PSL (F20) Project 1

Vijayakumar Sitha Mohan (VS24), Waitong Matthew Leung (wmleung2)

13-Sept-2020

Introduction

The goal is to predict the final price of a home (in log scale) using AMES housing dataset. As a project requirement we need to build Two prediction models and we chose Lasso Regression and XBGBoost as our model to predict the sale price.

The dataset has 2930 rows (i.e., houses) and 83 columns. The first column is “PID”, the Parcel identification number; The last column is the response variable, Sale_Price; The remaining 81 columns are explanatory variables describing (almost) every aspect of residential homes.

Finally we need to create two output files with the predictions prices from each model.

Method

Utility Functions

Data Read and Exploration

- Load Data

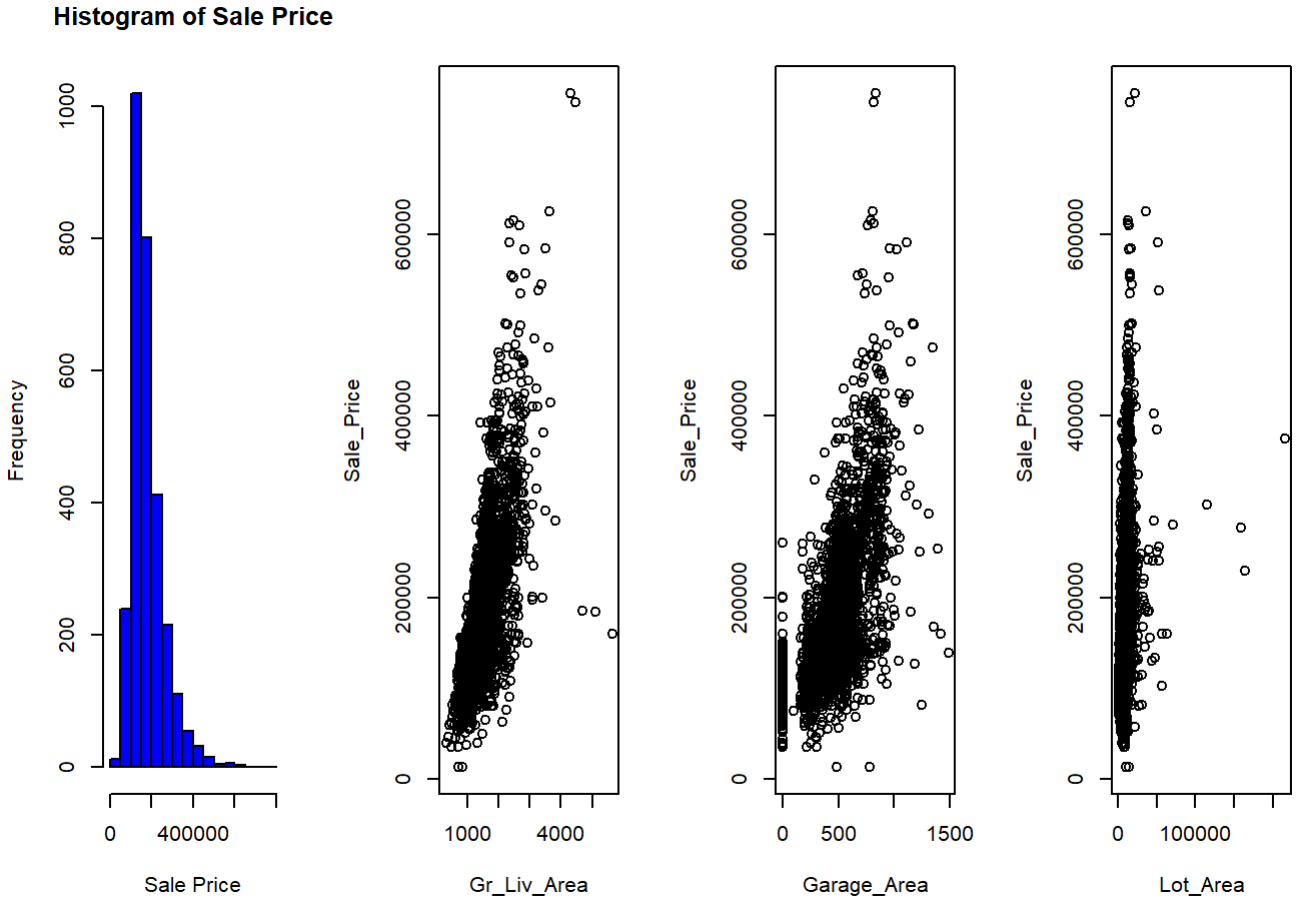
```
data = read.csv("Ames_data.csv")
```

- Analysis of Response Variable(Sale Price)

A mean and median sale price of $180,796 and $160,000 respectively, indicating a positive skew, which was confirmed by histogram of the Sale Price.

