HOUSE PRICE PREDICTION USING MACHINE LEARNING IN PYTHON

PHASE-1 DOCUMENT SUBMISSION

PROJECT: PROBLEM DEFINITION AT DESIGN THINKING

TEAM MEMBERS

110521106018 G.RAMYA

110521106110 R.KAVITHA

Autec22109 S.EZHILARASAN

Autec22129 S.SRIRAM

Title: House Price Prediction Using Python

Abstract: Predicting house prices is a crucial task in the real estate industry and has significant applications in financial planning and investment. In this project, we present a comprehensive guide to house price prediction using Python, including code examples and a high-level overview of the process. We employ a machine learning approach to build a predictive model that can estimate house prices accurately.

Introduction: House price prediction involves analyzing various factors such as location, size, number of bedrooms, and more to determine a property's market value. Machine learning algorithms have proven to be effective tools for this task. We will demonstrate how to implement a house price prediction model using Python, focusing on key libraries such as NumPy, pandas, scikit-learn, and Matplotlib.

Methods:

- **Data Preparation:**
- **Acquire and preprocess the dataset: We will use a real-world dataset containing information about houses, including features like square footage, number of bedrooms, and location.**

- **Data Exploration:**
- **Visualize the data: We will use Matplotlib and Seaborn to create informative plots that help us understand the data distribution and correlations among features.**
- **Feature Engineering:**
- **Select relevant features: Determine which features are most significant for predicting house prices.**
- **Handle missing data: Use techniques such as imputation or removal to address missing values.**
- **Model Building:**
- **Select a machine learning algorithm: We will use a regression model, such as linear regression, decision tree regression, or a more advanced algorithm like Random Forest or Gradient Boosting.**
- **Model Training:**
- **Split the data into training and testing sets: We will use techniques like k-fold cross-validation to evaluate our model's performance effectively.**
- **Train the model on the training data using scikit-learn.**
- **Model Evaluation:**
- **Assess the model's performance: We will use metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared to measure the accuracy of our predictions.**
- **Visualize the results: Create visualizations to showcase the model's predictions and actual prices.**
- **Hyperparameter Tuning:**
- **Optimize the model: Adjust hyperparameters to improve prediction accuracy.**

[Code Example:]

