**Housing Sales Price Predictor**

**Group 1**

Anshita Aishwarya

Unnati Ghodki

669-279-9234

669-294-6530

[aishwarya.a@northeastern.edu](mailto:aishwarya.a@northeastern.edu)

[ghodki.u@northeastern.edu](mailto:ghodki.u@northeastern.edu)

Percentage of Effort Contributed by Student 1: 50%

Percentage of Effort Contributed by Student 2: 50%

Signature of Student 1:

Signature of Student 2:

Submission Date: Apr 11th, 2022

1. **Problem Setting**

Housing sale is an ever-changing market, and its dynamics keep fluctuating with various factors like economic status, interest rates, etc. A housing sale forecast will benefit a large population such as potential buyers, sellers, investors, insurance agencies, bankers, real estate marketers etc., and allow them to understand the marketing conditions to expect in the coming time. This will also allow them to gain a perspective of how to sustain in this competitive market.

1. **Problem Definition**

This analysis will facilitate in predicting the housing sale prices taking into account housing attributes such as floor plan and area, number of bedrooms/ bathrooms, types of utilities offered, neighborhood etc. The aim is to project the sale prices with highest accuracy by identifying the best model using data mining techniques. Further, a house sale estimate calculator website will allow users to fill in their search parameters and trigger the model to predict the precise estimates.

1. **Data Source**

This dataset has been taken from a *Kaggle – a Data Science and Machine Learning Community.* It provides a platform to explore and publish the open source datasets on 1K+ datasets on projects in various domain, and collaborate with Data Professionals.

**Dataset** - <https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>

1. **Data Description**

This data contains housing sale prices with 79 explanatory variables describing almost every aspect of residential homes in Ames, Iowa. Through this project, we will be predicting the sale price (target variable) of the house based on various predictors including the categorical and numerical independent variables playing roles in pricing at various scale in market.

**The dataset contains the following attributes**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Variable Name** | **Description** |
| 1 | SalePrice | Property’s sale price (USD) - Output variable (Target) |
| 2 | MSSubClass | Identifies the type of dwelling involved in the sale |
| 3 | MSZoning | Identifies the general zoning classification - Agricultural, Commercial, Residential etc |
| 4 | LotFrontage | Linear feet of street connected to property |
| 5 | LotArea | Area of the lot (square feet) |
| 6 | Street | Type of road access - Gravel, Paved |
| 7 | Alley | Type of alley access - Gravel, Paved, No alley |
| 8 | LotShape | General shape of property - Regular, Irregular |
| 9 | LandContour | Flatness of the property - Level, Banked, Hillside, Depression |
| 10 | Utilities | Type of utilities available - Electricity, Gas, Water, All |
| 11 | LotConfig | Lot configuration - Inside, Corner, Frontage |
| 12 | LandSlope | Slope of property - Gentle, Moderate, Severe |
| 13 | Neighborhood | Physical locations within the city limits |
| 14 | Condition1 | Proximity to main road or railroad |
| 15 | Condition2 | Proximity to main road or railroad (if a second is present) |
| 16 | BldgType | Type of dwelling - Single family, Duplex, Townhouse |
| 17 | HouseStyle | Style of dwelling - One story, Two strory, Split foyer |
| 18 | OverallQual | Overall material and finish quality - Poor, Fair, Good, Excellent |
| 19 | OverallCond | Overall condition rating - Poor, Fair, Good, Excellent |
| 20 | YearBuilt | Original construction date |
| 21 | YearRemodAdd | Remodel date |
| 22 | RoofStyle | Type of roof - Flat, Gable, Hip, Shed |
| 23 | RoofMatl | Roof material - Membrane, Metal, Wood |
| 24 | Exterior1st | Exterior covering on house -Brick, Cinder, Cement etc. |
| 25 | Exterior2nd | Exterior covering on house (if more than one material) |
| 26 | MasVnrType | Masonry veneer type - Brick, Cinder, Stone |
| 27 | MasVnrArea | Masonry veneer area in square feet |
| 28 | ExterQual | Exterior material quality - Poor, Fair, Good, Excellent |
| 29 | ExterCond | Present condition of the material on the exterior |
| 30 | Foundation | Type of foundation - Brick, Cinder, Stone, Wood |
| 31 | BsmtQual | Height of the basement - No Basement, Poor, Good, Fair, Excellent |
| 32 | BsmtCond | General condition of the basement - No Basement, Poor, Good, Fair, Excellent |
| 33 | BsmtExposure | Walkout or garden level basement walls - No Basement, Poor, Good, Fair, Excellent |
| 34 | BsmtFinType1 | Quality of basement finished area |
| 35 | BsmtFinSF1 | Type 1 finished square feet |
| 36 | BsmtFinType2 | Quality of second finished area (if present) |
| 37 | BsmtFinSF2 | Type 2 finished square feet |
| 38 | BsmtUnfSF | Unfinished square feet of basement area |
| 39 | TotalBsmtSF | Total square feet of basement area |
| 40 | Heating | Type of heating - Floor, Gas, Hot water, Wall furnace |
| 41 | HeatingQC | Heating quality and condition - Poor, Good, Fair, Excellent |
| 42 | CentralAir | Central air conditioning - Yes, No |
| 43 | Electrical | Electrical system - Standard, Fuse Box, Mixed |
| 44 | 1stFlrSF | First Floor square feet |
| 45 | 2ndFlrSF | Second floor square feet |
| 46 | LowQualFinSF | Low quality finished square feet (all floors) |
| 47 | GrLivArea | Above grade (ground) living area square feet |
| 48 | BsmtFullBath | Basement full bathrooms |
| 49 | BsmtHalfBath | Basement half bathrooms |
| 50 | FullBath | Full bathrooms above grade |
| 51 | HalfBath | Half baths above grade |
| 52 | Bedroom | Number of bedrooms above basement level |
| 53 | Kitchen | Number of kitchens |
| 54 | KitchenQual | Kitchen quality - Poor, Fair, Good, Excellent |
| 55 | TotRmsAbvGrd | Total rooms above grade |
| 56 | Functional | Home functionality rating - Typical, Minor, Moderate, Major, Severe damage |
| 57 | Fireplaces | Number of fireplaces |
| 58 | FireplaceQu | Fireplace quality - Average, Good, Excellent |
| 59 | GarageType | Garage location - No garage, Detached, Built-in, Basement, More than 1 |
| 60 | GarageYrBlt | Year garage was built |
| 61 | GarageFinish | Interior finish of the garage - No garage, Unfinished, Rough finished, Finished |
| 62 | GarageCars | Size of garage in car capacity |
| 63 | GarageArea | Size of garage in square feet |
| 64 | GarageQual | Garage quality - No garage, Poor, Fair, Good, Excellent |
| 65 | GarageCond | Garage condition |
| 66 | PavedDrive | Paved driveway - Paved, Partial paved, Gravel |
| 67 | WoodDeckSF | Wood deck area in square feet |
| 68 | OpenPorchSF | Open porch area in square feet |
| 69 | EnclosedPorch | Enclosed porch area in square feet |
| 70 | 3SsnPorch | Three season porch area in square feet |
| 71 | ScreenPorch | Screen porch area in square feet |
| 72 | PoolArea | Pool area in square feet |
| 73 | PoolQC | Pool quality - No pool, Poor, Fair, Good, Excellent |
| 74 | Fence | Fence quality - No fence, Poor, Fair, Good, Excellent |
| 75 | MiscFeature | Miscellaneous feature not covered in other categories - Elevator, 2nd garage, Tennis court, None |
| 76 | MiscVal | Price Value of miscellaneous feature (USD) |
| 77 | MoSold | Month Sold |
| 78 | YrSold | Year Sold |
| 79 | SaleType | Type of sale - Warranty deed, Court office deed, Contract, Other |
| 80 | SaleCondition | Condition of sale - Normal, Abnormal, Partial, Allocation |

