INDEX

Chapter 1 : Introduction

About Data Science:

Data science combines the scientific method, math and statistics, specialized programming, advanced analytics, AI, and even storytelling to uncover and explain the business insights buried in data. Data science encompasses preparing data for analysis and processing, performing advanced data analysis, and presenting the results to reveal patterns and enable stakeholders to draw informed conclusions.

ABOUT PROJECT:

Buying a house is a stressful thing. Buyers are not generally aware of the factors that affect the house prices. Many problems are faced during buying a house and hence real estate agents are trusted between buyers and sellers as well as laying down a legal contract for the transfer. This just creates a middle man and increases the cost of houses.

They believe it depends upon:

- Square foot area
- Neighbourhood
- The no. of bedrooms.

But it depends upon other factors as well , such as:

- Area Outside the house
- No. of storeys
- No. of rooms on one floor

For so many years there is one thing that it's that housing and rental prices continue to rise. Since the housing crisis of 2008, housing prices have recovered remarkably well, especially in major housing markets. However, in the 4th quarter of 2016, I was surprised to read that housing prices had fallen the most in the last 4 years. In fact, median resale prices for condos and coops fell 6.3%, marking the first time there was a decline since Q1 of 2017. The decline has been partly attributed to political uncertainty domestically and abroad and the 2014 election. So, to maintain the transparency among customers and also the comparison can be made easy through this model. The average annual price change across 56 countries and territories was recorderd at 10.3%, according to the report.

- If customer finds the price of house at some given website higher than the price predicted by the model, so he can reject that house.

Chapter 2: Background

Problem Statement

Buying a house is a stressful thing. Buyers are not generally aware of the factors that affect the house prices. Many problems are faced during buying a house and hence a real estate agents are trusted between buyers and sellers as well as laying down a legal contract for the transfer. This just creates a middle man and increases the cost of houses.

Thousands of houses are sold every day.

There are some questions every buyer asks himself like :

- ► What is the actual price that this house deserves ?
- ► Am I paying a fair price ?

We are doing this for buyer welfare because they are facing so much difficulty in buying house. A house value is simply more than location and square footage. Like the features that make you a person, an educated party would want to know all aspects that give a house its value. For example, you want to sell a house and you don't know the price which you can take it can't be too low or too high, To find house price you usually try to find similar properties in your neighborhood and based on gathered data you will try to assess your house price.

Objectives:

To predict the price of house which is not too high or too low using regression models. The project challenges to predict the final price of each house.

Chapter 2: Background

Data set explanation

Explain the data set you will be using in your project

The dataset we are going to use in the present project is “House-Price-Prediction Cleaned Dataset” chose from the website called Kaggle.

The dataset we chose contains 1460 Rows and 32 Columns.

Our Predictions will be based on features like

- Overall quality, condition
- Lotarea etc.

Algorithm used on Dataset

Dataset Used	Algorithm Used	Accuracy
House price prediction cleaned dataset	XGBoost Regressor	89%

Key Features

- Proposed system is a online browser based application.
- The major objective of the system is house price prediction.
- Proposed system uses the parametres such as house size , balcony , number of bedrooms and bathrooms , location and some other parametres for house price prediction.
- System uses Machine Learning algorithms for price prediction.
- System helps real estate in faster decision making.
- System is useful for real estate bussiness and also for buyers and sellers.

Chapter 3: Proposed Framework

Software Required:

Technology Used

Machine Learning

Tool Used

- Google colaboratory
- Python

Libraries Used

- Pandas
- Numpy
- Matplotlib

Pseudo Code:

The steps are used to implement the house price prediction model is depicted as pseudo code

Pseudo Code: House price prediction

Step 1: Load the datasets.

Step 2: Summarize the data distribution range using Visualization tool.

Step 3: Identify the prevalent features using correlation tool.

Step 4: Split the input dataset into train and test.

Step 5: Transform the data to fed into machine learning models.

Step 6: Invoke XGBoost Regressor.

Step 7: Summarize the performance in terms rating and strength of a models using metrics.

