2022

A Report on Housing: Price Prediction

(A case Study.....)

Submitted to – Khusboo Garg
(SME of internship batch-28)

SUBMITTED BY- ABHISHEK BEHERA
DATA SCIENCE INTERN AT FLIP ROBO TECHNOLOGIES PVT LTD
INTERNSHIP BATCH-28

Content

INTRODUCTION

Business Problem Framing & problem Statement

Houses are one of the necessary needs of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain.

Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies. Our problem is related to one such housing company.

We are required to model the price of houses with the available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.

Conceptual Background of the Domain Problem

A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below.

The company is looking at prospective properties to buy houses to enter the market. You are required to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not. For this company wants to know:

1. Which variables are important to predict the price of a variable?
2. How do these variables describe the price of the house?

Literature Review

Based on the sample data provided to us from our client database where we have understood that the company is looking at prospective properties to buy houses to enter the market. The data set explains it is a regression problem as we need to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not. Also, we have other independent features that would help to decide which all variables are important to predict the price of the variable and how do these variables describe the price of the house.

Objective

Our main objective of doing this project is to build a model to predict the house prices with the help of other supporting features. We are going to predict by using Machine Learning algorithms.

The sample data is provided to us from our client database. In order to improve the selection of customers, the client wants some predictions that could help them in further investment and improvement in selection of customers.

House Price Index is commonly used to estimate the changes in housing price. Since housing price is strongly correlated to other factors such as location, area, population, it requires other information apart from HPI to predict individual housing price.

There has been a considerably large number of papers adopting traditional machine learning approaches to predict housing prices accurately, but they rarely concern themselves with the performance of individual models and neglect the less popular yet complex models.

As a result, to explore various impacts of features on prediction methods, this paper will apply both traditional and advanced machine learning approaches to investigate the difference among several advanced models. This paper will also comprehensively validate multiple techniques in model implementation on regression and provide an optimistic result for housing price prediction.

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

We are building a model in Machine Learning to predict the actual value of the prospective properties and decide whether to invest in them or not. So, this model will help us to determine which variables are important to predict the price of variables & also how do these variables describe the price of the house. This will help to determine the price of houses with the available independent variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns.

Regression analysis is a set of statistical processes for estimating the relationships between a dependent variable (often called the 'outcome variable') and one or more independent variables (often called 'predictors', 'covariates', or 'features'). The most common form of regression analysis is linear regression, in which one finds the line (or a more complex linear combination) that most closely fits the data according to a specific mathematical criterion. For specific mathematical reasons this allows the researcher to estimate the conditional expectation of the dependent variable when the independent variables take on a given set of values.

Regression analysis is also a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables.

Data Sources and their formats

Data set provided by Flip Robo technologies was in the format of CSV (Comma Separated Values). The dimension of data is 1168 rows and 81 columns. There are 2 data sets that are given. One is training data and one is testing data.

1) Train file will be used for training the model, i.e., the model will learn from this file. It contains all the independent variables and the target variable. Size of training set: 1168 records.

2) Test file contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the test data. Size of test set: 292 records.

Data Pre-processing

Data pre-processing in Machine Learning refers to the technique of preparing (cleaning and organizing) the raw data to make it suitable for a building and training Machine Learning models. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre- process our data before feeding it into our model. Therefore, it is the first and crucial step while creating a machine learning model. I have used some following pre-processing steps:

Loading the training dataset as a data frame

Used pandas to set display I ensuring we do not see any truncated information

Checked the number of rows and columns present in our training dataset

Checked for missing data and the number of rows with null values

Verified the percentage of missing data in each column and decided to discard the one's that have more than 50% of null values

Dropped all the unwanted columns and duplicate data present in our data frame

Separated categorical column names and numeric column names in separate list variables for ease in visualization

Checked the unique values information in each column to get a gist for categorical data

Performed imputation to fill missing data using mean on numeric data and mode for categorical data columns

Used Pandas Profiling during the visualization phase along with pie plot, count plot, scatter plot and the others

With the help of ordinal encoding technique converted all object data type columns to numeric data type

Thoroughly checked for outliers and skewness information

With the help of heat map, correlation bar graph was able to understand the Feature vs Label relativity and insights on multi collinearity amongst the feature columns Separated feature and label data to ensure feature scaling is performed avoiding any kind of biasness Checked for the best random state to be used on our Regression Machine Learning model pertaining to the feature importance details. Finally created a regression model function along with evaluation metrics to pass through various model formats

Data Inputs- Logic- Output Relationships

When we loaded the training dataset, we had to go through various data pre processing steps to understand what was given to us and what we were expected to predict for the project. When it comes to logical part the domain expertise of understanding how real estate works and how we are supposed to cater to the customers came in handy to train the model with the modified input data. In Data Science community there is a saying "Garbage In Garbage Out" therefore we had to be very cautious and spent almost 80% of our project building time in understanding each and every aspect of the data how they were related to each other as well as our target label.

With the objective of predicting hosing sale prices accurately we had to make sure that a model was built that understood the customer priorities trending in the market imposing those norms when a relevant price tag was generated. I tried my best to retain as much data possible that was collected but I feel discarding columns that had lots of missing data was good. I did not want to impute data and then cause a biasness in the machine learning model from values that did not come from real people.

State the set of assumptions (if any) related to the problem under consideration

The assumption part for me was relying strictly on the data provided to me and taking into consideration that the separate training and testing datasets were obtained from real people surveyed for their preferences and how reasonable a price for a house with various features inclining to them were.

Hardware and Software Requirements and Tools Used

Hardware Specification:

RAM: 8 GB

CPU: AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx 2.10 GHz

GPU: AMD Radeon ™ Vega 8 Graphics and NVIDIA GeForce GTX 1650 Ti

Software Specification:

Programming language: Python

Distribution: Anaconda Navigator

Browser based language shell: Jupyter Notebook

Libraries/Packages Used:

Pandas, NumPy, matplotlib, seaborn, scikit-learn and pandas_profiling

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

I have used both statistical and analytical approaches to solve the problem which mainly includes the pre-processing of the data and EDA to check the correlation of independent and dependent features. Also, before building the model, I made sure that the input data is cleaned and scaled before it was fed into the machine learning models.

