

# **Machine Learning Project Report**

## **Selected Project Idea:**

**“House Price Prediction”**

## **Team Members:**

1<sup>st</sup> Team Member Name:

Kareem Saeed Ragab

1<sup>st</sup> Team Member ID:

2018170282

2<sup>nd</sup> Team Member Name:

Kareem Sherif Fathy

2<sup>nd</sup> Team Member ID:

2018170283

3<sup>rd</sup> Team Member Name:

Abanoub Asaad Azab

3<sup>rd</sup> Team Member ID:

2018170001

4<sup>th</sup> Team Member Name:

Nada El Sayed Anies

4<sup>th</sup> Team Member ID:

2018170430

5<sup>th</sup> Team Member Name:

Nada Mohamed Abdelhamed

5<sup>th</sup> Team Member ID:

2018170434

# Milestone # 1: “Preprocessing and Regression”

## 1- Preprocessing:

### 1.1 Drop unwanted Data

- Drop ID column because it only refer to row index
- For this dataset we are considering eliminating those variables which have or have more than 45% of total data missing or showing no values. PoolQC, MiscFeature, Fence, and FireplaceQu attributes for predicting house prices have very large numbers of missing values, which are, 99%, 96%, 81% and 47% respectively. Since it covers almost whole columns, eliminating these attributes won't cause any significant loss of values to our project. It is also important to be aware that right evaluation is required to deal with such missing values because there may be a chance of loss of important data.

### 1.2 Solve missing values

- LotFrontage attribute will be considered in our dataset. It has 211 missing values, which is a significant number, and we will replace it with the previous used value in the same column.
- Replace null values in the other columns by the most frequent value.

### 1.3 Label encoding

### 1.4 Feature Selection

- Get the top 50% correlation features with the SalePrice (target).
- From the top features correlation we noticed that there is features that have high correlation for that reason we will remove the least correlation with SalePrice For Example: 1stFlrSF and TotalBsmtSF both of them have very high correlation for that reason we will remove one of them, we chose to remove 1stFlrSF because it has least correlation with SalePrice.

### 1.5 Feature Scaling by using Normalization

## **2- Regression techniques used:**

- 2.1 Gradient Boosting Regressor.
- 2.2 Ridge Regression.

## **3- Differences between each model:**

**3.1 Gradient Boosting Regressor:** Builds an additive model in a forward stage wise fashion, it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

- Accuracy: 88.3%
- Mean Square Error: 807478598.5
- Training Time: 1.2 s

**3.2 Ridge Regression:** Estimates the coefficients of multiple-regression models in scenarios where independent variables are highly correlated.

- Accuracy: 82.4%
- Mean Square Error: 1095906187.7
- Training Time: 1.0 s

## **4- Training & Testing Sizes:**

Training size is 80% of the total data size and the testing data size is 20%.

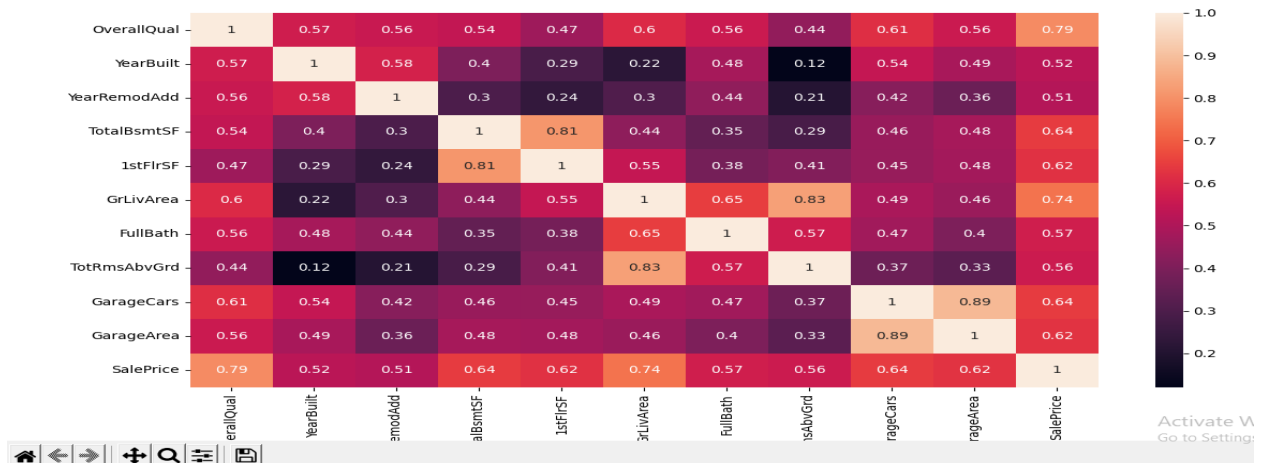
## **5- Conclusion:**

Prediction house prices are expected to help people who plan to buy a house so they can know the price range in the future, then they can plan their finance well.