The existing dataset has already been partitioned into training and test datasets into a 50-50 proportion. The training data has 80 columns (79 attributes, 1 target variable) and 1460 rows. The test data has 79 columns (79 attributes and no target variable) and 1459 rows. The test data is unlabeled that is, we do not have the actual sale price values. We will partition the train data for training the model and choosing the best model validation and further use the test data to validate the chosen model.

1. **Data Processing**
   1. ***Dealing with Date columns – Age Calculations***

We have 4 columns denoting the year the house was built, remodelled, sold and the year the garage was built. It makes more sense to calculate the respective ages of these Year columns as the year alone is not insightful. Hence, we have manipulated the data by calculating their ages up to today's year.

* **YearBuilt**
* **YearRemodAdd**
* **YrSold**
* **GarageYrBlt**
  1. ***Dealing with missing values***

We have ample of missing values in our dataset which has been visualized in the below heatmap and need to be imputed with relevant statistical measures for Machine Learning model fitting. We try to explore these missing values for numerical and categorical columns individually as the process of imputations are to be used for both data types.

A picture containing text, device

Description automatically generated

* + 1. ***Numerical columns***

We have 3 numerical columns with missing values. We have used the threshold as 17% for the missing values i.e., all the columns having missing values more than this threshold will be dropped. Thus, we are dropping 1 column (**LotFrontage**). Another column containing missing values is **MasVnrArea** where these missing values signify that such houses don't have masonry veneer. Thus, it makes sense to impute these NaNs with 0. As for the 3rd column **GarageAge**, missing value means that there is no garage present in the house. Thus, similarly, we have imputed these values with 0.

Dropped Numerical Columns:

* **LotFrontage**
* **MasVnrArea**
* **GarageAge**
  + 1. ***Categorical columns***

As for the categorical columns, they have NAs which do not actually represent missing values. For example, in the column **Alley**, the actual data contains NA which is abbreviated for No Alley Access, hence we cannot consider such values as missing values. Hence, in order to deal with such values, we have replaced all NA values with 'None'.

1. **Dimensionality Reduction**
   1. ***Correlation Matrix for Numeric Variables (Pearson's Coefficient)***

As part of dimension reduction, we have calculated the correlation between every 2 numerical columns using Pearson’s coefficient. Pearson’s correlation coefficient is a bivariate correlation that measures the linear correlation between two sets of data, whose value ranges from -1 to 1. Essentially, it is a normalized measurement of their covariances. If a pair of variables are highly correlated (here we have considered 0.8 as the cutoff for highly correlated variables), then we drop one of the column-pair.