CODE AND SIMULATION

Importing the Dependencies

```
[ ] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
    import sklearn.datasets
    from sklearn.model_selection import train_test_split
    from xgboost import XGBRegressor
    from sklearn import metrics
```

importing the house price dataset from kaggle

```
[ ] house_price_dataset = pd.read_csv("/content/House-Price-Prediction-clean.csv")
```

```
[ ] print(house_price_dataset)
```

```
          Id  MSSubClass  LotArea  OverallQual  OverallCond  YearBuilt  \
0          1          60     8450            7            5       2003
1          2          20     9600            6            8       1976
2          3          60    11250            7            5       2001
3          4          70     9550            7            5       1915
4          5          60    14260            8            5       2000
...      ...         ...      ...          ...          ...        ...
1455  1456          60     7917            6            5       1999
1456  1457          20    13175            6            6       1978
1457  1458          70     9042            7            9       1941
1458  1459          20     9717            5            6       1950
1459  1460          20     9937            5            6       1965

      YearRemodAdd  BsmtFinSF1  BsmtUnfSF  TotalBsmtSF  ...  WoodDeckSF  \
0             2003         706        150          856  ...           0
1             1976         978        284         1262  ...         298
2             2002         486        434          920  ...           0
3             1970         216        540          756  ...           0
4             2000         655        490         1145  ...         192
...            ...         ...        ...          ...  ...         ...
1455          2000           0        953          953  ...           0
```

```
1456        1988      790      589      1542  ...      349
1457        2006      275      877      1152  ...        0
1458        1996       49        0      1078  ...      366
1459        1965      830      136      1256  ...      736

      OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea  MiscVal  \
0              61              0          0            0         0        0
1               0              0          0            0         0        0
2              42              0          0            0         0        0
3              35            272          0            0         0        0
4              84              0          0            0         0        0
...           ...            ...        ...          ...       ...      ...
1455           40              0          0            0         0        0
1456            0              0          0            0         0        0
1457           60              0          0            0         0     2500
1458            0            112          0            0         0        0
1459           68              0          0            0         0        0

      MoSold  YrSold  SalePrice
0          2    2008     208500
1          5    2007     181500
2          9    2008     223500
3          2    2006     140000
4         12    2008     250000
...      ...     ...        ...
1455       8    2007     175000
1456       2    2010     210000
1457       5    2010     266500
1458       4    2010     142125
1459       6    2008     147500
```

```python
#print first 5 rows
house_price_dataset.head()
```

	Id	MSSubClass	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd
0	1	60	8450	7	5	2003	2003
1	2	20	9600	6	8	1976	1976
2	3	60	11250	7	5	2001	2002
3	4	70	9550	7	5	1915	1970
4	5	60	14260	8	5	2000	2000

5 rows × 32 columns

```python
#add the target (price) column to the Dataframe
house_price_dataset['price']=house_price_dataset.SalePrice
```

```
[ ] house_price_dataset.head()
```

Porch	3SsnPorch	ScreenPorch	PoolArea	MiscVal	MoSold	YrSold	SalePrice
0	0	0	0	0	2	2008	208500
0	0	0	0	0	5	2007	181500
0	0	0	0	0	9	2008	223500
272	0	0	0	0	2	2006	140000
0	0	0	0	0	12	2008	250000

```
#printing the no. of rows and columns
house_price_dataset.shape
```

```
(1460, 33)
```

```
#checking the missing values
house_price_dataset.isnull().sum()
```

```
Id               0
MSSubClass       0
LotArea          0
OverallQual      0
OverallCond      0
YearBuilt        0
YearRemodAdd     0
BsmtFinSF1       0
BsmtUnfSF        0
TotalBsmtSF      0
1stFlrSF         0
2ndFlrSF         0
GrLivArea        0
BsmtFullBath     0
FullBath         0
HalfBath         0
BedroomAbvGr     0
KitchenAbvGr     0
TotRmsAbvGrd     0
Fireplaces       0
GarageCars       0
```

```
WoodDeckSF       0
OpenPorchSF      0
EnclosedPorch    0
3SsnPorch        0
ScreenPorch      0
```

```
# statistical measures of the dataset
house_price_dataset.describe()
```