For this project we need to predict the sale price of houses, means our target column is continuous so this is a regression problem. I have used various regression algorithms and tested for the prediction. By doing various evaluations I have selected Extra Trees Regressor as best suitable algorithm for our final model as it is giving good r2-score and least difference in r2-score and CV-score among all the algorithms used. Other regression algorithms are also giving me good accuracy but some are over-fitting and some are with under-fitting the results which may be because of less amount of data.

In order to get good performance as well as accuracy and to check my model from over-fitting and under-fitting I have made use of the K-Fold cross validation and then hyper parameter tuned the final model.

Once I was able to get my desired final model I ensured to save that model before I loaded the testing data and started performing the data pre-processing as the training dataset and obtaining the predicted sale price values out of the Regression Machine Learning Model.

Testing of Identified Approaches (Algorithms)

The algorithms used on training and test data are as follows:

Linear Regression Model

Ridge Regularization Regression Model

Lasso Regularization Regression Model

Support Vector Regression Model

Decision Tree Regression Model

Random Forest Regression Model

K Nearest Neighbours Regression Model

Gradient Boosting Regression Model

Ada Boost Regression Model

Extra Trees Regression Model

Run and Evaluate selected models

I used a total of 10 Regression Models after choosing the random state amongst 1-1000 number. Then I even defined a function for getting the regression model trained and evaluated. The code for the models is listed below.

Random State:

Finding the best random state for building Regression Models

```
maxAccu=0
maxRS=0

for i in range(1, 1000):
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=i)
    lr=LinearRegression()
    lr.fit(X_train, Y_train)
    pred = lr.predict(X_test)
    r2 = r2_score(Y_test, pred)

    if r2>maxAccu:
        maxAccu=r2
        maxRS=i

print("Best R2 score is", maxAccu,"on Random State", maxRS)
```

```
Best R2 score is 0.8856355344351948 on Random State 340
```

Regression Model Function:

Machine Learning Model for Regression with Evaluation Metrics

```
# Regression Model Function

def reg(model, X, Y):
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=340)

    # Training the model
    model.fit(X_train, Y_train)

    # Predicting Y_test
    pred = model.predict(X_test)

    # RMSE - a lower RMSE score is better than a higher one
    rmse = mean_squared_error(Y_test, pred, squared=False)
    print("RMSE Score is:", rmse)

    # R2 score
    r2 = r2_score(Y_test, pred, multioutput='variance_weighted')*100
    print("R2 Score is:", r2)

    # Cross Validation Score
    cv_score = (cross_val_score(model, X, Y, cv=5).mean())*100
    print("Cross Validation Score:", cv_score)

    # Result of r2 score minus cv score
    result = r2 - cv_score
    print("R2 Score - Cross Validation Score is", result)
```

Linear Regression:

```
# Linear Regression Model

model=LinearRegression()
reg(model, X, Y)
```

```
RMSE Score is: 24876.373485691707
R2 Score is: 88.56355344351948
Cross Validation Score: 74.14529018813273
R2 Score - Cross Validation Score is 14.418263255386748
```

Ridge Regularization:

```
# Ridge Regularization

model=Ridge(alpha=1e-2, normalize=True)
reg(model, X, Y)
```

```
RMSE Score is: 24815.18998074428
R2 Score is: 88.6197402024921
Cross Validation Score: 74.45483255058483
R2 Score - Cross Validation Score is 14.16490765190727
```

Lasso Regularization:

```
# Lasso Regularization

model=Lasso(alpha=1e-2, normalize=True, max_iter=1e5)
reg(model, X, Y)
```

```
RMSE Score is: 24917.18385422086
R2 Score is: 88.52599905988447
Cross Validation Score: 74.1554161073105
R2 Score - Cross Validation Score is 14.370582952573969
```

Support Vector Regressor:

```
# Support Vector Regression

model=SVR(C=1.0, epsilon=0.2, kernel='poly', gamma='auto')
reg(model, X, Y)
```

```
RMSE Score is: 76592.05128076131
R2 Score is: -8.413750687388166
Cross Validation Score: -6.214424099645246
R2 Score - Cross Validation Score is -2.1993265877429202
```

Decision Tree Regressor:

```
# Decision Tree Regressor

model=DecisionTreeRegressor(criterion="poisson", random_state=111)
reg(model, X, Y)
```

```
RMSE Score is: 57727.62379648374
R2 Score is: 38.41366921116711
Cross Validation Score: 41.26696984258857
R2 Score - Cross Validation Score is -2.8533006314214617
```

Random Forest Regressor:

```
# Random Forest Regressor

model=RandomForestRegressor(max_depth=2, max_features="sqrt")
reg(model, X, Y)
```

```
RMSE Score is: 40625.4396140173
R2 Score is: 69.49906765983303
Cross Validation Score: 64.61456200338246
R2 Score - Cross Validation Score is 4.884505656450571
```

K Nearest Neighbours Regressor:

```
# K Neighbors Regressor

KNeighborsRegressor(n_neighbors=2, algorithm='kd_tree')
reg(model, X, Y)
```

```
RMSE Score is: 40466.730494501026
R2 Score is: 69.73691471173798
Cross Validation Score: 64.42251920085333
R2 Score - Cross Validation Score is 5.314395510884651
```

Gradient Boosting Regressor:

```
# Gradient Boosting Regressor

model=GradientBoostingRegressor(loss='quantile', n_estimators=200, max_depth=5)
reg(model, X, Y)
```

```
RMSE Score is: 34539.463803694656
R2 Score is: 77.95306863017093
Cross Validation Score: 78.2983938466606
R2 Score - Cross Validation Score is -0.34532521648966963
```

