FLIP ROBO

Housing project

Submitted by

Fenny Denny

ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped me and guided me in completion of the project.

I wish to express my sincere gratitude to Miss. Khushboo Garg, SME for providing me an opportunity to do my internship and project work in "FLIP ROBO".

It gives me immense pleasure in presenting this project report on "Micro Credit Defaulter Model". It has been my privilege to have a team of project guide who have assisted me from the commencement of this project. The success of this project is a result of sheer hard work, and determination put in by me with the help of You Tube videos, references taken from Kaggle.com, skikit-learn.org.. To know more about micro finance, I read

https://www.geeksforgeeks.org/

https://github.com/

https://www.mckinsey.com/

https://www.counterpointresearch.com/

I hereby take this opportunity to add a special note of thanks for to Miss. Khushboo Garg, who undertook to act as my mentor despite his many other professional commitments. Her wisdom, knowledge and commitment to the highest standards inspired and motivated me. Without his insight, support this project wouldn't have reached fruitfulness.

The project is dedicated to all those people of Fliprobo, Datatrained who helped me while doing this project.

INTRODUCTION

Houses are one of the necessary need of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies. Our problem is related to one such housing company. A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below. The company is looking at prospective properties to buy houses to enter the market. You are required to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not.

For this company wants to know:

- Which variables are important to predict the price of variable?
- How do these variables describe the price of the house?

Conceptual Background of the Domain Problem

The company wants to enter the Australian Market and hence are looking at prospective properties to buy. They want to understand what are the factors affecting the prices and how exactly those factors are influencing it. The company would then manipulate the strategy of the firm and concentrate on areas that will yield high return.

Motivation for the Problem Undertaken

As I am a part of the Internship Process of RoboFlip Technologies, this project was given to me as the first assignment after getting accepted as a Data Science Intern in RoboFlip Technologies. I took the project as a challenge for myself to see how I have improved and upgraded from the day I started learning with Data-Trained.

Analytical Problem Framing

• Mathematical/ Analytical Modeling of the Problem

As from the Problem statement we can understand that this Dataset needs Regression for Model Building.

So what is Regression? Regression in machine learning consists of mathematical methods that allow data scientists to predict a continuous outcome (y) based on the value of one or more predictor variables (x). Linear regression is probably the most popular form of regression analysis because of its ease-of-use in predicting and forecasting.

Few of the terminologies encountered in machine learning – classification:

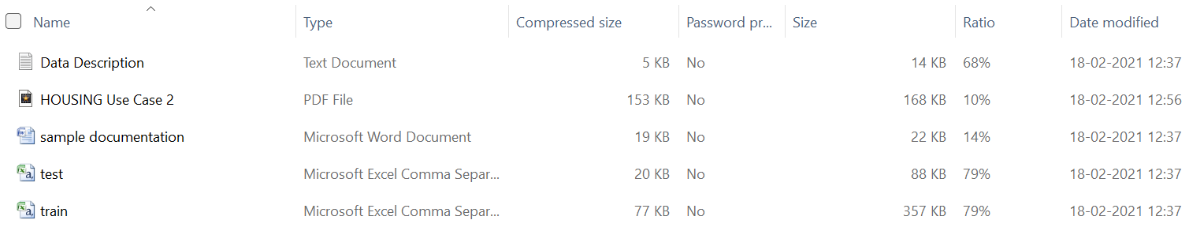
- Classifier: An algorithm that maps the input data to a specific category.
- Regression model: A classification model tries to draw some conclusion from the input values given for training. It will predict the class labels/categories for the new data.
- Feature: A feature is an individual measurable property of a phenomenon being observed.

The following are the steps involved in building a Regression model:

- Initialize the Regressor to be used.
- Train the Regressor: All classifiers in scikit-learn uses a fit(X, y) method to fit the model (training) for the given train data X and train label y.
- Predict the target: Given an unlabelled observation X, the predict(X) returns the predicted label y.
- Evaluate the Regressor model So we have somewhat understand what Regression is and why we will be using this in Model building of our Dataset.

Data Sources and their formats

The PFA Housing Project Dataset was sourced from RoboFlip Technologies as a part of their internship project. - The Dataset is in a .csv (comma separated value). - The general information about the dataset is given in a documentation format. - Information about the dependent and independent variables are given in an Excel Format.



□ Name	Type	Compressed size	Password pr...	Size	Ratio	Date modified
Data Description	Text Document	5 KB	No	14 KB	68%	18-02-2021 12:37
HOUSING Use Case 2	PDF File	153 KB	No	168 KB	10%	18-02-2021 12:56
sample documentation	Microsoft Word Document	19 KB	No	22 KB	14%	18-02-2021 12:37
test	Microsoft Excel Comma Separ...	20 KB	No	88 KB	79%	18-02-2021 12:37
train	Microsoft Excel Comma Separ...	77 KB	No	357 KB	79%	18-02-2021 12:37

Data Preprocessing Done

This is the most crucial step of all in Building a Machine Learning Model. So first, what is Data Pre-processing? Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, lacking in certain behaviours or trends, and is likely to contain many errors.

Data pre-processing is a proven method of resolving such issues. Data pre-processing prepares raw data for further processing. Data pre-processing is used in database-driven applications such as customer relationship management and rule-based applications (like neural networks).

In Machine Learning (ML) processes, data pre-processing is critical to encode the dataset in a form that could be interpreted and parsed by the algorithm. Data goes through a series of steps during pre-processing:

Data Cleaning:

Data is cleansed through processes such as filling in missing values or deleting rows with missing data, smoothing the noisy data, or resolving the inconsistencies in the data. Smoothing noisy data is particularly important for ML datasets, since machines cannot make use of data they cannot interpret. Data can be cleaned by dividing it into equal size segments that are thus smoothed (binning), by fitting it to a linear or multiple regression function (regression), or by grouping it into clusters of similar data (clustering). Data inconsistencies can occur due to human errors (the information was stored in a wrong field). Duplicated values should be removed through reduplication to avoid giving that data object an advantage (bias).

Data Integration:

Data with different representations are put together and conflicts within the data are resolved.

