HOUSING: PRICE PREDICTION

Submitted by:

Vijay Ashley Rodrigues K.

ACKNOWLEDGMENT

- I would like to thank FlipRobo Technologies for providing me this opportunity and guidance throughout the project and all the steps that are implemented.
- I have primarily referred to various articles scattered across various websites for the purpose of getting an idea on Housing related in general.
- I would like to thank the technical support team also for helping me out and reaching out to me on clearing all my doubts as early as possible
- I would like to thank my project SME Sajid Choudhary for providing the flexibility in time and also for giving us guidance in creating the project.
- The following are some of the articles I referred to in this project.

- https://bestplaces.net/cost_of_living/state/iowa
- https://www.investopedia.com/articles/economics/09/financial-crisis-review.asp
- https://onproperty.com.au/invest-in-us-or-australia/
- https://www.noradarealestate.com/blog/housing-market-predictions/

INTRODUCTION

- Business Problem Framing

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.

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:

• Which variables are important to predict the price of variable?

• How do these variables describe the price of the house?

Business Goal:

You 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

    - Houses are one of the important needs of every single individual all throughout the planet and in this way lodging and land market is one of the business sectors which is one of the significant supporters on the planet's economy.
    - It is an exceptionally enormous market what's more, there are different organizations working in the area. Information science comes as a vital instrument to tackle issues in the area to help the organizations increment their general income, benefits, further developing their promoting procedures and zeroing in on changing patterns in house deals and buys. Prescient demonstrating, Market blend displaying, suggestion frameworks are a portion of the AI strategies utilized for accomplishing the business objectives for lodging organizations. Our concern is identified with one such lodging organization.

- Motivation for the Problem Undertaken

    - Record-low home loan rates and deficiency of stock are keeping the US real estate market solid to the extent request is thought of. Home costs have been flooding month-over-month breaking new records. While moderateness issues deteriorate, low home loan rates, developing investment funds, and a fortifying position market join to save homeownership accessible for some likely purchasers.

- For a very long time, home costs have been filling in the mid-single digits. The new cost expansions in the twofold digits mirror the juncture of phenomenal interest and steadily low inventory. Costs are ascending as there is a lot of capital uninvolved, just as exceptionally modest home loan rates.
- Because of millennial homeownership and different reasons, for example, developing development expenses and land financial backers gathering up starter houses, lodging supply is by and by at its most minimal level since the 1970s.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

    - You are needed to show the cost of houses with the accessible autonomous factors. This model will then, at that point be utilized by the administration to see how precisely the costs shift with the factors. They can in like manner control the system of the firm and focus on regions that will yield significant yields. Further, the model will be a decent way for the the executives to comprehend the estimating elements of another market.
    - A US-based lodging organization named Surprise Housing has chosen to enter the Australian market. The organization employments information investigation to buy houses at a cost underneath their real qualities and flip them at a more exorbitant cost. For the equivalent reason, the organization has gathered an informational collection from the offer of houses in Australia. The information is given in the CSV record beneath.

- Data Sources and their formats

    - The dataset is provided by FlipRobo and is available for academic purpose only and not for any kind of commercial activities.
    - The dataset contains the Housing related data with 1460 records (rows) and 81 features (columns).
    - The file is in .csv format and we have 2 files. One for train data and another test data.

- The dataset is in both numerical and categorical data.

	I	J	K	L	M	N
1	LandCont	Utilities	LotConfig	LandSlope	Neighborh	Conditio
2	HLS	AllPub	Corner	Gtl	StoneBr	Norm
3	Lvl	AllPub	CulDSac	Gtl	StoneBr	Norm
4	Lvl	AllPub	Inside	Gtl	CollgCr	Norm
5	Bnk	AllPub	Inside	Gtl	Crawfor	Norm

- Data Pre-processing Done

1. <u>Acquire the dataset</u>

   - We have received this dataset from FlipRobo Technologies which is related to a Housing data company from US.
   - The sample data is provided to us from our client database. It is hereby given for this exercise. The customer wishes to get into Australian market based on US Housing model.

2. <u>Import all the crucial libraries</u>

   - For this project I have used the following major libraries like Pandas-profiling, Pandas, Matplotlib and Seaborn that are used for EDA or any other pre-processing done in this scenario.

```
: import numpy as np
  import pandas as pd

  import matplotlib.pyplot as plt
  import seaborn as sns

  from sklearn.tree import DecisionTreeRegressor
  from sklearn.ensemble import RandomForestRegressor
  from sklearn.ensemble import ExtraTreesRegressor
  from sklearn.neighbors import KNeighborsRegressor
  from sklearn.experimental import enable_hist_gradient_boosting
  from sklearn.ensemble import HistGradientBoostingRegressor
  from sklearn.ensemble import GradientBoostingRegressor

  from sklearn.metrics import mean_squared_error, mean_absolute_error
  from statsmodels.stats.outliers_influence import variance_inflation_factor

  from sklearn.model_selection import train_test_split
  from scipy.stats import zscore
  from sklearn.metrics import r2_score

  import warnings
  warnings.filterwarnings("ignore")
```

3. Import the dataset

- The dataset is in CSV format and it is imported using Pandas library in Jupyter Notebook.

```
In [2]: pd. set_option("display.max_columns", None)
        pd. set_option("display.max_rows", None)
        df_train = pd.read_csv("D:\housing_train.csv")
        df_test = pd.read_csv("D:\housing_test.csv")
```

- The statement **pd. set_option("display.max_columns", None)** simply allows us to physically see all the feature columns.
- The statement **pd. set_option("display.max_rows", None)** simply allows us to physically see all the feature rows.
- By default, Jupyter Notebook doesn't display all the rows and columns at the same time and only selected portion from starting and ending of dataset are displayed.

4. Identifying and handling the missing values

- The dataset appears to have a total of 1460 records (rows) and 81 features (columns) including 1 target column "SalesPrice"

```
In [5]: print("Train Data:", df_train.shape)
        print("Test Data:", df_test.shape)

        Train Data: (1168, 81)
        Test Data: (292, 80)
```

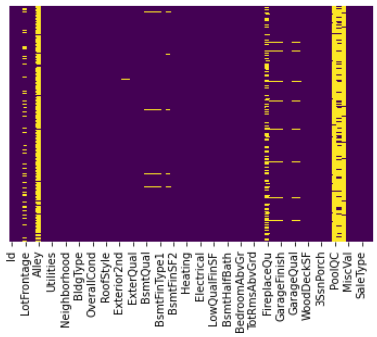
- The following columns are dropped as there are more number of missing values than it could be replaced.

```
In [11]: df_train.drop(columns = ["Alley", "PoolQC", "Fence", "MiscFeature"], axis=1, inplace=True)

In [12]: df_test.drop(columns = ["Alley", "PoolQC", "Fence", "MiscFeature"], axis=1, inplace=True)
```

- We have missing values and following plot shows the same in both train and test dataset.