Ada Boost Regressor:

```
# Ada Boost Regressor

model=AdaBoostRegressor(n_estimators=300, learning_rate=1.05, random_state=42)
reg(model, X, Y)
```

```
RMSE Score is: 31820.346272586143
R2 Score is: 81.28771728128767
Cross Validation Score: 79.16566313678824
R2 Score - Cross Validation Score is 2.1220541444994296
```

Extra Trees Regressor:

```
# Extra Trees Regressor

model=ExtraTreesRegressor(n_estimators=200, max_features='sqrt', n_jobs=6)
reg(model, X, Y)
```

```
RMSE Score is: 23816.88408105236
R2 Score is: 89.51696939850329
Cross Validation Score: 84.8703100074016
R2 Score - Cross Validation Score is 4.646659391101693
```

Key Metrics for success in solving problem under consideration

The key metrics used here were r2_score, cross_val_score, MAE, MSE and RMSE. We tried to find out the best parameters and also to increase our scores by using Hyper parameter Tuning and we will be using Grid Search CV method.

1. Cross Validation:

Cross-validation helps to find out the over fitting and under fitting of the model. In the cross validation the model is made to run on different subsets of the dataset which will get multiple measures of the model. If we take 5 folds, the data will be divided into 5 pieces where each part being 20% of full dataset. While running the Cross-validation the 1st part (20%) of the 5 parts will be kept out as a holdout set for validation and everything else is used for training data. This way we will get the first estimate of the model quality of the dataset.

In the similar way further iterations are made for the second 20% of the dataset is held as a holdout set and remaining 4 parts are used for training data during process. This way we will get the second estimate of the model quality of the dataset. These steps are repeated during the cross-validation process to get the remaining estimate of the model quality.

2. R2 Score:

It is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted.

3. Mean Squared Error (MSE):

MSE of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors — that is, the average squared difference between the estimated values and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss. RMSE is the Root Mean Squared Error.

4. Mean Absolute Error (MAE):

MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

5. Hyper parameter Tuning:

There is a list of different machine learning models. They all are different in some way or the other, but what makes them different is nothing but input parameters for the model. These input parameters are named as Hyper parameters. These hyper parameters will define the architecture of the model, and the best part about these is that you get a choice to select these for your model. You must select from a specific list of hyper parameters for a given model as it varies from model to model.

We are not aware of optimal values for hyper parameters which would generate the best model output. So, what we tell the model is to explore and select the optimal model architecture automatically. This selection procedure for hyper parameter is known as Hyper parameter Tuning. We can do tuning by using Grid Search CV.

Grid Search CV is a function that comes in Scikit-learn (or SK-learn) model selection package. An important point here to note is that we need to have Scikit-learn library installed on the computer. This function helps to loop through predefined hyper parameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyper parameters.

Hyper parameter tuning

```
# Choosing Extra Trees Regressor

fmod_param = {'n_estimators' : [100, 200, 300],
              'criterion' : ['squared_error', 'mse', 'absolute_error', 'mae'],
              'n_jobs' : [-2, -1, 1],
              'random_state' : [42, 111, 340]
             }
```

```
GSCV = GridSearchCV(ExtraTreesRegressor(), fmod_param, cv=5)
```

I am using the Grid Search CV method for hyper parameter tuning my best model.

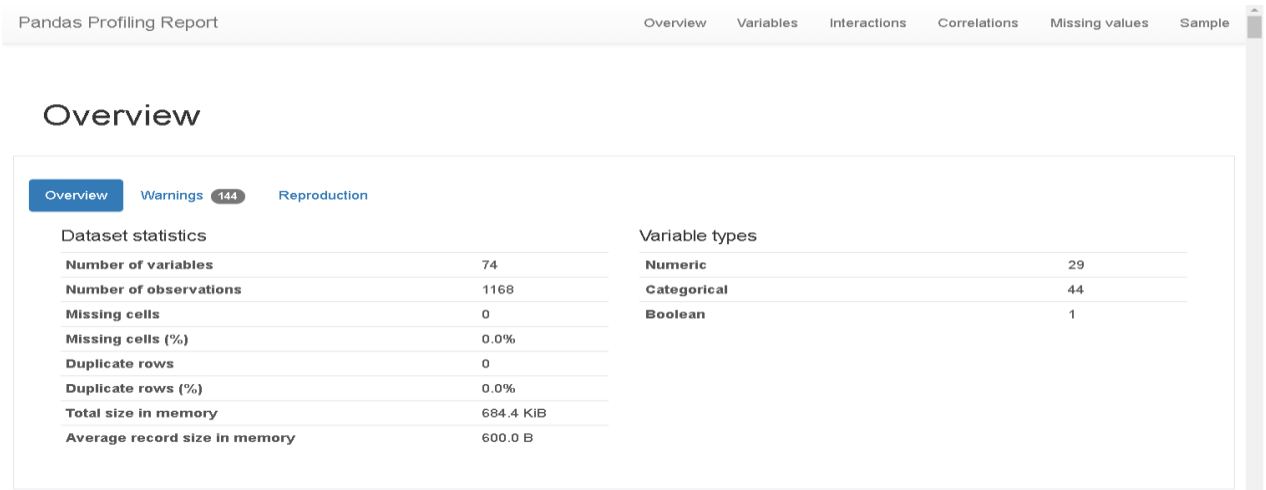
```
Final_Model = ExtraTreesRegressor(criterion='mse', n_estimators=100, n_jobs=-2, random_state=42)
Model_Training = Final_Model.fit(X_train, Y_train)
fmod_pred = Final_Model.predict(X_test)
fmod_r2 = r2_score(Y_test, fmod_pred, multioutput='variance_weighted')*100
print("R2 score for the Best Model is:", fmod_r2)
```

```
R2 score for the Best Model is: 83.64443386563624
```

It is possible that there are times when the default parameters perform better than the parameters list obtained from the tuning and it only indicates that there are more permutations and combinations that one needs to go through for obtaining better results.