Data Transformation:

Data is normalized and generalized. Normalization is a process that ensures that no data is redundant, it is all stored in a single place, and all the dependencies are logical.

Data Reduction:

When the volume of data is huge, databases can become slower, costly to access, and challenging to properly store. Data reduction step aims to present a reduced representation of the data in a data warehouse. There are various methods to reduce data.

For example, once a subset of relevant attributes is chosen for its significance, anything below a given level is discarded. Encoding mechanisms can be used to reduce the size of data as well. If all original data can be recovered after compression, the operation is labelled as lossless. If some data is lost, then it's called a lossy reduction.

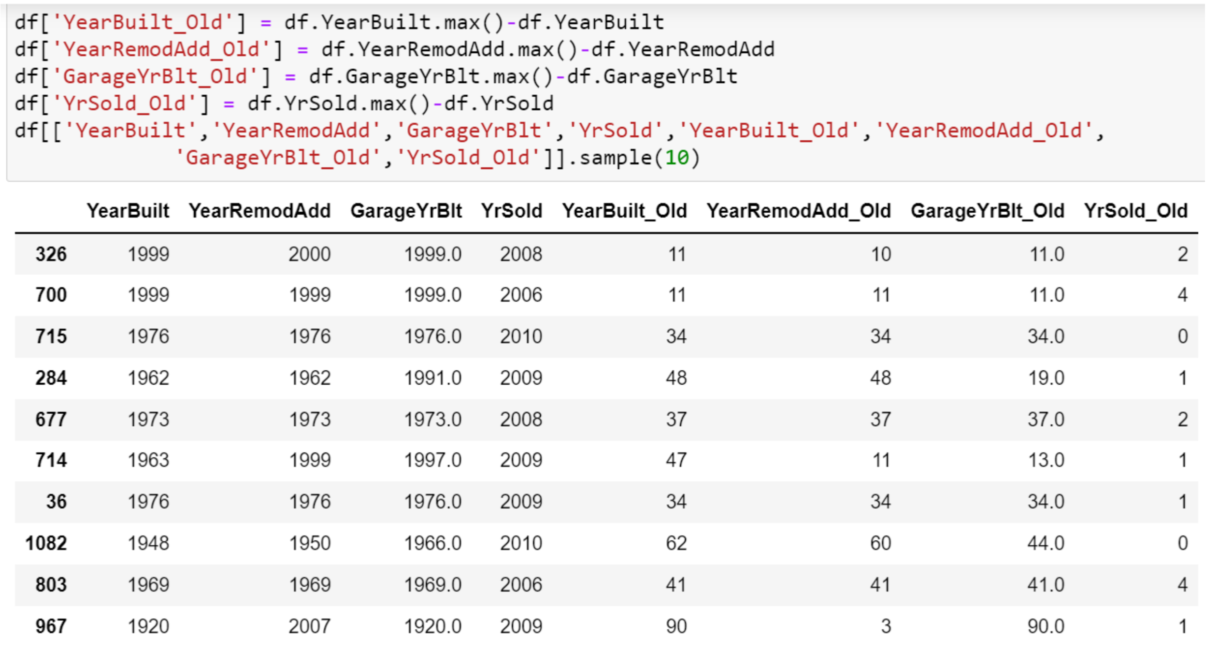
Aggregation can also be used, for example, to condense countless transactions into a single weekly or monthly value, significantly reducing the number of data objects.

Data Discretization:

Data could also be discretized to replace raw values with interval levels. This step involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals.

Data Sampling:

Sometimes, due to time, storage or memory constraints, a dataset is too big or too complex to be worked with. Sampling techniques can be used to select and work with just a subset of the dataset, provided that it has approximately the same properties of the original one.



```
df['YearBuilt_Old'] = df.YearBuilt.max()-df.YearBuilt
df['YearRemodAdd_Old'] = df.YearRemodAdd.max()-df.YearRemodAdd
df['GarageYrBlt_Old'] = df.GarageYrBlt.max()-df.GarageYrBlt
df['YrSold_Old'] = df.YrSold.max()-df.YrSold
df[['YearBuilt','YearRemodAdd','GarageYrBlt','YrSold','YearBuilt_Old','YearRemodAdd_Old',
             'GarageYrBlt_Old','YrSold_Old']].sample(10)
```

	YearBuilt	YearRemodAdd	GarageYrBlt	YrSold	YearBuilt_Old	YearRemodAdd_Old	GarageYrBlt_Old	YrSold_Old
326	1999	2000	1999.0	2008	11	10	11.0	2
700	1999	1999	1999.0	2006	11	11	11.0	4
715	1976	1976	1976.0	2010	34	34	34.0	0
284	1962	1962	1991.0	2009	48	48	19.0	1
677	1973	1973	1973.0	2008	37	37	37.0	2
714	1963	1999	1997.0	2009	47	11	13.0	1
36	1976	1976	1976.0	2009	34	34	34.0	1
1082	1948	1950	1966.0	2010	62	60	44.0	0
803	1969	1969	1969.0	2006	41	41	41.0	4
967	1920	2007	1920.0	2009	90	3	90.0	1

Checking the missing values

Missing values in the dataset can be checked by below python code:-

missing_values=[x for x in df.columns if df[x].isnull().sum()>1]

print(‘Number of missing variable columns:’, len(missing_values))

print(“Missing values in the dataset : \n “, missing_values)

print(“-”*125) df[missing_values].head()

Checking the percentage of the missing values

Missing value can be checked using the following code:-

For feature in missing_values:

print(feature,np.round(df[feature].isnull().mean()*100,4), “% Missing Values”)

Extracting all the numerical feature:

numerical_features=[x for x in df.columns if df[x].dtypes != “O”] print(“The number of the numerical columns in the dataset:”, len(numerical_features))

print(“Numerical columns in the dataset:\n”, numerical_features)

```
print(“-”*125) df[numerical_features].head()
```

