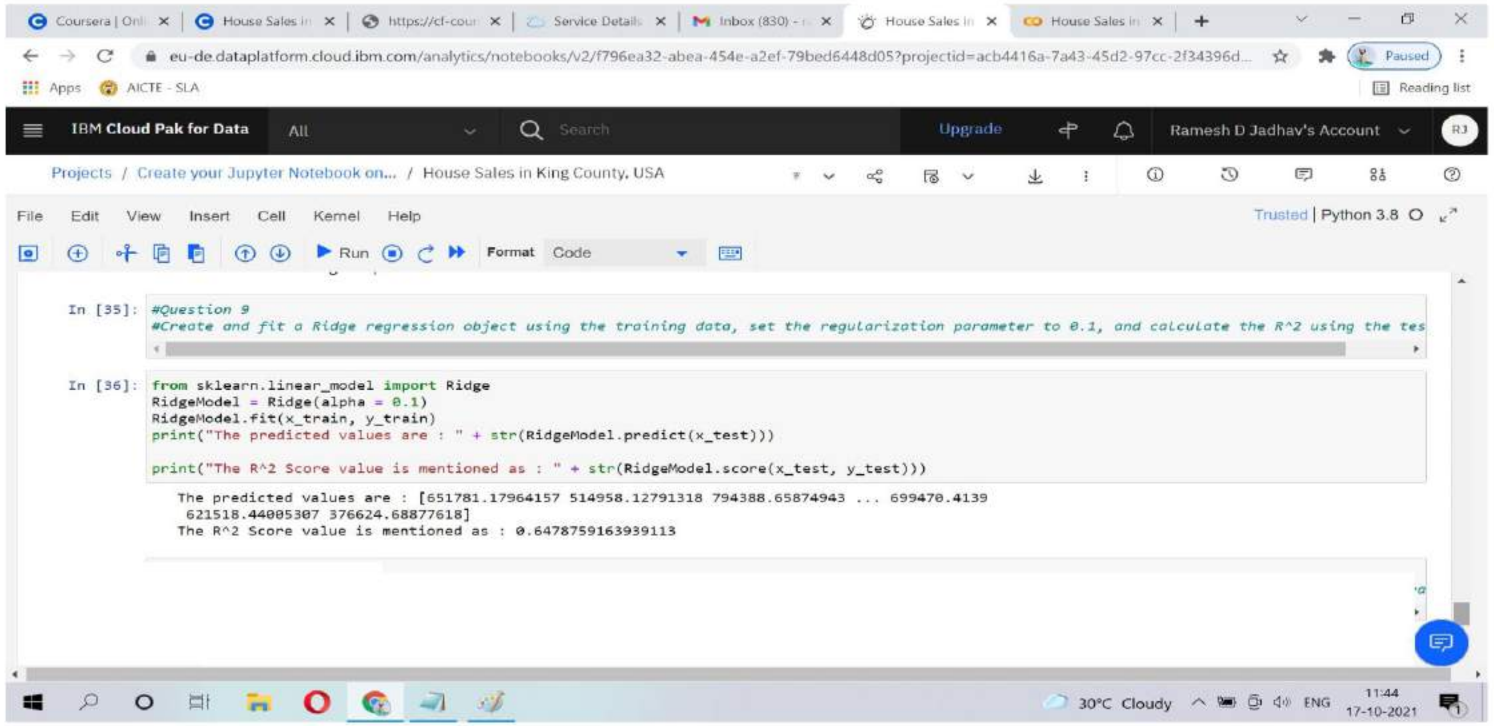
```
In [27]: #Question 7:Fit a linear regression model to predict the 'price' using the list of features:

In [28]: features = ["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view" ,"bathrooms","sqft_living15","sqft_above","grade","sqft_living"]

In [29]: Y = df['price']
         lm2 = LinearRegression()
         lm2.fit(df[features], Y)

         print("The predicted values are mentioned as : " + str(lm2.predict(df[features])))

         print("The R^2 value is : " + str(lm2.score(df[features], Y)))

    The predicted values are mentioned as : [284903.35248866 661991.35248866 309223.35248866 ... 303847.35248866
     428231.35248866 303625.35248866]
    The R^2 value is : 0.6576569675583581

In [30]: Input=[('scale',StandardScaler()),('polynomial', PolynomialFeatures(include_bias=False)),('model',LinearRegression())]
```