## **6- Work Screenshots:**

### **5.1 Dataset Features Correlation:**

Figure 1



## 5.2 Gradient Boosting Regressor Model Output Information:

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help ML-HousePricePrediction [F:\Drasna\Fourth Year\First... ML-HousePricePrediction [F:\Drasna\Fourth Year\First... ML-HousePricePrediction [F:\Drasna\Fourth Year\First... ML-HousePricePrediction [F:\Drasna\Fourth Year\First...
Project ML-HousePricePrediction RegressionGradientBoostingModel.py RegressionDataProcessing.py ClassificationPoly_SVM_Model.py
.gitignore
Classification_Preprocessed_House_Data.csv
Classification_saved_model_1.sav
ClassificationDataProcessing.py
ClassificationPoly_SVM_Model.py
House_Data.csv
HouseDataClassification.csv
README.md
Regression_Preprocessed_House_Data.csv
regression_saved_model_1.sav
regression_saved_model_2.sav
RegressionDataProcessing.py
RegressionGradientBoostingModel.py
RegressionRidgeModel.py
script.py
External Libraries
Scratches and Consoles

12 saved_model_filename = 'regression_saved_model_2.sav'
13 start_preprocessing(dataset_name='House_Data.csv')
14 data = pd.read_csv('Regression_Preprocessed_House_Data.csv')
15
16 X = data.iloc[:, 0:15] # Features
17 Y = data['SalePrice'] # Label
18
19 # Apply GradientBoostingRegressor on the selected features
20 gradient_boosting_start_time = time.time()
21
22 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, shuffle=True)
23 cls = GradientBoostingRegressor()
24 cls.fit(X_train, Y_train)
25 pickle.dump(cls, open(saved_model_filename, 'wb'))
26 prediction = cls.predict(X_test)
27
28 time.sleep(1)
29 gradient_boosting_end_time = time.time()
30
31
Run: RegressionGradientBoostingModel (1) x
-----
Mean Square Error: 807478598.5411919
Accuracy: 88.3073443503813
Runtime of Gradient Boosting Regressor is 1.1727957725524902
Process finished with exit code 0

```

## 5.3 Ridge Regression Model Output Information:

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help ML-HousePricePrediction [F:\Machine Learning\Project\Milestone #2\Project\ML-HousePricePrediction] ... RegressionRidgeModel.py
Project ML-HousePricePrediction RegressionRidgeModel.py RegressionDataProcessing.py ClassificationPoly_SVM_Model.py
.gitignore
Classification_Preprocessed_House_Data.csv
Classification_saved_model_1.sav
ClassificationDataProcessing.py
ClassificationPoly_SVM_Model.py
House_Data.csv
HouseDataClassification.csv
README.md
Regression_Preprocessed_House_Data.csv
regression_saved_model_1.sav
regression_saved_model_2.sav
RegressionDataProcessing.py
RegressionGradientBoostingModel.py
RegressionRidgeModel.py
script.py
External Libraries
Scratches and Consoles

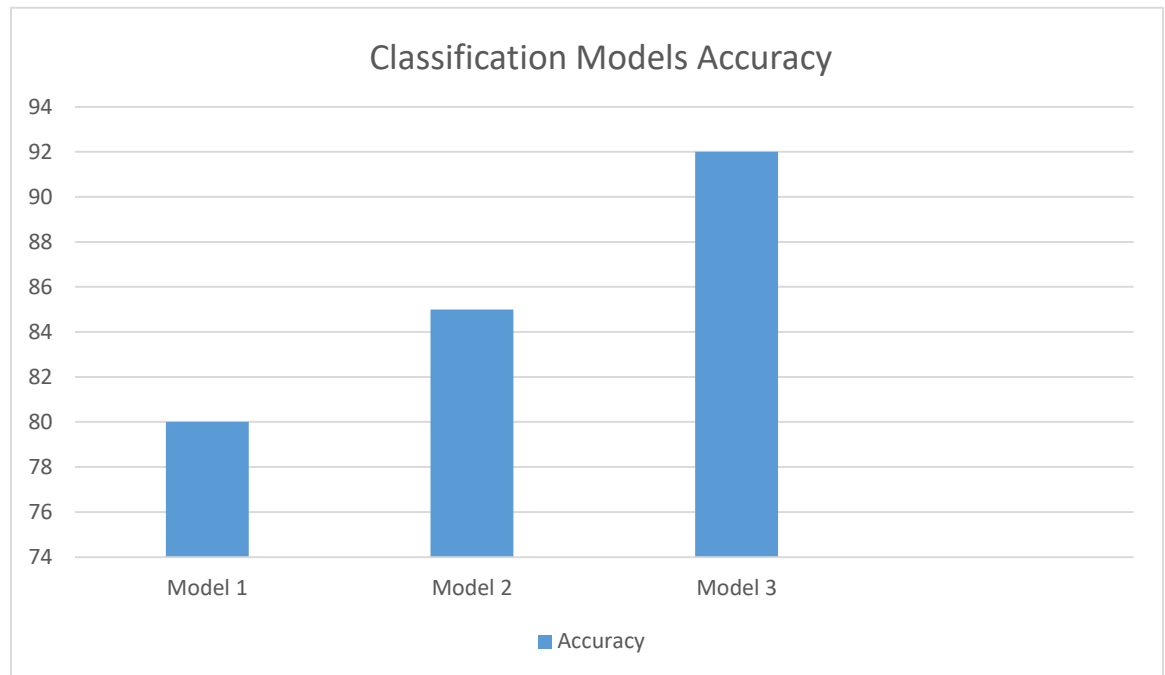
9 from RegressionDataProcessing import start_preprocessing
10
11 # Load house data
12
13 saved_model_filename = 'regression_saved_model_1.sav'
14 start_preprocessing(dataset_name='House_Data.csv')
15 data = pd.read_csv('Regression_Preprocessed_House_Data.csv')
16
17 X = data.iloc[:, 0:15] # Features
18 Y = data['SalePrice'] # Label
19
20 # Apply Ridge Regression on the selected features
21 ridge_start_time = time.time()
22
23 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, shuffle=True)
24 cls = RidgeCV(alphas=range(1, 40))
25 cls.fit(X_train, Y_train)
26 pickle.dump(cls, open(saved_model_filename, 'wb'))
27 prediction = cls.predict(X_test)
28
Run: RegressionRidgeModel x
-----
Mean Square Error: 1095906187.6758213
Accuracy: 82.38631886295238
Runtime of Ridge regression is 1.0226621627807617
Process finished with exit code 0