Extract the year column from the dataset:

```
year_feature=[x for x in df.columns if ‘Yr’ in x or ‘Year’ in x]

print(“The number of Year column in the dataset :”,len(year_feature))

print(“Year columns in the dataset :\n”,year_feature)

print(“-”*125) df[year_feature].head()
```

Checking the unique items in date time columns:

checking the unique items in the datetime columns for feature in year_feature:

```
print(“The unique items in the colunmn”, feature, “:\n”, df[feature].unique())
```

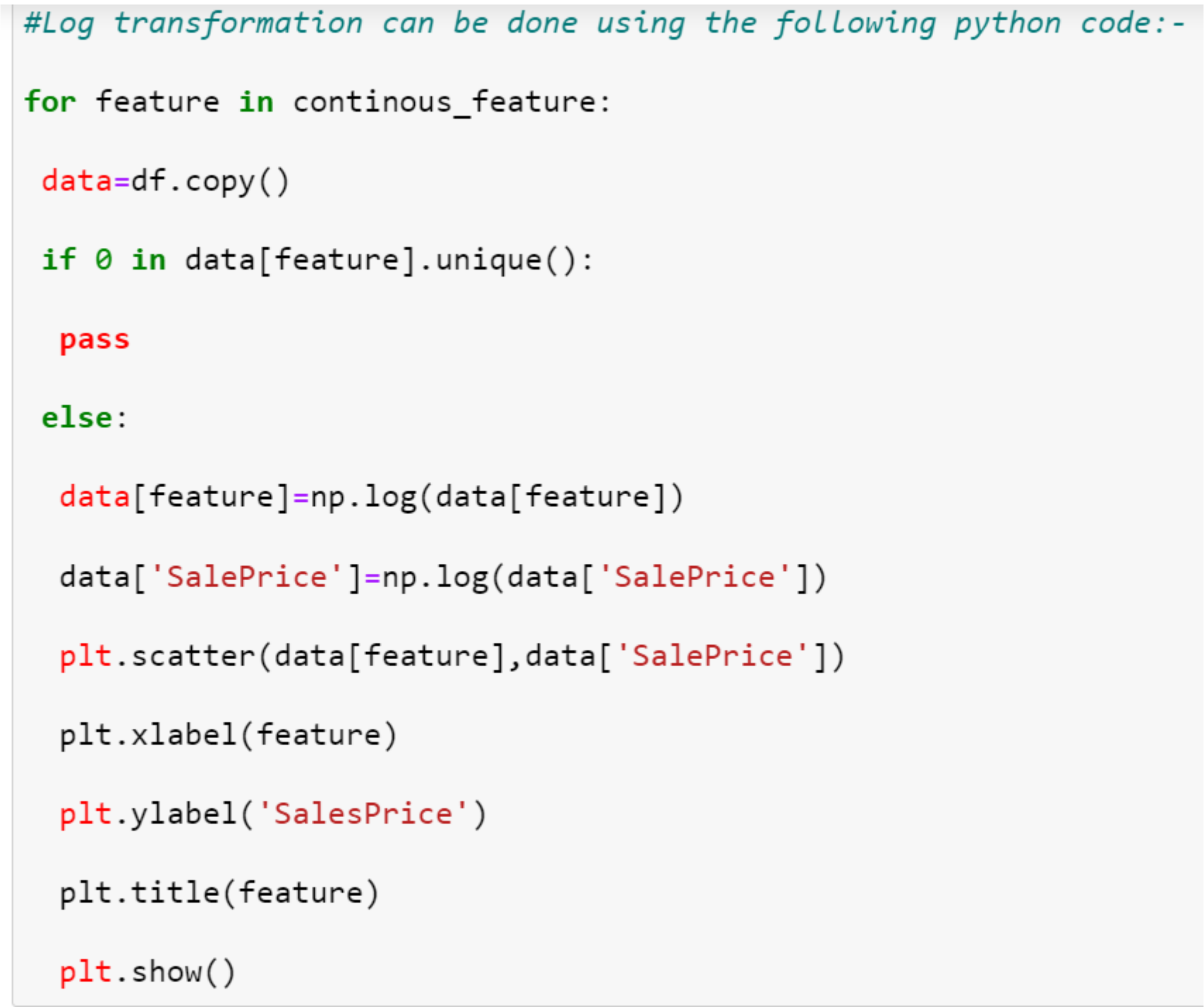
So now we have established what is Data pre-processing,

I will let know all the steps I took to clean the Data before proceeding: -

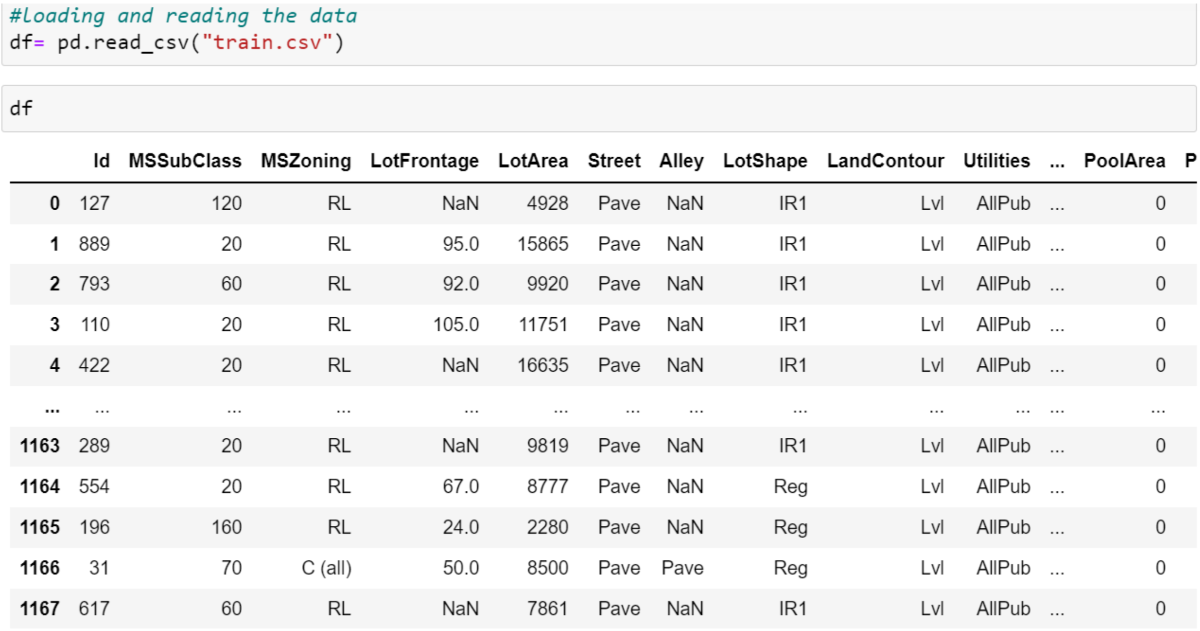
Firstly I checked for Null Values, there were so we have to treat them later. –