pythonCopy code

```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv('house_data.csv')

# Data preprocessing and feature engineering here...
# Split the data into training and testing sets
X = data[['feature1', 'feature2', ...]]  # Replace with relevant features
y = data['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Visualize results (e.g., scatter plot of actual vs. predicted prices)
plt.scatter(y_test, y_pred)
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.title('Actual vs. Predicted House Prices')
plt.show()

# Print performance metrics
print(f'Mean Absolute Error: {mae}')
print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

Installation Guide

1. **Clone or Fork the Project**
2. **Create a Virtual Enviroment**
3. **go to same virtual enviroment and write below cmd**
4. **pip install -r requirements.txt**

Project Highlights

5. **Research Paper**
6. **User Friendly**

7. Accuracy
8. Open Source

Table Of Content

1. Project Description

A. Problem Statement

Thousands of houses are sold everyday. There are some questions every buyer asks himself like: What is the actual price that this house deserves? Am I paying a fair price?

B. Best Possible Solutions

a.Housing Expert
b.Intuition About House
c.Using Machine Learning

C. Introduction About Project

House Price prediction are very stressful work as we have to consider different things while buying a house like the structure and the rooms kitchen parking space and gardens. People don’t know about the factor which influence the house price. But by using the Machine learning we can easily find the house which is to be prefect for us and helps to predict the price accurately.

D. Tools and Libraries

Tools

a.Python
b.Jupyter Notebook
c. Flask
d. HTML
e. CSS
f. JS
g. Heroku
h. GitHub

Libraries

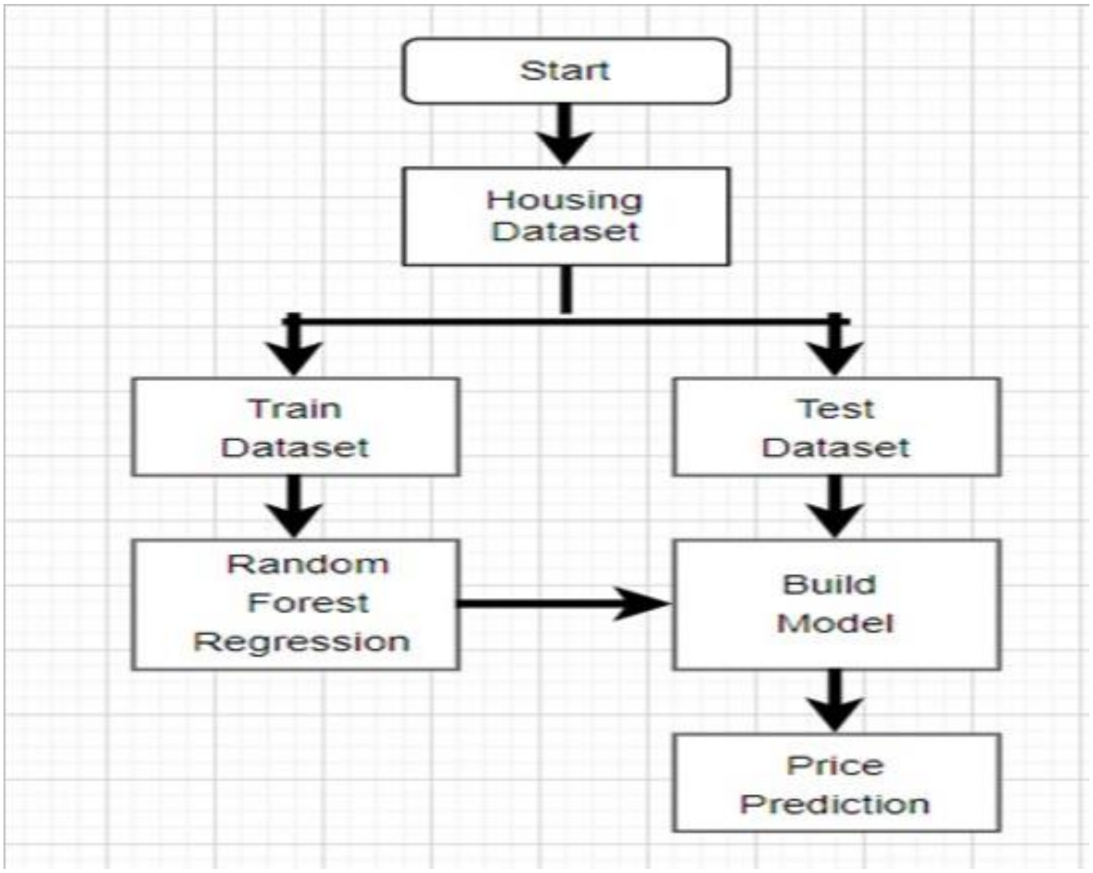
a.Pandas
b.Scikit Learn
c.Numpy
d.Seaborn
e.Matpoltlib

2. Data Collection

For this project we used the data that is available on kaggle.(click here for data).There are 26 columns and 318851 Rows. These are the major point about the data set.
url: the url which fetches the data
id: the id of transaction
Lng: and Lat coordinates, using the BD09 protocol.
Cid: community id
tradeTime: the time of transaction

DOM: active days on market
followers: the number of people follow the transaction.
totalPrice: the total price
price: the average price by square
square: the square of house
livingRoom: the number of living room
drawingRoom: the number of drawing room
kitchen: the number of kitchen
bathroom the number of bathroom
floor: the height of the house. I will turn the Chinese characters to English in the next version.
buildingType: including tower(1) , bungalow(2)， combination of plate and tower(3), plate(4).
constructionTime: the time of construction
renovationCondition: including other(1), rough(2),Simplicity(3), hardcover(4)
buildingStructure: including unknow(1), mixed(2), brick and wood(3), brick and concrete(4),steel(5) and steel-concrete composite (6).
ladderRatio: the proportion between number of residents on the same floor and number of elevator of ladder. It describes how many ladders a resident have on average.
elevator: have (1) or not have elevator(0)
fiveYearsProperty: if the owner have the property for less than 5 years.

3. Generic Flow Of Project



4. EDA

