NORTHEASTERN UNIVERSITY -INFO-7290- ADVANCE DATA SCIENCE AND ARCHITECTURE - SUMMER -2017

**TEAM -9**

ASSIGNMENT - 2 - DaaS - Zillow DataSet

horizontal line

# 

Instructor- Srikanth Krishnamurthy

TA- Pranjal Jain

Team Members:

Khusboo Parekh

Tapadyuti Maiti

# Problem Statement

The Zillow Dataset contains full list of real estate properties in three counties (Los Angeles, Orange and Ventura, California) data in 2016.Following are main problem statements that we have worked upon it:

1.**Data Ingestion , ED , Wrangling** -

In - Depth Exploratory Data Analysis ,Data Cleansing (for automation), Taking care of the missing values, Moving the Raw Data data to AWS S3, Dockerize the process,Push images to DockerHib and Github

2.**Creating Data Base As a Service** (DBaaS)-

Moving Clean Data to Cloud Base Database (AWS - EC2 Instance - DB - PostgresSQL DB ) ,Script that would automate the uploading of file to AWS EC2 instance, Jupyter notebook featuring all the details of the steps involved.

3.**Rest API to Serve the Data:**

Creating REST API on FLASK Hosted on AWS Cloud, Jupyter notebook featuring all the details of the steps involved.

4.**GeoSpatial Search** -

Create a REST API that given a Lat and Long, should return the 10 closest homes, Jupyter notebook featuring all the details of the steps involved.

DataSet Used - properties.csv and train.csv form the below link: <https://www.kaggle.com/c/zillow-prize-1>

## 

# Solution:

**Part 1 : Data Ingestion , ED , Wrangling**

1.1 In - Depth Exploratory Data Analysis

We would try first check how many predictors are present in the data set.

Few insights of the data:  
Range Index: 2985217 entries, 0 to 2985216  
Data columns (total 58 columns)

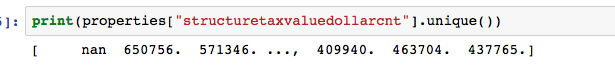
dtypes: float64(52), int64(1), object(5)

|  |  |
| --- | --- |
| parcelid | pooltypeid10 |
| airconditioningtypeid | pooltypeid2 |
| architecturalstyletypeid | pooltypeid7 |
| basementsqft | propertycountylandusecode |
| bathroomcnt | propertylandusetypeid |
| bedroomcnt | propertyzoningdesc |
| buildingclasstypeid | rawcensustractandblock |
| buildingqualitytypeid | regionidcity |
| calculatedbathnbr | regionidcounty |
| decktypeid | regionidneighborhood |
| finishedfloor1squarefeet | regionidzip |
| calculatedfinishedsquarefeet | roomcnt |
| finishedsquarefeet12 | storytypeid |
| finishedsquarefeet13 | threequarterbathnbr |
| finishedsquarefeet15 | typeconstructiontypeid |
| finishedsquarefeet50 | unitcnt |
| finishedsquarefeet6 | yardbuildingsqft17 |
| fips | yardbuildingsqft26 |
| fireplacecnt | yearbuilt |
| fullbathcnt | numberofstories |
| garagecarcnt | fireplaceflag |
| garagetotalsqft | structuretaxvaluedollarcnt |
| hashottuborspa | taxvaluedollarcnt |
| heatingorsystemtypeid | assessmentyear |
| latitude | landtaxvaluedollarcnt |
| longitude | taxamount |
| lotsizesquarefeet | taxdelinquencyflag |
| poolcnt | taxdelinquencyyear |
| poolsizesum | Censustractandblock |

We would try to reduce number of Dimensions. We would handle this situation by reducing number of Columns(Predictors).

Before proceeding further for the Data Wrangling Process , we found that the data from October are not present in the dataset.So , we have decided to remove all the rows after 01-08-2016.

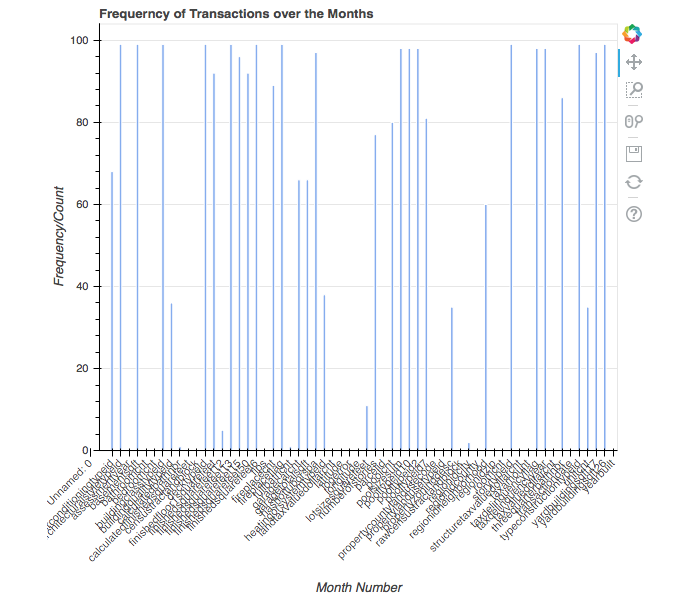
We also saw manually in the predictors that the data does not contain any junk values except Nan ( where the null values are present )

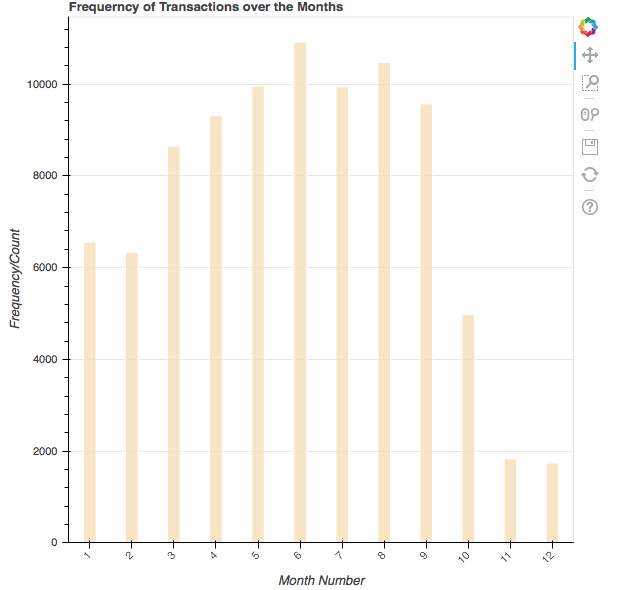


With this step Our Data ingestion step is Finished.

**Data Wrangling**

We have selected **VARIABLE SELECTION** method to reduce the Dimensions.

In this process we are looking at all the variables . As a First Step we would seeing how many Null values are present in each of columns.Following Plot shows in terms of percentage. 

We are interested to plot how the sales of the Homes have been happening in the 2016 year as per the given the data.

We can easily see in the month of 11, 12 there are drastic fall of the sales of home.