Secondly I checked for any Nan values in the dataset which I found, so I filled all the Nan values with zero rather than removing it. –

Thirdly I found outliers and skewness in the Dataset which I dealt with by using median of the columns and replacing it with respective medians. - I also found some ‘-‘sign in the dataset. Which I removed with the help of replace statement.



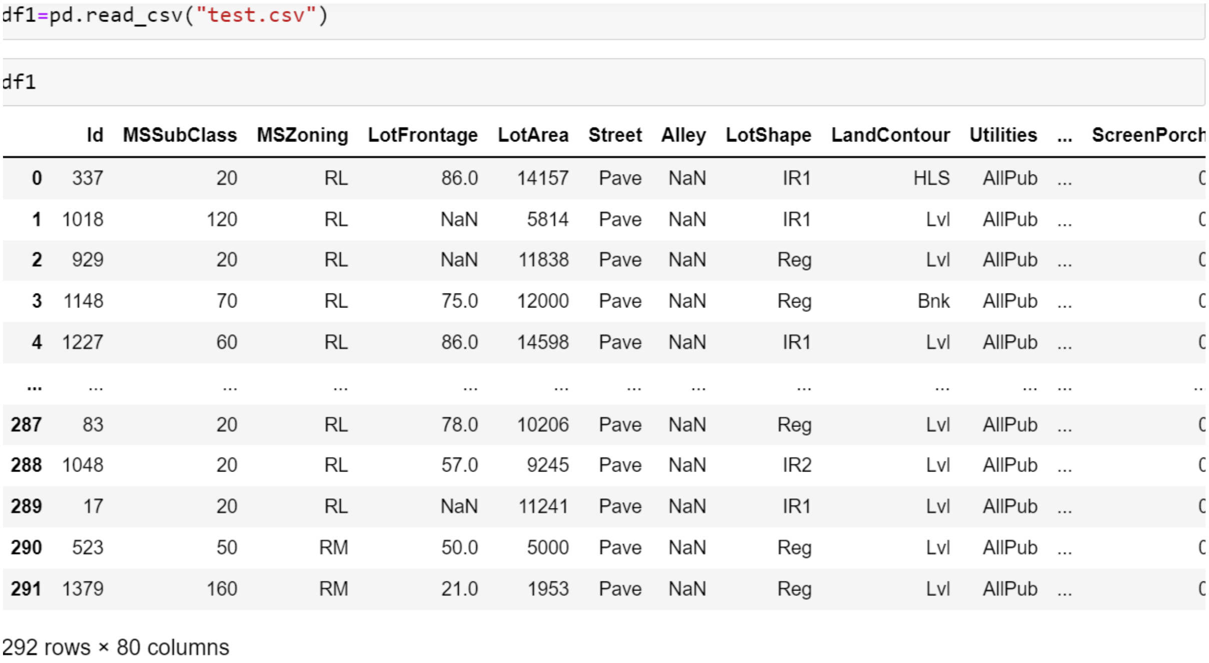
```
#Log transformation can be done using the following python code:-

for feature in continous_feature:

 data=df.copy()

 if 0 in data[feature].unique():

  pass

 else:

  data[feature]=np.log(data[feature])

  data['SalePrice']=np.log(data['SalePrice'])

  plt.scatter(data[feature],data['SalePrice'])

  plt.xlabel(feature)

  plt.ylabel('SalesPrice')

  plt.title(feature)

  plt.show()
```



```
#loading and reading the data
df= pd.read_csv("train.csv")
```

```
df
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	P
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	...	0	
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	...	0	
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	...	0	
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	...	0	
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
1163	289	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	...	0	
1164	554	20	RL	67.0	8777	Pave	NaN	Reg	Lvl	AllPub	...	0	
1165	196	160	RL	24.0	2280	Pave	NaN	Reg	Lvl	AllPub	...	0	
1166	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	...	0	
1167	617	60	RL	NaN	7861	Pave	NaN	IR1	Lvl	AllPub	...	0	

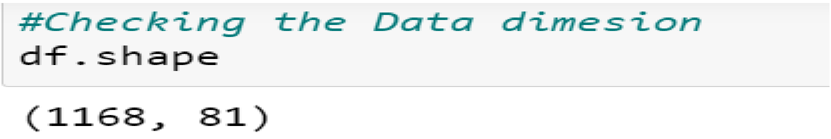