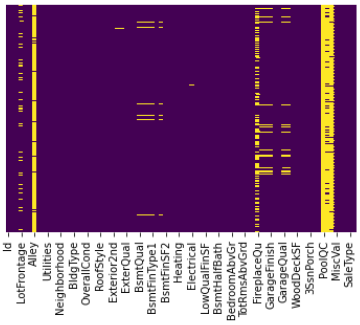
```
sns.heatmap(df_train.isnull(), yticklabels=False, cbar=False, cmap="viridis")
```

```
<AxesSubplot:>
```




```
sns.heatmap(df_test.isnull(), yticklabels=False, cbar=False, cmap="viridis")
```

```
<AxesSubplot:>
```



Missing values for both Test and Train dataset are filled accordingly. Numerical values are filled with mean and categorical values are filled with mode.

[14]:
```
# Fill the columns with mean as its continous data

df_train["LotFrontage"].fillna(df_train["LotFrontage"].mean(), inplace=True)
df_train["MasVnrArea"].fillna(df_train["MasVnrArea"].mean(), inplace=True)
```

[15]:
```
# Fill the columns with mode as its categorical data

df_train["MasVnrType"].fillna(df_train["MasVnrType"].value_counts().index[0], inplace=True)
df_train["BsmtQual"].fillna(df_train["BsmtQual"].value_counts().index[0], inplace=True)
df_train["BsmtCond"].fillna(df_train["BsmtCond"].value_counts().index[0], inplace=True)
df_train["BsmtExposure"].fillna(df_train["BsmtExposure"].value_counts().index[0], inplace=True)
df_train["BsmtFinType1"].fillna(df_train["BsmtFinType1"].value_counts().index[0], inplace=True)
df_train["BsmtFinType2"].fillna(df_train["BsmtFinType2"].value_counts().index[0], inplace=True)
df_train["FireplaceQu"].fillna(df_train["FireplaceQu"].value_counts().index[0], inplace=True)
df_train["GarageType"].fillna(df_train["GarageType"].value_counts().index[0], inplace=True)
df_train["GarageYrBlt"].fillna(df_train["GarageYrBlt"].value_counts().index[0], inplace=True)
df_train["GarageFinish"].fillna(df_train["GarageFinish"].value_counts().index[0], inplace=True)
df_train["GarageQual"].fillna(df_train["GarageQual"].value_counts().index[0], inplace=True)
df_train["GarageCond"].fillna(df_train["GarageCond"].value_counts().index[0], inplace=True)
```

:
```
# Fill the columns with mean as its continous data

df_test["LotFrontage"].fillna(df_test["LotFrontage"].mean(), inplace=True)
df_test["MasVnrArea"].fillna(df_test["MasVnrArea"].mean(), inplace=True)
```

:
```
# Fill the columns with mode as its categorical data

df_test["MasVnrType"].fillna(df_test["MasVnrType"].value_counts().index[0], inplace=True)
df_test["BsmtQual"].fillna(df_test["BsmtQual"].value_counts().index[0], inplace=True)
df_test["BsmtCond"].fillna(df_test["BsmtCond"].value_counts().index[0], inplace=True)
df_test["BsmtExposure"].fillna(df_test["BsmtExposure"].value_counts().index[0], inplace=True)
df_test["BsmtFinType1"].fillna(df_test["BsmtFinType1"].value_counts().index[0], inplace=True)
df_test["BsmtFinType2"].fillna(df_test["BsmtFinType2"].value_counts().index[0], inplace=True)
df_test["FireplaceQu"].fillna(df_test["FireplaceQu"].value_counts().index[0], inplace=True)
df_test["GarageType"].fillna(df_test["GarageType"].value_counts().index[0], inplace=True)
df_test["GarageYrBlt"].fillna(df_test["GarageYrBlt"].value_counts().index[0], inplace=True)
df_test["GarageFinish"].fillna(df_test["GarageFinish"].value_counts().index[0], inplace=True)
df_test["GarageQual"].fillna(df_test["GarageQual"].value_counts().index[0], inplace=True)
df_test["GarageCond"].fillna(df_test["GarageCond"].value_counts().index[0], inplace=True)
df_test["Electrical"].fillna(df_test["Electrical"].value_counts().index[0], inplace=True)
```

- But if we observe, the std values are lower than the mean values for almost all the features and data appears to be normally distributed.

```
df_train.describe()
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCon
count	1168.000000	1168.000000	954.00000	1168.000000	1168.000000	1168.00000
mean	724.136130	56.767979	70.98847	10484.749144	6.104452	5.59589
std	416.159877	41.940650	24.82875	8957.442311	1.390153	1.12434
min	1.000000	20.000000	21.00000	1300.000000	1.000000	1.00000
25%	360.500000	20.000000	60.00000	7621.500000	5.000000	5.00000
50%	714.500000	50.000000	70.00000	9522.500000	6.000000	5.00000
75%	1079.500000	70.000000	80.00000	11515.500000	7.000000	6.00000
max	1460.000000	190.000000	313.00000	164660.000000	10.000000	9.00000

5. Encoding the categorical data

- I have used LabelEncoder as the data is categorical and is not ordinal in nature.

```
In [18]: from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()

# Encode the training dataset

df_train.MSZoning = encoder.fit_transform(df_train.MSZoning)
df_train.Street = encoder.fit_transform(df_train.Street)
df_train.LotShape = encoder.fit_transform(df_train.LotShape)
df_train.LandContour = encoder.fit_transform(df_train.LandContour)
df_train.Utilities = encoder.fit_transform(df_train.Utilities)
df_train.LotConfig = encoder.fit_transform(df_train.LotConfig)
df_train.LandSlope = encoder.fit_transform(df_train.LandSlope)
df_train.Neighborhood = encoder.fit_transform(df_train.Neighborhood)
df_train.Condition1 = encoder.fit_transform(df_train.Condition1)
df_train.Condition2 = encoder.fit_transform(df_train.Condition2)
df_train.BldgType = encoder.fit_transform(df_train.BldgType)
df_train.HouseStyle = encoder.fit_transform(df_train.HouseStyle)
```

6. Splitting the dataset

- Splitting up of dataset between x (features) and y (target column)

```
# train dataset with featurs only
x = df_train.drop(columns = ["SalePrice"], axis=1)

y = df_train["SalePrice"]

# test dataset with featurs only
x1 = df_test
```

- Split the dataset into train and test data set.
- I have chosen 200 random state and 30% of data is divided in text dataset.