	Id	MSSubClass	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	BsmtFinSF1
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000
mean	730.500000	56.897260	10516.828082	6.099315	5.575342	1971.267808	1984.865753	443.639726
std	421.610009	42.300571	9981.264932	1.382997	1.112799	30.202904	20.645407	456.098091
min	1.000000	20.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.000000
25%	365.750000	20.000000	7553.500000	5.000000	5.000000	1954.000000	1967.000000	0.000000
50%	730.500000	50.000000	9478.500000	6.000000	5.000000	1973.000000	1994.000000	383.500000
75%	1095.250000	70.000000	11601.500000	7.000000	6.000000	2000.000000	2004.000000	712.250000
max	1460.000000	190.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	5644.000000

8 rows × 33 columns

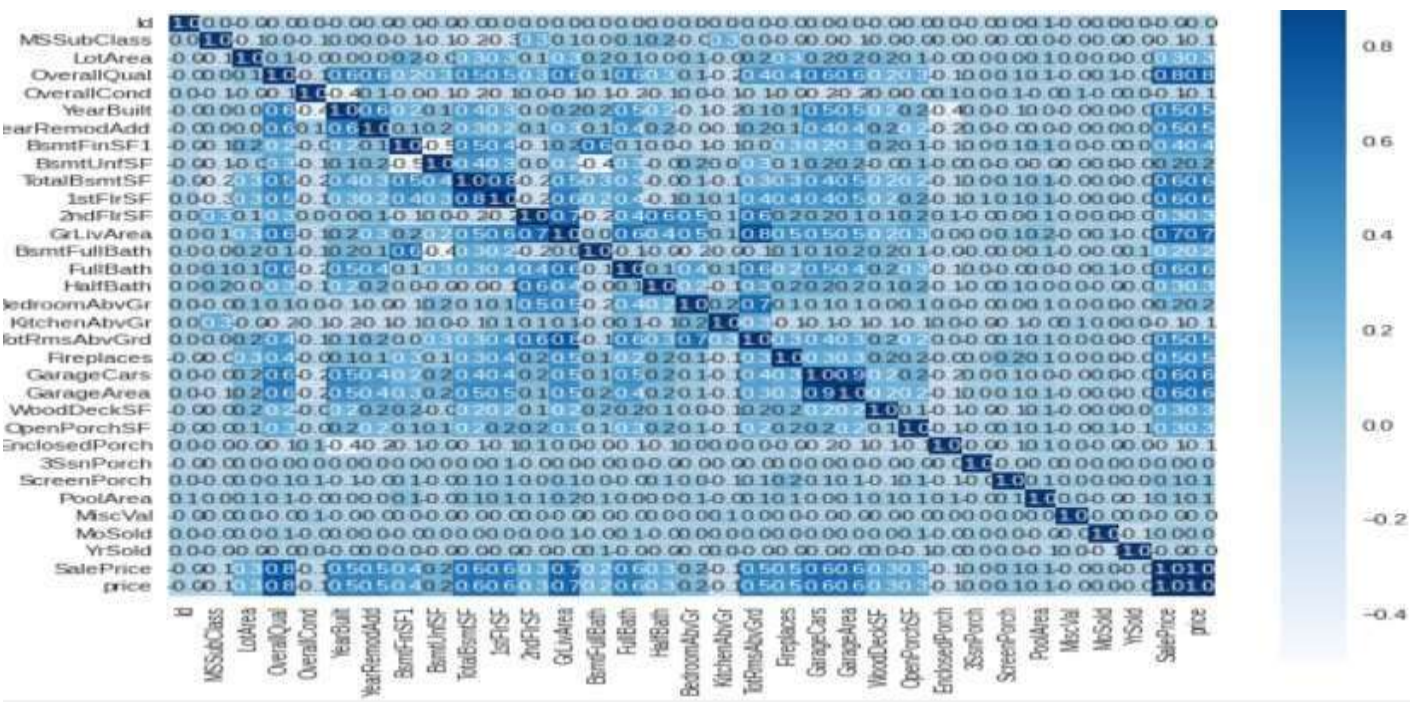
understanding the correlation between various features in the dataset