```
df1=pd.read_csv("test.csv")
```

```
df1
```

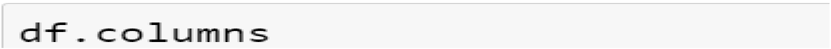
	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch
0	337	20	RL	86.0	14157	Pave	NaN	IR1	HLS	AllPub	...	0
1	1018	120	RL	NaN	5814	Pave	NaN	IR1	Lvl	AllPub	...	0
2	929	20	RL	NaN	11838	Pave	NaN	Reg	Lvl	AllPub	...	0
3	1148	70	RL	75.0	12000	Pave	NaN	Reg	Bnk	AllPub	...	0
4	1227	60	RL	86.0	14598	Pave	NaN	IR1	Lvl	AllPub	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
287	83	20	RL	78.0	10206	Pave	NaN	Reg	Lvl	AllPub	...	0
288	1048	20	RL	57.0	9245	Pave	NaN	IR2	Lvl	AllPub	...	0
289	17	20	RL	NaN	11241	Pave	NaN	IR1	Lvl	AllPub	...	0
290	523	50	RM	50.0	5000	Pave	NaN	Reg	Lvl	AllPub	...	0
291	1379	160	RM	21.0	1953	Pave	NaN	Reg	Lvl	AllPub	...	0

292 rows × 80 columns

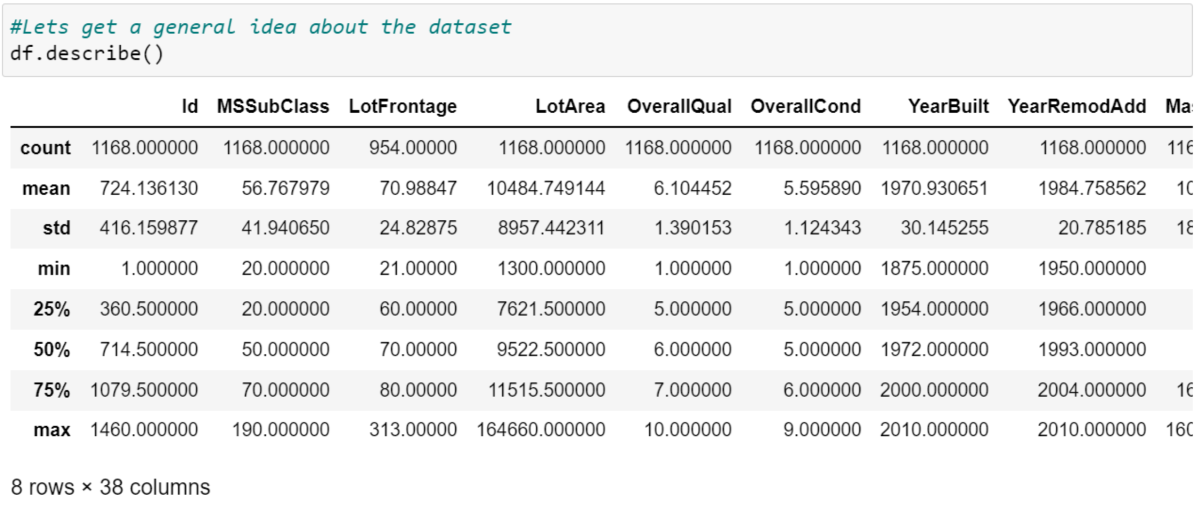


```
#Checking the Data dimesion
df.shape
```

```
(1168, 81)
```



```
df.columns
```



```
#Lets get a general idea about the dataset
df.describe()
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	Ma
count	1168.000000	1168.000000	954.00000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	116
mean	724.136130	56.767979	70.98847	10484.749144	6.104452	5.595890	1970.930651	1984.758562	10
std	416.159877	41.940650	24.82875	8957.442311	1.390153	1.124343	30.145255	20.785185	18
min	1.000000	20.000000	21.00000	1300.000000	1.000000	1.000000	1875.000000	1950.000000	
25%	360.500000	20.000000	60.00000	7621.500000	5.000000	5.000000	1954.000000	1966.000000	
50%	714.500000	50.000000	70.00000	9522.500000	6.000000	5.000000	1972.000000	1993.000000	
75%	1079.500000	70.000000	80.00000	11515.500000	7.000000	6.000000	2000.000000	2004.000000	16
max	1460.000000	190.000000	313.00000	164660.000000	10.000000	9.000000	2010.000000	2010.000000	160

8 rows × 38 columns

So from both the tables above we can conclude that:

- There are no null values in the dataset that we have to deal with.

- There also seems to be outliers for some of the features.

- And some kind of skewness can also be seen from the description which we will see it more clearly later on.

- State the set of assumptions (if any) related to the problem under consideration

As from the Problem statement which states that we have to build a model which will predict the average price. Since the target variable is continuous we can readily presume that it is a Regression Problem for which we will be using Regression algorithms to build our model from the dataset given. Secondly as we know Data is very expensive I tried those methods for Data cleaning which doesn’t require removal of bulk of Data. Like for example I did not use IQR method for Outliers as it would have erased huge amounts of Data. Instead I used Median method for Outlier treatment. Lastly I will be using RandomSearch CV for hyper parameter tuning as it uses least amount of memory and processing power to run the algorithms as compared to that of GridSearch CV.

Hardware & Software Requirements & Tools Used While taking up the project

Data Science task should be done with sophisticated machine with high end machine configuration. The machine which I'm currently using is powered by intel core i5 processor with 8GB of RAM. With this above-mentioned configuration, I managed to work with the data set in Jupyter Notebook which help us to write Python codes. As I'm using low configuration machine so it took more time then usual to execute codes. The library used for the assignment are Numpy, Pandas, Matplotlib, Seaborn, Scikit learn

Model/s Development and Evaluation Identification of possible problem-solving approaches (methods)

So below are the few things that I faced during the project:

- There are null values in the dataset.
- For some features, there may be values which might not be realistic.
- I also came across outliers in some features.
- Testing of Identified Approaches (Algorithms) So as we know it is a Regression problem we will use classification algorithms to train our Dataset. Following are some regression Algorithms:
- Simple Linear Regression model
- Lasso Regression
- Logistic regression
- Support Vector Machines
- Multivariate Regression algorithm & Ridge Regression

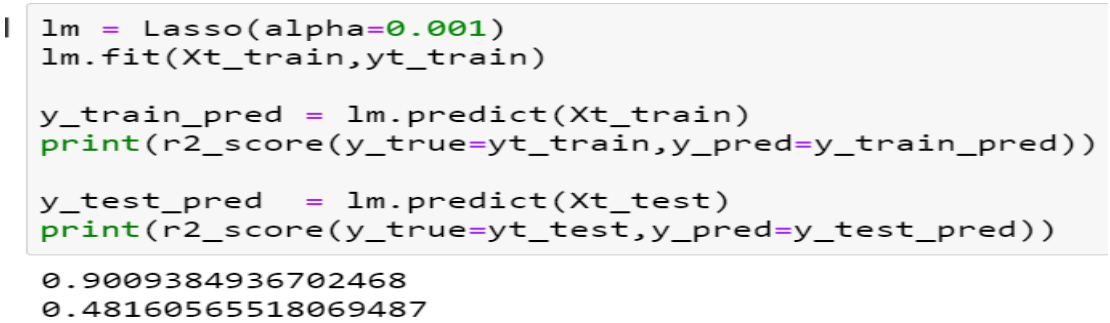
Let’s understand the main two regression algorithms briefly: LASSO - Lasso regression is a regularization technique. It is used over regression methods for a more accurate prediction. This model uses shrinkage. Shrinkage is where data values are shrunk towards a central point as the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of multicollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination. Lasso Regression uses L1 regularization technique (will be discussed later in this article). It is used when we have more number of features because it automatically performs feature selection.