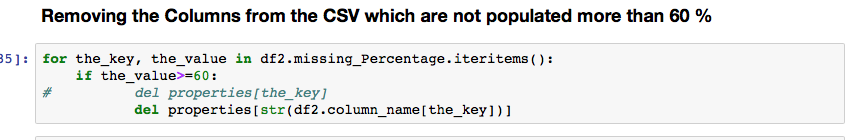
AS the data after the November has the drastic fall , we would try focus on the hottest Sale happening months , also we notice that there a lots of Null values present in the month of Oct- Dec.

So we would remove the data from November.

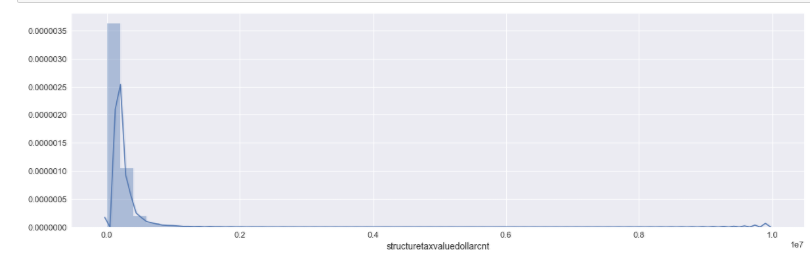


We now also following the **Backward Selection Process** .   
Note : Backward Elimination Process , we have decided to take all the columns and then we are eliminating all the possible Columns.

We have decided to remove the columns whose total Percentage of Null value exceeds 60%, as these columns won’t be of much importance(probably the reason of being empty in the dataset)



So now we Have only 32 Columns to analyse.   
We will review our most important column i.e assessed value of the Parcel



Assessed Value of Parcel is Positively Skewed Data

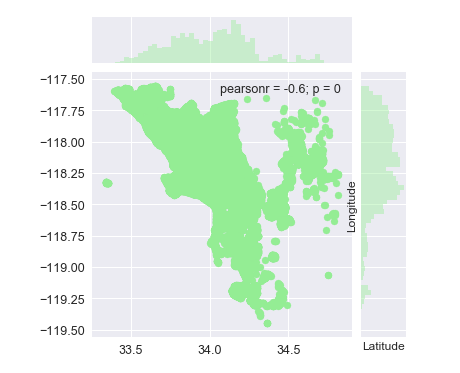
For normally distributed data, the skewness should be about 0. A skewness value > 0 means that there is more weight in the left tail of the distribution.

Krtosis: the sharpness of the peak of a frequency-distribution curve.

Normalizing the data for Latitude and Longitude



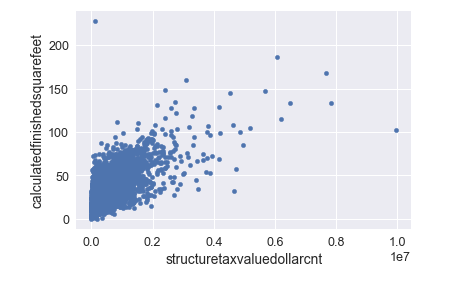
**Visualizing the Houses** on map :



**REGRESSION :**

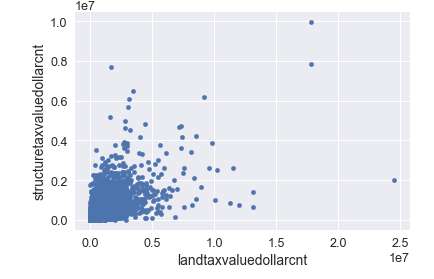
We would Now do few regression on most important factors that we think to decide the market price of the House.

Plotting Assessed Value and Calculated Total Area of the Parcel



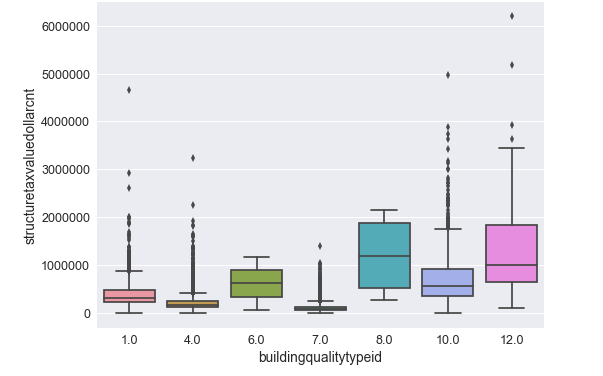
There are outliers in our data and we have roughly direct relationship , although not true for most of the parcels

Plot for Assessed Value and Land Tax Value of all parcels



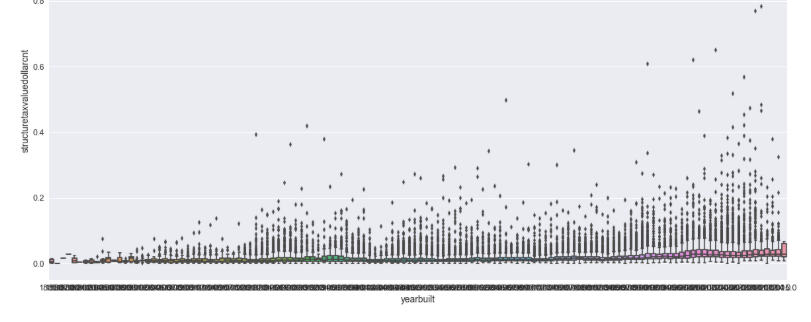
Visualization of Categorical DATA

Plotting Building Type VS Assesed Value of the parcel Rating 1 (Worst) to 10(Supreme)



It is kind of not following the trend as we expect that if the House is Supreme , its price would be also high. For Example Rating with 4 and 7 follow the same price.   
Although we see that Rating with 8 and 12 considerably follow the pattern.

Following Graph would tell us about how the (Sell Price of each houses has been traded in 2016



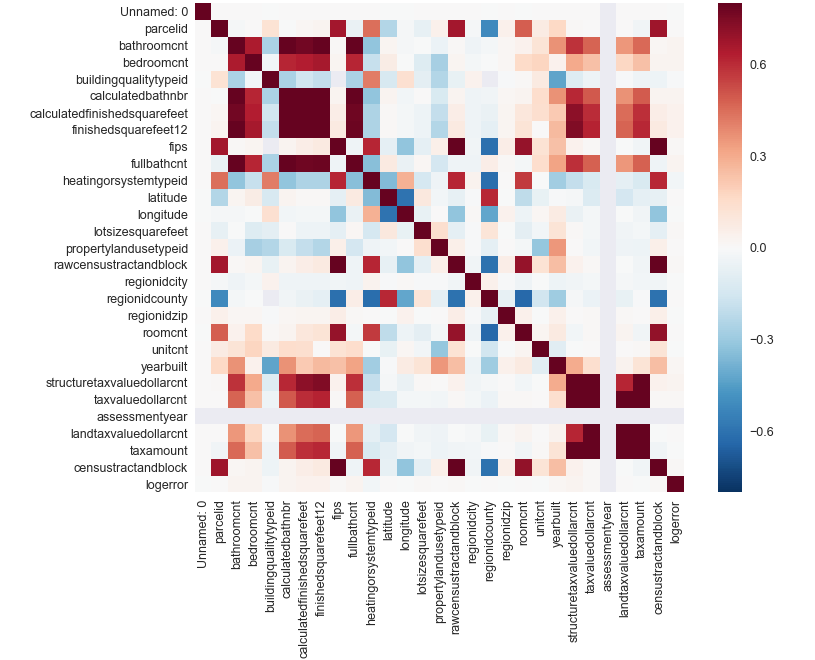
we can clearly see the number of selling high parcel value have been in rising,though half of data for each box plot is on average having same assessed Value

Now we will try to Correlate statistically how the variables are related with each other numerically.

