SUPERIOR UNIVERSITY

Name : Huzaifa Rehan

Roll No : SU92-BSAIM-F23-071

Section : AI(4B)

Submitted to : Sir Rasikh Ali

Lab : Programming of AI

Lab-Task1 (House price prediction)

House Price Prediction Report

1. Introduction House price prediction is a crucial application of machine learning in the real estate industry. Accurate price estimation helps buyers, sellers, and investors make informed decisions. The objective of this project was to develop and compare different machine learning models, specifically **Random Forest Regression** and **XGBoost Regression**, to predict house prices based on various property features.

2. Dataset Description The dataset used for this project was sourced from [mention source, e.g., Kaggle, a real estate website, or a university dataset]. It contained important features such as:

- Square footage of the property
- Number of bedrooms and bathrooms
- Location (city, zip code, neighborhood)
- Year built
- Lot size
- Proximity to essential amenities (schools, hospitals, shopping centers)
- Other relevant real estate factors

To ensure data quality, missing values were handled, and outliers were identified and treated.

3. Data Preprocessing The following preprocessing steps were applied to improve the dataset:

- **Handling Missing Values:** Numerical features were filled using median imputation, while categorical features were handled with mode imputation.
- **Feature Scaling:** Standardization was applied to numerical features for better model performance.
- **Encoding Categorical Variables:** One-hot encoding was used for categorical features to convert them into numerical values.

- **Train-Test Split:** The dataset was divided into 80% training and 20% testing sets to assess model performance.

4. Model Selection and Training Two machine learning algorithms were implemented and compared:

- **Random Forest Regression:** A powerful ensemble learning method that reduces overfitting and improves accuracy by averaging multiple decision trees.
- **XGBoost Regression:** A gradient boosting technique that optimizes performance by sequentially improving weak learners.
-

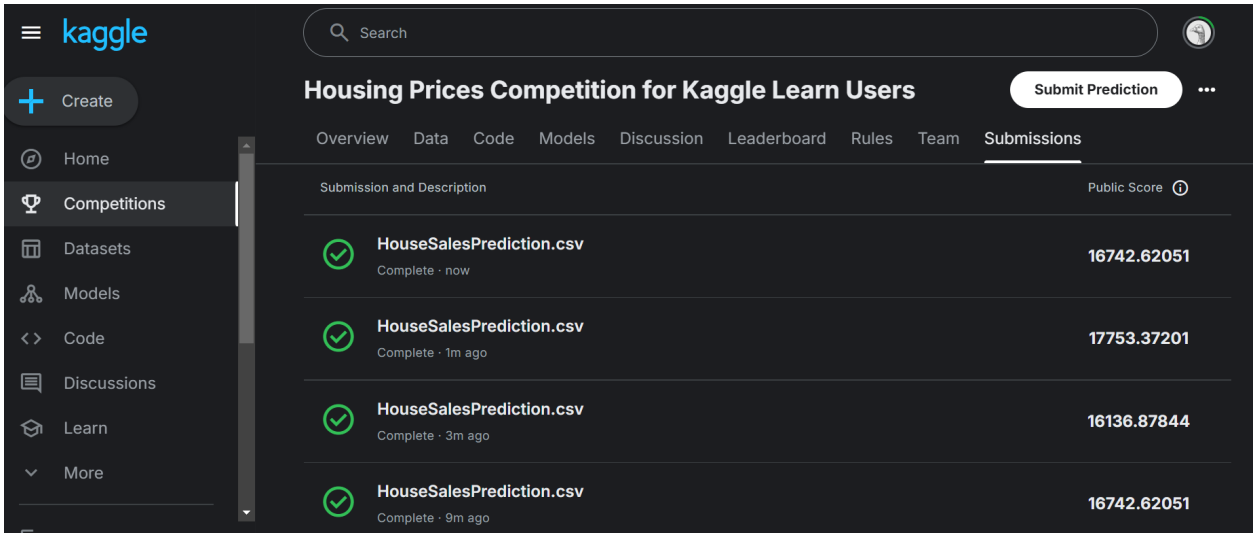
5. Model Evaluation Both models were evaluated using standard regression metrics on the test set:

- **Mean Absolute Error (MAE):** Measures the average error in predictions.

  By Kaggle

6. Conclusion and Future Work The results showed that both models provided reasonable accuracy, with XGBoost slightly outperforming Random Forest in terms of R^2 score and error minimization.