Visualizations

I used pandas profiling to get the over viewed visualization on the pre-processed data. pandas-profiling is an open-source Python module with which we can quickly do an exploratory data analysis with just a few lines of code. It generates interactive reports in web format that can be presented to any person, even if they don't know programming. It also offers report generation for the dataset with lots of features and customizations for the report generated. In short, what pandas-profiling does is save us all the work of visualizing and understanding the distribution of each variable. It generates a report with all the information easily available.



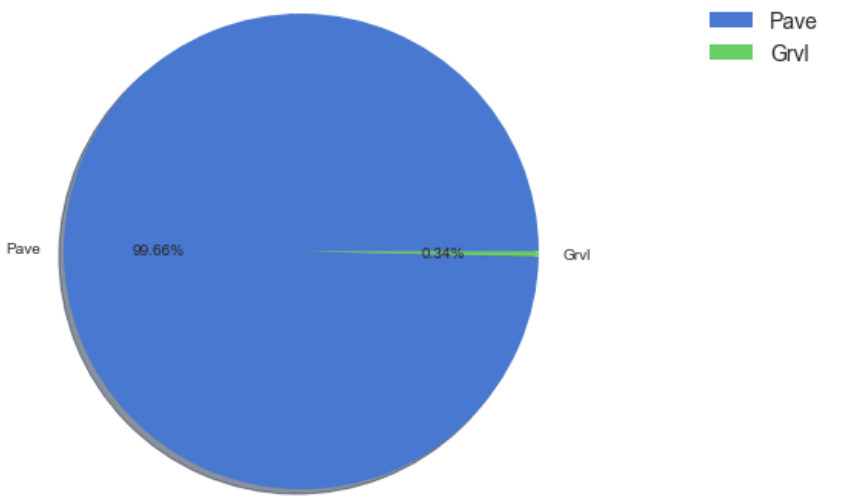
Then I have created pie plots, count plots and scatter plots to get further visual insights on our training dataset feature values.

Code:

```
plt.style.use('seaborn-muted')
def generate_pie(x):
    plt.style.use('seaborn-white')
    plt.figure(figsize=(10,5))
    plt.pie(x.value_counts(), labels=x.value_counts().index, shadow=True, autopct='%1.2f%%')
    plt.legend(prop={'size':14})
    plt.axis('equal')
    plt.tight_layout()
    return plt.show()

for i in train_df[single]:
    print(f"Single digit category column name:", i)
    generate_pie(train_df[i])
```

Output:



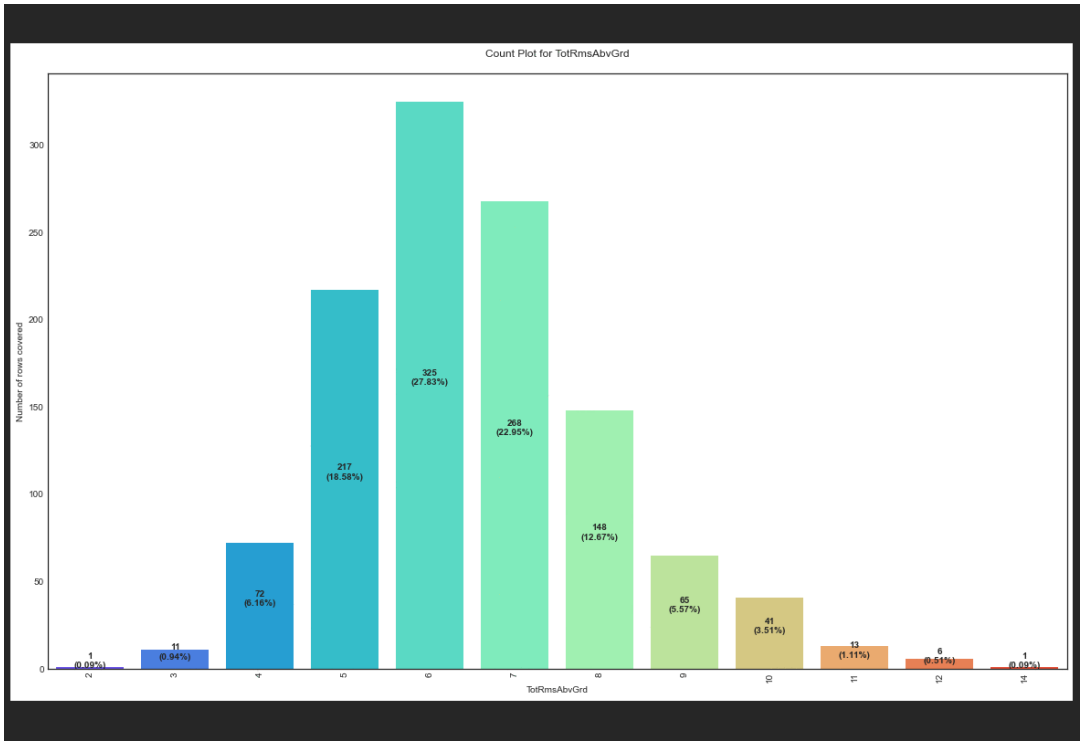
Code:

```
for col in train_df[double]:
    plt.figure(figsize=(20,12))
    col_name = col
    values = train_df[col_name].value_counts()
    index = 0
    ax = sns.countplot(train_df[col_name], palette="rainbow")

    for i in ax.patches:
        h = i.get_height() # getting the count of each value
        t = len(train_df[col_name]) # getting the total number of records using length
        s = f"{h}\n({round(h*100/t,2)}%)" # making the string for displaying in count bar
        plt.text(index, h/2, s, ha="center", fontweight="bold")
        index += 1

    plt.title(f"Count Plot for {col_name}\n")
    plt.xlabel(col_name)
    plt.ylabel(f"Number of rows covered")
    plt.xticks(rotation=90)
    plt.show()
```