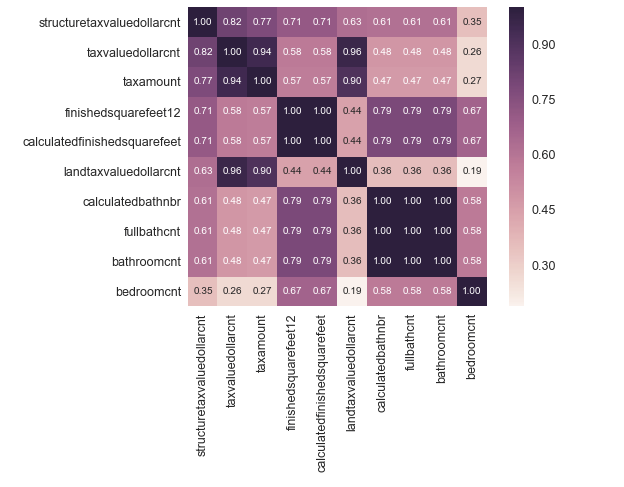
WE would also follow FACTOR ANALYSIS , where in we can easily find the duplicate values in the dataset.

It's clear from the Heat map that we may have some duplicate columns present tin the data as many of the columns shows Deep Red </p>

Let's Take top most Scored from the data Correlativity matrix



We would try to see in more closer way with top most 10 variable under consideration (Whose correlation with Structure Tax Value Dolar Count are high)

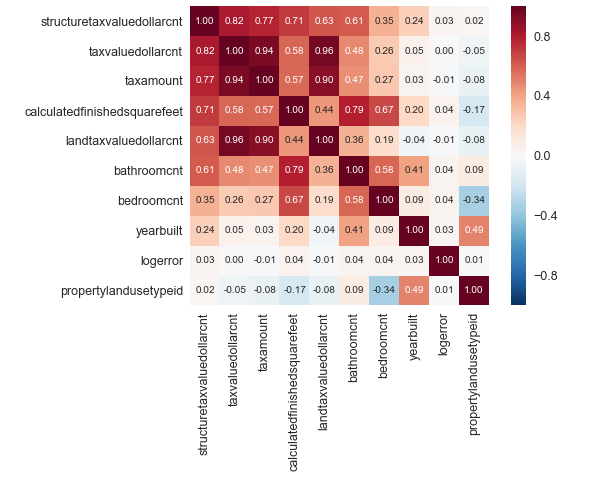


We can easily find that the below Columns are same in the dataset

1. finishedsquarefeet12 and calculatedfinishedsquarefeet

2.fullbathcnt, calculatedbathnbr and bathroomcnt

So we would remove those columns and make the Correlativity matrix again



CONCLUSION :

There are some high correlation between :

Positive Corealtion :

Assessed Value and Tax Value Dollar Count.

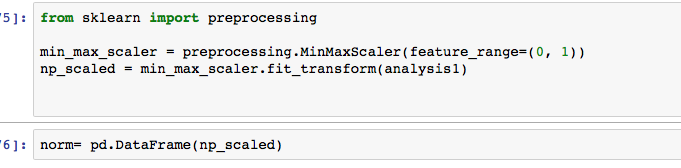
Tax-value-dollar-count- and tax-amount and land -tax-dollar-value-count

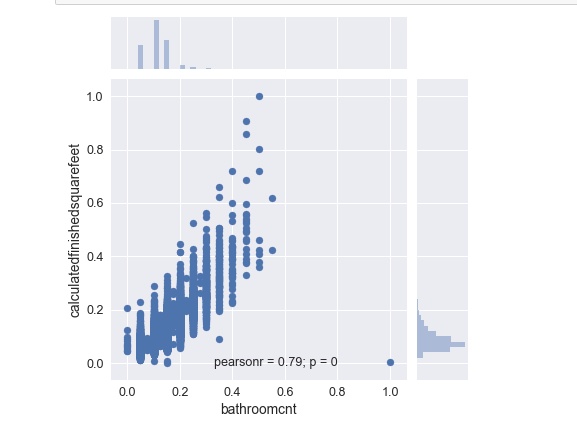
Negative Correlation:

Bedroom count and property-land-use-ID :   
That’s true as the bathroom count increases, automatically the House’ Value and Use type would be upgraded. Property - Use-type-id is rated Worst to Best .

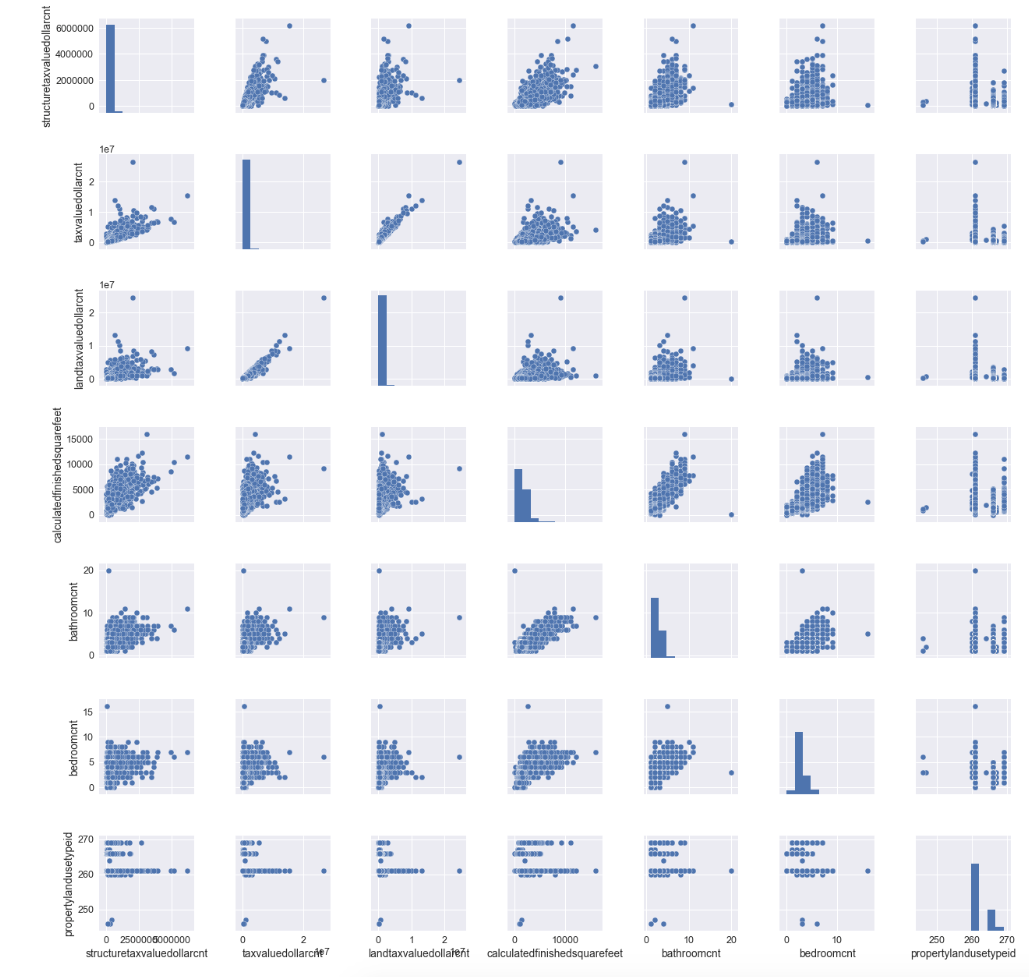
Analysing Relationship between Total Area VS No of Bathroom present in the Parcel