A long right tail tells that there are outlier data points. max = 755,000 and min = 12,789

- Skewness in Data



Data PreProcessing

- Remove features with Zero variance

```
# Find zero varience factors using caret
nzv.data = nearZeroVar(data, saveMetrics = TRUE)

# Remove zero variance factors
drop.cols = rownames(nzv.data)[nzv.data$nzv == TRUE]
remove.var = c(drop.cols, "Longitude","Latitude")
data = removeVars(data,remove.var)
```

- Impute Missing Values with Preprocess using Caret package

```
d <- preProcess(data, "medianImpute")
data = predict(d,data)
```

- Log transformation for skewed features

```
skewed.vars = getSkewedVars(data)
data_encoded = hotEncoding(data)
for(i in skewed.vars){
  if(0 %in% data_encoded[, i]){
    data_encoded[,i] <- log(1+data_encoded[,i])
  }
  else{
    data_encoded[,i] <- log(data_encoded[,i])
  }
}
data_encoded$PID = data$PID
data_encoded$Sale_Price = data$Sale_Price
data_encoded$Sale_Price_Log = log(data$Sale_Price)
```

- Remove Outlier Data points

```
fit = lm(Sale_Price_Log ~ Lot_Area + Mas_Vnr_Area  + Bsmt_Unf_SF + Total_Bsmt_SF +
Second_Flr_SF + First_Flr_SF + Gr_Liv_Area + Garage_Area + Wood_Deck_SF   , data =
data_encoded)
fit_cd = cooks.distance(fit)
data_encoded = data_encoded[fit_cd < 4 / length(fit_cd),]
```

- Check for missing value

```
## [1] "Number of Missing Values 0"
```

```
## [1] "Number of Missing Values 0"
```

- Train and Test Data Split Split the dataset into Train and Test by using given test project ids.

Model Building

- LASSO Regression Model
- Cross Validation to choose Lamda hyperparameter

```
set.seed(8742)
X = trainData[,!names(trainData) %in% c("Sale_Price","Sale_Price_Log","PID")]
cv_lasso=cv.glmnet(as.matrix(X),trainData$Sale_Price_Log, nfolds = 10, alpha = 1)
cv_lasso$lambda.min
```

```
## [1] 0.001423
```

- Feature Engineering

```
# select variables
sel.vars <- predict(cv_lasso, type="nonzero", s = cv_lasso$lambda.min)$X1
```

- Fit & Predict using selected Model

```
## [1] 0.08108
```

```
## [1] 0.0911
```

- XGBoost - Boosting Model
- Cross Validation to choose Lamda, subsample, max_depth hyperparameter
- Fit & Predict using selected Model

```
## [1] 0.02331
```

```
## [1] 0.09255
```

Results

```
##    Lasso Train Error Lasso Test Error XGBoost Train Error XGBoost Test Error
## 1            0.08308          0.11321             0.02158            0.11406
## 2            0.07671          0.07300             0.02433            0.07924
## 3            0.07739          0.08771             0.02337            0.09828
## 4            0.08116          0.07374             0.02251            0.07468
## 5            0.08656          0.08447             0.02321            0.09786
## 6            0.08791          0.08698             0.02644            0.08527
## 7            0.07905          0.08794             0.02235            0.09343
## 8            0.08364          0.11218             0.02470            0.10332
## 9            0.07516          0.09781             0.02230            0.10050
## 10           0.07963          0.08599             0.02246            0.07549
```

Discussion

- There are two important observations we had during model building.

    1. Without outlier removal Lasso Model didn't meet the criteria specified by the project requirement. When we handled the outlier data points by doing linear regression and using cooks distiance metrics then the performance was greatly improved and both train and test errors for Lasso Regression was below 10.

    2. The hyperparameter tuning for XGBoost runs too long and to tune Lambda, MaX_Depth and SubSample, took 11 hours to find the optimal value using my laptop.