RIDGE - Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. When the issue of multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values to be far away from the actual values. The cost functions for ridge regression: Min (||Y – X(theta)||^2 + λ||theta||^2) Lambda is the penalty term. λ given here is denoted by an alpha parameter in the ridge function. So, by changing the values of alpha, we are controlling the penalty term. Higher the values of alpha, bigger are the penalty and therefore the magnitude of coefficients is reduced.

- It shrinks the parameters. Therefore, it is used to prevent multicollinearity

- It reduces the model complexity by coefficient shrinkage Run and Evaluate selected models So for the project PFA Housing, I will be using these following algorithms which I choose by keeping in mind the time taken to iterate and what is better for this Dataset: -

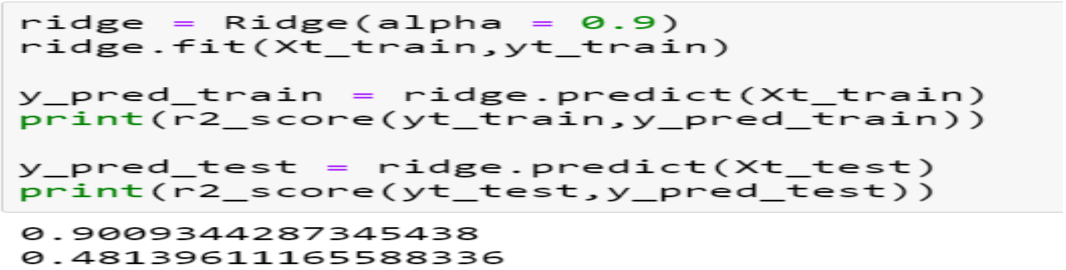
So the first Model which I am going to use is Lasso Regression.



```
lm = Lasso(alpha=0.001)
lm.fit(Xt_train,yt_train)

y_train_pred = lm.predict(Xt_train)
print(r2_score(y_true=yt_train,y_pred=y_train_pred))

y_test_pred  = lm.predict(Xt_test)
print(r2_score(y_true=yt_test,y_pred=y_test_pred))
```

```
0.9009384936702468
0.48160565518069487
```

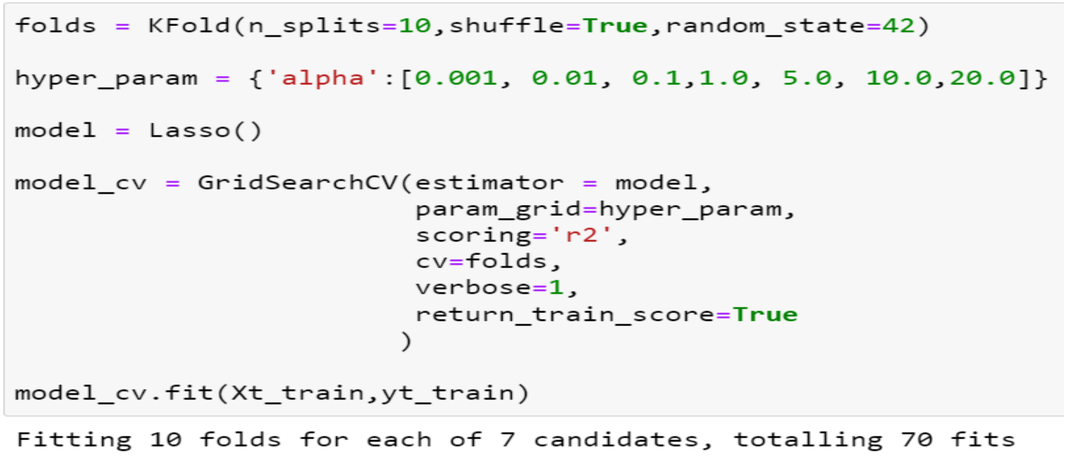
The second Model which I am going to use is Ridge Regressor.



```
ridge = Ridge(alpha = 0.9)
ridge.fit(Xt_train,yt_train)

y_pred_train = ridge.predict(Xt_train)
print(r2_score(yt_train,y_pred_train))

y_pred_test = ridge.predict(Xt_test)
print(r2_score(yt_test,y_pred_test))
```

```
0.9009344287345438
0.48139611165588336
```