Rescaling the data before plotting





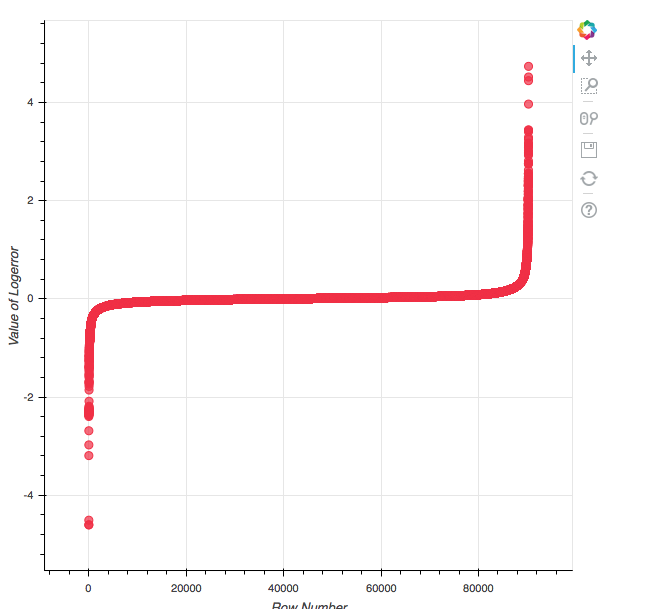
We would visualize all the important parameter that came up in the above matrix :



1.2 Data Cleansing

LOG ERROR present in the dataset for all the parcel.

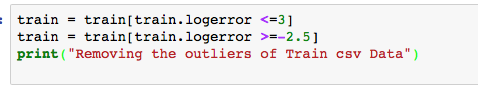
We are also seeing the value of the Log Error , how it varies.



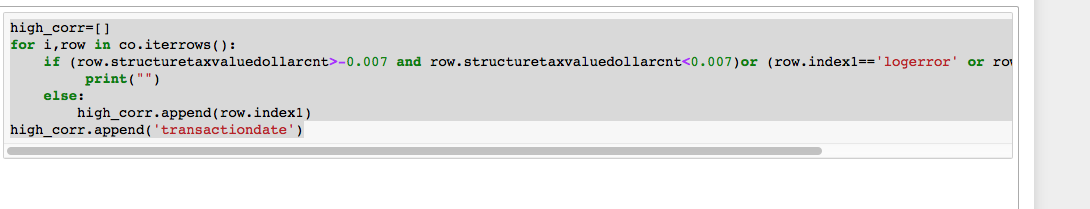
So we would try to remove the Outliers form the above graph on the below criteria:

1.The Values which are less than 3

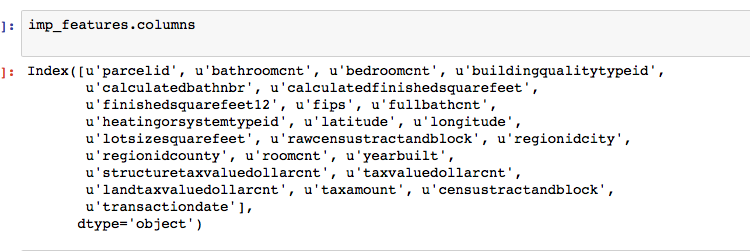
2. The Value which are more than -2.5



Removing features with low variance with respect to structuretaxvaluedollarcnt :



So these are below final predictors that we going to focus to analyse.   
We have columns which are highly correlated statistically with the assessed value of each house



By studying the values manually we found out that few columns have same value through the data , so its better to remove those columns from our analaysis



After analysing we dropping few more columns whose Correlation and the data which are insignificant to our analysis

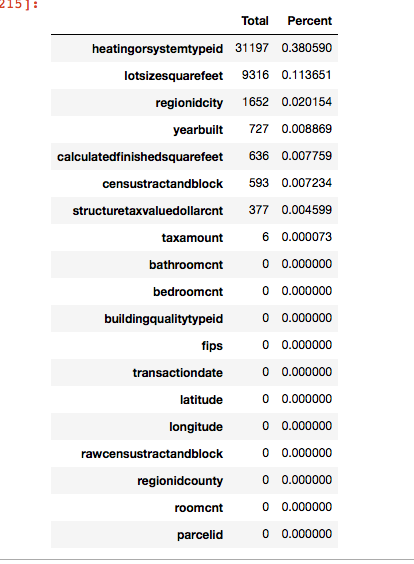


1.3 Taking care of the missing values

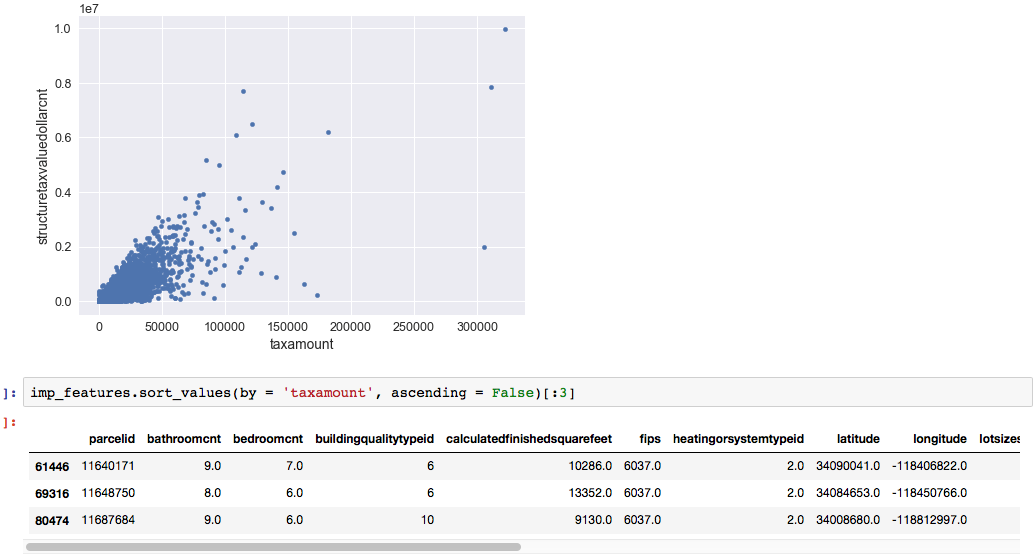
We are filling the null values present in the buildingqualitytypeid with the average ID avlaible i.e 6 Range is from 1(best)- 12(worst)