* This method gives us a correlation matrix which has been represented using the below heatmap.
* Based on the obtained correlation matrix, the below 3 pairs of variables are highly correlated.

|  |  |  |
| --- | --- | --- |
| **Predictor1** | **Predictor2** | **Correlation Coefficient** |
| TotalBsmtSF | 1stFlrSF | 0.81953 |
| TotRmsAbvGrd | GrLivArea | 0.825489 |
| GarageCars | GarageArea | 0.882475 |

A screenshot of a computer

Description automatically generated with low confidence

Due to the higher correlation between the pair of predictors we have dropped one of them to avoid the presence of unwanted bias in the predictions as mentioned below -

* **TotalBsmtSF**
* **TotRmsAbvGrd**
* **GarageCars**
  1. ***Correlation Matrix for Categorical Variables (Crammer's Rule)***

***Definition***

This rule is used to calculate the correlation between two categorical variables for selecting one category among the pairs in case of a higher correlation between them. The coefficient’s value varies from 0 to 1.

*We have defined a function for calculating the correlation matrix between categorical columns of the dataset using the Crammer’s Rule. This user-defined function has been obtained from Kaggle:* [*https://www.kaggle.com/chrisbss1/cramer-s-v-correlation-matrix*](https://www.kaggle.com/chrisbss1/cramer-s-v-correlation-matrix)

The function’s output returns a confusion matrix including all the categorical columns and their respective correlation values. Based on this, we have identified the following pairs of highly correlated columns (with correlation values greater than 0.6):

|  |  |  |
| --- | --- | --- |
| **Predictor1** | **Predictor2** | **Correlation Coefficient** |
| BsmtQual | BsmtFinType | 0.607478 |
| GarageFinish | GarageQual | 0.610604 |
| GarageCond | GarageFinish | 0.613192 |
| MSZoning | Neighborhood | 0.644037 |
| GarageType | GarageFinish | 0.743423 |
| GarageCond | GarageQual | 0.799799 |
| Exterior2nd | Exterior1st | 0.851769 |

In order to identify the categorical columns that can be dropped, we used this matrix of highly correlated columns and checked the number of categories for each of the pair of columns. The category having more number of categorical values is chosen to be dropped so that while encoding the categorical columns into numerical, a lesser number of newer columns are created.

1. **Data Exploration**
   1. ***Combining Categories – by plotting histogram of different categories***

We have a lot of categorical columns containing rare categories that can be combined together under a single category. This is done so that during encoding these categorical columns into numerical values, lesser number of new columns are created. In order to visualize which categories can be combined together, we created the following bar chart. Further, we have fixed a threshold of 20% of the most dominant category, that is, if any category has records 20% less than the most dominant category, then we combine such categories under "Other" category. This is done for all the 37 categorical columns.

Chart, waterfall chart

Description automatically generated

Chart, bar chart, waterfall chart

Description automatically generated

Chart, bar chart

Description automatically generated A picture containing shape

Description automatically generated

By combining the small number categories, we have decreased 104 categories (cumulative sum of all categorical columns combined).

* 1. ***Assessing Correlation between Predictors and Target variable:***

As we know that the presence of uncorrelated predictors with the target variable increases the variance in the predicted output, we are eliminating the weakly correlated variables.

* + 1. ***Numerical Predictor and Target Variable***

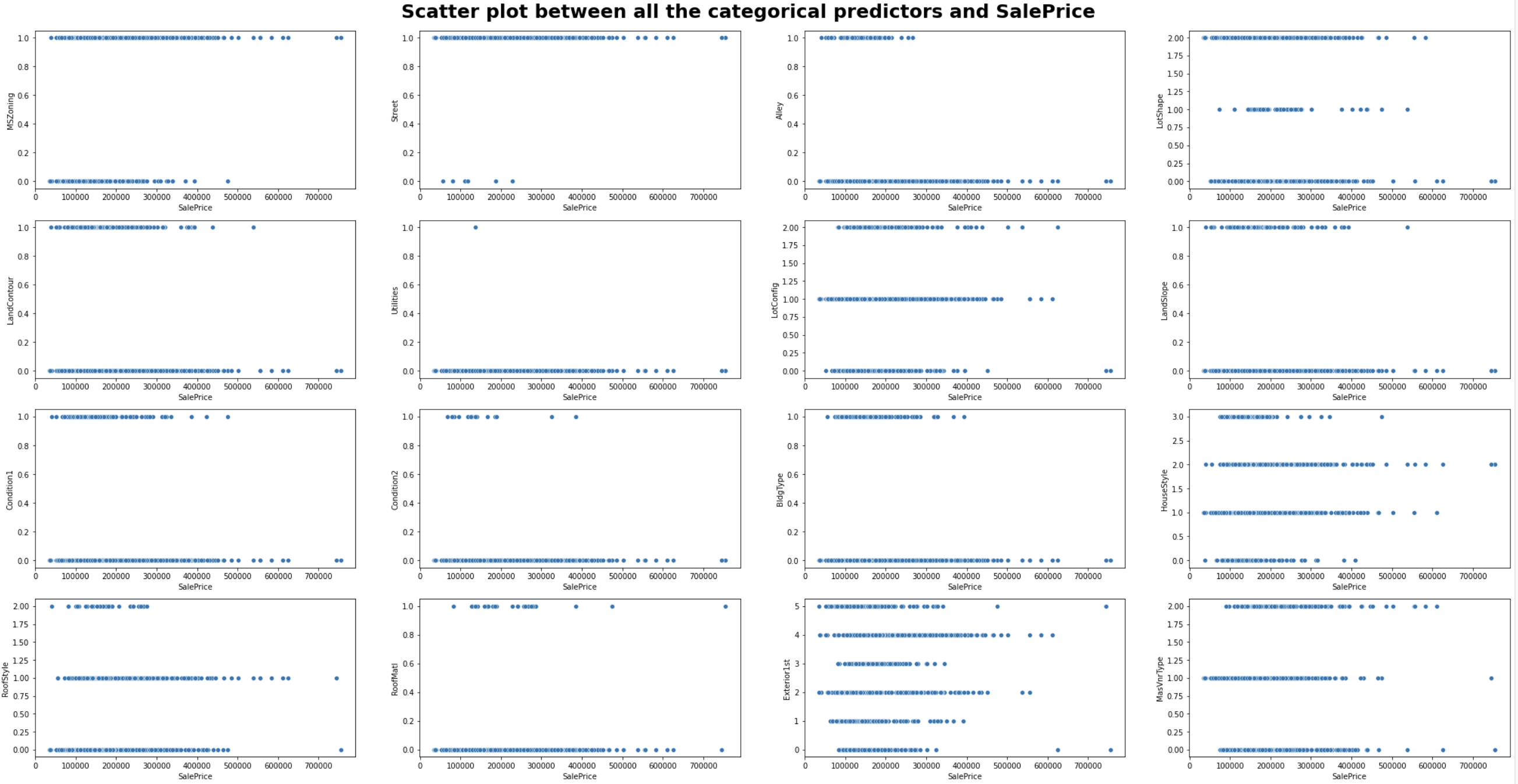
We have performed a correlation coefficient analysis between all the numerical predictors and the target variable. This has been visualized with the help of the scatter plots.