```
Index(['url', 'id', 'Lng', 'Lat', 'Cid', 'tradeTime', 'DOM', 'followers',
       'totalPrice', 'price', 'square', 'livingRoom', 'drawingRoom', 'kitchen',
       'bathRoom', 'floor', 'buildingType', 'constructionTime',
       'renovationCondition', 'buildingStructure', 'ladderRatio', 'elevator',
       'fiveYearsProperty', 'subway', 'district', 'communityAverage'],
      dtype='object')
```

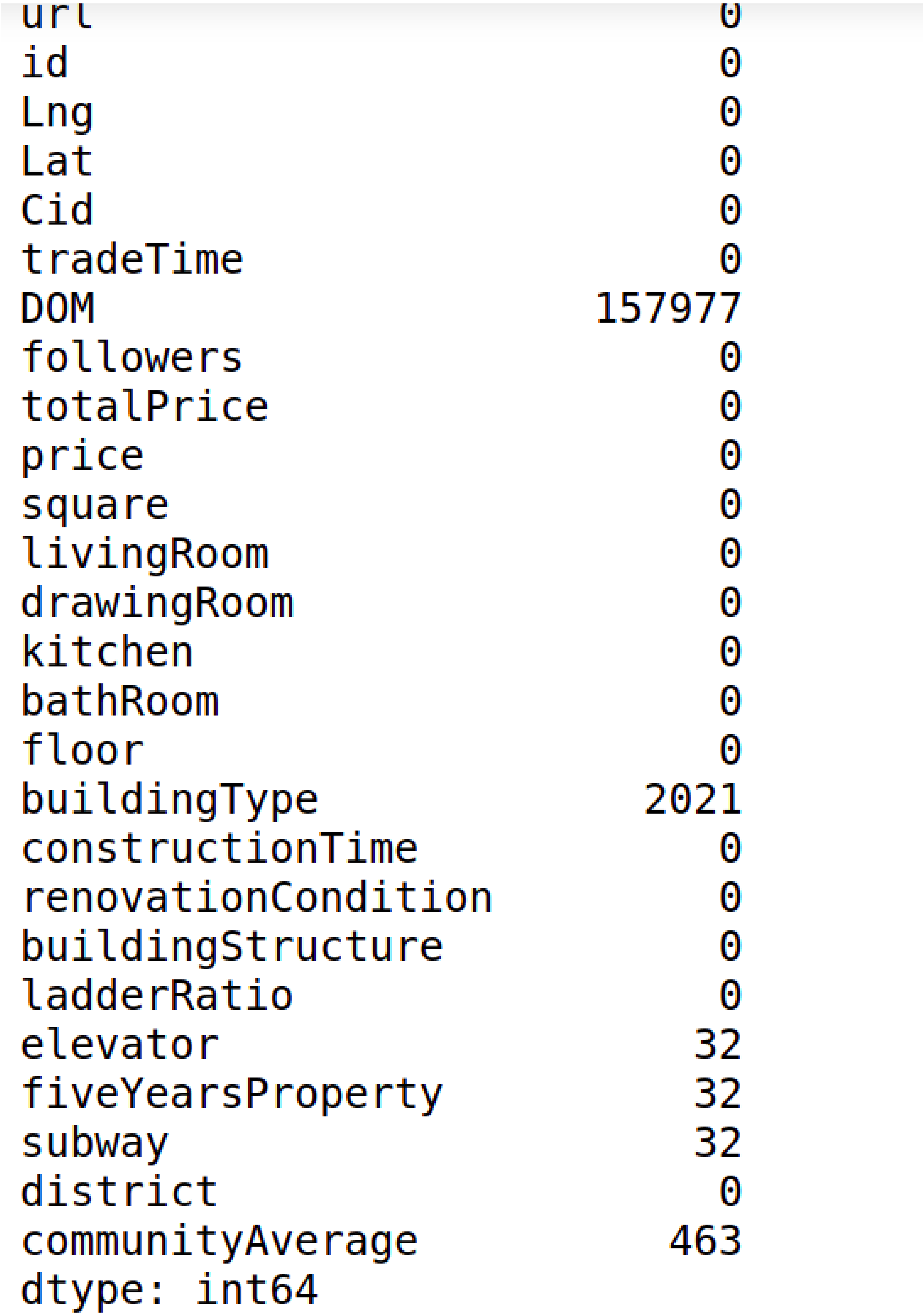
A.Data Cleaning

we have 26 columns ,from these we don't want some column(i.e. url,id,cid) then we will perform data cleaning wich involve following steps. our target variable is totalPrice

a. Impute/Remove missing values or Null values (NaN)

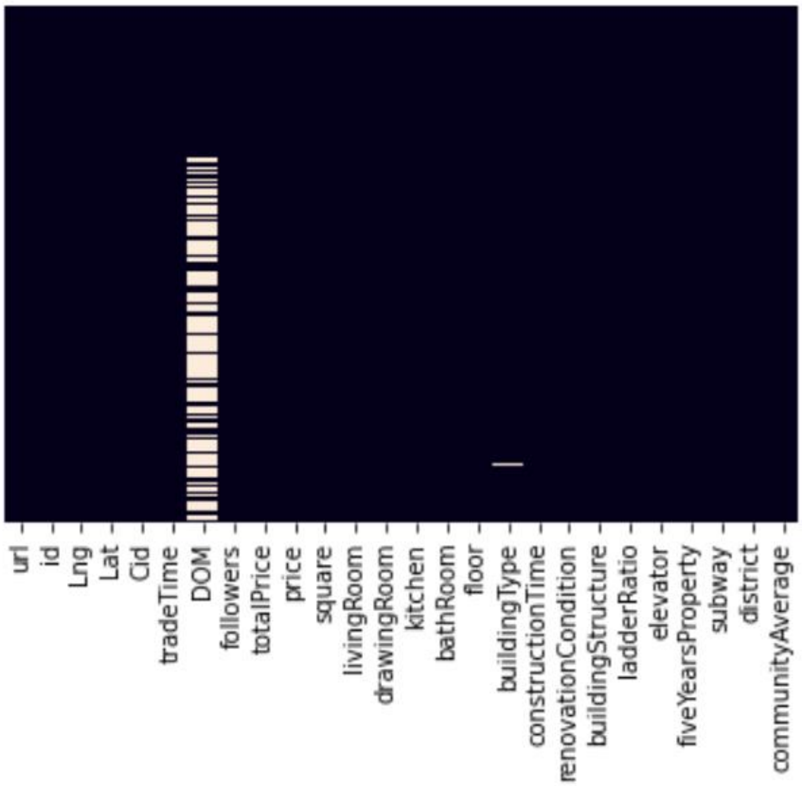
b. Remove unnecessary and corrupted data.

c. Date/Text parsing if required.