We are removing the rest of the Rows which are having Null values as the there is very small percentage of Null values present for the rest of the Fields



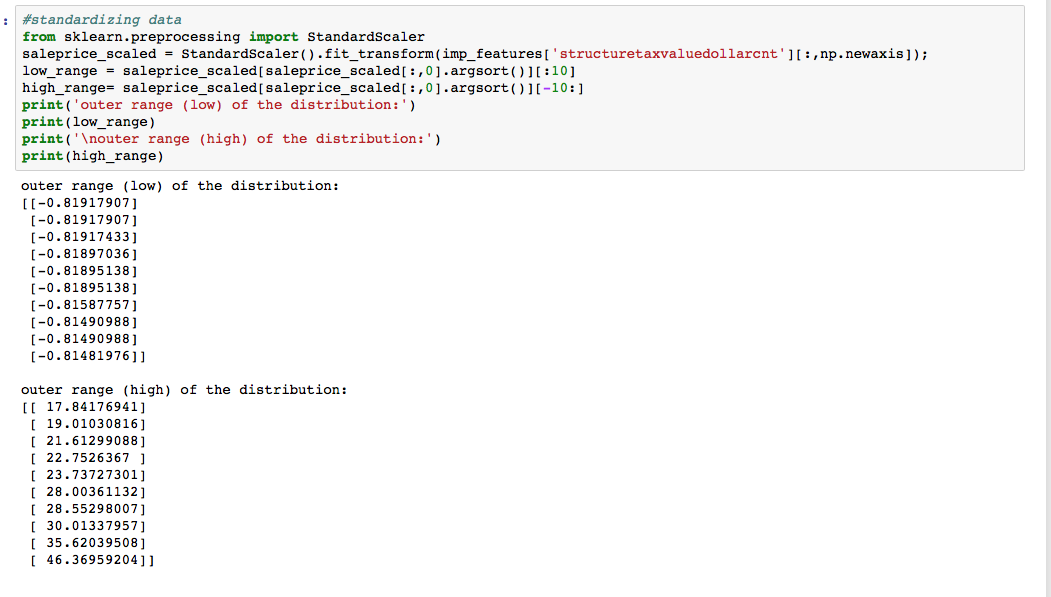
**Removing the Outliers Bivariate analysis**

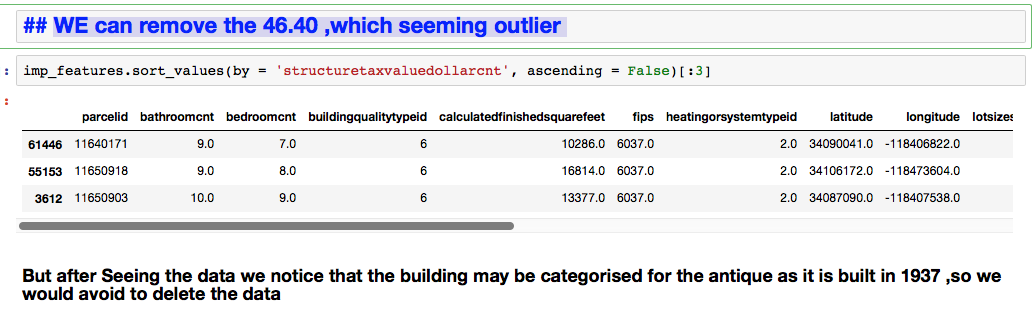


So we are removing the outlier 11648750 -Parcelid, as it shows to high amount of tax for year built - 2011.



**Univariate Analysis on structuretaxvaluedollarcnt column**

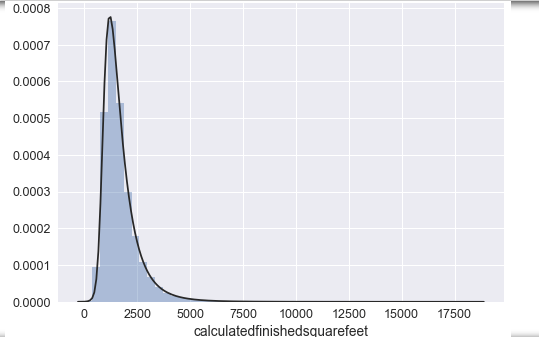




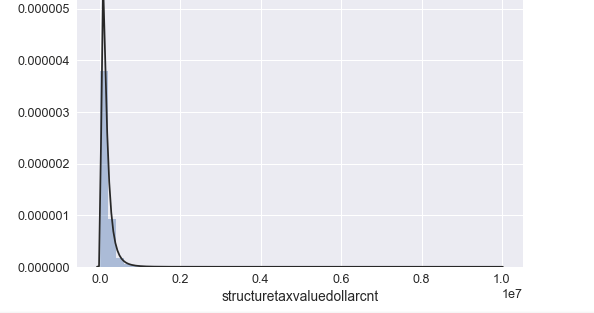
**NORMALIZING DATA:**

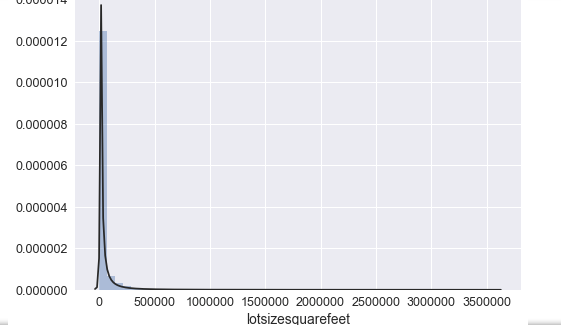
We have chosen Johnson distribution for better accuracy with same alignment as of the original data

We apply to the quantitative columns available





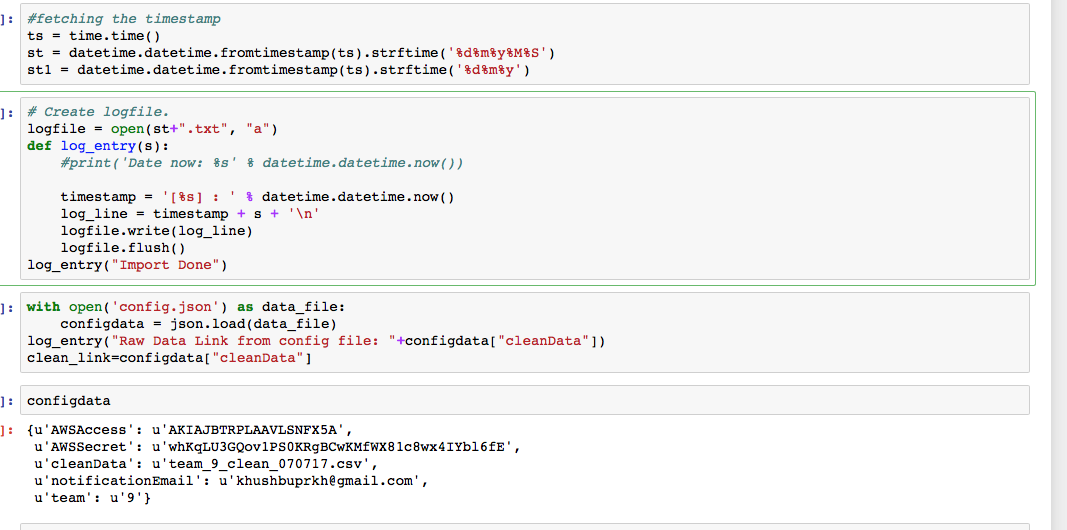




1.4 Moving the Raw Data data to AWS S3

WE will first import all the required packages and create the log file to create the report LOG.