```
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.30, random_state = 200)
```

7. Feature scaling

- Finding variance inflation factor in each scaled column
- This gives us relationship between feature vs feature and we can drop if necessary to avoid multicollinearity.
- From the below observation, I have not considered this step.
- Also without dropping the features I got even better accuracy in the final model.
- Let's us now Scale the data for further processing.
- we have used StandardScaler for further scaling up of data

```
: from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)

x_scaled1 = scaler.fit_transform(x1)
```

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

- Data contains 1460 entries each having 81 variables.
- Data contains Null values. You need to treat them using the domain knowledge and your own understanding.
- Extensive EDA has to be performed to gain relationships of important variable and price.
- Data contains numerical as well as categorical variable. You need to handle them accordingly.

- You have to build Machine Learning models, apply regularization and determine the optimal values of HyperParameters.
- you need to find important features which affect the price positively or negatively.
- Two datasets are being provided to you (test.csv, train.csv). You will train on train.csv dataset and predict on
- test.csv file.

- Hardware and Software Requirements and Tools Used

Hardware / Software specifications

Windows 10 64bit

Anaconda 2021.05

Python version – Python 3.9.5

Jupyter Notebook 6.4 and Google Colab

Pandas-profiling – package that performs simple EDAs for distribution of variables. This helped me give basic details about what the dataset consists of and its correlations with each other. The report is displayed using **.to_widgets()**

Pandas – This is used in the data manipulation, processing and cleaning and also to get description, stats and almost everywhere in the project.

Matplotlib and **Seaborn** - Majority of the data visualizations are plotted using Seaborn and to some extent only Matplotlib is used here.

LabelEncoder - I have used this **Skipy** library to convert all the non-ordered categorical data into numerical data. In our case it's only one feature "pcircle".

train_test_split module from **sklearn.model_selection** to split the data into train and test and then used **StandardScaler** to bring the values to one level before imputing to model.

Warnings: I have used "ignore" to avoid the general errs that may occur and used "FutureWarning" to avoid errors that I got when running algorithms on Google Colab. To have a generic and efficient notebook file I used this as well.

Sciypy Used xgboost to import XGBClassifier and remaining algorithms including ensemble are part of Scipy.

Explained Dashboard – This library helps in the creation of dashboards automatically specially to help laymen understand how the model works. I have used this to explain my ML model in the end of this project.

Model/s Development and Evaluation

- Testing of Identified Approaches (Algorithms)

For the model building I have considered the following 6 algorithms for the training and testing.

DecisionTreeRegressor
RandomForestRegressor
ExtraTreesRegressor
KNeighborsRegressor
HistGradientBoostingRegressor
GradientBoostingRegressor

- Run and Evaluate selected models

  - I have used a total of 6 machine learning algorithms to find the best and suited model which also includes ensemble algorithms.
  - I have considered Adjusted R2 Score for model evaluation.
  - But we cannot simply rely on these scores as we cannot have any scope for assumption. Hence post this I have also performed Cross Validation for all these algorithms to find the estimated performance metric when it's actually used in production.

1) DecisionTreeRegressor

  - From the below algorithm we can see the accuracy score for DecisionTreeRegressor is approximately **100%** better.

```
from sklearn.tree import DecisionTreeRegressor

dt_reg = DecisionTreeRegressor()
dt_reg.fit(x_train,y_train)

y_pred = dt_reg.predict(x_test)

print("Adjusted R2 squared : ",dt_reg.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  1.0
Mean Absolute Error (MAE):  25578.723646723647
Mean Squared Error (MSE):  1359837277.3618233
Root Mean Squared Error (RMSE):  36875.97154464982
```

2) RandomForestRegressor

- From the below algorithm we can see the accuracy score for RandomForestRegressor is approximately **97.64%** better.

```
from sklearn.ensemble import RandomForestRegressor

rf_reg = RandomForestRegressor()
rf_reg.fit(x_train,y_train)

y_pred = rf_reg.predict(x_test)

print("Adjusted R2 squared : ",rf_reg.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  0.9764839971754542
Mean Absolute Error (MAE):  18871.76133903134
Mean Squared Error (MSE):  1091641912.9978378
Root Mean Squared Error (RMSE):  33040.004736649746
```

3) ExtraTreesRegressor

- From the below algorithm we can see the accuracy score for ExtraTreesRegressor is approximately **100%** better.

```
from sklearn.ensemble import ExtraTreesRegressor

extra_reg = ExtraTreesRegressor()
extra_reg.fit(x_train,y_train)

y_pred = extra_reg.predict(x_test)

print("Adjusted R2 squared : ",extra_reg.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  1.0
Mean Absolute Error (MAE):  19839.853931623933
Mean Squared Error (MSE):  1241526745.211117
Root Mean Squared Error (RMSE):  35235.305379847596
```

4) KNeighborsRegressor

- From the below algorithm we can see the accuracy score for KNeighborsRegressor is approximately **94.49%** better.

```
: from sklearn.neighbors import KNeighborsRegressor

k_neigh = KNeighborsRegressor()
k_neigh.fit(x_train,y_train)

y_pred = k_neigh.predict(x_test)

print("Adjusted R2 squared : ",k_neigh.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  0.8303046044408277
Mean Absolute Error (MAE):  27257.960113960115
Mean Squared Error (MSE):  2474058856.8460402
Root Mean Squared Error (RMSE):  49739.91211136224
```

5) HistGradientBoostingRegressor

- From the below algorithm we can see the accuracy score for HistGradientBoostingRegressor is approximately **97.80%** better.

```
from sklearn.experimental import enable_hist_gradient_boosting
from sklearn.ensemble import HistGradientBoostingRegressor

hist_grad = HistGradientBoostingRegressor()
hist_grad.fit(x_train,y_train)

y_pred = hist_grad.predict(x_test)

print("Adjusted R2 squared : ",hist_grad.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  0.97802661116812
Mean Absolute Error (MAE):  19504.41223151312
Mean Squared Error (MSE):  1260819820.6607265
Root Mean Squared Error (RMSE):  35508.02473611742
```

6) GradientBoostingRegressor

- From the below algorithm we can see the accuracy score for GradientBoostingRegressor is approximately **97.45%** better.