```
url                         0
id                          0
Lng                         0
Lat                         0
Cid                         0
tradeTime                   0
DOM                    157977
followers                   0
totalPrice                  0
price                       0
square                      0
livingRoom                  0
drawingRoom                 0
kitchen                     0
bathRoom                    0
floor                       0
buildingType             2021
constructionTime            0
renovationCondition         0
buildingStructure           0
ladderRatio                 0
elevator                   32
fiveYearsProperty          32
subway                     32
district                    0
communityAverage          463
dtype: int64
```

we handle NAN value using appropriate solutions.



DOM Column have more than 50% value are missing it's better to delete that column

```
data.floor.unique()
#so,floor have a chinese character...
```

```
array(['高 26', '高 22', '中 4', '底 21', '中 6', '中 8', '高 6', '高 10', '中 23',
       '底 11', '底 3', '高 24', '低 23', '中 19', '高 18', '低 25', '中 12',
       '中 14', '中 30', '中 27', '中 5', '低 18', '底 28', '中 11', '低 9',
       '顶 7', '顶 27', '低 6', '中 17', '顶 6', '中 24', '中 15', '底 5', '中 29',
       '顶 19', '顶 5', '中 9', '低 22', '顶 18', '低 16', '高 13', '高 9',
       '高 17', '底 6', '中 28', '低 26', '底 15', '高 16', '底 2', '低 7',
       '中 13', '低 33', '底 14', '高 15', '底 4', '顶 11', '中 32', '顶 16',
       '底 18', '顶 17', '低 14', '低 10', '底 20', '高 12', '低 31', '低 30',
       '低 19', '低 12', '中 10', '中 16', '顶 20', '底 19', '中 31', '低 13',
       '底 10', '高 25', '中 21', '中 20', '高 20', '低 21', '低 24', '顶 4',
```

some column have unique character. we solve these problem using split method and create seprate column for unique character.