```

## Milestone # 2: “Classification and Hyperparameter tuning”

### 1- Three Classification Models Summary:

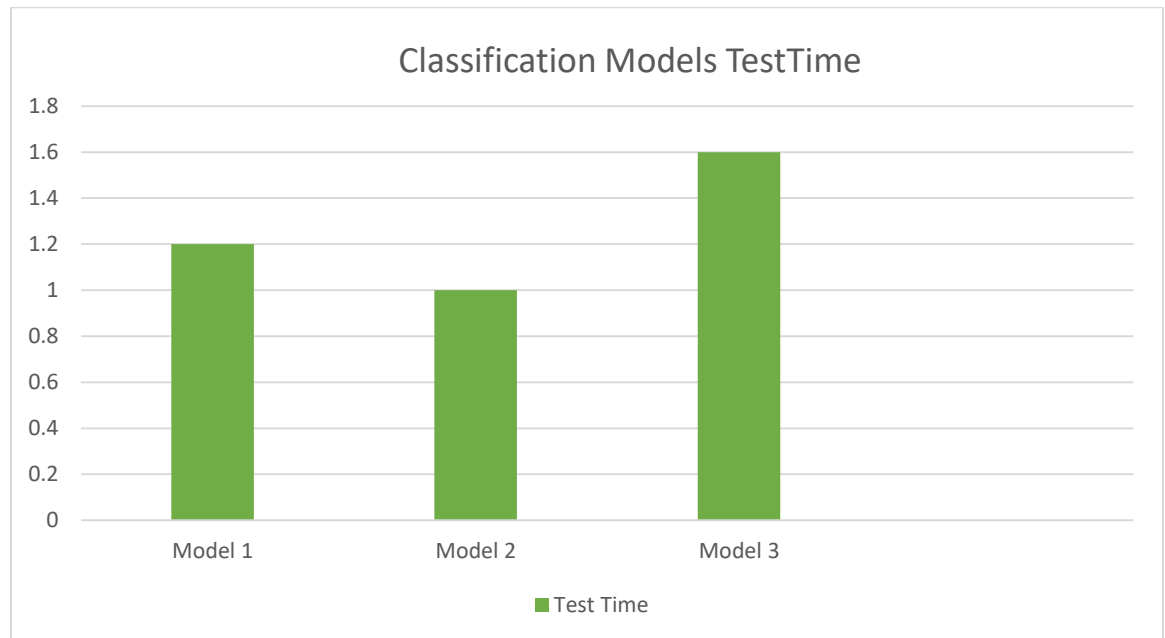
#### 1.1 Accuracy



#### 1.2 Training Time



### 1.3 Test Time



### 2- Feature Selection:

In this phase the feature selection differ from the previous phase So, Get the top 50% correlation features with the PriceRate (target).

HouseStyle and BsmtHalfBath features have the high correlation with PriceRate (target).

### 3- Hyperparameter Tuning:

#### **4- Conclusion:**