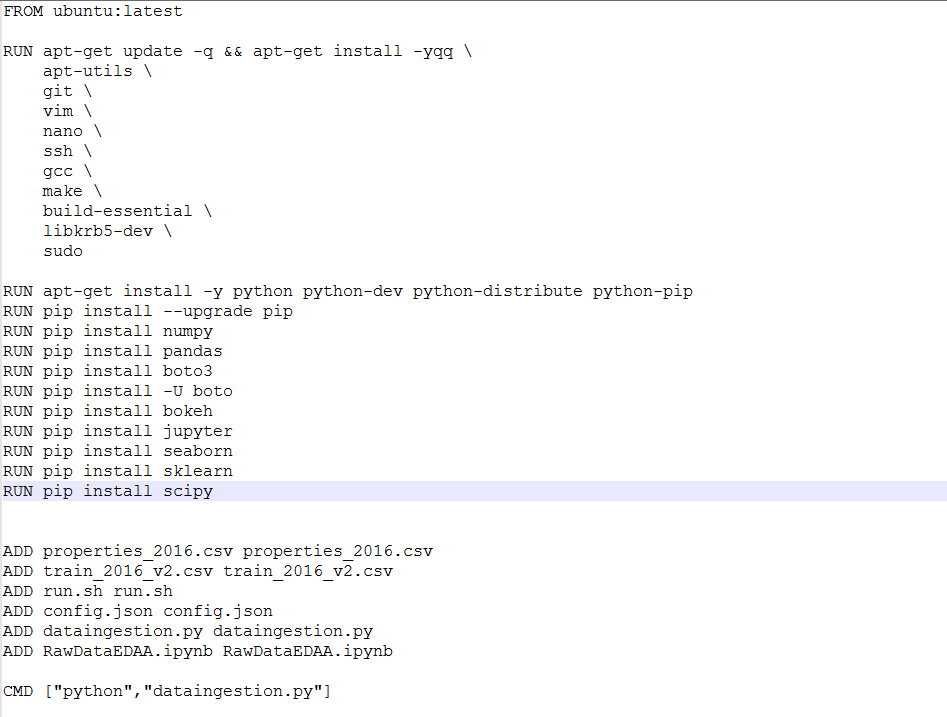
Below Codes Snippet explains this process:



We will then check in the AWS Bucket S3 , whether the data is present or not, if it is not present , then we would download from the S3.   
Below Code Snippet explains :



1.5 Dockerize the process



1.6 Pushing the docker images to DockerHub and Github

**Steps to execute a Docker File:**

1. Command to build the docker image

*docker build -t path1 .*

1. Command to run the docker image

*docker run -ti path1*

1. Command to enter into the bash of a container

*docker exec -it containerid bash*

1. Command to create a directory to include all the files required while executing docker image

*docker mkdir assignment1*

1. Command to exit bash

*exit*

1. Command to commit changes

*docker commit containerid khushbuprkh/path1*

1. Command to login to docker hub

*docker login*

1. Command to push to docker image form local to docker hub

*docker login*

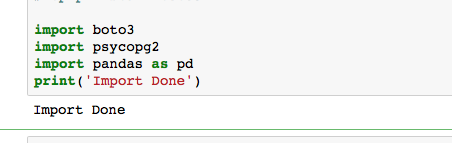
**Part 2.Creating Data Base As a Service** (DBaaS)-

2.1 Moving Clean Data to Cloud Base Database (AWS - EC2 Instance - DB - PostgresSQL DB )

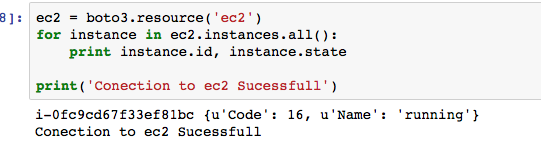
Note : Please Make sure to start the Postgres SQL DB Server , before connection of DBaaS

Import of Packages

**1.Connecting to Ec2 instance**



**2.Defining the Public DNS of ec2 instance**



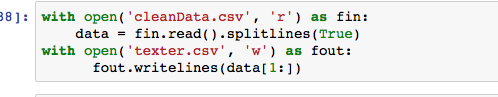
**3.Connecting PostgresSQL Host in EC2**



**4.Creating Table in Postgres**



**5.Removing the first column of the clean csv file , so that we can easily import the data into the Table**



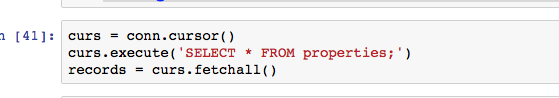
**6.Opening the fitered data**



**7.Exporting the the local datafile to postgressql residing in the EC2 instance**



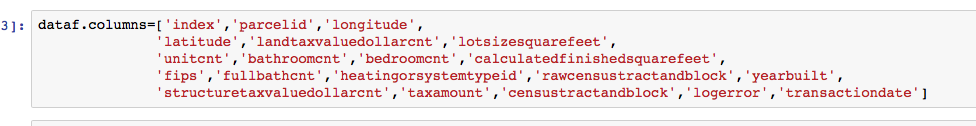
**8.Getting back the data formthe PostSQL DB server to local Dataframe**



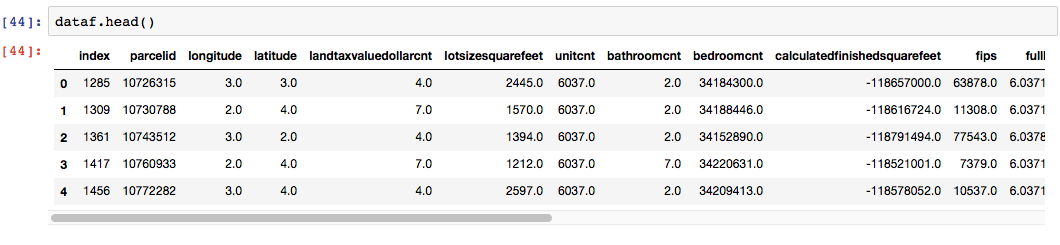
**9.Converting to Dataframe to further analysis as required**



**10.Renaming the columns as per fetched , which will help ahead for querying purpose**



**11.Sample Data fetchefrom the EC2 instance (Postgres SQL DB server )**



2.2 Script that would automate the uploading of file to AWS EC2 instance,

The Whole Python Script would be run from the Shell Script.