We also have a categorical data we handle such kind of data using dummies variable concept. following are the columns which have categorical data.

a. renovationCondition

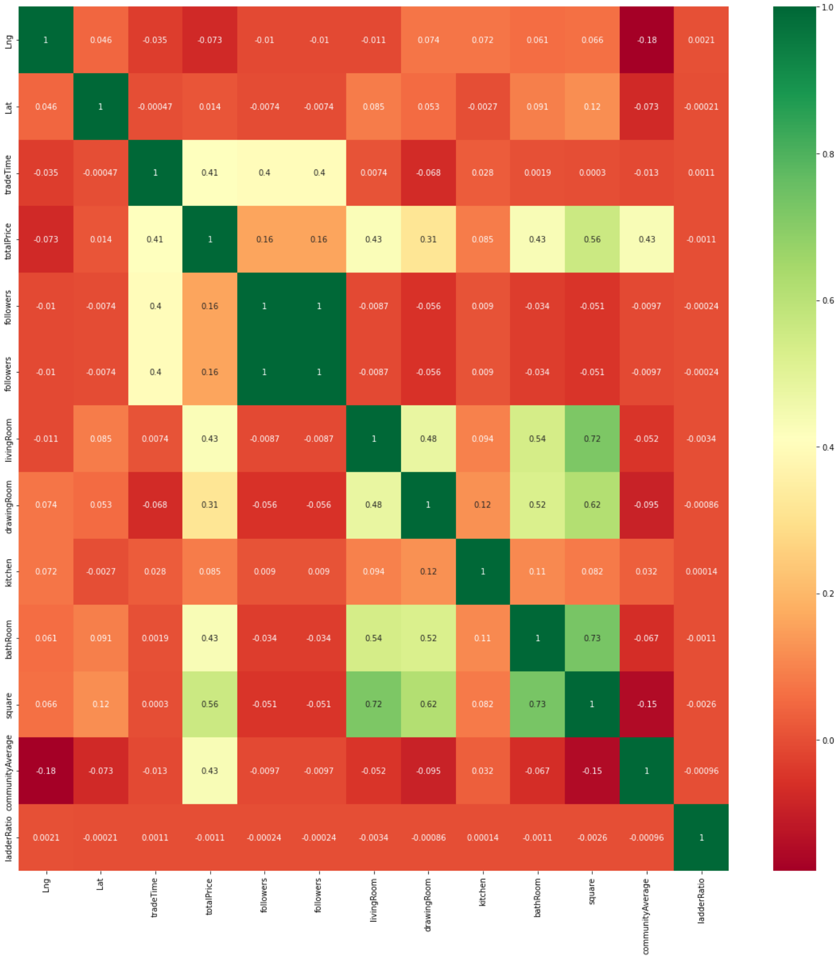
b. buildingStructure

c. buildingType

d. district

e. elevator

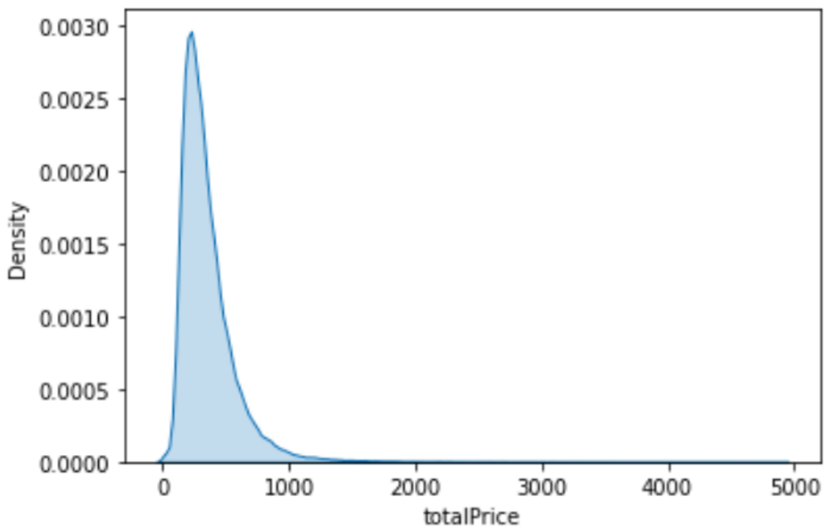
f. floor_type



Summary of the Heat-Map

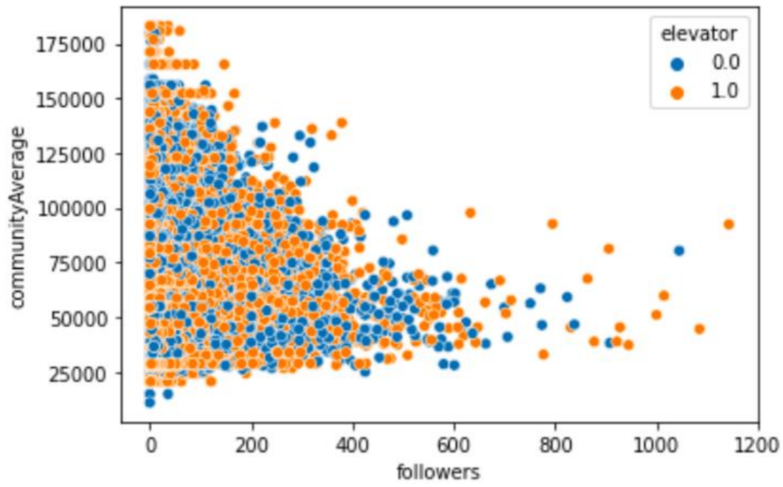
a. totalPrice is highly corellated with community average,square,bathroom,livingroom and Trde Time.

b. totalprice is highly negative corellated with ladderRatio,lat and lng.



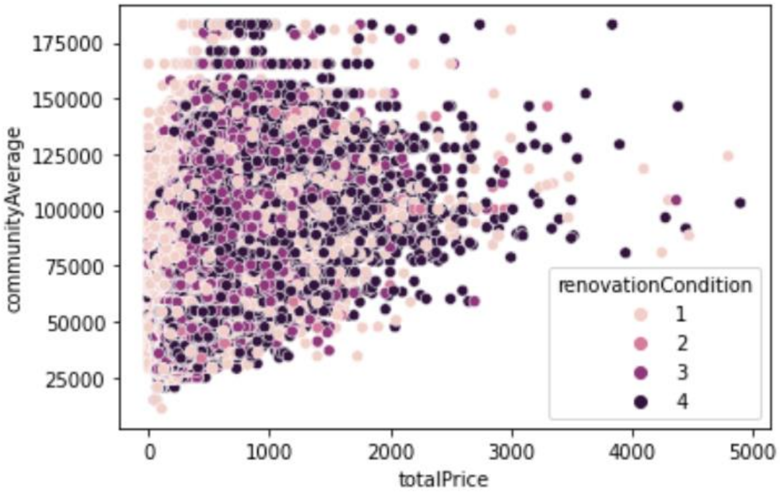
Summary of the Density Plot

a. most of the output features is lies between 0-2500



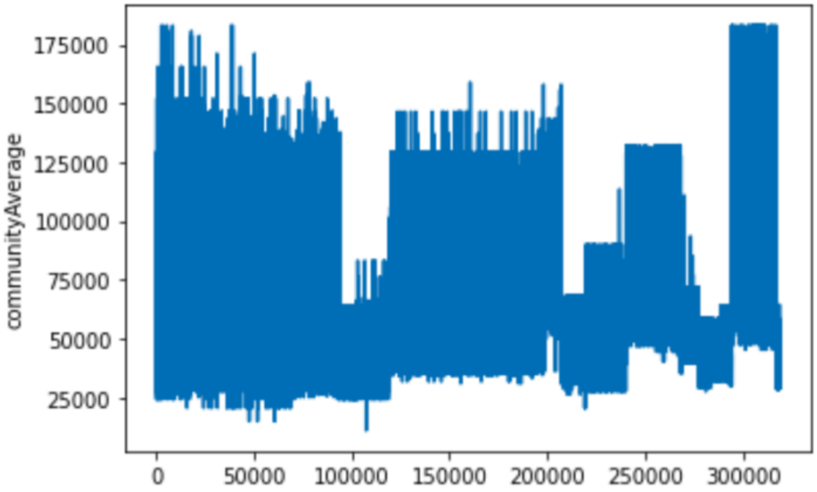
Summary of Scatterplot

a. Most of the House Followers 0-400.



Summery of Scatterplot with respect to renovationCondition

a. most of the expensive houses have HardCover as a renovation condition

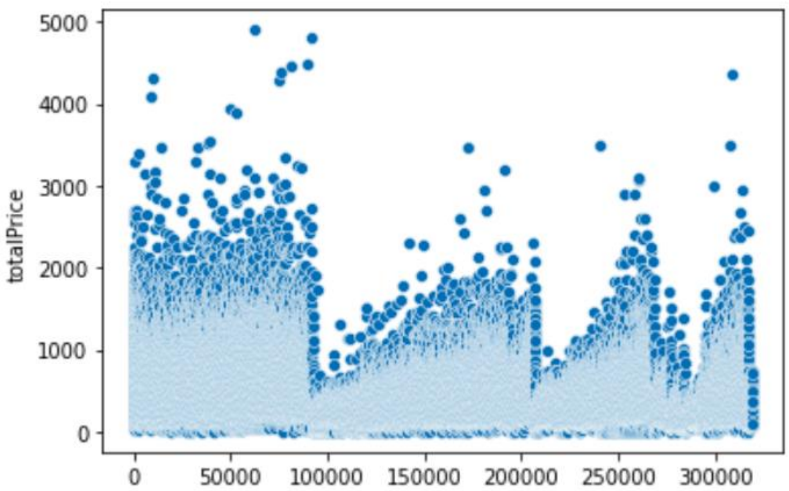


Summary of lineplot

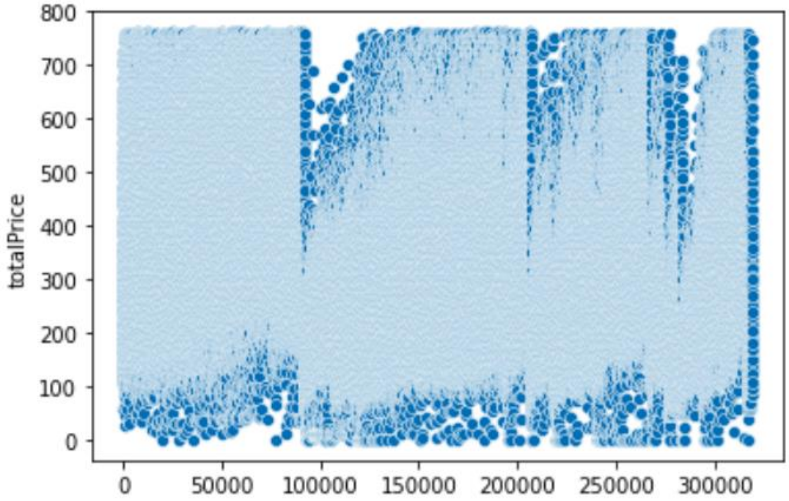
a. Most of the peoples average are lies in 12500-150000 ...

B. Feature Engineering

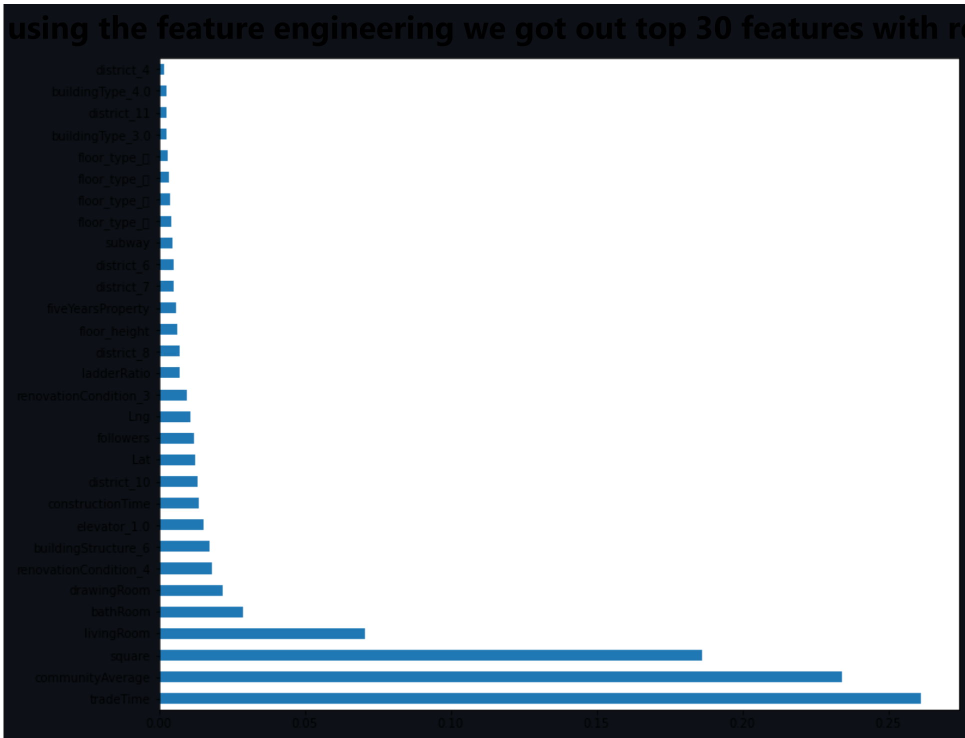
[redacted]



from the above figure we can notice that we have an outlier present in our dataset.
for outlier we can use IQR method and after using IQR method.Now, our data looks fine.



using the feature engineering we got out top 30 features with respect to totalPrice .



So,these are the top 20 features for our model

a. tradeTime

b. CommunityAverage

c. square

d. livingRoom

e. bathRoom

f. drawingRoom

g. renovationCondition

h. buildingStructure

i. elevator

j. constructionTime

k. Followers

C. Data Normalization

Normalization (min-max Normalization)

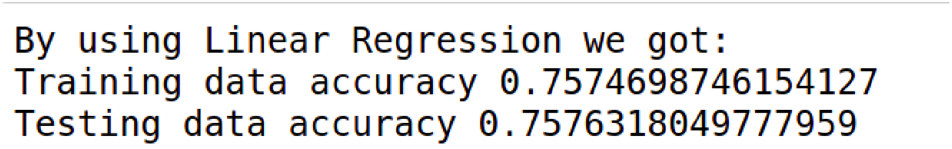
In this approach we scale down the feature in between 0 to 1

we have numerical column where we can apply min-max Normalization.