1. Positive correlation
2. Negative correlation

```
[ ]  correlation = house_price_dataset.corr()
```

```
[ ]  # constructing a heatmap to understand the correlation
     plt.figure(figsize=(10,10))
     sns.heatmap(correlation,cbar = True,square = True ,fmt= '.1f',annot = True,annot_kws={'size':10},cmap= 'Blues'
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f66f998ab50>
```



splitting the data and target

```
[1] #x = house_price_dataset.drop([0],axis = 0)
    #y = house_price_dataset['price']
```

```
#print(x)
#print(y)
```

```
[ ] x = house_price_dataset.drop(['price'],axis = 1)
    y = house_price_dataset['price']
```

```
[ ] print(x)
    print(y)
```

```
        Id  MSSubClass  LotArea  OverallQual  OverallCond  YearBuilt  \
0        1          60     8450            7            5       2003
1        2          20     9600            6            8       1976
2        3          60    11250            7            5       2001
3        4          70     9550            7            5       1915
4        5          60    14260            8            5       2000
...    ...         ...      ...          ...          ...        ...
1455  1456          60     7917            6            5       1999
1456  1457          20    13175            6            6       1978
1457  1458          70     9042            7            9       1941
1458  1459          20     9717            5            6       1950
1459  1460          20     9937            5            6       1965
```

Splitting the data into training set and test set

```
[ ] x_train ,x_test,y_train, y= train_test_split(x,y,test_size = 0.2,random_state=2)
```

```
[ ] print(x.shape,x_train.shape,x_test.shape)

    (1460, 32) (1168, 32) (292, 32)
```

Model Training

XGBoost Regression

```
[ ] # loading the model
    model = XGBRegressor()
```

```
# training the model with x_train
model.fit(x_train,y_train)
```

```
[08:48:44] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0,
             importance_type='gain', learning_rate=0.1, max_delta_step=0,
             max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
```

Evaulation

prediction on training data

```
[ ] #accuracy for prediction on training model data
    training_data_prediction=model.predict(x_train)
```

```
[ ] print(training_data_prediction)
```