Python Part2.py

2.3 Jupyter notebook featuring all the details of the steps involved.

As Attached in the Assignment Folder , you can find the Steps in Jupyter notebook.

**Part 3.Rest API to Serve the Data:**

3.1 Creating REST API on FLASK Hosted on AWS Cloud,

3.2 Jupyter notebook featuring all the details of the steps involved.

As Attached in the Assignment Folder , you can find the Steps in Jupyter notebook.

**Part 4.GeoSpatial Search** -

4.1 Create a REST API that given a Lat and Long, should return the 10 closest homes,

4.2 Jupyter notebook featuring all the details of the steps involved.

As Attached in the Assignment Folder , you can find the Steps in Jupyter notebook.

**Part 3. REST API**

**Python-Flask Framework**

Flask was created by Armin Ronacher in 2000 as a small, minimalistic and light-weight Python Webapp framework. It is so small to be called a micro-framework. Flask is actually a glue that sticks together two popular frameworks:

Werkzeug: a WSGI (Web Server Gateway Interface) library for Python, which includes a URL routing system, fully featured request and response objects and a powerful debugger. (WSGI is a specification for simple and universal interface between web servers and Python web applications.)

Jinja2: a full-feature template engine for Python.

High-level tasks like database access, web form and user authentication are supported through "extensions".

Within the scope of MVC (Model-View-Controller) architecture, Werkzeug covers the Controller (C) and Jinja2 covers the View (V). Flask does not provide an integrated Model (M) layer, and lets you pick your database solution. A popular choice is Flask-SQLAlchemy with a ORM (Object-Relational Mapper) over a relational database such as MySQL or PostgreSQL.

In summary, the Flask framework provides:

1. a WSGI compliance interface.
2. URL routing and Request dispatching.
3. Secure cookies and Sessions.
4. a built-in Development Web Server and Debugger.
5. Unit test client for unit testing that facilitates write-test-first.
6. Jinja2 templates (tags, filters, macros, etc).
7. Via Flask, you can handle HTTP and AJAX requests, user sessions between requests, route requests to controllers, evaluate and validate the request data, and response with HTML or JSON, and so on.

**Getting Started with flask**

**cd /path/to/project-directory** # Choose your project directory

$ **virtualenv -p python3 venv** # For Python 3, or

$ **virtualenv venv** # For Python 2

$ **source venv/bin/activate** # Activate the virual environment

(venv)$ **pip install flask** # Install flask using 'pip' which is symlinked to pip2 or pip3 (no sudo needed)

......

Successfully installed Jinja2-2.9.5 MarkupSafe-0.23 Werkzeug-0.11.15 click-6.7 flask-0.12 itsdangerous-0.24

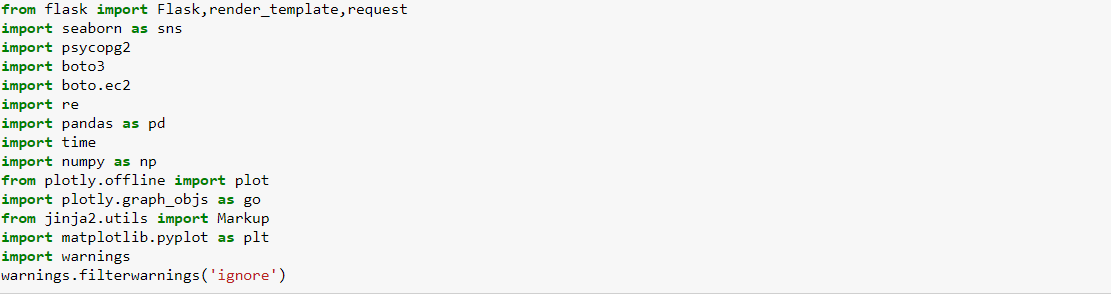
(venv)$ **pip show flask** # Check installed packages

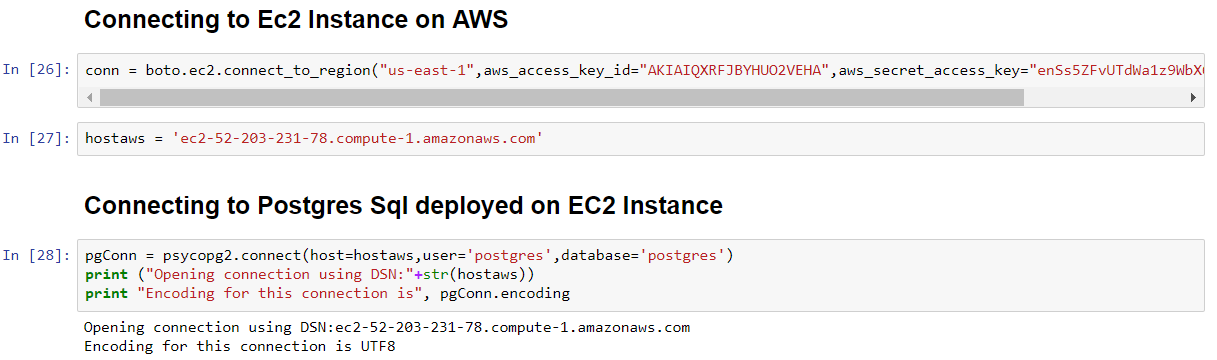
Name: Flask

Version: **0.12**

We have created an app folder on our virtual environment which has a templates folder and app.py file. The templates folder contains all the html (static) file. But we have made it dynamic by using jinja2 to populate data received form the database Postgres deployed on EC2.