```
col_for_normalization=['Lng', 'Lat','followers','square','livingRoom', 'drawingRoom',
                       'kitchen', 'bathRoom','ladderRatio', 'fiveYearsProperty',
                       'subway', 'communityAverage','floor_height']
```

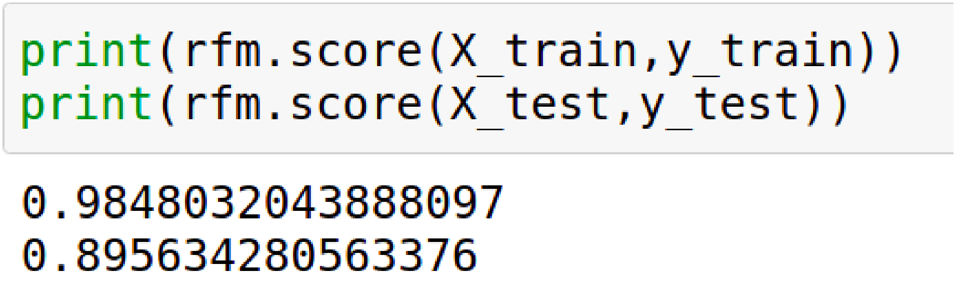
5. Choosing Best ML Model

List of the model that we can use for our problem
a. LinearRegression model
b. KNN Model
c. Decesion Tree
d. Random Forest