```
: from sklearn.ensemble import GradientBoostingRegressor

grid_reg = GradientBoostingRegressor()
grid_reg.fit(x_train,y_train)

y_pred = grid_reg.predict(x_test)

print("Adjusted R2 squared : ",grid_reg.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
Adjusted R2 squared :  0.9745853853585651
Mean Absolute Error (MAE):  19063.775356416183
Mean Squared Error (MSE):  1519508101.2212322
Root Mean Squared Error (RMSE):  38980.86840003994
```

- The below code shows us the cross validation performed over all the algorithms and I have used the CV values as 5.
- If you observe the below screenshot, we get the difference in the values.

```
: from sklearn.model_selection import cross_val_score
```

```
: scr = cross_val_score(dt_reg, x, y, cv=5)
print("Cross Validation score of DecisionTreeRegressor model is:", scr.mean())
```

```
Cross Validation score of DecisionTreeRegressor model is: 0.7225223404501427
```

```
: scr = cross_val_score(rf_reg, x, y, cv=5)
print("Cross Validation score of RandomForestRegressor model is:", scr.mean())
```

```
Cross Validation score of RandomForestRegressor model is: 0.8512853972653488
```

```
: scr = cross_val_score(extra_reg, x, y, cv=5)
print("Cross Validation score of ExtraTreesRegressor model is:", scr.mean())
```

```
Cross Validation score of ExtraTreesRegressor model is: 0.8346042437410823
```

```
: scr = cross_val_score(k_neigh, x, y, cv=5)
print("Cross Validation score of KNeighborsRegressor model is:", scr.mean())
```

```
Cross Validation score of KNeighborsRegressor model is: 0.6146861409759754
```

```
: scr = cross_val_score(hist_grad, x, y, cv=5)
print("Cross Validation score of HistGradientBoostingRegressor model is:", scr.mean())
```

```
Cross Validation score of HistGradientBoostingRegressor model is: 0.8381162602258746
```

```
: scr = cross_val_score(grid_reg, x, y, cv=5)
print("Cross Validation score of GradientBoostingRegressor model is:", scr.mean())
```

```
Cross Validation score of GradientBoostingRegressor model is: 0.863916928260322
```

- From the above algorithms GradientBoostingRegressor seems to be an ideal algorithm in this scenario and for this type of dataset.
- The difference between the accuracy score and cross validation for this model is very less compared to other models.

Sr.No	Models used	Adjusted R2 score	CV score	Difference output
1	DecisionTreeRegressor	100	0.722522340450142	99.2774776595499
2	RandomForestRegressor	0.976483997175454	0.851285397265348	0.125198599910106
3	ExtraTreesRegressor	100	0.834604243741082	99.1653957562589
4	KNeighborsRegressor	0.830304604440827	0.614686140975975	0.215618463464852
5	HistGradientBoostingRegressor	0.97802661116812	0.838116260225874	0.139910350942246
6	GradientBoostingRegressor	0.974585385358565	0.863916928260322	0.110668457098243

- Let us try to tune the proposed model (GradientBoostingRegressor) to get better accuracy, if possible.
- The "parameters" have been selected from the skicit library and I have considered 6 parameters.

```
parameters = {"loss":["ls", "lad", "huber", "quantile"],
              "criterion":["friedman_mse", "mse", "mae"],
              "max_features":["auto", "sqrt", "log2"],
              "n_estimators":[50, 70, 90, 100, 130, 150],
              "random_state":[50, 70, 90, 100, 130, 150],
              "tol":[1e-1, 1e-2, 1e-3, 1e-4, 1e-5]
              }
```

- RandomizedSearchCV is used to tune the parameters by fitting the same to the training dataset and used the best parameters after selection

```
from sklearn.model_selection import RandomizedSearchCV
RCV = RandomizedSearchCV(GradientBoostingRegressor(), parameters, cv=5, n_iter=10)
```

```
RCV.fit(x_train, y_train)
```

```
RandomizedSearchCV(cv=5, estimator=GradientBoostingRegressor(),
                   param_distributions={'criterion': ['friedman_mse', 'mse',
                                                      'mae'],
                                        'loss': ['ls', 'lad', 'huber',
                                                 'quantile'],
                                        'max_features': ['auto', 'sqrt',
                                                         'log2'],
                                        'n_estimators': [50, 70, 90, 100, 130,
                                                         150],
                                        'random_state': [50, 70, 90, 100, 130,
                                                         150],
                                        'tol': [0.1, 0.01, 0.001, 0.0001,
                                                1e-05]})
```

```
RCV.best_params_
```

```
{'tol': 0.0001,
 'random_state': 100,
 'n_estimators': 130,
 'max_features': 'auto',
 'loss': 'ls',
 'criterion': 'friedman_mse'}
```

- Rebuild the model using the appropriate params we received from best_params_

```
mod_grid_reg = GradientBoostingRegressor(learning_rate= 0.0001, random_state= 100, n_estimators= 130, max_features= "auto",
                                         loss= "ls", criterion= "friedman_mse")

mod_grid_reg.fit(x_train,y_train)
pred = mod_grid_reg.predict(x_test)
pred
```

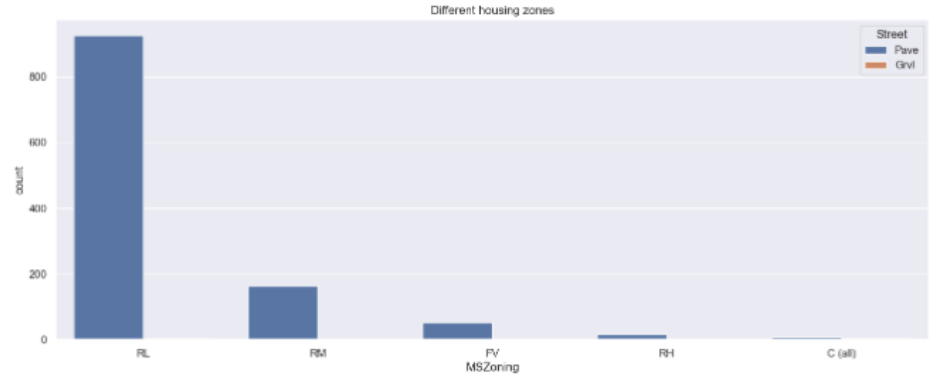
- Key Metrics for success in solving problem under consideration

  - Using sklearn.metrics I have used Adjusted R2 squared, Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) to check for all possible.

- Visualizations

  - From the below plot we can identify the zoning area of the properties / apartments that are up for sale.
  - We can see from this plot that "RL" (Residential Low Density) areas are up for sale more than other zones.
  - The low thickness private zones are planned to make, keep up with, and advance neighborhoods with bigger parcel sizes where the land use is fundamentally single-family abodes. They take into account some non household living uses however keep up with the general person of a solitary family area.
  - It makes sense as if the zones that fall in green belt or agriculture then using it for commercial purpose is not possible unless the land is infertile. This appears to be a large residential neighbourhood.