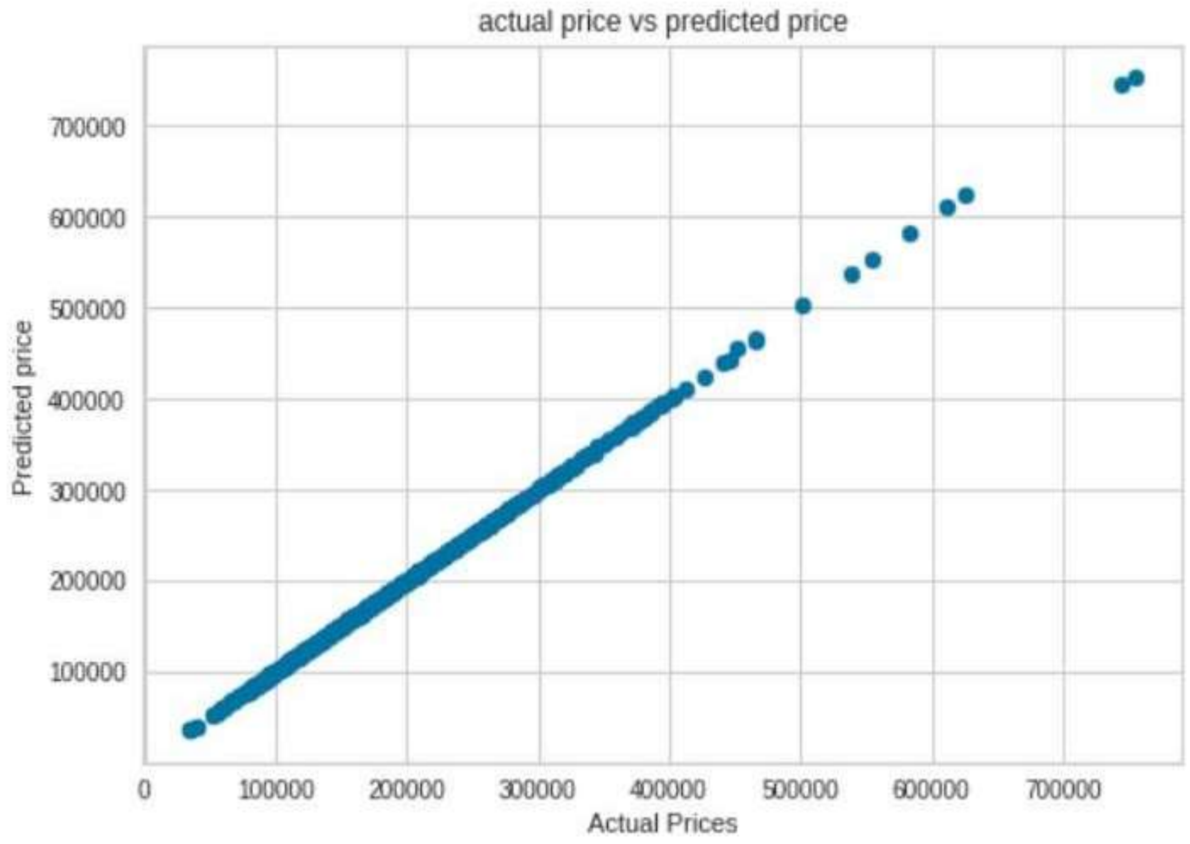
```
[ ] #r squared error
    score_1=metrics.r2_score(y_train,training_data_prediction)
    #mean absolute error
    score_2=metrics.mean_absolute_error(y_train,training_data_prediction)

    print("R square error:",score_1)
    print("mean absolute error:",score_2)
```

```
R square error: 0.9999336479149868
mean absolute error: 445.80381795804794
```

Prediction on test data

visualizing the actual prices and predicted price

```
[ ] plt.scatter(y_train,training_data_prediction)
    plt.xlabel("Actual Prices")
    plt.ylabel("Predicted price")
    plt.title("actual price vs predicted price")
    plt.show()
```




```
#accuracy for prediction on test model data
test_data_prediction=model.predict(x_test)
```

```
[ ] #r squared error
    score_1=metrics.r2_score(y,test_data_prediction)
    #mean absolute error
    score_2=metrics.mean_absolute_error(y,test_data_prediction)

    print("R square error:",score_1)
    print("mean absolute error:",score_2)

    R square error: 0.9997083836488491
    mean absolute error: 676.9227846746576
```

```
[ ] pip install pycaret
```

```
[ ] from pycaret.regression import *
    model_setup=setup(data=house_price_dataset,target='SalePrice',silent=True)
    cm=compare_models()
```

