HOUSE PRICE PREDICTION USING MACHINE LEARNING

Introduction

House price prediction can help the developer determine the selling price of a house and can help the customer to arrange the right time to purchase a house. There are three factors that influence the price of a house which include physical conditions, concept and location. Here we have to study about Selection and training of the dataset.

A property's value is important in real estate transactions. Housing price trends are not only the concern of buyers and sellers, but it also indicates the current economic situation. Therefore, it is important to predict housing prices without bias to help both the buyers and sellers make their decisions. This project development may help to predict the house price.

• Developing an accurate prediction model for housing prices is always needed for socio-economic development and well-being of citizens.

• In this paper, a diverse set of machine learning algorithms such as XGBoost, CatBoost, Random Forest, Lasso, Voting Regressor, and others, are being employed to predict the housing prices using public available datasets.

• The housing datasets of 62,723 records from January 2015 to November 2019 is obtained from the Florida's Volusia County Property Appraiser website.

• The records are publicly available and include the real estate/economic database, maps, and other associated information.

• The database is usually updated weekly according to the State of Florida regulations. Then, the housing price prediction models using machine learning techniques are developed and their regression model performances are compared.

• Finally, an improved housing price prediction model for assisting the housing market is proposed. Particularly, a house seller/buyer or a real estate broker can get insight in making better-informed decisions considering the housing price prediction.

• The empirical results illustrate that based on prediction model performance, Coefficient of Determination (R2), Mean Square Error (MSE), Mean Absolute Error (MAE), and computational time, the XGBoost algorithm performs superior than the other models to predict the house price.

TECHNIQUES:

Housing Price Prediction(Machine Learning Algorithms, XGBoost Method,Target Binning) Case Study and Modeling Framework

•In this section, a general overview of the case study and a real-world case of house pricing problem is presented.

• In addition, this section includes the information coming from the property sales datasets of the Volusia County Property Appraiser and the proposed modeling framework to analyse the datasets. Problem Description and Research Framework:

• The primary aim of this case study is to predict housing price for the given features to maximize the prediction accuracy by employing the proposed methodology.

• This housing problem can be considered both as a regression or a classification housing problem.

• Since the classification problem was previously reported in the literature, this research considers several regression models with target variable binning which are applied on the housing market data to predict the property price.

Removing Duplications:

• For regression modeling, if some data points are replicated by being present more than once in the dataset, they are more strongly represented in the underlying data, so the regression algorithm treats them as having more importance.

• It can be thought of each occurrence of a data point as pulling the regression line towards it with the same force.

• If there are two data points at a given point in the regression model plane, they will pull the line towards them twice as hard.

• Therefore, it is indicated to remove the duplicate values from data itself.

• Duplication removal should be done carefully though, as these duplicate data points may cause what is called data leakage, when splitting the entire data into training and validation sets.

• If a data point in the training set has a duplicate value in the validation set, then the model will give biased prediction toward these duplicate data points, which is not desirable since will bias the entire prediction model.

• The duplicate entries and null entries from the housing sale transactions dataset were removed.

Feature Engineering:

• The main purpose of feature engineering is to find the most influential or partially important features of the dataset and detect the less valuable features to be removed from the dataset.

• The process results in a highly efficient and less complex model.

• Following this process requires domain expert knowledge in identifying a set of key features of the dataset and performing feature analysis, which is described in the next subsection.

Feature Analysis:

• The main purpose of feature engineering is to find the most influential or partially important features of the dataset and detect the less valuable features to be removed from the dataset.

• The process results in a highly efficient and less complex model.

• Following this process requires domain expert knowledge in identifying a set of key features of the dataset and performing feature analysis, which is described in the next subsection.

Extreme Gradient Boosting (XGBoost)

• XGBoost regression is short form for extreme gradient boost regression.

• XGBoost is a high-performance machine learning algorithm based on gradient boosting, where multiple weak prediction models, typically decision trees, are sequentially trained to correct preceding errors and create a strong predictive model.

• XGBoost introduces regularization into the objective function to control complexity, reducing overfitting.