```
plt.figure(figsize=(16, 6))
sns.set_theme(style="darkgrid")
ax = sns.countplot("MSZoning", data=df_train, hue= "Street").set(title='Different housing zones')
plt.show()
```



- The below plot gives us idea on the locality of neighbourhood of the residences within Ames city limits.
- Ames is a city in Story County, Iowa, United States and the properties belong to the "Northwest Ames" in majority followed by "College Creek" and these areas are also known to have given properties on rent, lease etc. which included both apartments and condos.
- As per a couple of surveys, cost of living anywhere in USA above 100 points is considered expensive and Iowa's cost of living is 83.7 which includes all essentials like job, grocery, housing etc. This states that it's one of the most sought of place in the USA if budget friendly home is a priority.

```
plt.figure(figsize=(16, 6))
sns.set_theme(style="darkgrid")
ax = sns.countplot("Neighborhood", data=df_train).set(title='Neighbourhood assosiated around the property')
plt.xticks(rotation=70)
plt.show()
```

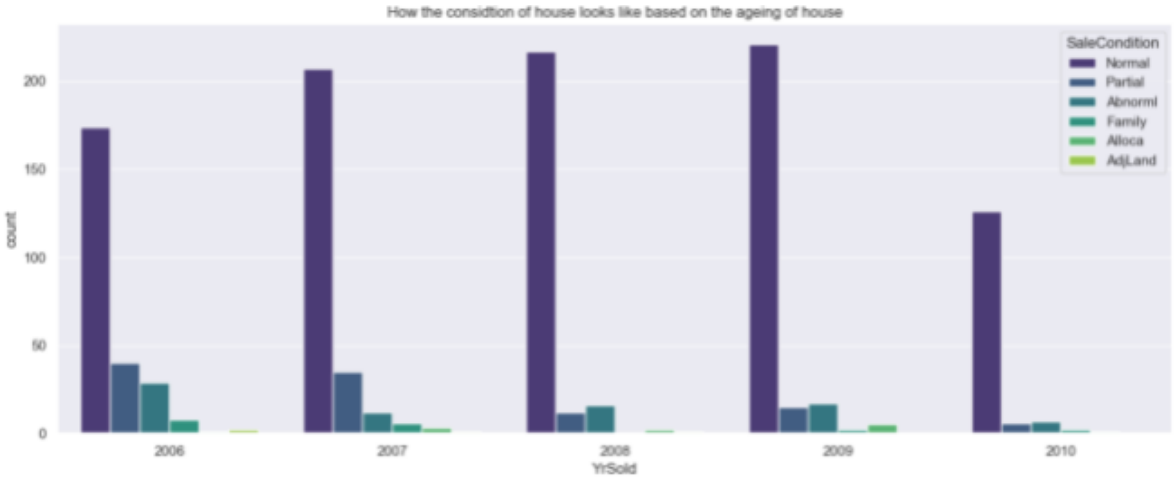
- Let's observe if there is any relation between sales price of the properties over the years.
- We can see that majority of high property sales have occurred in the year 2010, especially if the properties are either "New" or if it's on a Contract 15% Down payment regular terms.
- A such a down payment is possible if the person looking forward to purchase has a steady income.
- Higher the down payment, better the loan rates and since cost of living is so important. This could be an ideal option for people who cannot afford or do not wish to spend all at once.
- We can also see the sales prices although less for the year 2007 when compared now, it was highest for that period of time. This shows how real

estate’s values have increased over period of time for certain type of properties all over Iowa.

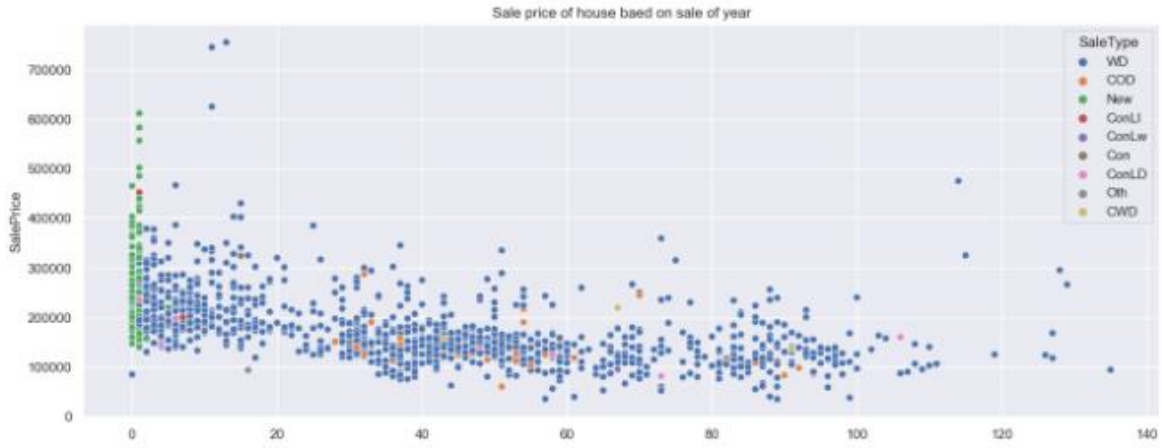
```
plt.figure(figsize=(16, 6))
sns.set_theme(style="darkgrid")
sns.lineplot(data=df_train, x="SaleType", y="SalePrice", hue = "YrSold", ci=None).set(title='Sales price based on the sale type
plt.show()
```

- The below plot shows us the years when properties were sold the highest.
- And it appears in the year 2006 and 2007 sales were high but gradually decreased in the year 2008 until 2010.
- The monetary emergency of 2007-2008 was a very long time really taking shape.
- By the late spring of 2007, monetary business sectors all throughout the planet were giving indications that the retribution was late for quite some time long gorge on modest credit.

- We can see from the below plot that thought the years only "Normal sales" houses were bought compared to already existing.
- Customers did not seem to go for options like trade, foreclosure, short sale or sale between families and seldom preferred houses that were partially unfinished.
- This is a clear indication that when a person gets on a market price without middleman or brokers customers tend to get direct and better deals thus saving money to some extent.