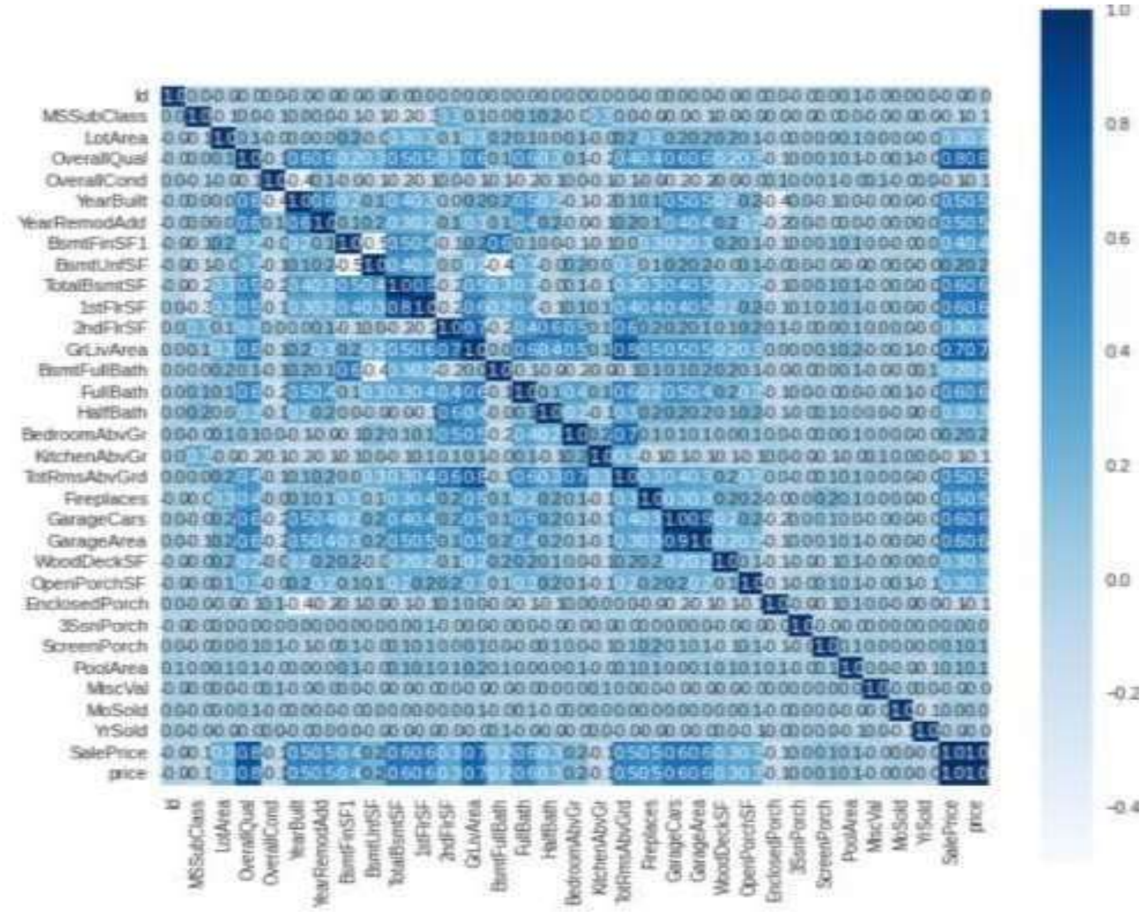
	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
gbr	Gradient Boosting Regressor	1.754217e+04	8.023047e+08	2.741774e+04	8.751000e-01	0.1449	1.055000e-01	0.393
lightgbm	Light Gradient Boosting Machine	1.838153e+04	9.736401e+08	3.059019e+04	8.453000e-01	0.1519	1.080000e-01	0.189
rf	Random Forest Regressor	1.866265e+04	9.960439e+08	3.055201e+04	8.402000e-01	0.1553	1.111000e-01	1.203
et	Extra Trees Regressor	2.010975e+04	1.004066e+09	3.107752e+04	8.387000e-01	0.1643	1.206000e-01	1.005
ada	AdaBoost Regressor	2.682064e+04	1.406433e+09	3.698439e+04	7.749000e-01	0.2170	1.786000e-01	0.244
omp	Orthogonal Matching Pursuit	1.934358e+04	1.300021e+09	3.373518e+04	7.662000e-01	0.1605	1.177000e-01	0.017
llar	Lasso Least Angle Regression	1.834798e+04	1.314280e+09	3.370806e+04	7.655000e-01	0.1536	1.072000e-01	0.045
lasso	Lasso Regression	1.845972e+04	1.318754e+09	3.376184e+04	7.647000e-01	0.1551	1.084000e-01	0.071
ridge	Ridge Regression	1.918466e+04	1.406448e+09	3.524982e+04	7.484000e-01	0.1587	1.115000e-01	0.018
lr	Linear Regression	2.166457e+04	1.559880e+09	3.740996e+04	7.270000e-01	0.1956	1.313000e-01	0.367
dt	Decision Tree Regressor	2.719511e+04	1.950058e+09	4.304578e+04	6.790000e-01	0.2284	1.602000e-01	0.031
en	Elastic Net	2.312687e+04	1.862486e+09	4.078307e+04	6.674000e-01	0.1804	1.339000e-01	0.078
br	Bayesian Ridge	2.483421e+04	2.061659e+09	4.318555e+04	6.320000e-01	0.1929	1.428000e-01	0.034
knn	K Neighbors Regressor	3.067496e+04	2.445849e+09	4.843479e+04	6.247000e-01	0.2314	1.792000e-01	0.028
huber	Huber Regressor	2.835213e+04	2.311750e+09	4.654043e+04	5.928000e-01	0.2206	1.691000e-01	0.173
par	Passive Aggressive Regressor	4.478774e+04	4.441206e+09	6.242942e+04	2.181000e-01	0.3581	2.676000e-01	0.025
dummy	Dummy Regressor	5.780656e+04	6.410081e+09	7.939172e+04	-7.700000e-03	0.4099	3.692000e-01	0.013
lar	Least Angle Regression	6.090247e+23	3.713769e+50	6.096078e+24	-5.656227e+40	24.0498	4.512275e+18	0.056