```
By using Linear Regression we got:
Training data accuracy 0.7574698746154127
Testing data accuracy 0.7576318049777959
```

Using the linearRegression we got only 75 % accuracy.



```
print(rfm.score(X_train,y_train))
print(rfm.score(X_test,y_test))
```

```
0.9848032043888097
0.895634280563376
```

Using the Random Forest we got 98 % accuracy on train data and 89 % on test data .so,we can consider RandomForest as a Best Algorithm for this problem.

6. Model Creation

So,using a RandomForest we got good accuracy , we can Hyperparameter tuning for best accuracy.

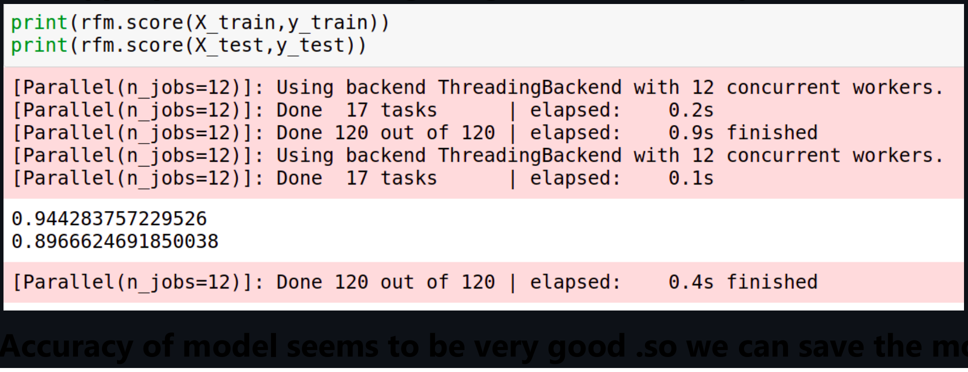
Algorithm that can be used for Hyperparameter tuning are :-

a. GridSearchCV
b. RandomizedSearchCV
c. Bayesian Optimization-Automate Hyperparameter Tuning (Hyperopt)
d. Sequential model based optimization
e. Optuna-Automate Hyperparameter Tuning
f. Genetic Algorithm

Main parameters used by RandomForest Algorithm are :-

a. n_estimators ---> The number of trees in the forest.
b. criterion--->{"mse", "mae"}-->The function to measure the quality of a split
c. max_features--->{"auto", "sqrt", "log2"}--> The number of features to consider when looking for the best split:

So, After Hyperparameter Tuning we got 90 % accuracy on test data and 94 % accuracy on train data.