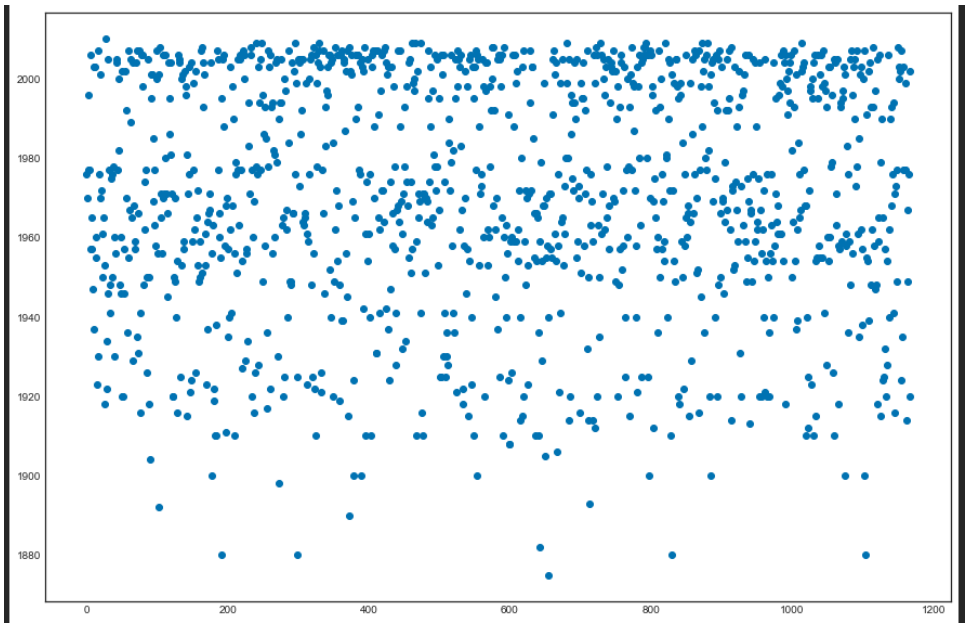
Output:



Code:

```
plt.style.use('seaborn-colorblind')
for j in train_df[triple]:
    plt.figure(figsize=(15,10))
    print(f"Scatter plot for {j} column with respect to the rows covered ->")
    plt.scatter(train_df.index, train_df[j])
    plt.show()
```

Output:



Result Interpretation

Visualizations:

It helped to understand the correlation between independent and dependent features. Also, helped the with feature importance and to check for multi co linearity issues. Detected outliers/skewness with the help of box plot and distribution plot. I got to know the count of a particular category for each feature by using count plot and most importantly with predicted target value distribution as well as scatter plot helped me to select the best model.

Pre-processing: Basically, before building the model the dataset should be cleaned and scaled by performing few steps. As I mentioned above in the pre-processing steps where all the important features are present in the dataset and ready for model building.

Model Creation:

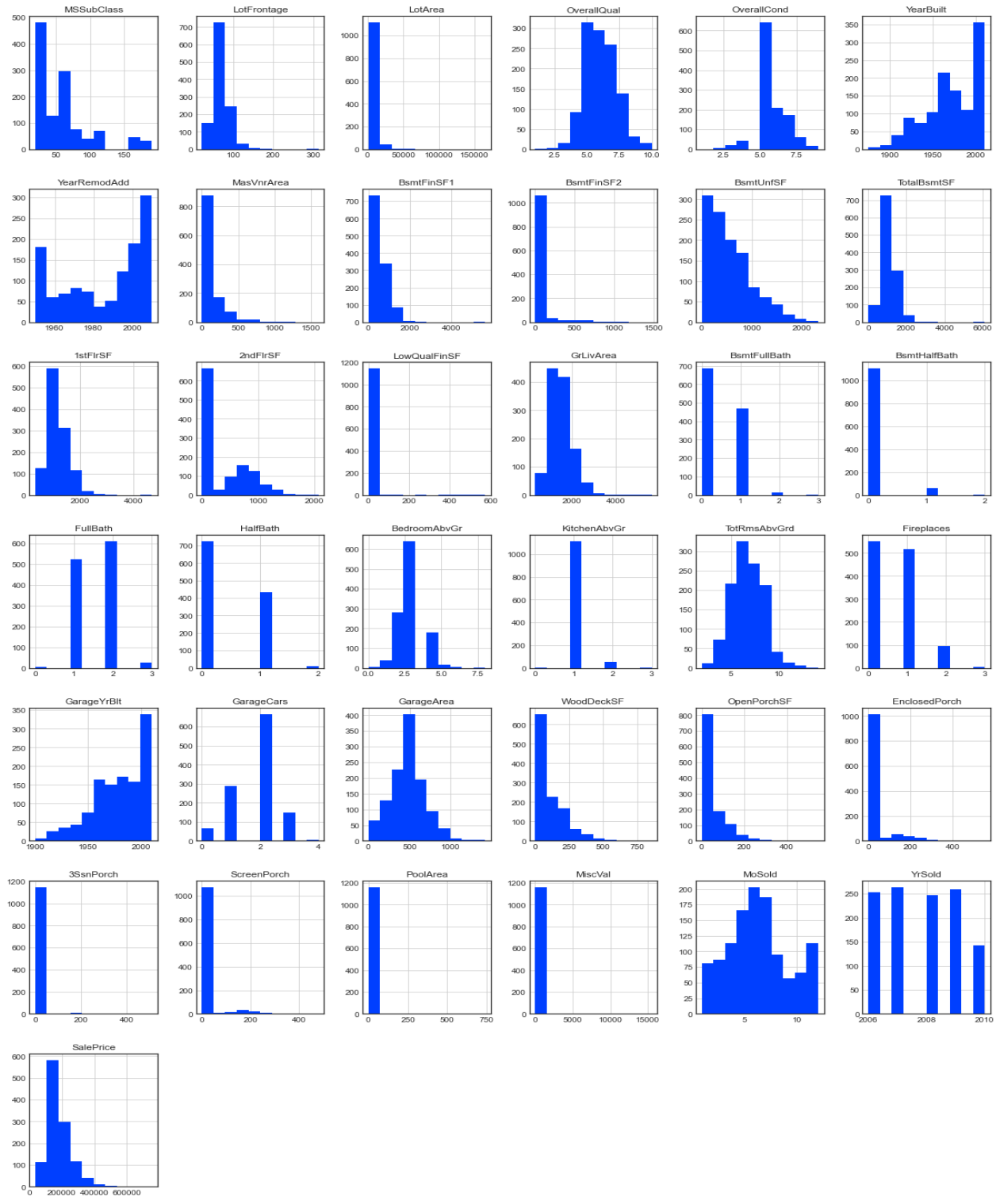
Now, after performing the train test split, I have x_train, x_test, y_train & y_test, which are required to build Machine learning models. I have built multiple regression models to get the best R2 score, MSE, RMSE & MAE out of all the models.

