HOUSING PRICE PREDICTION

PRESENTED

BY

MD SALEEM PATEL

Agenda:

- Introduction
- Problem Statement
- Objective
- Exploratory Data Analysis (EDA)
- Visualization
- Inference
- Future Work

INTRODUCTION

PROBLEM STATEMENT:

- Houses are one of the necessary need 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.

EDA(Exploratory Data Analysis)

Data Description

The dataset contains 1460 records (rows) and 81 features (columns).
Here, we will provide a brief description of dataset features. Since the number of features is large (81), we will attach the original data description file to this study for more information about the dataset. Now, we will mention the feature name with a short description of its meaning.

Target Variable

- **Sale Price** : It’s continuous type of data, so the model approach is carried out for Regression analysis.

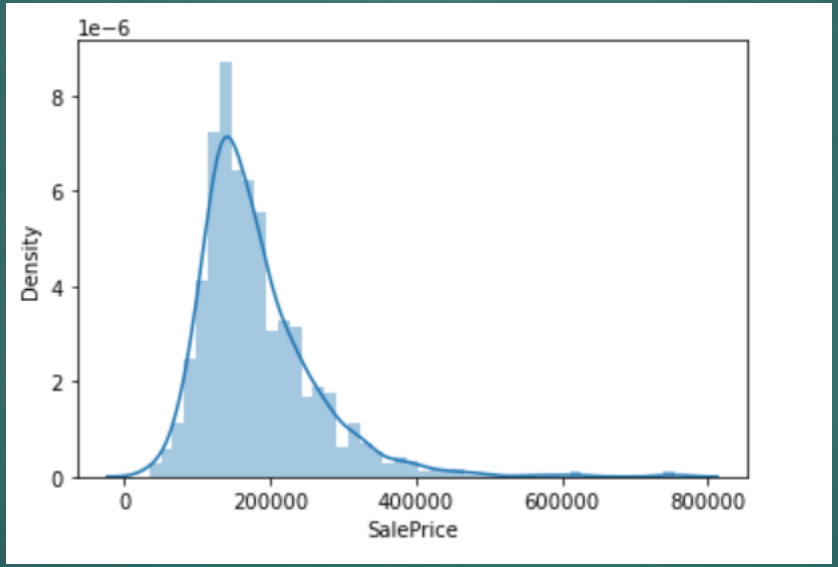
Regression:

It’s an analysis is used when you want to predict a continuous dependent variable from a number of independent variables.

Independent variables with more than two levels can also be used in regression analysis.

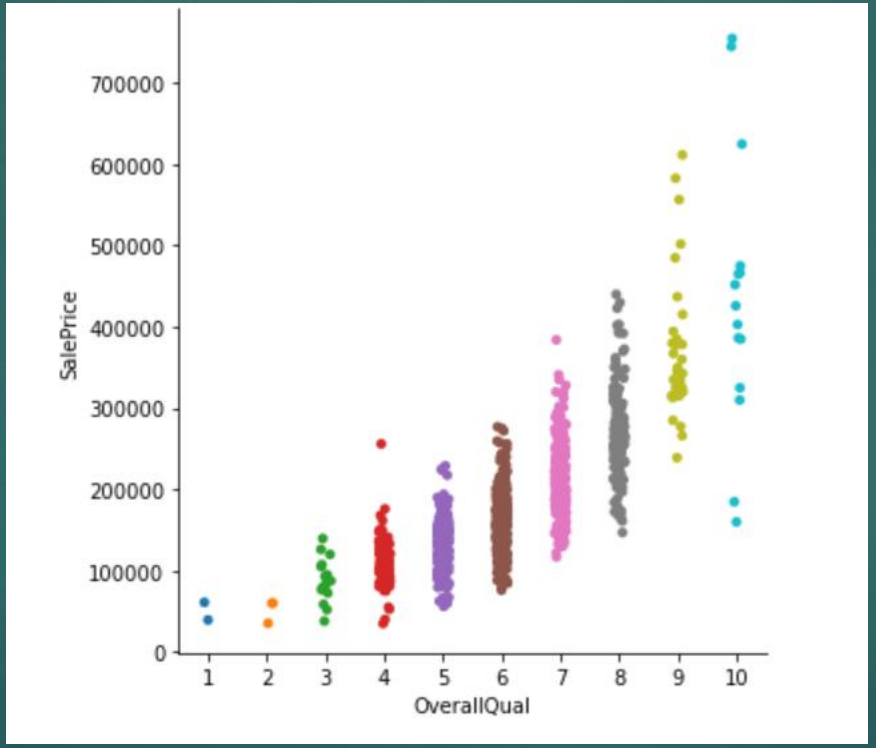
Visualization

Target Variable (Sale Price Distribution)