- Also, in general people prefer buying houses that are pose less threat and hassles, means less and concrete paperwork's, easier way of getting loans for houses which seems ideal when going with Normal Sales compared to other type of sales conditions.



- An assurance deed is a kind of deed where the grantor (seller) guarantees that the person being referred to holds clear title to a real estate parcel and has a choice to bring to the table it to the grantee (buyer), instead of a quitclaim deed, where the vendor doesn't guarantee that the individual holds title to a land parcel.
- An assurance deed is a record routinely used in land that gives the best proportion of protection to the purchaser of the property. The deed pledges or warrants that the owner cases the property totally free in regards to any exceptional liens, contracts, or various encumbrances
- It appears Warranty Deed seems to be the most sought-after payment or purchasing method as it's clearly in favour of both the parties as and when the duration of built year and sale year increases.

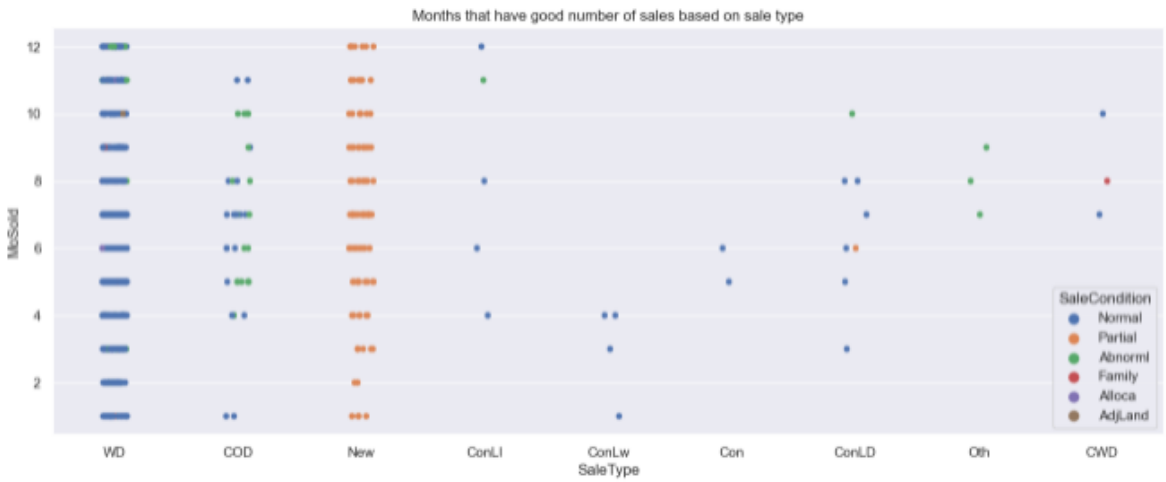


- It appears from the below plot that number of years taken to remodel doesn't really seem to affect the sale type.
- Apart from New, in such cases also people seem to prefer WD over other sale types. Hence in a way, both selling of properties with or without remodelling completely or to some extent doesn't really seem to affect the value.
- Maybe because of abundance of properties across neighbourhood, a customer has multiple options to go with.
- But if the house is newly built that has not completed i.e. partially completed, then only it seems to fetch good amount if remodelled as the market value doesn't seem to diminish the actual value.
- This could be common amongst those who buy houses at a cheaper rate and rent it to other parties to get a potential and steady income.

- We can see that the roadways or streets seems to play an important role.
- Houses / properties that are in neighbourhoods, condos, villa's etc fetch good number of values and if it was remodelled or partially completed it seems to get even higher sale to some extent.

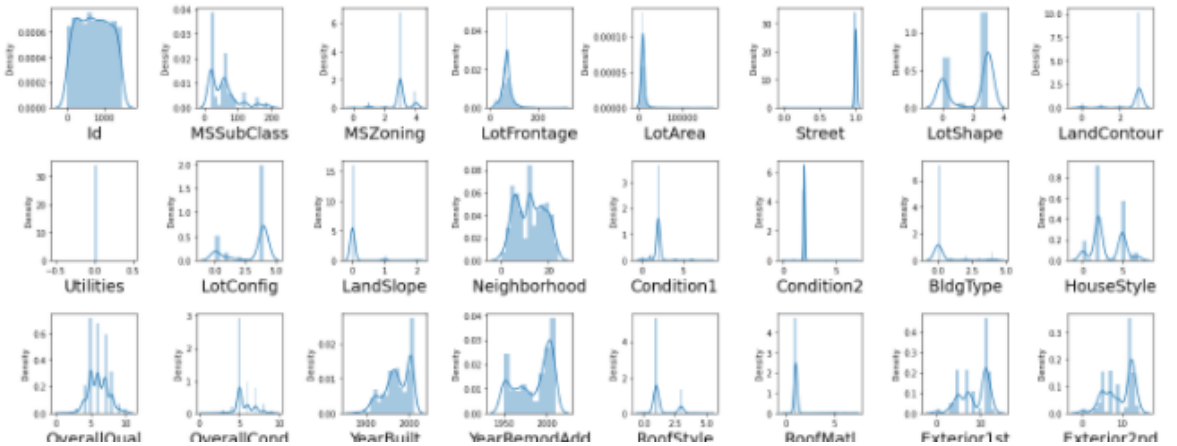
- But apart from these, customers may also prefer to go to areas that are near to roads, hence connectivity could be a key factor.
- Properties near to feeder street (inner / interior roads) and also on arterial street (junction roads) are the second sought after properties and majority in all cases normal sale type is highly preferred.

- We can see from the below plot that "Warranty Deed" and "New" are favourable throughout the year and since it's a procedure, it doesn't really seem to be affected by months.
- But we can see that apart from these options, people may also prefer "COD" (Court Officer Deed) and it seems to appear often between the months May to October.
- Court Officer Deed - A deed where an agent or director of a home or a conservator of the property of a ward moves title for the benefit of a bequest or conservatorship. Court endorsement might be important before the State can acknowledge this sort of transport.

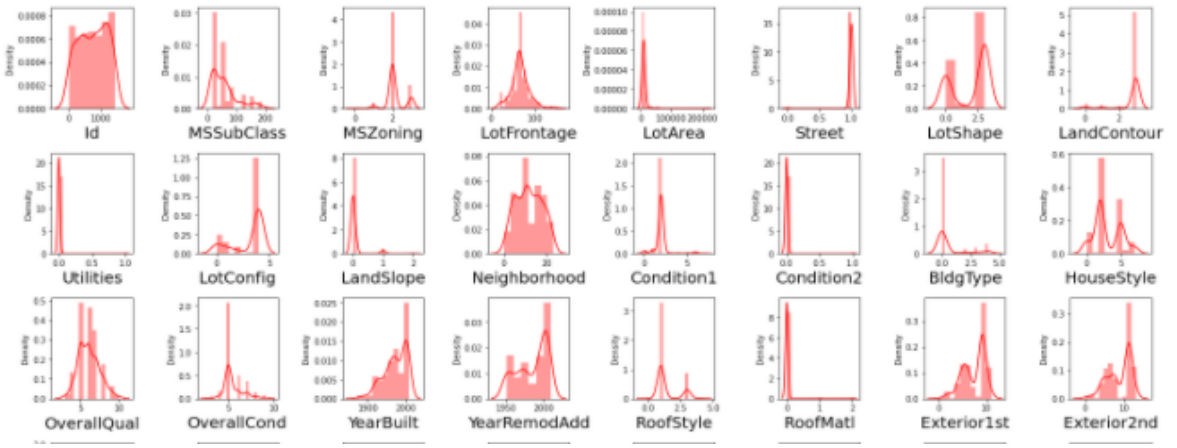


Distribution of the datasets