• It also has built-in handling of missing values, learning the best imputation during training. An advantage over other boosting methods is its tree pruning techniqe which enhances efficiency by eliminating non-beneficial splits.

• Additionally, XGBoost leverages a Column Block structure for efficient sparse data handling, and employs parallel computing for speed.

It works well compared to others machine learning algorithms.XGBoost is one of the best supervised learning algorithms which can inferred by the way it flows, it consists of objective function and base learners.

• Loss function is present in objective function which shows the difference between actual values and predicted values whereas regularisation term is used for showing how far is actual value away from predicted value.

• Ensemble learning used in XGBoost considers many models which are known as base learners for predicting a single value.

• Not all base learners are expected to have bad prediction so that after summing up all of them bad prediction cancelled out by good prediction.

• A regressor is the one that fits a model using given features and predicts the unknown output value. XG Boost Regression Algorithm for House Price Prediction Input: House attributes dataset.

Output: Price of house.

1. Check input dataset for missing values and calculate d mean is replaced in place of missing value.

2. Divide attributes based on values in data fields as categorical and non-categorical rows.

3. Check Non categorical rows for outliers using outlier detection techniques and remove all outliers.

4. Convert categorical rows into binary vectors using one hot encoding.

5. Divide dataset for cross validation using train test split.

6. Apply Ensemble learning through training and combining individual models termed as base learners in order to derive a single prediction.

a) Calculate Mean Squared Error (MSE) with true values to predicted values.

b) Classify independent models as weak-learners and strong-learners using error detection.

c) Total mean cancels bad prediction with good prediction.

7. Objective function contains the loss function and regularisation term to calculate difference between actual value and predicted value.

Data pre-processing is a process used for refining data before fed into model. Data pre-processing is vaguely divided into four stages called

• Data cleaning

• Data Editing

• Data reduction

• Data wrangling

Data cleaning is process where inaccurate data or if a data field is empty, then value is filled using mean or median or entire record is deleted from data. If data is recorded manually these problems tend to happen. Calculate the mean value considering the value of attributes and number of records in the data. Data editing is process where outliers are picked from data and eradicated. Outliers are mainly recorded in data mainly due to experimental errors produced by machined due to malfunctioning or due to some other parameters.

Data reduction is termed as the process of reducing data using some kind of normalisation for easy process of data. Z score is one of processes used for normalisation. Data wrangling is termed as a process where data is transformed or mapped. Data munging, data visualisation and data aggregation comes under this process. Data visualisation is process where statistics are used for producing graphs. Data aggregation is process where data is filtered before fed into model.

Extreme Gradient Boosting (XGBoost)

- XGBoost regression is short form for extreme gradient boost regression.
- XGBoost is a high-performance machine learning algorithm based on gradient boosting, where multiple weak prediction models, typically decision trees, are sequentially trained to correct preceding errors and create a strong predictive model.
- XGBoost introduces regularization into the objective function to control complexity, reducing overfitting.
- It also has built-in handling of missing values, learning the best imputation during training. An advantage over other boosting methods is its tree pruning techniqe which enhances efficiency by eliminating non-beneficial splits.

Implementation of XGBoost

- Extreme Gradient Boosting (XGBoost) is an enhanced gradient boosting machine using the tree ensemble boosting process.
- This process ends in the sum of the outputs from all the trees.
- The XGBoost algorithm used the XGBoost package in Python to evaluate house prices.
- The sample data allocation scheme used in this model is the same as the previous model which is LightGBM algorithm.
- Analysis of the feature importance with XGBoost model shows the rank of features (in sequence, highest to lowest): size of the house, the number of parking lots provided for a house, and the nearest distance to the public transport feature.
- The selection of the features is quite different compared to the previous two models (Multiple Linear Regression and Ridge Regression).
- The proposed XGBoost model is the first application of XGBoost to the study of the Kuala Lumpur housing market.
- The model used in this analysis was able to tackle the problems of the housing market in Kuala Lumpur as the XGBoost model has a better fitting and predictive abilities.
- The XGBoost model was able to generate results that were more consistent and justifiable than other models used for housing market data.
- The XGBoost model achieved better predictive ability, with the lowest mean absolute error

(MAE) and root mean squared error (RMSE), and adjusted R-squared value closest to 1, which indicates the most accurate model.