Catplot Distribution

overall Qualification vs Sale Price(Target Variable)



Column Dropped

THE COLUMNS THAT ARE GOING TO BE DROP ARE UTILITIES. THEY ARE STRINGS , CANNOT BE CATEGORIZED AND DON'T CONTRIBUTE MUCH TO THE OUTCOME.

The columns that are going to be drop are Utilities.They are strings,cannot be categorized and dont contribute much to the outcome.

```
df.drop(['Utilities'],axis=1,inplace=True)
```

Data Pre-processing

Preprocessing

In [57]:
```
df.describe() #statistics summary for numerical columns
```

Out[57]:

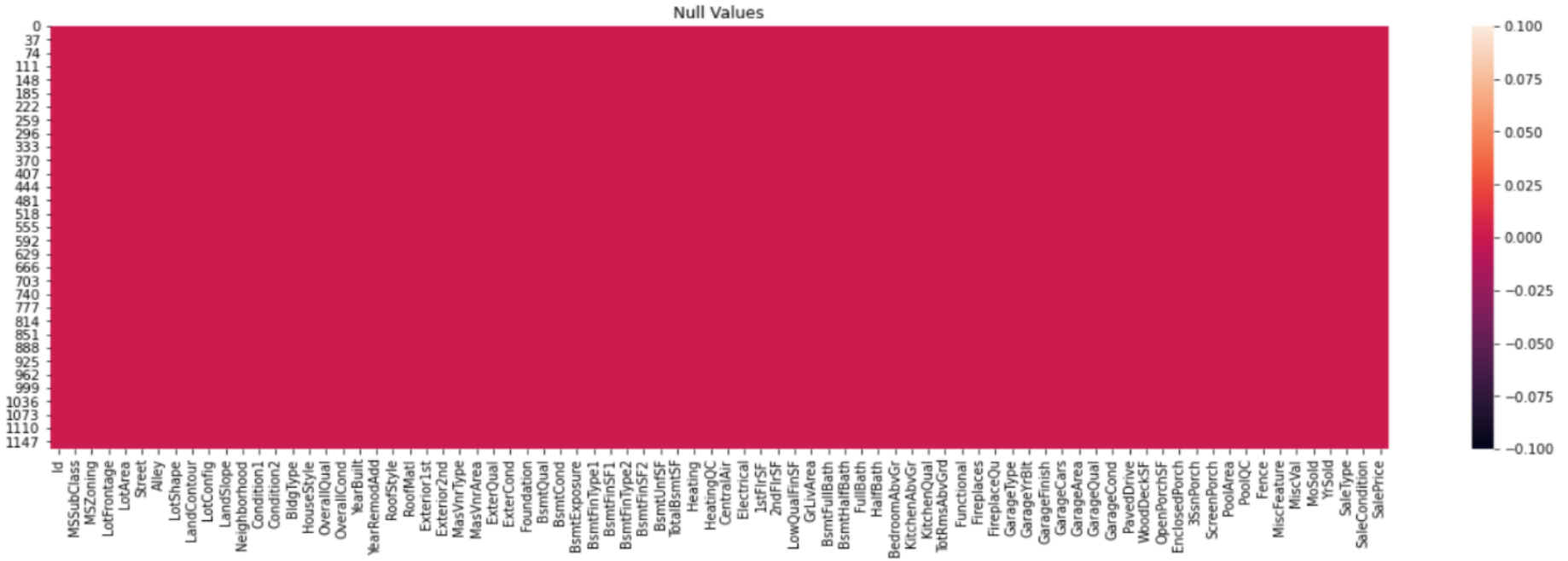
	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2
count	1168.000000	1168.000000	954.00000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1161.000000	1168.000000	1168.000000
mean	724.136130	56.767979	70.98847	10484.749144	6.104452	5.595890	1970.930651	1984.758562	102.310078	444.726027	46.647260
std	416.159877	41.940650	24.82875	8957.442311	1.390153	1.124343	30.145255	20.785185	182.595606	462.664785	163.520016
min	1.000000	20.000000	21.00000	1300.000000	1.000000	1.000000	1875.000000	1950.000000	0.000000	0.000000	0.000000
25%	360.500000	20.000000	60.00000	7621.500000	5.000000	5.000000	1954.000000	1966.000000	0.000000	0.000000	0.000000
50%	714.500000	50.000000	70.00000	9522.500000	6.000000	5.000000	1972.000000	1993.000000	0.000000	385.500000	0.000000
75%	1079.500000	70.000000	80.00000	11515.500000	7.000000	6.000000	2000.000000	2004.000000	160.000000	714.500000	0.000000
max	1460.000000	190.000000	313.00000	164660.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000	1474.000000