```
print(rfm.score(X_train,y_train))
print(rfm.score(X_test,y_test))
```

```
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  17 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 120 out of 120 | elapsed:    0.9s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  17 tasks      | elapsed:    0.1s
```

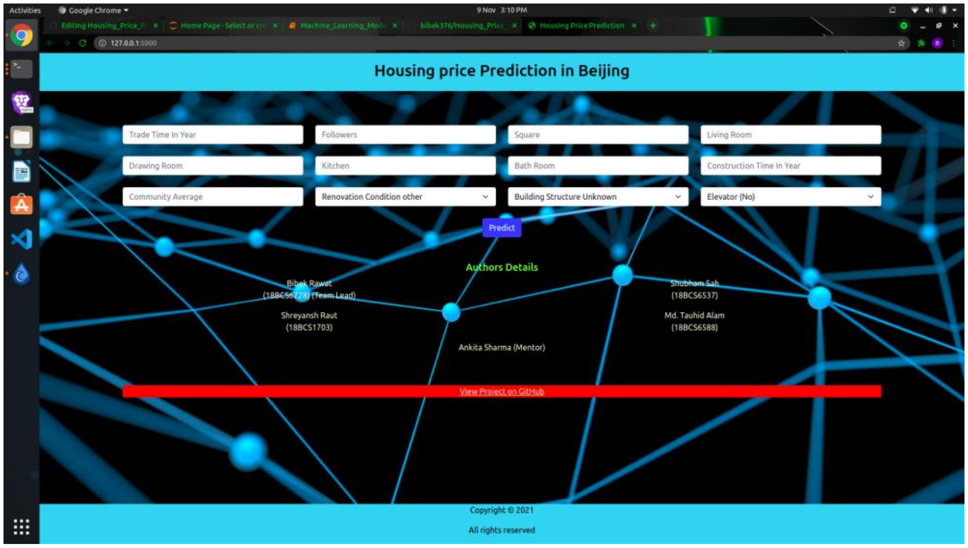
```
0.944283757229526
0.8966624691850038
```

```
[Parallel(n_jobs=12)]: Done 120 out of 120 | elapsed:    0.4s finished
```

Now,Accuracy of model seems to be very good .so we can save the model using pickle.

7. Model Deployment

After creating model ,we integrate that model with beautiful UI. for the UI part we used HTML,CSS,JS and Flask.



Housing price Prediction in Beijing

Authors Details

Bibek Rawat
(18BCS6728) (Team Lead)

Shreyansh Raut
(18BCS1703)

Ankita Sharma (Mentor)

Shubham Sah
(18BCS6537)

Md. Tauhid Alam
(18BCS6588)

Copyright © 2021
All rights reserved

8. Model Conclusion

Model predict 90% accurately on test data and 94% accurately on train data .

9. Project Innovation

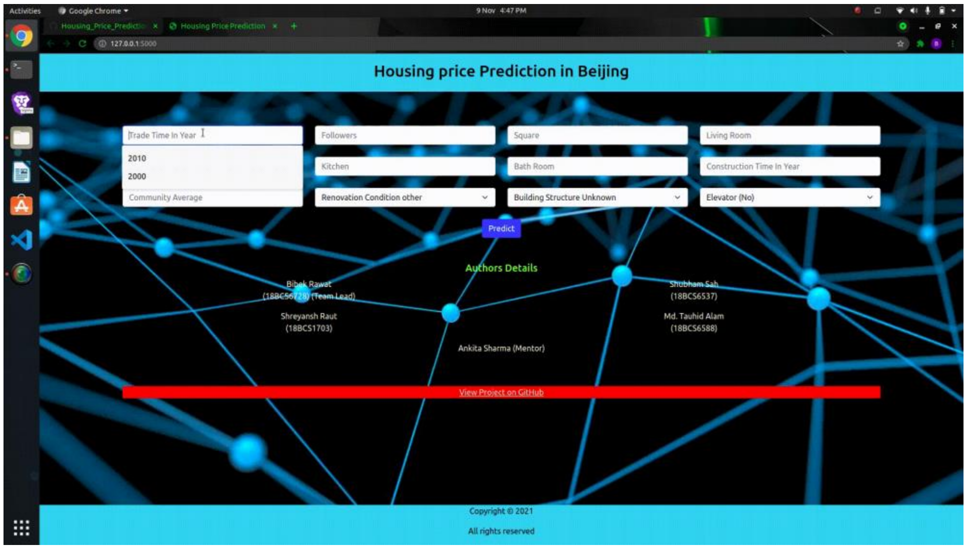
a. Easy to use
b. open source
c. Best accuracy
d. GUI Based Application

10. Limitation And Next Step

Limitation are :-
a. Mobile Application
b. Accuracy can be improve
c. Model Size is heavy(~310 mb)
d. Feature is limited

Next Step are :-
a. we will work on mobile application
b. we will reduce the size of model using PCA .

11. Working Project Video



Housing price Prediction in Beijing

Renovation Condition other
Building Structure Unknown
Elevator (No)

Authors Details

Bibek Rawat
(18BCS6728) (Team Lead)

Shreyansh Raut
(18BCS1703)

Shubham Sah
(18BCS6537)

Md. Tauhid Alam
(18BCS6588)

Ankita Sharma (Mentor)

Copyright © 2021
All rights reserved

Example: Suppose we apply the above code to a dataset of house prices. After training the model and evaluating its performance, we might obtain an R-squared value of 0.85, indicating a strong

correlation between predicted and actual prices. The scatter plot visualizes how close the predicted prices are to the actual prices, with most points lying close to the diagonal line.

Results: We will present the results of our house price prediction model, including performance metrics and visualizations. High-quality predictions will demonstrate the effectiveness of the implemented approach.

Conclusion: In this project, we have demonstrated how to predict house prices using Python and machine learning techniques. Accurate house price predictions are essential for both buyers and sellers in the real estate market. The code examples and examples provided here can serve as a foundation for building more advanced models and making informed real estate decisions.