Using this correlation analysis, we can identify that, columns such as **‘MSSubClass’**, ‘**OverallQual’**, **‘OverallCond’** etc. are weakly correlated with target. We have used the threshold as 0.2 (Pearson Correlation coefficient) which means all the predictors having correlation coefficient values with target of 0.2 or lesser will be dropped. Using this, we successfully dropped 14 numerical columns.

A picture containing window, surrounded

Description automatically generated

* + 1. ***Categorical Predictor and Target Variable***
* We have used the ***Label Encoder*** from the “pre-processing” library to perform the encoding of categorical variables in order to calculate the correlation between all the categorical predictors and Target – **SalePrice** and plotted the scatter plot.
* Calculated the Pearson correlation coefficient between all the predictors and Target and dropped the variables having coefficients less than 0.2.
* Using the above criterion, we were able to remove 25 categorical columns which would help us acquire lesser variance in the predicted output.

****

**A screenshot of a computer

Description automatically generated with low confidenceGraphical user interface, application

Description automatically generated**

* 1. **Data Encoding**

Performed the **“Target Encoding”** as part of data encoding, a technique that results in only 1 dummy variable for a categorical column based on the mean value of the **Target** variable.

“Target Encoder“ function has been used from the “category\_encoders” library which is replacing categorical values with the respective mean of Target Variable.

We have chosen **Target Encoding** over **One- Hot Encoding** due to its benefit of not adding to the dimensionality of the dataset. Although “One-Hot Encoding” is an extremely easy technique to understand, it significantly increases the dimensionality of a dataset depending on the number of categories present in all the categorical columns.

1. **Data Preparation/Modelling**

The final data that we have obtained contains 30 features and 1460 records.

* 1. ***Standardization***

Our data contains values ranging from 1,300 to 215,245 square feet for the column ‘LotArea’. On the other hand, it also contains single-digit values for ‘OverallQual’ denoting the overall quality of the house. Based on the fact that our data has a very large-scale difference, we have standardized the data to bring the values on the same scale.

* 1. ***Data Partitioning***

Further, before model exploration, we perform a split in the dataset to create partitioning into training and validation data. We have used the 75-25% ratio for data partition. All the 30 features are collectively stored in variable X and the target variable is stored in y. After performing data partitioning, we have obtained X\_train, X\_test, y\_train, y\_test.

**X\_train** = contains data of independent variables used for training model.

**y\_train** = contains data of dependent variable used for training model.

**X\_test** = contains data of independent variables used for testing model.

**y\_test** = contains data of dependent variable used for testing model.

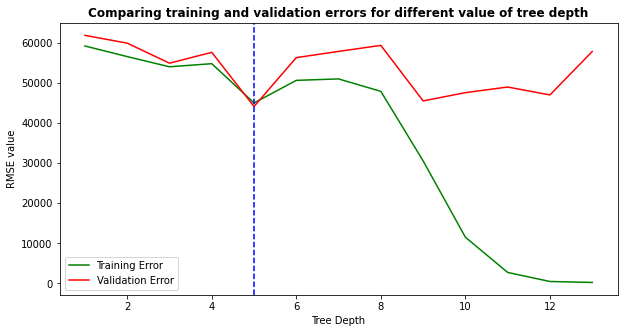
Post performing split, our data looks like this:

|  |  |  |
| --- | --- | --- |
| **Training** |  |  |
| X\_train | 1095 | 30 |
| y\_train | 1095 | 1 |
|  |  |  |
| **Validation** |  |  |
| X\_test | 365 | 30 |
| y\_test | 365 | 1 |