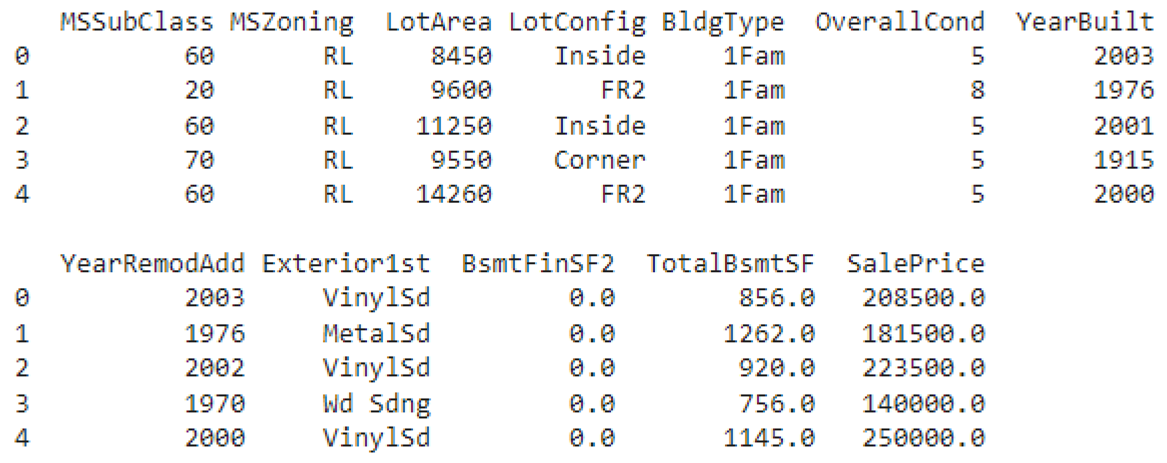
• In addition, consistent model performance was found in the XGBoost model as XGBoost outperformed other models in the training and testing R-squared value.

• The proposed XGBoost model is, therefore, effective in predicting housing prices, which favor not only future house buyers but also investors and policymakers in the real estate industry. Conclusion A model for house price prediction that assists both buyer and seller.

Code:

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


dataset = pd.read_excel("HousePricePrediction.xlsx")


# Printing first 5 records of the dataset

print(dataset.head(5))
```

Output:



```
   MSSubClass MSZoning  LotArea LotConfig BldgType  OverallCond  YearBuilt
0          60       RL     8450    Inside     1Fam            5       2003
1          20       RL     9600       FR2     1Fam            8       1976
2          60       RL    11250    Inside     1Fam            5       2001
3          70       RL     9550    Corner     1Fam            5       1915
4          60       RL    14260       FR2     1Fam            5       2000

   YearRemodAdd Exterior1st  BsmtFinSF2  TotalBsmtSF  SalePrice
0          2003     VinylSd         0.0        856.0   208500.0
1          1976     MetalSd         0.0       1262.0   181500.0
2          2002     VinylSd         0.0        920.0   223500.0
3          1970     Wd Sdng         0.0        756.0   140000.0
4          2000     VinylSd         0.0       1145.0   250000.0
```

FEATURE SELECTION :

Code:

```
# Import starting libraries
import numpy as np
import pandas as pd
import seaborn as sns
import
```

```
# Separate temporal features
feature_with_year = []

for feature in X_train.columns:
    if "Yr" in feature or "Year" in feature:
        feature_with_year.append(feature)
```