gbr regression is the best model when compared to other models because its r2 value is 0.875

Chapter 4: Results

1.Distribution plot

2.Heat Map:



3.Error:

```
#r squared error
score_1=metrics.r2_score(y_train,training_data_prediction)
#mean absolute error
score_2=metrics.mean_absolute_error(y_train,training_data_prediction)

print("R square error:",score_1)
print("mean absolute error:",score_2)
```

```
R square error: 0.9999336479149868
mean absolute error: 445.80381795804794
```

Chapter 5: Conclusion and Future Scope

In order to purchase a house accurate estimation of house price is necessary. A house property contains various factors. In order to predict house prices, machine learning algorithms are considered to be efficient techniques. This paper provides insight of XGBoost algorithm as one of the useful algorithm for house price prediction and to provide flexible and efficient results. Dataset used for the experiment was obtained from Kaggle. Features includes LOT, Coast, number of bedrooms, bathrooms, price. Model accuracy and Mean absolute error are being calculated using XGBoost algorithm. Observations made from the obtained results shows that compared to all models used in predictions, XGBoost model out performed and provided with high rate of accuracy.

This experiment of predicting house price has been developed using XGBoost algorithm on python notebook. XGBoost is an application of gradient boosting decision tree algorithm. It was designed to push the computational limits of boosted tree algorithm. Idea of selecting optimized distributed gradient boosting library is being its fast and flexible nature and best used for tabular dataset and classification and regression model. XGBoost allows parallel processing that makes it 10 times faster than other models.

In order to predict house prices several metrics are used such as feature selection. This dataset comprise of 23 features and 21613 records. Feature selection is a procedure used in this process that required to manually or automatically selecting attributes that contributes to predict variable. Initially this feature selection technique is used in preprocessing stage where number of features are omitted on the basis of their less association with predicting attribute. Later on while implementing XGBoost model more feature were dropped to assure the efficient results. Data set is being tested using various testing and training ratios to obtained multiple Model accuracy values and Mean Absolute errors.

References:

1.https://ieeexplore.ieee.org/abstract/docum ent/8882834/

2. https://link.springer.com/article/10.1007/s 11146-007-9036-8

3. https://www.divaportal.org/smash/get/diva 2:1456610/FULLTEXT01.pdf

4.http://103.47.12.35/bitstream/handle/1/96 51/BT3083_RPT%20-%20Amit%20Kumar.pdf?sequence=1&isAllow ed=y

5. https://arxiv.org/abs/1709.08432

6.https://www.sciencedirect.com/science/art icle/pii/S0957417414007325

7.https://www.cse.ust.hk/~rossiter/independ ent_studies_projects/real_estate_prediction/ real_estate_report.pdf

8.https://ieeexplore.ieee.org/abstract/docum ent/8962443/

9. https://www.researchgate.net/publication/ 347584803_House_Price_Prediction_using_a _Machine_Learning_Model_A_Survey_of_Lite rature

10. https://kalaharijournals.com/resources/AP RIL_15.pdf