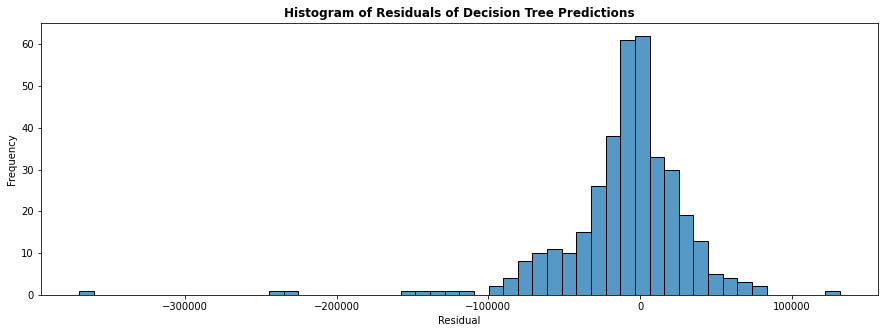
1. **Model Exploration**
   1. ***Decision Trees***

Using decision trees, we have tried to predict the value of the target variable (Sales Price). Decision tree is an algorithm that tries to strategically split a node into 2 or more sub-nodes in order to increase homogeneity of the resulting sub-nodes. Each node in the decision tree acts as a test case for some feature and based on the decision made at every node, the data is split into 2 or more sub-nodes. Decision trees end with leaf nodes that attempt to achieve maximum homogeneity without overfitting the data.

In our case, firstly, we tried to obtain the tree depth (number of edges from leaf node to the tree’s root node). This has been done by calculating the mean squared error obtained by predicting the target values with varying tree depths. Ultimately, we obtained 5 as the optimal tree depth.



Using this, we predicted the Sales Price and corresponding residual values (y actual – y predicted). This has been demonstrated using the histogram plot.



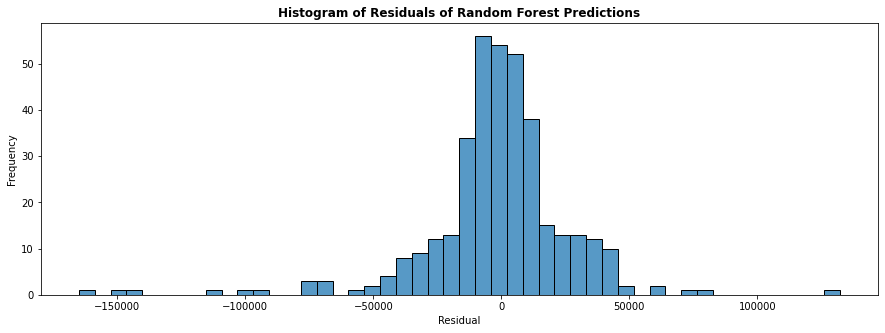
Finally, we evaluated the performance of the model using various prediction measures as below:

|  |  |
| --- | --- |
| **R2 score** | 0.721739 |
| **Mean Squared Error** | 1949306000 |
| **Root Mean Squared Error** | 44150.94421 |
| **Mean Absolute Error** | 27318.75069 |

* 1. ***Random Forest***

Random Forest is an ensemble of decision trees created using the bagging method (which states that a combination of learning models improves the overall method). Random forest adds randomness to the model while growing the trees, that is, while splitting a node, it searches for the best feature among a random subset of features. Therefore, in random forest, only a random subset of features are taken into consideration while fitting the model. This allows us to obtain better models as compared to decision trees.

We have applied the Random Forest regressor for our dataset, obtaining predictions on the target variable. The residuals obtained are plotted in the below histogram:



Moreover, the performance of the predictions is measured using the following parameters:

|  |  |
| --- | --- |
| **R2 score** | 0.884731 |
| **Mean Squared Error** | 807492500 |
| **Root Mean Squared Error** | 28416.41284 |
| **Mean Absolute Error** | 18047.65975 |

* 1. ***Linear Regression***

Linear Regression is the machine learning model which fits the data assuming the linear relationship between predictors and target variable. Our will the multiple linear regression as it has more than one input variable. We have used “LinearRegression” which uses “ordinary least squares” procedure to estimate the coefficients by minimizing the sum of squared errors between the predicted output and the actual output.