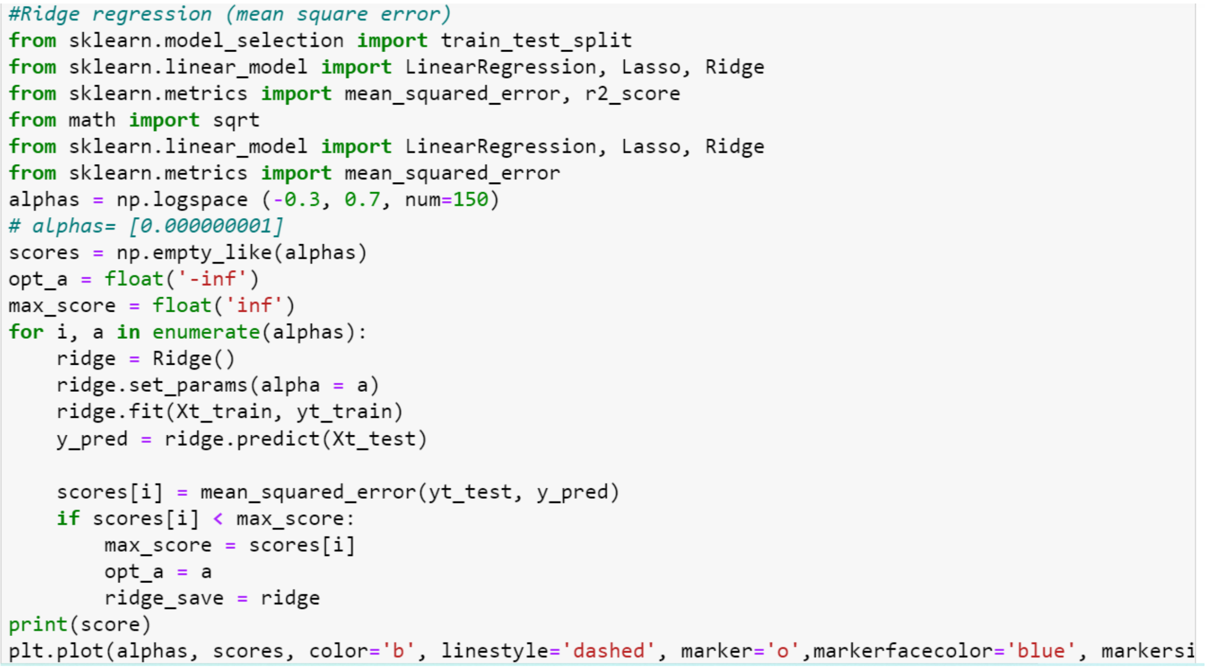
USING GRIDSEARCH CV FOR HYPER-PARAMETER TUNING:



```
folds = KFold(n_splits=10,shuffle=True,random_state=42)

hyper_param = {'alpha':[0.001, 0.01, 0.1,1.0, 5.0, 10.0,20.0]}

model = Lasso()

model_cv = GridSearchCV(estimator = model,
                        param_grid=hyper_param,
                        scoring='r2',
                        cv=folds,
                        verbose=1,
                        return_train_score=True
                       )

model_cv.fit(Xt_train,yt_train)
```

```
Fitting 10 folds for each of 7 candidates, totalling 70 fits
```

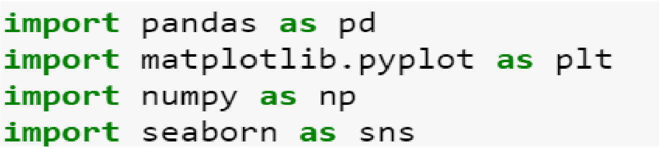
Now let’s check for Lasso



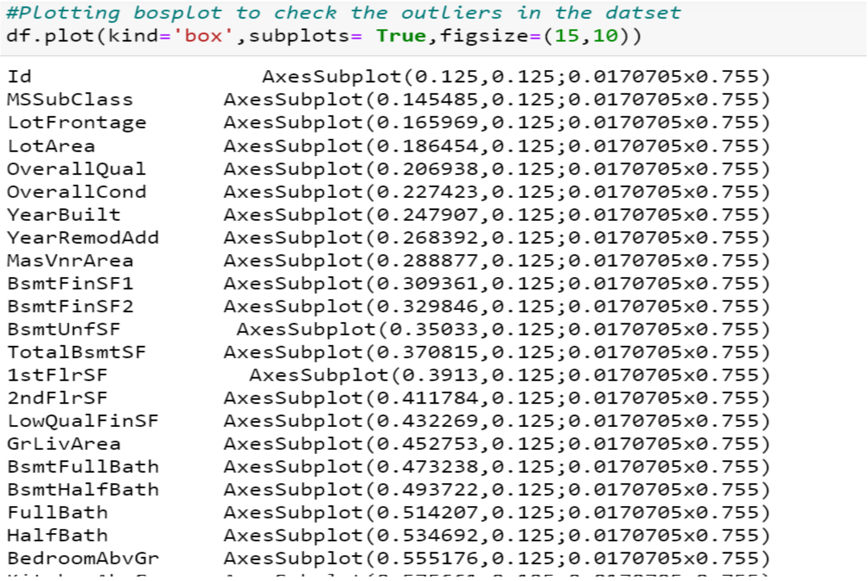
```
#Ridge regression (mean square error)
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.metrics import mean_squared_error, r2_score
from math import sqrt
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.metrics import mean_squared_error
alphas = np.logspace (-0.3, 0.7, num=150)
# alphas= [0.000000001]
scores = np.empty_like(alphas)
opt_a = float('-inf')
max_score = float('inf')
for i, a in enumerate(alphas):
    ridge = Ridge()
    ridge.set_params(alpha = a)
    ridge.fit(Xt_train, yt_train)
    y_pred = ridge.predict(Xt_test)

    scores[i] = mean_squared_error(yt_test, y_pred)
    if scores[i] < max_score:
        max_score = scores[i]
        opt_a = a
        ridge_save = ridge
print(score)
plt.plot(alphas, scores, color='b', linestyle='dashed', marker='o',markerfacecolor='blue', markersi
```

Visualizations (EDA) & Interpretation of Results:

We will be using Libraries such as Seaborn, matplotlib, plotly. At first we will be importing the libraries so that we can use visualizations for our Dataset.



```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

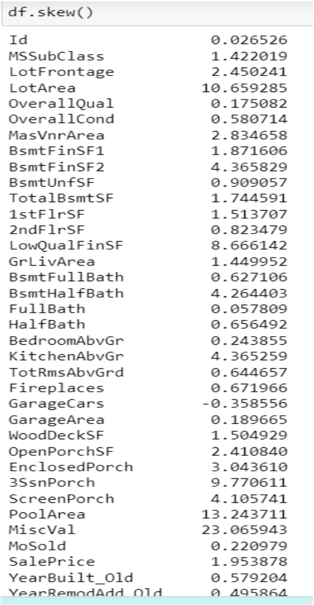


```
#Plotting bosplot to check the outliers in the datset
df.plot(kind='box',subplots= True,figsize=(15,10))
```