```
]: # Let us now see the distribution of the "Train dataset"

plt.figure(figsize=(20,25), facecolor="white")
plotnumber = 1

for column in df_train:
    if plotnumber <=77:
        ax = plt.subplot(10,8, plotnumber)
        sns.distplot(df_train[column])
        plt.xlabel(column, fontsize=20)
    plotnumber+=1
plt.tight_layout()
```

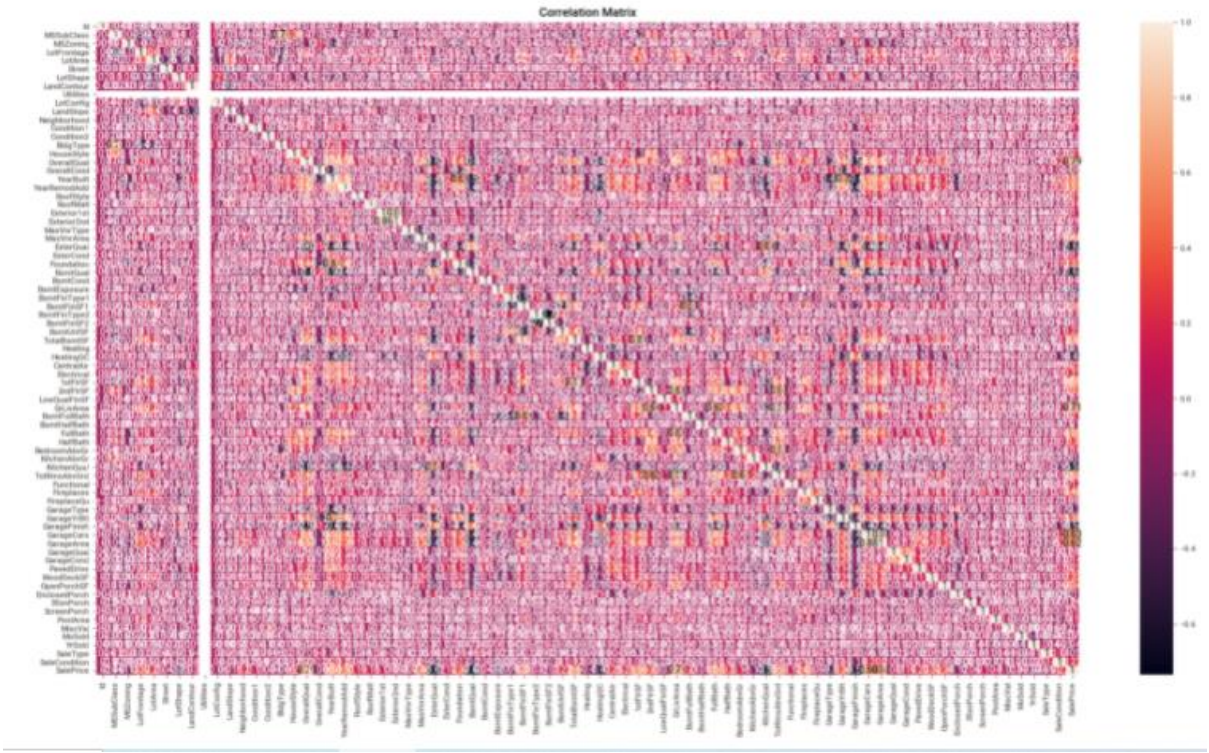



```
]: # Let us now see the distribution of the "Test dataset"