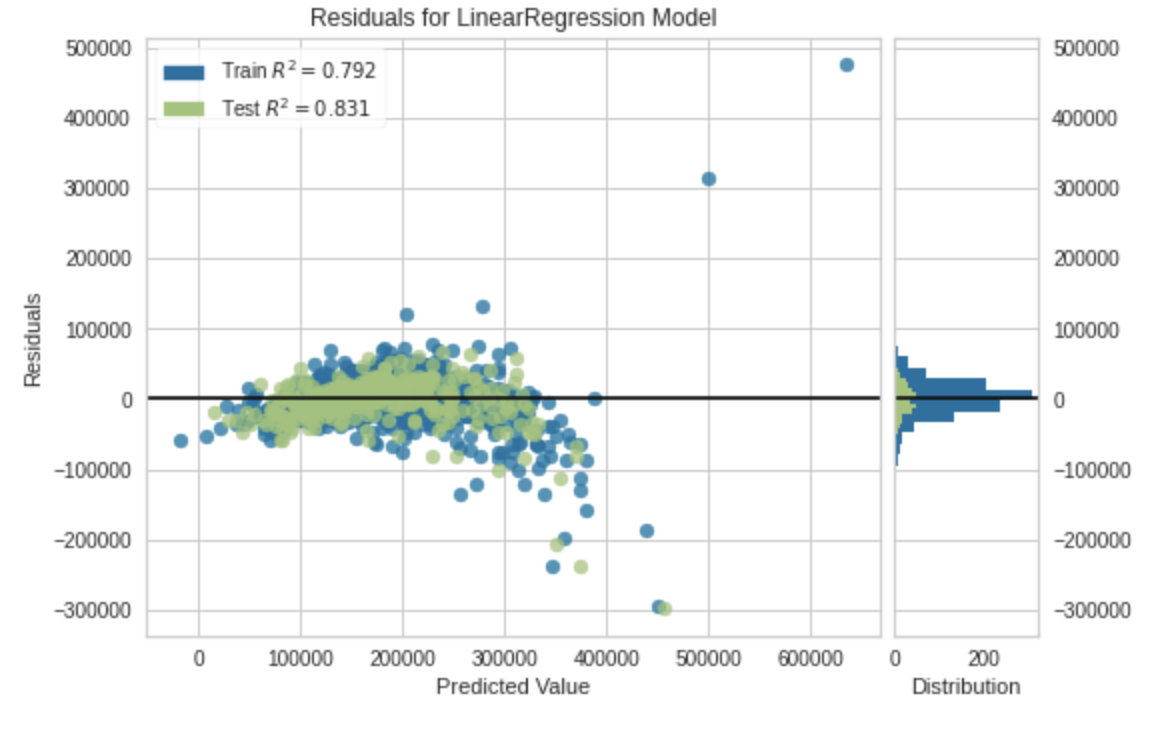
The performance evaluation of the model has been calculated as below-

| **S.No** | **Scores** | **Values** |
| --- | --- | --- |
| **0** | Coefficient of determination (R2) | 0.791584 |
| **1** | Intercept | -29220.373756 |
| **2** | Slope | [0.4637986707855082, 19837.66387180761, 18.840... |

The model parameters have been estimated as below –

| **S.No** | **Predictor** | **Coefficient** |
| --- | --- | --- |
| **0** | LotArea | 0.463799 |
| **1** | OverallQual | 19837.663872 |
| **2** | MasVnrArea | 18.840037 |
| **3** | BsmtFinSF1 | 10.126599 |
| **4** | BsmtUnfSF | 1.098707 |
| **5** | 1stFlrSF | 25.218931 |
| **6** | 2ndFlrSF | 18.424189 |
| **7** | GrLivArea | 18.499338 |
| **8** | BsmtFullBath | 9585.848658 |
| **9** | FullBath | 1806.924592 |
| **10** | HalfBath | 1538.740196 |
| **11** | Fireplaces | 7205.601757 |
| **12** | GarageArea | 43.725333 |
| **13** | WoodDeckSF | 26.213349 |
| **14** | OpenPorchSF | -2.811547 |
| **15** | HouseAge | -57.088840 |
| **16** | HouseRemodelAge | -284.855130 |
| **17** | GarageAge | -31.380153 |
| **18** | MSZoning | 8244.225018 |
| **19** | LotShape | -1996.866399 |
| **20** | ExterQual | -3406.733330 |
| **21** | Foundation | 3823.788226 |
| **22** | BsmtQual | 4228.652945 |
| **23** | HeatingQC | -1833.759747 |
| **24** | CentralAir | 922.628887 |
| **25** | Electrical | -6848.371418 |
| **26** | KitchenQual | 1358.450718 |
| **27** | GarageFinish | -2180.189378 |
| **28** | PavedDrive | 2006.939843 |
| **29** | SaleType | -11730.081550 |

We have used “ResidualsPlot” from “YellowBrick.regressor” library to visualize the scores of the fitted model on the validation and training dataset.



The various errors are calculated using “RegressionSummary” function –

Regression statistics

Mean Error (ME) : 3629.0352

Root Mean Squared Error (RMSE) : 34390.7353

Mean Absolute Error (MAE) : 21610.8772

Mean Percentage Error (MPE) : 0.5491

Mean Absolute Percentage Error (MAPE) : 12.7155

*Training Residual Plot Validation Residual Plot*

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

* 1. ***Regularized Linear Regression Model – Lasso***

Lasso is a type of linear regression model which uses “shrinkage” wherein data points are shrunk towards any statistical measure. It uses **L1** regularization and applies penalty based on the sum of absolute values of coefficients.

The performance evaluation of the model has been calculated as below-

|  |  |  |
| --- | --- | --- |
| **S.No** | **Scores** | **Values** |
| **0** | Coefficient of determination (R2) | 0.791572 |
| **1** | Intercept | -28930.649996 |
| **2** | Slope | [0.46126203383172, 19843.38400698893, 18.99501... |



We can see that the R2 is not improved using Lasso Model which uses alpha = 1 as default.