Interactive map with Folium


eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05?projectid=acb4416a-7a43-45d2-97cc-2f34396d...
IBM Cloud Pak for Data
Ramesh D Jadhav's Account
Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
File Edit View Insert Cell Kernel Help
Trusted | Python 3.8
Run
Format Code
30°C Cloudy
11:43
17-10-2021

```
In [30]: #Question 8)Use the list to create a pipeline object to predict the 'price', fit the object using the features in the list features, and calcul

In [31]: pipe = Pipeline(Input)

         pipe.fit(df[features], Y)
         print("The predicted values are : " + str(pipe.predict(df[features])))
         print("The R^2 score value is : " + str(pipe.score(df[features], Y)))

            The predicted values are : [349662.25  559094.    449572.5   ... 417216.5   459667.875 417162.5  ]
            The R^2 score value is : 0.7513417707683823

In [32]: #Module 5: Model Evaluation and Refinement :Import the necessary modules:

In [33]: from sklearn.model_selection import cross_val_score
         from sklearn.model_selection import train_test_split
         print("done")

            done

In [34]: features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view" ,"bathrooms","sqft_living15","sqft_above","grade","sqft_living"]
```

Interactive map with Folium

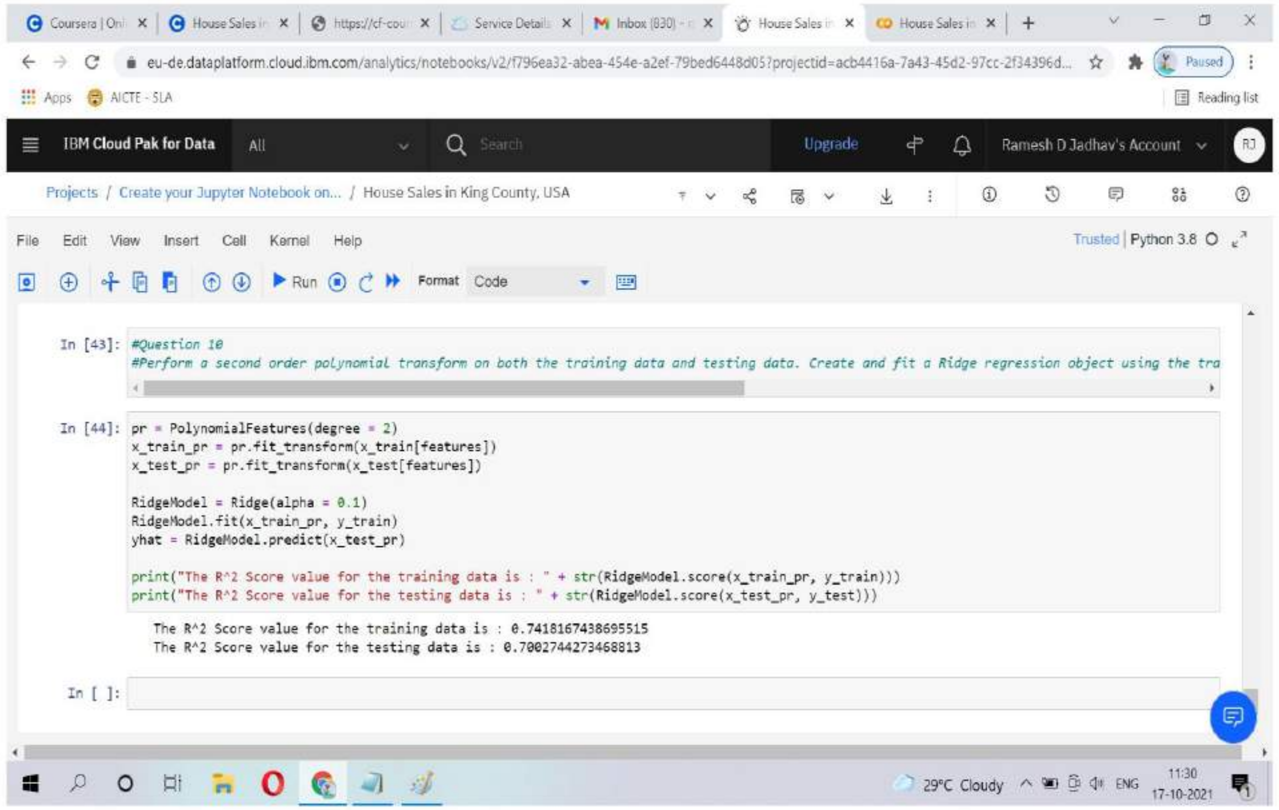


```
In [35]: #Question 9
         #Create and fit a Ridge regression object using the training data, set the regularization parameter to 0.1, and calculate the R^2 using the tes
```

```
In [36]: from sklearn.linear_model import Ridge
         RidgeModel = Ridge(alpha = 0.1)
         RidgeModel.fit(x_train, y_train)
         print("The predicted values are : " + str(RidgeModel.predict(x_test)))

         print("The R^2 Score value is mentioned as : " + str(RidgeModel.score(x_test, y_test)))
```

```
The predicted values are : [651781.17964157 514958.12791318 794388.65874943 ... 699470.4139
 621518.44005307 376624.68877618]
The R^2 Score value is mentioned as : 0.6478759163939113
```