CONCLUSION

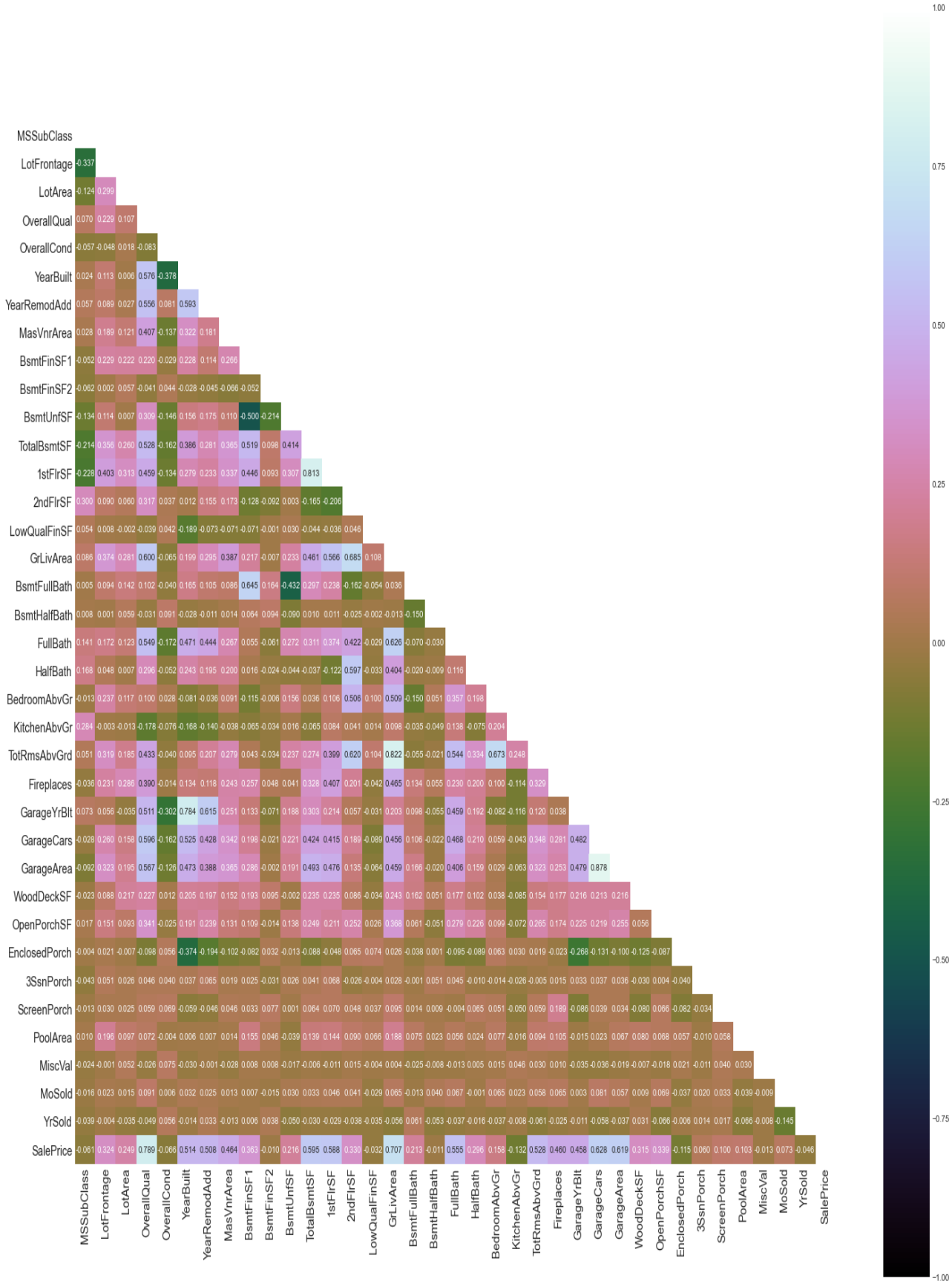
Key Findings and Conclusions of the Study

It is observed that all the encoded dataset information by plotting various graphs and visualised further insights.