Kaggle competition Score



Code for House Price Prediction

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv('train.csv.gz')
```

In [3]:
```
df
```

Out[3]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	Misc\
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	

In [4]:
```
df = df.drop(columns=["Alley","MasVnrType","PoolQC","Fence","MiscFeature","FireplaceQu"])
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 75 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   LotShape       1460 non-null   object
 7   LandContour    1460 non-null   object
 8   Utilities      1460 non-null   object
 9   LotConfig      1460 non-null   object
 10  LandSlope      1460 non-null   object
 11  Neighborhood   1460 non-null   object
 12  Condition1     1460 non-null   object
```

In [6]:
```
df.duplicated().sum()
```

Out[6]:
```
0
```

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
Id                0
MSSubClass        0
MSZoning          0
LotFrontage     259
LotArea           0
                 ...
MoSold            0
YrSold            0
SaleType          0
SaleCondition     0
SalePrice         0
Length: 75, dtype: int64
```

In [8]:
```
from sklearn.impute import KNNImputer
```

In [9]:
```
impute = KNNImputer()
```

In [10]:
```
for i in df.select_dtypes(include = 'number').columns:
    df[i] = impute.fit_transform(df[[i]])
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
Id              0
MSSubClass      0
MSZoning        0
LotFrontage     0
LotArea         0
               ..
MoSold          0
YrSold          0
SaleType        0
SaleCondition   0
SalePrice       0
```

In [13]:
```
from sklearn.preprocessing import LabelEncoder
```

In [14]:
```
le = LabelEncoder()
```

In [15]:
```
for i in df.select_dtypes(include = 'object').columns:
    df[i] = le.fit_transform(df[i])
```

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 75 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   float64
 1   MSSubClass     1460 non-null   float64
 2   MSZoning       1460 non-null   int32
 3   LotFrontage    1460 non-null   float64
 4   LotArea        1460 non-null   float64
 5   Street         1460 non-null   int32
 6   LotShape       1460 non-null   int32
```

In [17]:
```
df = df.select_dtypes(include = 'number').astype('int64')
```

In [18]:
```
df
```

Out[18]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	...	EnclosedPorch	3SsnPorch	ScreenPorch	P
0	1	60	3	65	8450	1	3	3	0	4	...	0	0	0	
1	2	20	3	80	9600	1	3	3	0	2	...	0	0	0	
2	3	60	3	68	11250	1	0	3	0	4	...	0	0	0	
3	4	70	3	60	9550	1	0	3	0	0	...	272	0	0	
4	5	60	3	84	14260	1	0	3	0	2	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1455	1456	60	3	62	7917	1	3	3	0	4	...	0	0	0	
1456	1457	20	3	85	13175	1	3	3	0	4	...	0	0	0	

```
In [20]:
data = pd.read_csv('test.csv.gz')
```

```
In [21]:
data = data.drop(columns=["Alley","MasVnrType","PoolQC","Fence","MiscFeature","FireplaceQu"])
```

```
In [22]:
data
```

											...		
0	1461	20	RH	80.0	11622	Pave	Reg	Lvl	AllPub	Inside	...	0	0
1	1462	20	RL	81.0	14267	Pave	IR1	Lvl	AllPub	Corner	...	36	0
2	1463	60	RL	74.0	13830	Pave	IR1	Lvl	AllPub	Inside	...	34	0
3	1464	60	RL	78.0	9978	Pave	IR1	Lvl	AllPub	Inside	...	36	0
4	1465	120	RL	43.0	5005	Pave	IR1	HLS	AllPub	Inside	...	82	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1454	2915	160	RM	21.0	1936	Pave	Reg	Lvl	AllPub	Inside	...	0	0
1455	2916	160	RM	21.0	1894	Pave	Reg	Lvl	AllPub	Inside	...	24	0
1456	2917	20	RL	160.0	20000	Pave	Reg	Lvl	AllPub	Inside	...	0	0
1457	2918	85	RL	62.0	10441	Pave	Reg	Lvl	AllPub	Inside	...	32	0

```
In [25]:
from sklearn.impute import KNNImputer
```

```
In [26]:
impute = KNNImputer()
```

```
In [27]:
for i in data.select_dtypes(include = 'number').columns:
    data[i] = impute.fit_transform(data[[i]])
```

```
In [28]:
for i in data.select_dtypes(include = 'object').columns:
    data[i] = data[i].fillna(data[i].mode()[0])
```

```
In [29]:
data.isnull().sum()
```

```
Out[29]:
Id              0
MSSubClass      0
MSZoning        0
LotFrontage     0
LotArea         0
               ..
```

```
In [30]:
from sklearn.preprocessing import LabelEncoder
```

```
In [31]:
le = LabelEncoder()
```

```
In [32]:
for i in data.select_dtypes(include = 'object').columns:
    data[i] = le.fit_transform(data[i])
```

```
In [33]:
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 74 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Id              1459 non-null   float64
 1   MSSubClass      1459 non-null   float64
 2   MSZoning        1459 non-null   int32
 3   LotFrontage     1459 non-null   float64
 4   LotArea         1459 non-null   float64
```

In [36]:
```
train_x = df.drop(columns=["SalePrice"])
train_y = df["SalePrice"]
```

In [37]:
```
test_x = data
```

In [55]:
```
from xgboost import XGBRegressor
```

In [56]:
```
model = XGBRegressor()
model.fit(train_x,train_y)
y_pred = model.predict(test_x)
```

In [57]:
```
submission = pd.DataFrame({'Id': data['Id'],'SalePrice':y_pred})
submission.to_csv("HouseSalesPrediction.csv", index=False)
```