Interactive map with Folium



```
In [43]: #Question 10
         #Perform a second order polynomial transform on both the training data and testing data. Create and fit a Ridge regression object using the tra
```

```
In [44]: pr = PolynomialFeatures(degree = 2)
         x_train_pr = pr.fit_transform(x_train[features])
         x_test_pr = pr.fit_transform(x_test[features])

         RidgeModel = Ridge(alpha = 0.1)
         RidgeModel.fit(x_train_pr, y_train)
         yhat = RidgeModel.predict(x_test_pr)

         print("The R^2 Score value for the training data is : " + str(RidgeModel.score(x_train_pr, y_train)))
         print("The R^2 Score value for the testing data is : " + str(RidgeModel.score(x_test_pr, y_test)))

  The R^2 Score value for the training data is : 0.7418167438695515
  The R^2 Score value for the testing data is : 0.7002744273468813

In [ ]:
```

eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05?projectid=acb4416a-7a43-45d2-97cc-2f34396d...
IBM Cloud Pak for Data
Ramesh D Jadhav's Account
Projects / Create your Jupyter Notebook on... / House Sales in King County, USA
File Edit View Insert Cell Kernel Help
Trusted | Python 3.8
Run Format Code
29°C Cloudy ENG 11:30 17-10-2021

Folium:

Folium is a Python library used for visualizing geospatial data. It is easy to use and yet a powerful library. Folium is a Python wrapper for Leaflet.js which is a leading open-source JavaScript library for plotting interactive maps.
It has the power of Leaflet.js and the simplicity of Python, which makes it an excellent tool for plotting maps. Folium is designed with simplicity, performance, and usability in mind. It works efficiently, can be extended with a lot of plugins, has a beautiful and easy-to-use API.

Installing Folium on our Machine
Folium comes installed on Google Colab and Kaggle kernels. But, if you don’t have Folium installed by default on your system, you can install it using the following command:

e.g. pip install folium
For this article, I am using Folium 0.10.1, which is the latest version of Folium at the time of writing this article. I suggest you use the same version as me so that all your code works properly. You can install this specific version by running the below command:

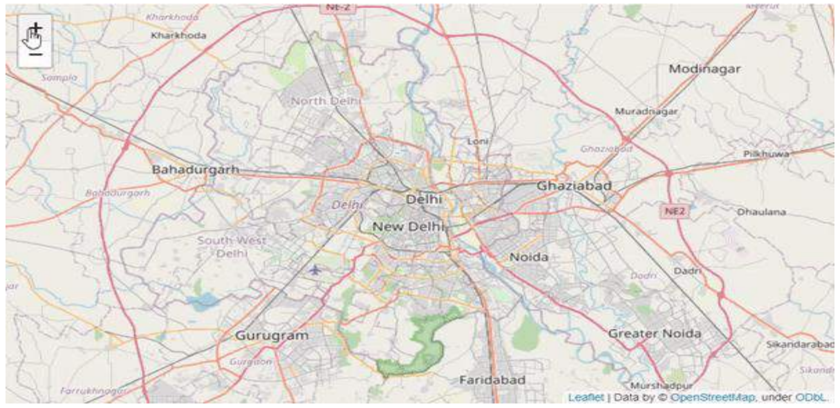
Folium:

e.g. pip install folium == 0.10.1 ----Right, let's start plotting maps!

Plotting Maps with Folium

Plotting maps with Folium is easier than you think. Folium provides the **folium.Map()** class which takes location parameter in terms of latitude and longitude and generates a map around it. So, let's plot a map of Delhi with latitude and longitude as 28.644800 and 77.216721 respectively:

e.g. import folium
m=folium.Map(location=[28.644800, 77.216721])
m



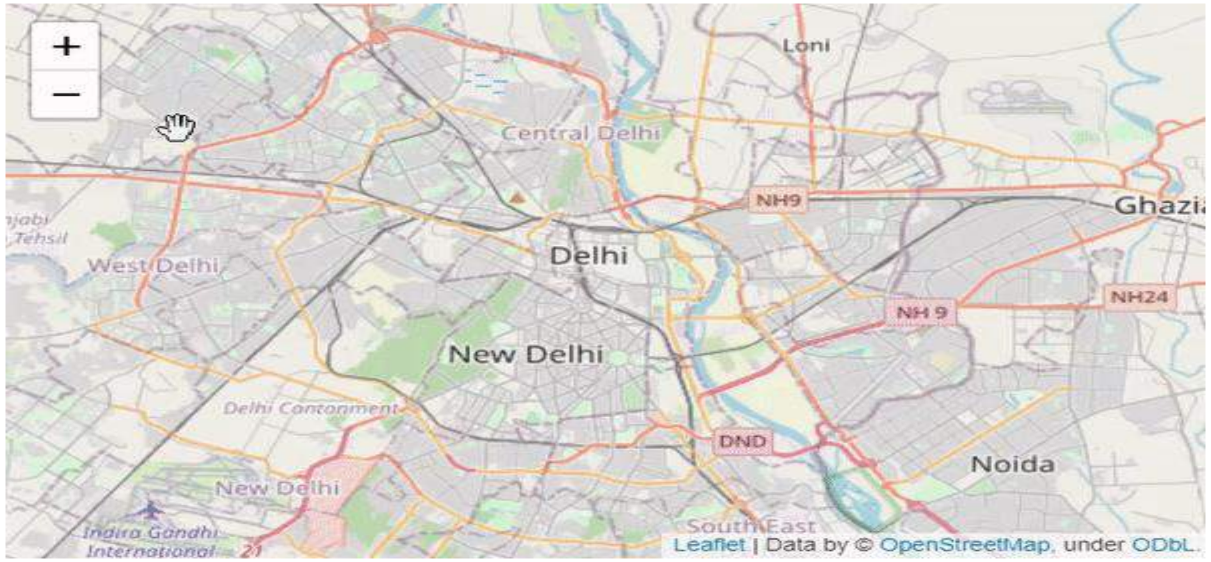
Folium:

- You can see that these maps are interactive. You can zoom in and out by clicking the positive and negative buttons in the top-left corner of the map. You can also drag the map and see different regions.
- Let's try to customize this map now. First, we'll reduce the height and width of the map, and then we'll change the zoom level.
- We can resize our map by using the branca library in Python. It is a spinoff from Folium that hosts the non-map specific features. We can use its **Figure** class for resizing our maps and pass the desired width and height in pixels:
- E.g. from branca.element import Figure
- fig=Figure(width=550,height=350)

- Next, let's move to our second problem and change the zoom level. By default, you can zoom in and out as much as you want. Also, the starting zoom level is fixed to 10 (the map is plotted with a 10x zoom). Now, you might be thinking – how can this be a problem?
- Well, imagine you want to plot a map of a small area. Here, you need to mention the appropriate starting zoom level so that the map focuses only on that region. Also, if you don't want them to zoom in or out much and lose focus on the map, then you can restrict it. For doing this, Folium gives us three parameters – zoom_start, min_zoom, and max_zoom.
- So, let us again plot the map of Delhi but this time of fixed size and with a restricted zoom level:

Folium:

- m1=folium.Map(width=550,height=350,location=[28.644800, 77.216721],zoom_start=11,min_zoom=8,max_zoom=14)
- fig.add_child(m1)
- M1



Plotly Dash dashboard

- As y’all might have come across authentication in Dash that is implemented using the dash-auth library, and are aware that this method is a popup login method. But what if I want a login page as we have in HTML, JavaScript, PHP, etc? Here’s the answer to your question.
- So, I created a dash app and added the user authentication using dash-auth. I wanted to perform testing operations for the same. But as it was a popup I wasn’t able to get the id, class names or any related information that could be used for testing the login module.
- Also, I tried applying some methods that were given but still nothing. Even after searching over the internet for hours, there was no significant answer to my question. That’s when I knew I had to find a way around to circumvent this problem. I knew about dcc.Link() and thought of using it for making the login page.
- This blog contains the solution on how to create an actual login page that is just like HTML page and where you can jump from one page to another using links.

Importing Libraries:

- First things first, importing necessary libraries. You all must be familiar with these.
- *E.g. import dash*
- *import dash_core_components as dcc*
- *import dash_html_components as html*
- *from dash.dependencies import Output,Input,State*

Conclusion

- I have Completed IBM Data Engineering Professional Certificate as you provided 10 modules as per coursera. I learn so much from this course. I very happy to thanks coursera team and associated with IBM.
- This course very informative and I have also completed some assignment and project.
- https://colab.research.google.com/drive/1f3eaq-sQTAcxGLtOu_O0c6pUyA-zBCwA
- https://coursera-assessments.s3.amazonaws.com/assessments/1636885531447/6d05cd9b-7119-44c2-b2d5-af7dfbe21892/6th.pdf
- https://eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/f796ea32-abea-454e-a2ef-79bed6448d05/view?access_token=c96b9a11bee88b9a51dd2d4c7f70aed86a8463b95e87a0b3b2bfb94de808f20e
- https://colab.research.google.com/drive/1ZbS1YUOkr1-jm7-IUQYKHebYNeRS7vdF#scrollTo=053600XdCy8f
-