Data Cleaning

- Dealing with Missing Values:
- Filling the missing values using fillna method.
- Check if there is any remaining missing value in our dataset

To sh

```
In [62]: # Heatmap for Null value for data.
         plt.figure(figsize=[22,6])
         sns.heatmap(df.isnull())
         plt.title('Null Values')
         plt.show()
```



Encoding of Data Frame:

Since the dataset has a lot string values. We will use the ordinal encoding techniques to convert the string data to numerical one.

Encoding of DataFrame:

In [63]:
```
from sklearn.preprocessing import OrdinalEncoder
enc=OrdinalEncoder()
```

In [64]:
```
for i in df.columns:
    if df[i].dtypes=="object":
        df[i]=enc.fit_transform(df[i].values.reshape(-1,1))
```

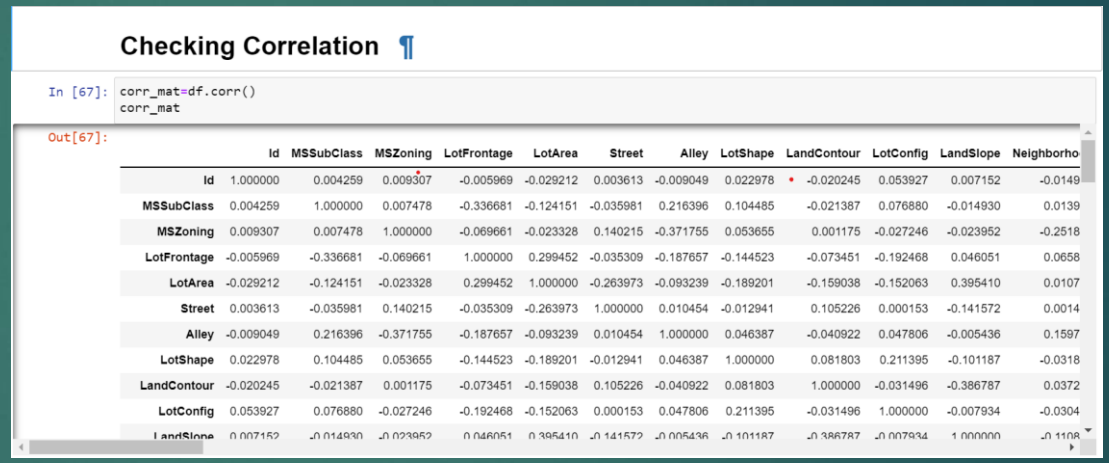
In [65]:
```
df.head()    # informtion about top of the data after Ordinal encoder
```

Out[65]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	Bl
0	127	120	3.0	70.98847	4928	1.0	0.0	0.0	3.0	4.0	0.0	13.0	2.0	2.0	
1	889	20	3.0	95.00000	15865	1.0	0.0	0.0	3.0	4.0	1.0	12.0	2.0	2.0	
2	793	60	3.0	92.00000	9920	1.0	0.0	0.0	3.0	1.0	0.0	15.0	2.0	2.0	
3	110	20	3.0	105.00000	11751	1.0	0.0	0.0	3.0	4.0	0.0	14.0	2.0	2.0	
4	422	20	3.0	70.98847	16635	1.0	0.0	0.0	3.0	2.0	0.0	14.0	2.0	2.0	

Correlation matrix

A correlation matrix is simply a table which displays the correlation. The measure is best used in variables that demonstrate a linear relationship between each other. The fit of the data can be visually represented in a heatmap.