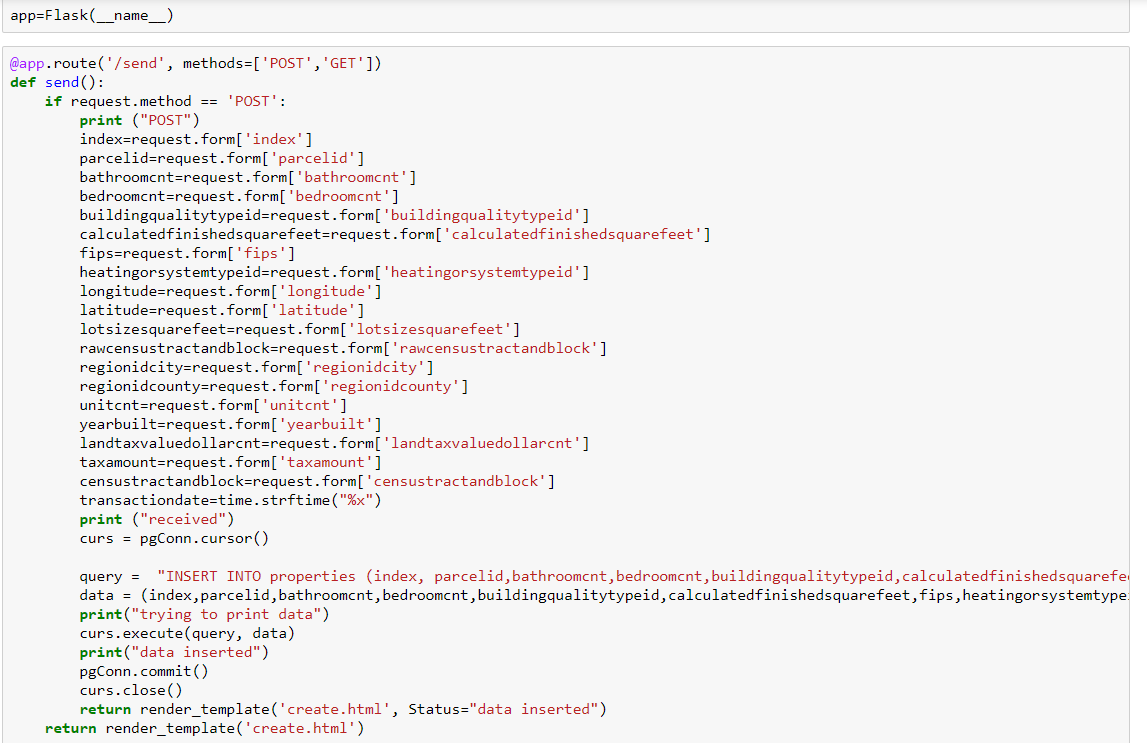
Import statements of our app.py file





We have created pages to perform CRUD operations

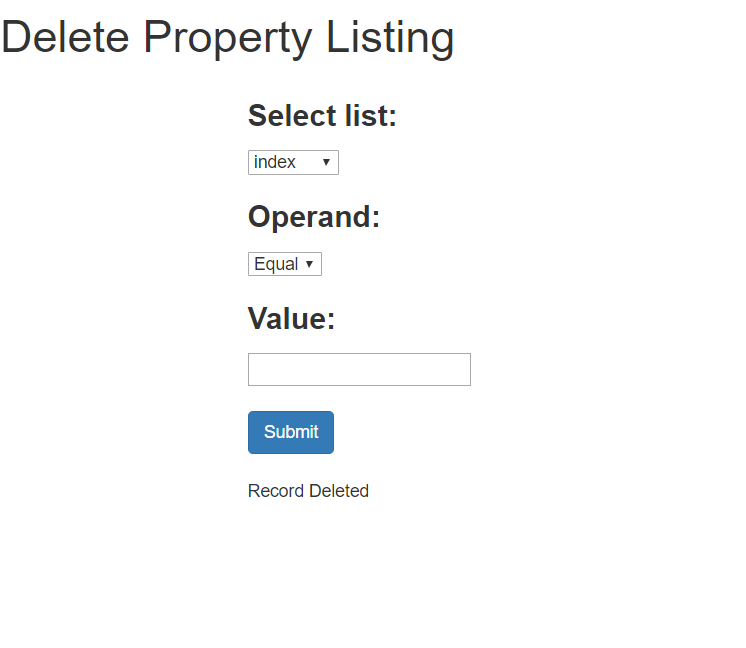
1. Insert Data





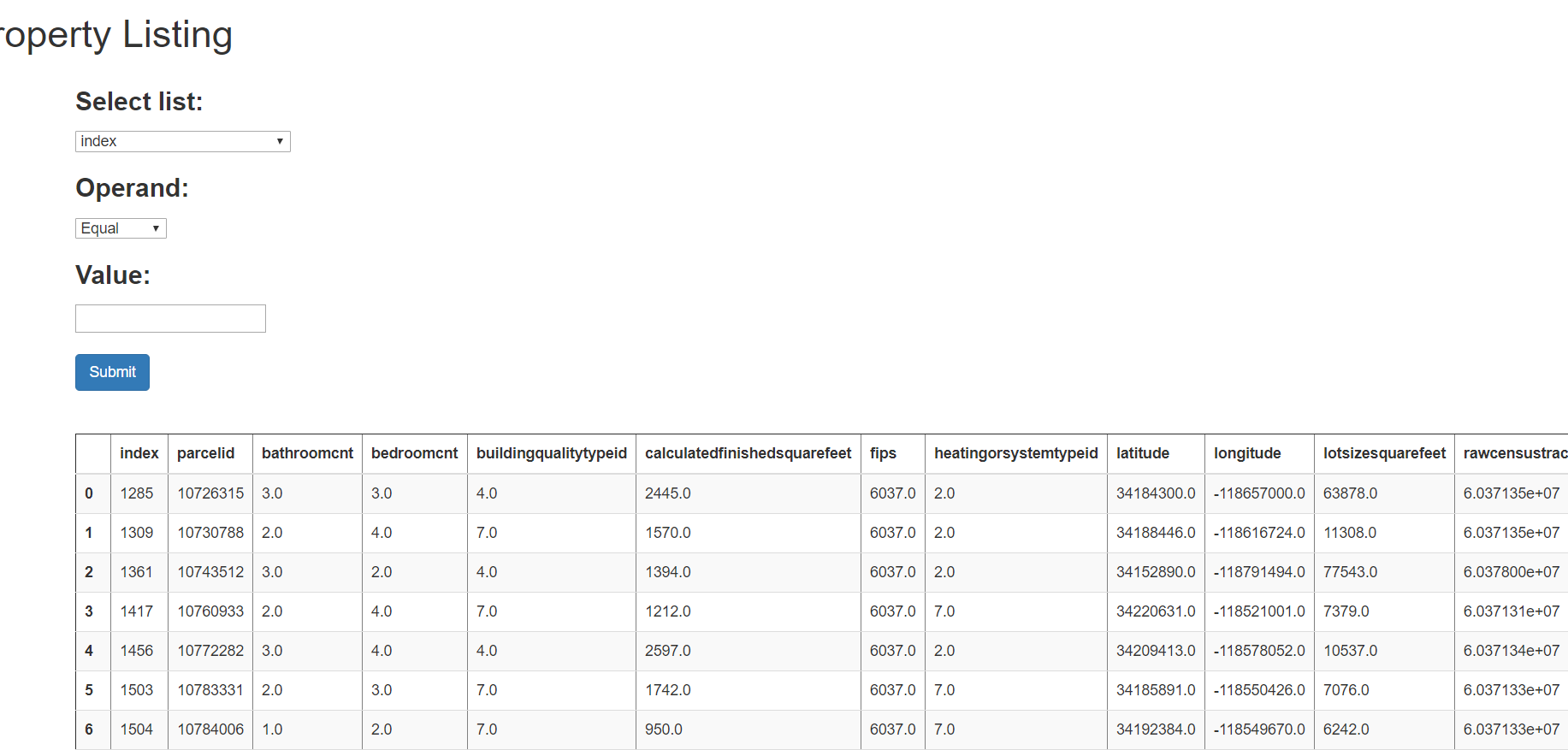
1. Delete Data





1. Fetch Data





**Part 4 : Enhancing your REST API: Geospatial search**

We have created a Rest API where User can pass the input and get the nearest Location of it.

We have used the geopy Library to calculate the distance of each location of house measured with the user input.

Below is the screen shot of Code snippet

Our Run time complexity O(n) where n is Number of Homes.