```
Id                 AxesSubplot(0.125,0.125;0.0170705x0.755)
MSSubClass       AxesSubplot(0.145485,0.125;0.0170705x0.755)
LotFrontage      AxesSubplot(0.165969,0.125;0.0170705x0.755)
LotArea          AxesSubplot(0.186454,0.125;0.0170705x0.755)
OverallQual      AxesSubplot(0.206938,0.125;0.0170705x0.755)
OverallCond      AxesSubplot(0.227423,0.125;0.0170705x0.755)
YearBuilt        AxesSubplot(0.247907,0.125;0.0170705x0.755)
YearRemodAdd     AxesSubplot(0.268392,0.125;0.0170705x0.755)
MasVnrArea       AxesSubplot(0.288877,0.125;0.0170705x0.755)
BsmtFinSF1       AxesSubplot(0.309361,0.125;0.0170705x0.755)
BsmtFinSF2       AxesSubplot(0.329846,0.125;0.0170705x0.755)
BsmtUnfSF         AxesSubplot(0.35033,0.125;0.0170705x0.755)
TotalBsmtSF      AxesSubplot(0.370815,0.125;0.0170705x0.755)
1stFlrSF           AxesSubplot(0.3913,0.125;0.0170705x0.755)
2ndFlrSF         AxesSubplot(0.411784,0.125;0.0170705x0.755)
LowQualFinSF     AxesSubplot(0.432269,0.125;0.0170705x0.755)
GrLivArea        AxesSubplot(0.452753,0.125;0.0170705x0.755)
BsmtFullBath     AxesSubplot(0.473238,0.125;0.0170705x0.755)
BsmtHalfBath     AxesSubplot(0.493722,0.125;0.0170705x0.755)
FullBath         AxesSubplot(0.514207,0.125;0.0170705x0.755)
HalfBath         AxesSubplot(0.534692,0.125;0.0170705x0.755)
BedroomAbvGr     AxesSubplot(0.555176,0.125;0.0170705x0.755)
```

So we can see a lot of outliers in most of the variables, which we will deal later by using median method to replace the outliers.

Visualisation for some Independent Variables along with their skewness:



```
df.skew()
```

```
Id                  0.026526
MSSubClass          1.422019
LotFrontage         2.450241
LotArea            10.659285
OverallQual         0.175082
OverallCond         0.580714
MasVnrArea          2.834658
BsmtFinSF1          1.871606
BsmtFinSF2          4.365829
BsmtUnfSF           0.909057
TotalBsmtSF         1.744591
1stFlrSF            1.513707
2ndFlrSF            0.823479
LowQualFinSF        8.666142
GrLivArea           1.449952
BsmtFullBath        0.627106
BsmtHalfBath        4.264403
FullBath            0.057809
HalfBath            0.656492
BedroomAbvGr        0.243855
KitchenAbvGr        4.365259
TotRmsAbvGrd        0.644657
Fireplaces          0.671966
GarageCars         -0.358556
GarageArea          0.189665
WoodDeckSF          1.504929
OpenPorchSF         2.410840
EnclosedPorch       3.043610
3SsnPorch           9.770611
ScreenPorch         4.105741
PoolArea           13.243711
MiscVal            23.065943
MoSold              0.220979
SalePrice           1.953878
YearBuilt_Old       0.579204
YearRemodAdd_Old    0.495864
```

So from the above we can see that many of the variables are rightly skewed which we will fix when we are treating outliers.

Key Findings of the Study My Key Findings:

After comparing both models we can see that the below are the best features of the Dataset MiscVal: $Value of miscellaneous feature

BsmtHalfBath: Basement half bathrooms

LowQualFinSF: Low quality finished square feet (all floors)

BsmtFullBath: Basement full bathrooms

HalfBath: Half baths above grade

Best alpha value for Lasso: {'alpha': 0.001}

Best alpha value for Ridge: {'alpha': 0.9}

Learning Outcomes of the Study in respect of Data Science

The amount and complexity of information produced in science, engineering, business, and everyday human activity is increasing at staggering rates. Good visualizations not only present a visual interpretation of data, but do so by improving comprehension, communication, and decision making. The importance of visualization is a topic taught to almost every data scientist in an entry-level course at university but is mastered by very few individuals. It is often regarded as obvious or unimportant due to its inherently subjective nature. In this article, I hope to dispel some of those thoughts and show you that visualization is incredibly important, not just in the field of data science, but for communicating any form of information.

Visualisation through Seaborn and many other libraries helped me in gaining insight of the Dataset. –

It also helped me in gaining insight of outliers through boxplot method.

It helped me in finding skewness of a particular variable with the help of seaborn.

Univariate Analysis of the target variable showed us that it is a highly imbalanced Dataset.

Multivariate Analysis showed us the relation between all the variables with each other.

Limitations of this work and Scope for Future Work

What are the limitations of this solution provided the future scope? The most important task for contractor of a property is to find the right price for their property. An accurate prediction can help in balancing risk for the contractor. - There are some limitations of the solutions provided by me as I do not have access to powerful laptops, for that I have to use algorithms which consumes less power and memory. The Future scope for this is infinite; the manual method which is currently used in the market is out dated and has high risk. So as to overcome this fault, there is a need for an updated and automated system. Data mining algorithms can be used to help investors to invest in an appropriate estate according to their mentioned requirements. Also the new system will be cost and time efficient. This will have simple operations.

Conclusions

In today's real estate world, it has become tough to store such huge data and extract them for one's own requirement. Also, the extracted data should be useful. The system makes optimal use of these advanced regression Algorithms. The system makes use of such data in the most efficient way. The algorithm helps to fulfil customers by increasing the accuracy of estate choice and reducing the risk of investing in an estate. A lot's of features that could be added to make the system more widely acceptable. One of the major future scopes is adding estate database of more cities which will provide the user to explore more estates and reach an accurate decision. More factors like recession that affect the house prices shall be added. In-depth details of every property will be added to provide ample details of a desired estate. This will help the system to run on a larger level.

Thank you

****