```
# Separate numerical and categorial features
categorical_features = []
numerical_features = []
discrete_features = []
continuous_features = []

for feature in X_train.columns:
    if X_train[feature].dtypes == "O":
        categorical_features.append(feature)
    else:
        numerical_features.append(feature)
        if len(X_train[feature].unique()) <= 20 and feature not in feature_with_year:
            discrete_features.append(feature)
        else:
            continuous_features.append(feature)
# Separate numerical and categorial features
categorical_features = []
numerical_features = []discrete_features = []
continuous_features = []

for feature in X_train.columns:
    if X_train[feature].dtypes == "O":
        categorical_features.append(feature)
    else:
        numerical_features.append(feature)
        if len(X_train[feature].unique()) <= 20 and feature not in feature_with_yerr:
          discrete_features.append(feature)
        else:
          continuous_features.append(feature)
```

OUTPUT:

Numerical Features ['Id', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SalePrice']

Discrete Features ['MSSubClass', 'OverallQual', 'OverallCond', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageCars', '3SsnPorch', 'PoolArea', 'MoSold']

Continuous Features ['Id', 'LotFrontage', 'LotArea', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'GarageYrBlt', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', 'ScreenPorch', 'MiscVal', 'YrSold', 'SalePrice']

CODE:

```
# Mutual information on Numerical Input
from sklearn.feature_selection import mutual_info_regression

y_train = X_train['SalePrice']
final_columns = discrete_features
for i in continuous_features:
  if i not in feature_with_year:
      final_columns.append(i)

print(final_columns)
mi_scores = mutual_info_regression(X_train[final_columns], y_train)
mi_scores = pd.Series(mi_scores, name="MI Scores", index=final_columns)
mi_scores = mi_scores.sort_values(ascending=False)

mi_scores
```

```
['MSSubClass', 'OverallQual', 'OverallCond', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageCars', '3SsnPorch', 'PoolArea', 'MoSold', 'Id', 'LotFrontage', 'LotArea', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', 'ScreenPorch', 'MiscVal', 'SalePrice']
```

OUTPUT:

```
SalePrice       5.453403
OverallQual     0.543599
GrLivArea       0.420799
GarageCars      0.353652
GarageArea      0.333527
TotalBsmtSF     0.323180
1stFlrSF        0.265578
FullBath        0.258839
MSSubClass      0.246294
2ndFlrSF        0.183458
LotFrontage     0.181488
TotRmsAbvGrd    0.174985
LotArea         0.168340
Fireplaces      0.162931
OpenPorchSF     0.138124
BsmtFinSF1      0.131653
BsmtUnfSF       0.116379
WoodDeckSF      0.089007
HalfBath        0.075320
OverallCond     0.075303
BedroomAbvGr    0.062036
MasVnrArea      0.043270
BsmtFullBath    0.036779
ScreenPorch     0.021023
LowQualFinSF    0.019126
EnclosedPorch   0.016887
BsmtFinSF2      0.008468
KitchenAbvGr    0.002491
PoolArea        0.002446
BsmtHalfBath    0.001572
MoSold          0.000000
3SsnPorch       0.000000
Id              0.000000
MiscVal         0.000000
Name: MI Scores, dtype: float
```

TRAINING AND TESTING OF DATA

Training data is an extremely large dataset that is used to teach a Machine learning model. Training data is used to teach prediction models That use machine learning algorithms how to extract features that are Relevant to specific business goals.

The test data set used to provide an unbiased evaluation of a final model fit on the training d ata set. If the data in the test data set has never been used in training , the test data set is also called a holdout dataset.

```
from sklearn.metrics import mean_absolute_error

from sklearn.model_selection import train_test_split



X = df_final.drop(['SalePrice'], axis=1)

Y = df_final['SalePrice']



# Split the training set into

# training and validation set

X_train, X_valid, Y_train, Y_valid = train_test_split(

    X, Y, train_size=0.8, test_size=0.2, random_state=0)
```

```
plt.figure(figsize=(18, 36))

plt.title('Categorical Features: Distribution')

plt.xticks(rotation=90)

index = 1



for col in object_cols:

    y = dataset[col].value_counts()

    plt.subplot(11, 4, index)

    plt.xticks(rotation=90)

    sns.barplot(x=list(y.index), y=y)

    index += 1
```

Conclusion:

This House price prediction project help us to predict the price of the house and detecting the quality of the house. By including some features we have able to measure the price approximately not be the decimal categorization.