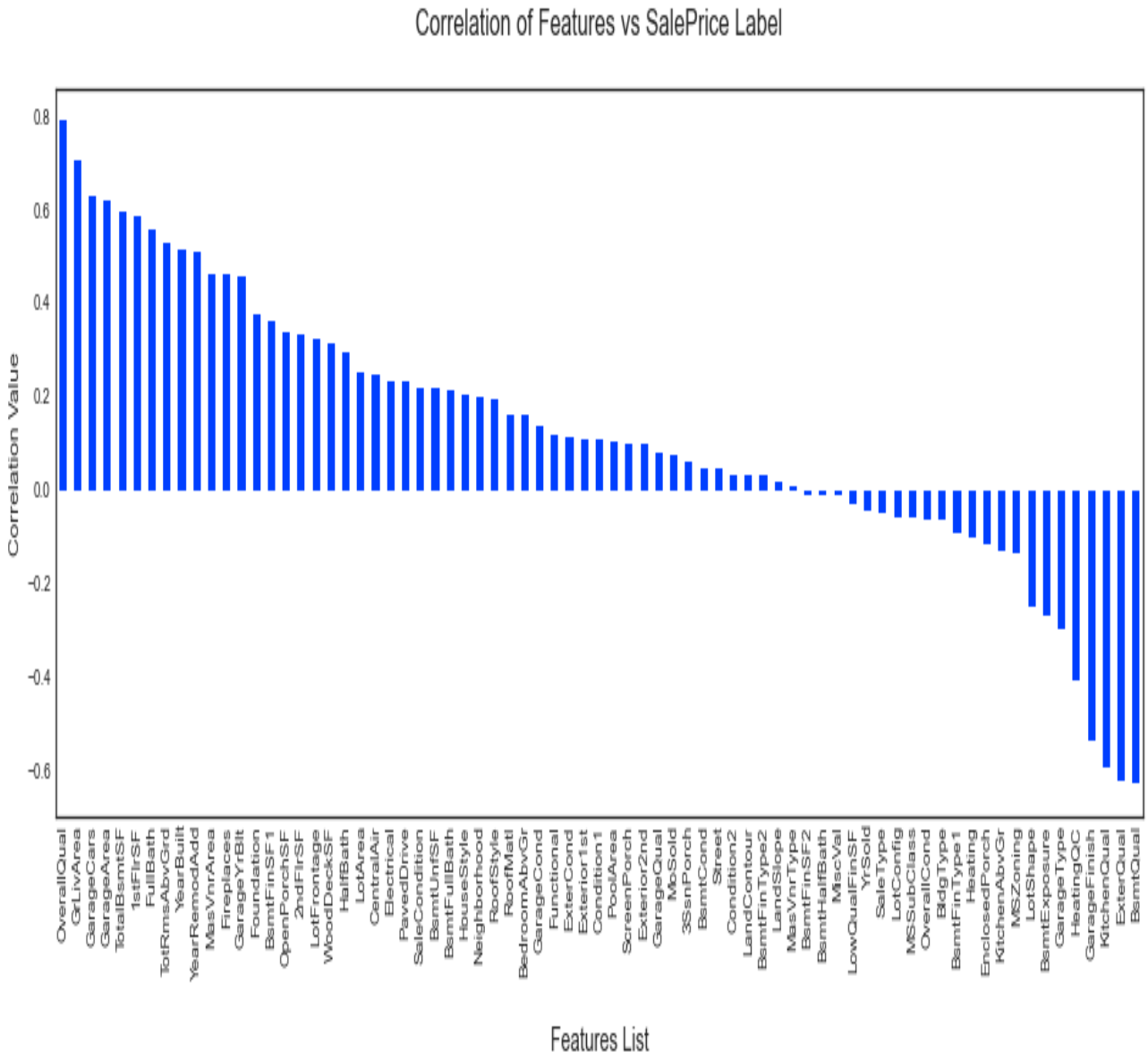
Histogram:



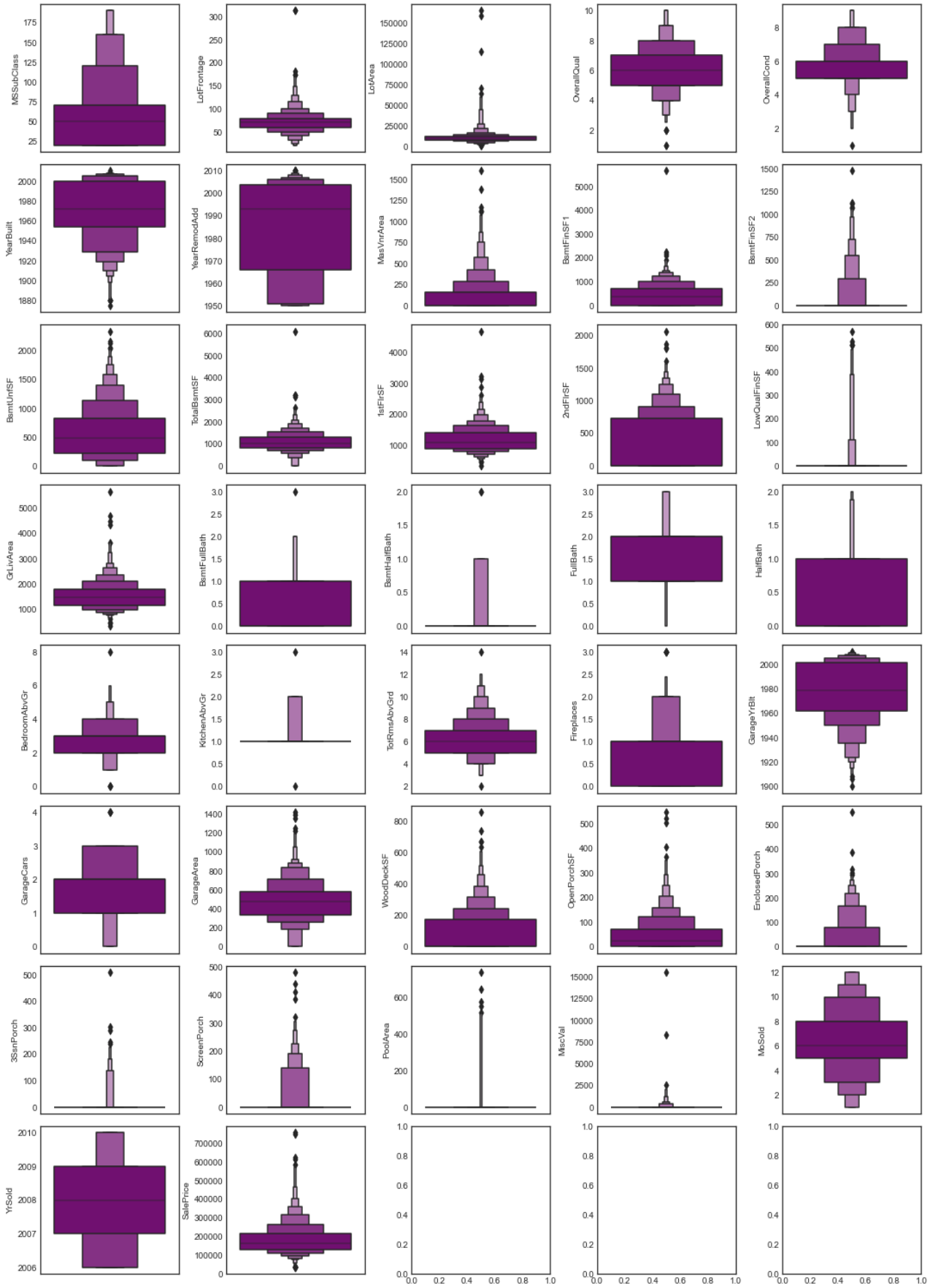
Heat map:


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
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00

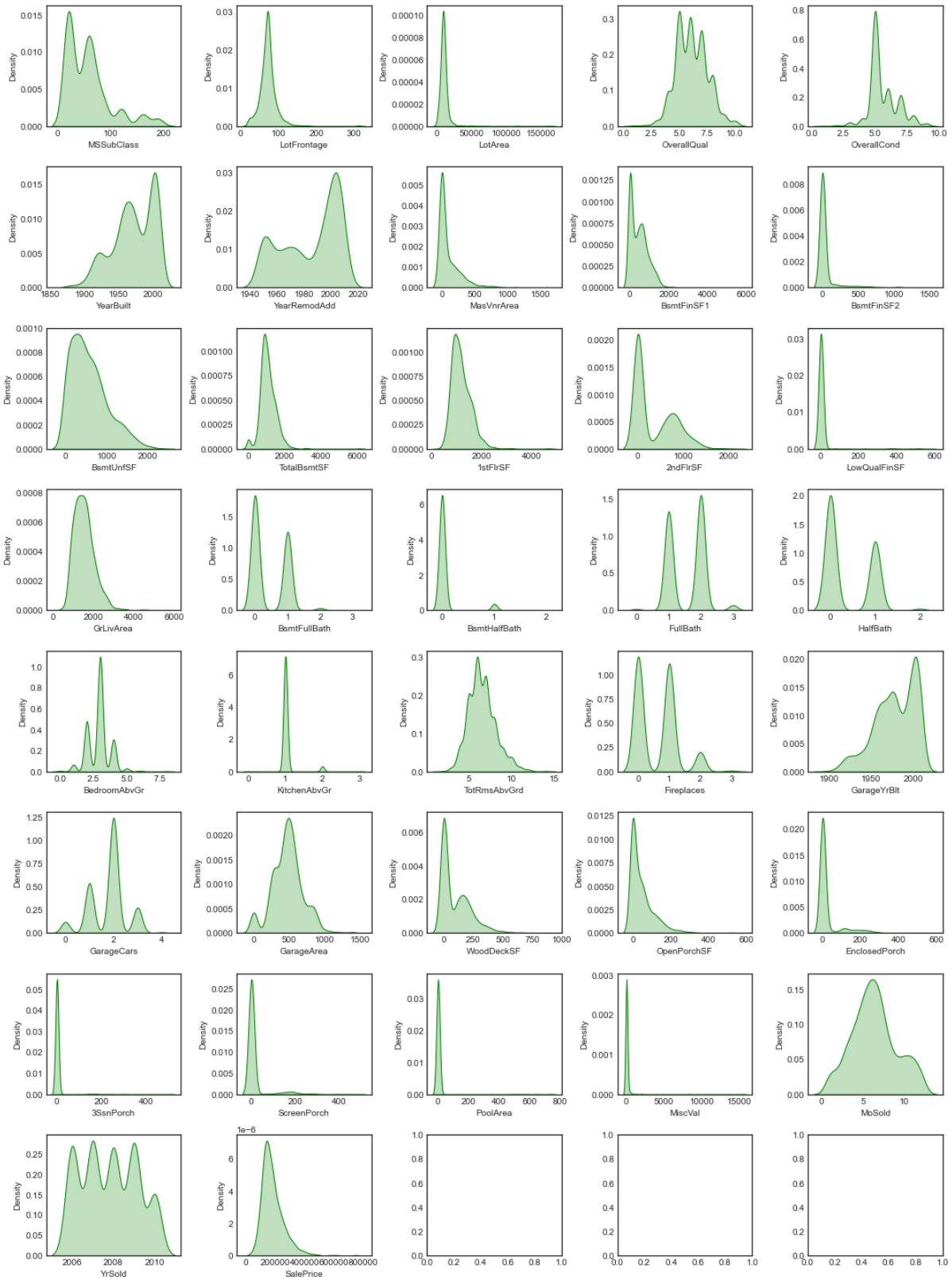
Correlation:



Boxen Plot:


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

Distribution Plot:



After completion of Post model building and choosing the appropriate model I went ahead and loaded the testing dataset. After applying all the data pre processing steps as the training dataset I was then able to get the predicted sale price results. Since the values were in

array format, converted them into a data frame and merged it with the original testing data frame that consisted only our feature columns. Once the testing dataset with feature columns and predicted label was formed, I exported the values in a comma separated values file to be accessed as needed.

Learning Outcomes of the Study in respect of Data Science

The above study helps one to understand the business of real estate. How the price is changing across the properties. With the Study we can tell how multiple real estate amenities like swimming pool, garage, pavement and lawn size of Lot Area, and type of Building raise decides the cost. With the help of the above analysis, one can sketch the needs of a property buyer and according to need we can project the price of the property.

Limitations of this work and Scope for Future Work

During this project I have faced a problem of low amount of data. Many columns are with same entries in more than 80% of rows which lead to reduction in our model performance. One more issue is there are large number of missing values presents in this data set, so we have to fill those missing values in correct manner. We can still improve our model accuracy with some feature engineering and by doing some extensive hyper parameter tuning on it.

****

References:

1) https://www.google.com/
2) https://www.youtube.com/
3) https://scikit-learn.org/stable/user_guide.html
4) https://github.com/
5) https://www.kaggle.com/
6) https://medium.com/
7) https://towardsdatascience.com/
8) https://www.analyticsvidhya.com/