plt.figure(figsize=(20,25), facecolor="white")
plotnumber = 1

for column in df_test:
    if plotnumber <=77:
        ax = plt.subplot(10,8, plotnumber)
        sns.distplot(df_test[column], color="red")
        plt.xlabel(column, fontsize=20)
    plotnumber+=1
plt.tight_layout()
```

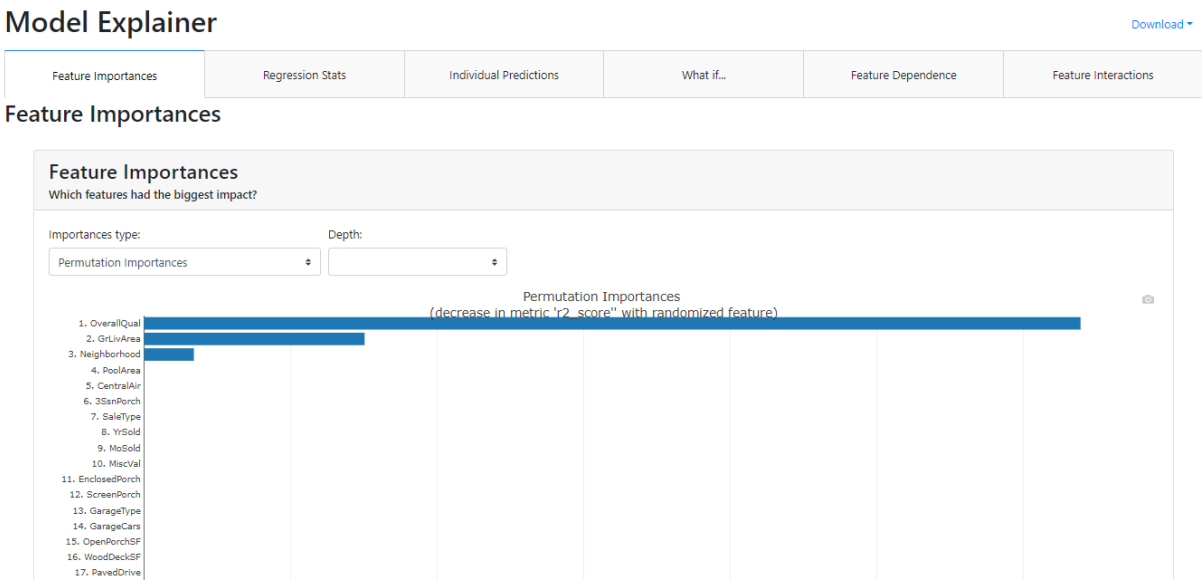


- Let’s us now examine correlation using a "heatmap" for further clarification
- Since there are multiple features, going through this plot may seem difficult.

```
plt.figure(figsize=(22,12))
sns.heatmap(corr_matrix, annot=True)
plt.title("Correlation Matrix")
plt.show()
```



Model Dashboard



Model Explainer

Download

Feature Importances | Regression Stats | Individual Predictions | What if... | Feature Dependence | Feature Interactions

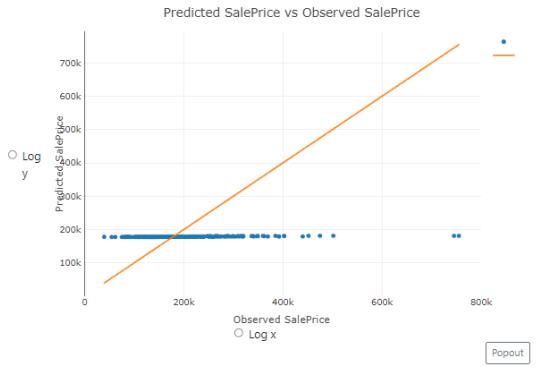
Model Summary

Quantitative metrics for model performance

metric	Score
mean-squared-error	7550502648.441
root-mean-squared-error	86893.628
mean-absolute-error	60818.541
mean-absolute-percentage-error	0.35
R-squared	0.004

Predicted vs Actual

How close is the predicted value to the observed?



Model Explainer

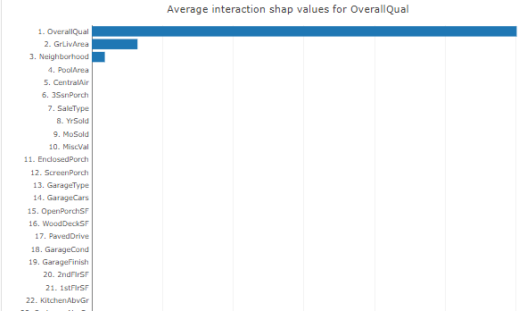
Download

Feature Importances | Regression Stats | Individual Predictions | What if... | Feature Dependence | Feature Interactions

Interactions Summary

Ordering features by shap interaction value

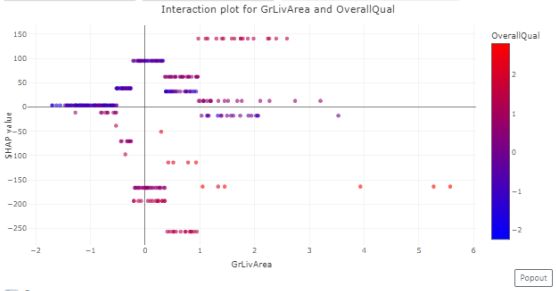
Feature: OverallQual | Depth: | Summary Type: Aggregate



Interaction Dependence

Relation between feature value and shap interaction value

Feature: OverallQual | Interaction: GrLivArea | Index: Select...



Remove outliers

Model Explainer

Download

Feature Importances | Regression Stats | Individual Predictions | What if... | Feature Dependence | Feature Interactions

Select Index

Select from list or pick at random

50 | Random Index

Predicted range: 177925.78 – 181392.64

Absolute residuals: 418.69 – 573607.36

Range: Predicted | Residuals: Absolute Residuals

Prediction

	SalePrice
Predicted	177925.783

Feature Input

Adjust the feature values to change the prediction

OverallQual	GrLivArea	Neighborhood	BedroomAbvGr
-0.7948227396909927	-0.7825948915504617	-0.35712764782952444	0.14149241142104663
Range: -2.23–2.8	Range: -1.7–5.59	Range: -2.02–1.97	Range: -3.53–6.26
FireplaceQu	Fireplaces	Functional	TotRmsAbvGrd
-0.43019297347912605	-0.9492505443085474	0.26114553852193356	-1.5914442640034545

CONCLUSION

- Key Findings and Conclusions of the Study

    - From the EDA , I could observe safety is very much import for customers and hence “Warranty Deed” is mostly preferred.
    - From couple of articles I found that Housing in USA is much easier than Australia.
    - Albeit the real estate market seems, by all accounts, to be cooling, it stays cutthroat.
    - The National Association of Realtors detailed those deals expanded 2% from the earlier month to an occasionally changed yearly pace of 5.99 million, up from an overhauled 1.6 percent increment in June.

- Interpretation of the Results

    - It’s a small dataset and prediction made may or may not be reliable based on the economic conditions of the region.
    - Majority of articles I referred to help me understand housing market of US more than that of Australia.

- Learning Outcomes of the Study in respect of Data Science

    - With the help of this dataset, I was able to work on multiple regression algorithms and also got to work on real-world dataset.
    - The dashboard that I built using ExplainerDashbaord took me approximately 16 hours. This could be because I use older system and processing power is very less for such applications as the dataset is very small.

- Limitations of this work and Scope for Future Work

    - Also, this dataset contains information only for past years and not current year and I believe having at least an annual information can help in formulating even better approaches.

    - The customer is looking forward to build / invest in housing in Australia and since the market is very different the predictions may vary dues to economic and other external factors.