*Validation Residual Plot*

*Chart, histogram

Description automatically generated*

Regression statistics

Mean Error (ME) : 3631.2808

Root Mean Squared Error (RMSE) : 34407.5348

Mean Absolute Error (MAE) : 21603.3905

Mean Percentage Error (MPE) : 0.5390

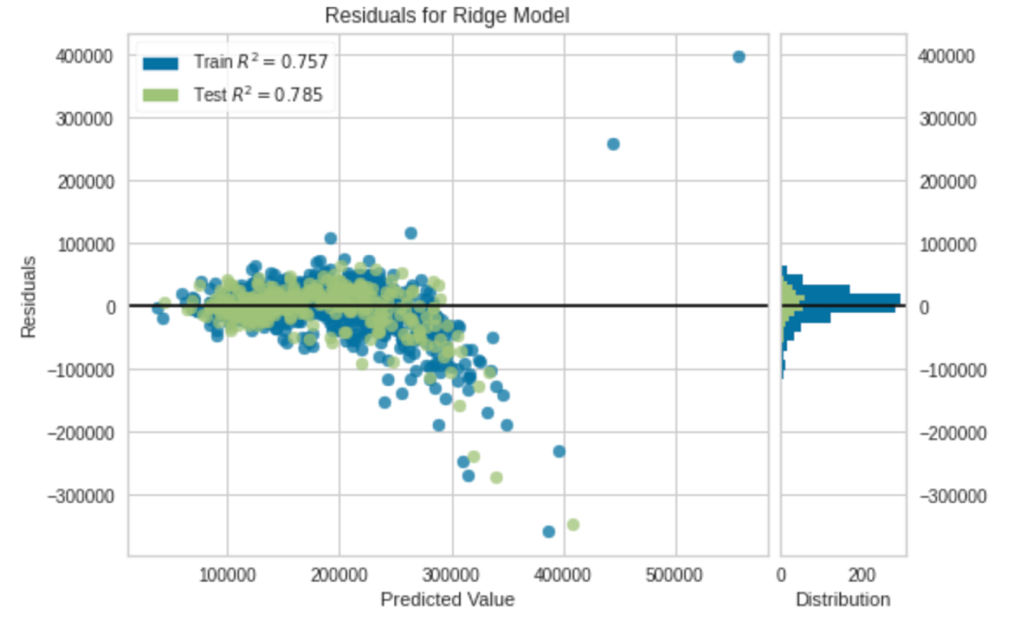
Mean Absolute Percentage Error (MAPE) : 12.7077

* 1. ***Regularized Linear Regression Model – Ridge***

Lasso is a type of linear regression model which uses “shrinkage” wherein data points are shrunk towards any statistical measure. It uses **L2** regularization and applies a penalty based on the sum of squared coefficients.

The performance evaluation of the model has been calculated as below-

|  |  |  |
| --- | --- | --- |
| **S.No** | **Scores** | **Values** |
| **0** | Coefficient of determination (R2) | 0.757312 |
| **1** | Intercept | 43234.690169 |
| **2** | Slope | [0.3315319382310712, 8459.551315963707, 28.038... |



We can see that the R2 score of Test dataset is reduced

*Validation Residual Plot*

*Chart, histogram

Description automatically generated*

Regression statistics

Mean Error (ME) : 1934.4418

Root Mean Squared Error (RMSE) : 38836.1704

Mean Absolute Error (MAE) : 22705.8181

Mean Percentage Error (MPE) : -3.1038

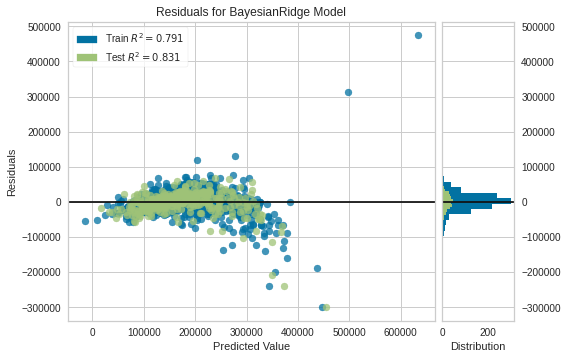
Mean Absolute Percentage Error (MAPE) : 12.3542

* 1. ***Regularized Linear Regression Model – Bayesian***

It uses Gaussian distribution to make the predictions instead of point estimates. We use probability distribution to estimate the target variable and the regression model is sampled from Normal distribution.

The output variable is generated using Normal distribution metrics – means and variances

|  |  |  |
| --- | --- | --- |
| **S.No** | **Scores** | **Values** |
| **0** | Coefficient of determination (R2) | 0.791394 |
| **1** | Intercept | -23290.7 |
| **2** | Slope | [0.45192006229283005, 18676.54858595858, 20.30... |



Regression statistics

Mean Error (ME) : 3499.7098

Root Mean Squared Error (RMSE) : 34416.9572

Mean Absolute Error (MAE) : 21512.8742

Mean Percentage Error (MPE) : 0.3556

Mean Absolute Percentage Error (MAPE) : 12.5695

*Validation Residual Plot*

Chart, histogram

Description automatically generated

The below table summarizes the performance of various machine learning models on validation dataset.

1. **Performance Evaluation and Comparison**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **MSE** | **RMSE** | **MAE** | **R2** | |
| Linear Regression | 1.182723e+09 | 34390.735 | 21610.877 | 0.791584 | |
| Lasso Regression | 1.183878e+09 | 34407.535 | 21603.39 | 0.831003 | |
| Ridge Regression | 1.508248e+09 | 38836.17 | 22705.818 | 0.784699 | |
| Bayesian Regression | 1.184527e+09 | 34416.957 | 21512.874 | 0.83091 | |
| Decision Tree | 1949306000 | 44150.94421 | 27318.75069 | 0.721739 | |
| ***Random Forest*** | | ***807492500*** | ***28416.41284*** | ***18047.65975*** | ***0.884731*** |

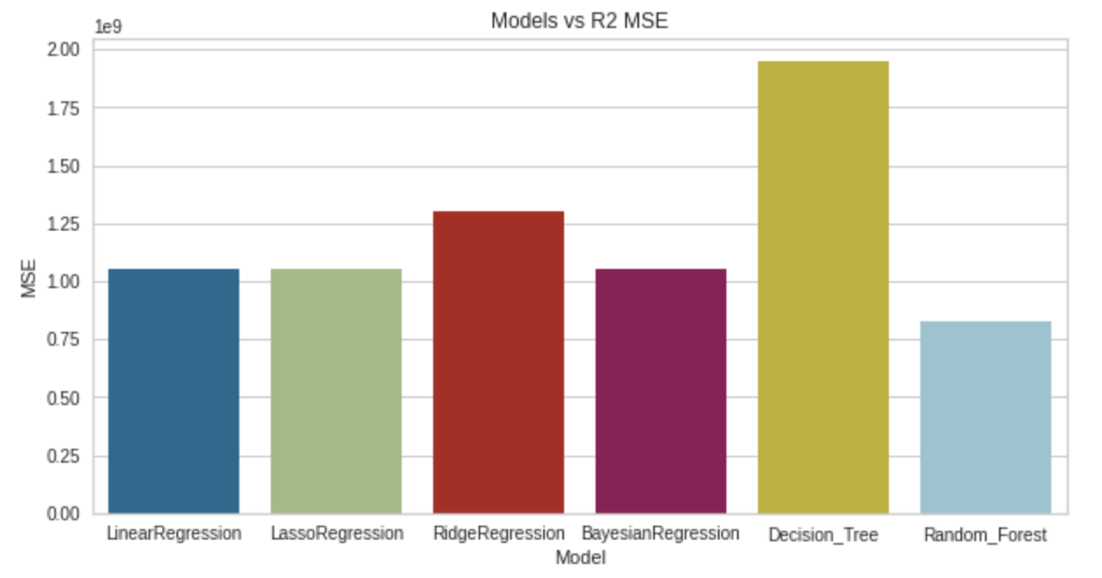
We have plotted the different measures of performance – R2 Score, RMSE, MSE, MAE for every model and selected the Random Forest as the best performing model.

* 1. ***Coefficient of Determination (R-Squared)***

Using the Barplot to visualize the different metrics



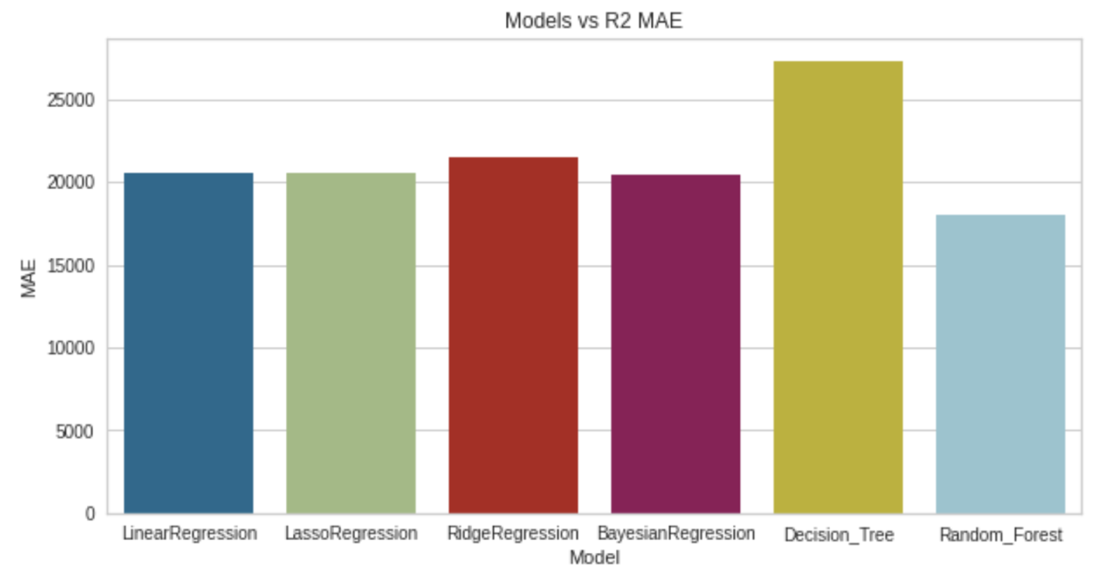
* ***As per the above figure, Random Forest has the highest R2- score.***
  1. ***Mean Squared Error***



* ***As per the above figure, Random Forest has the lowest MSE***
  1. ***Root Mean Squared Error***



* ***In the above barplot Random Forest can be seen with the lowest RMSE***
  1. ***Mean Absolute Error***



* ***Even MAE is lowest for Random Forest***

By assessing all the measures, we observed that the **Random Forest** has been performing in the best way by reducing all kind of errors.