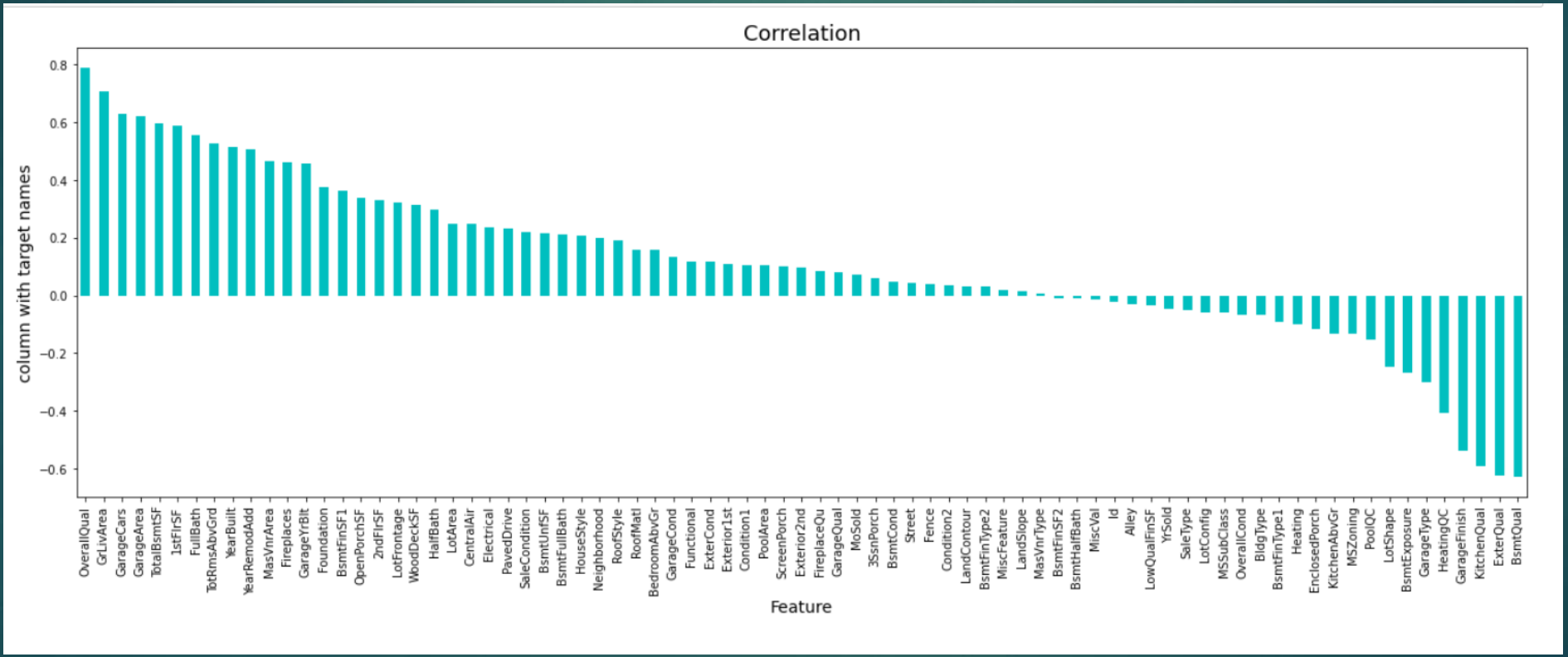
Checking Correlation ¶

In [67]:
```
corr_mat=df.corr()
corr_mat
```

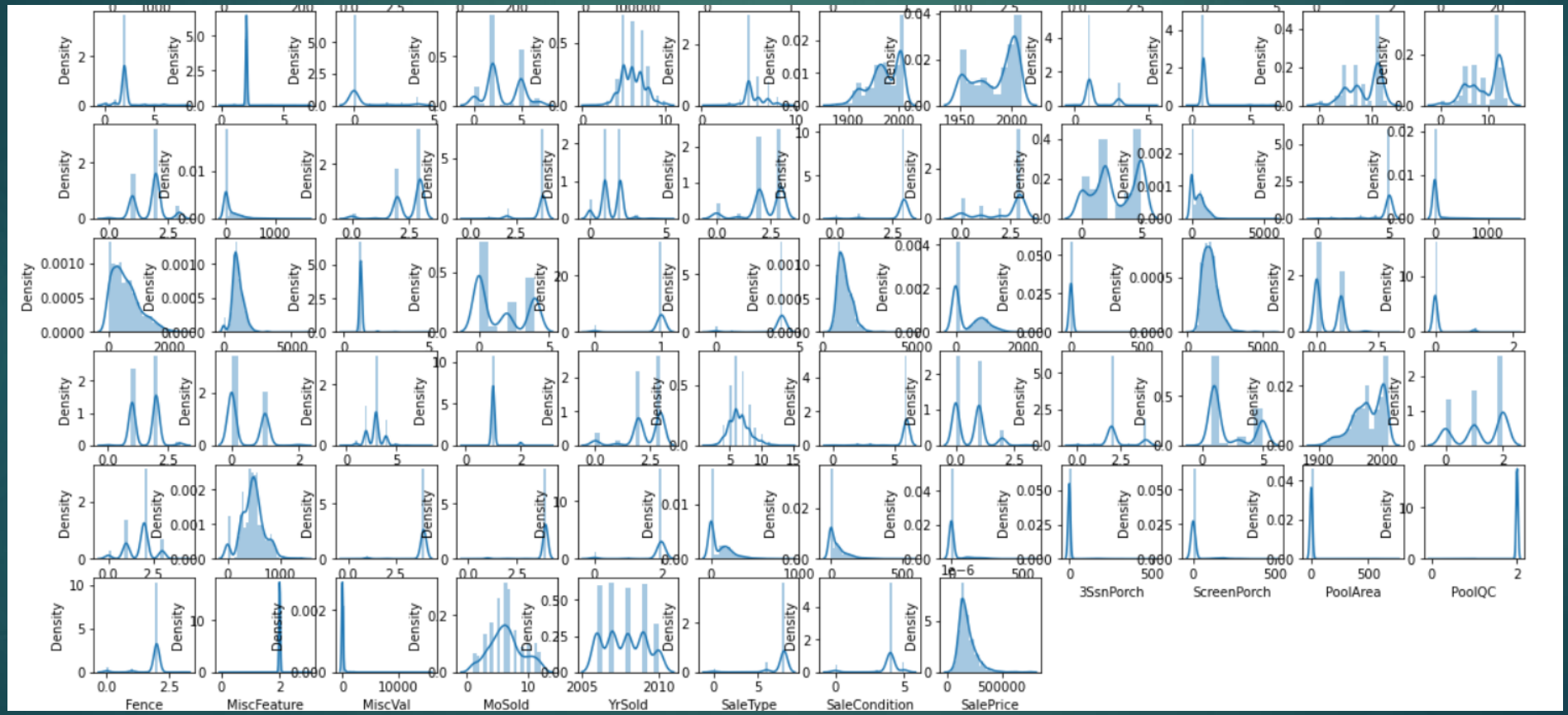
Out[67]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	LotConfig	LandSlope	Neighborho
Id	1.000000	0.004259	0.009307	-0.005969	-0.029212	0.003613	-0.009049	0.022978	-0.020245	0.053927	0.007152	-0.0149
MSSubClass	0.004259	1.000000	0.007478	-0.336681	-0.124151	-0.035981	0.216396	0.104485	-0.021387	0.076880	-0.014930	0.0139
MSZoning	0.009307	0.007478	1.000000	-0.069661	-0.023328	0.140215	-0.371755	0.053655	0.001175	-0.027246	-0.023952	-0.2518
LotFrontage	-0.005969	-0.336681	-0.069661	1.000000	0.299452	-0.035309	-0.187657	-0.144523	-0.073451	-0.192468	0.046051	0.0658
LotArea	-0.029212	-0.124151	-0.023328	0.299452	1.000000	-0.263973	-0.093239	-0.189201	-0.159038	-0.152063	0.395410	0.0107
Street	0.003613	-0.035981	0.140215	-0.035309	-0.263973	1.000000	0.010454	-0.012941	0.105226	0.000153	-0.141572	0.0014
Alley	-0.009049	0.216396	-0.371755	-0.187657	-0.093239	0.010454	1.000000	0.046387	-0.040922	0.047806	-0.005436	0.1597
LotShape	0.022978	0.104485	0.053655	-0.144523	-0.189201	-0.012941	0.046387	1.000000	0.081803	0.211395	-0.101187	-0.0318
LandContour	-0.020245	-0.021387	0.001175	-0.073451	-0.159038	0.105226	-0.040922	0.081803	1.000000	-0.031496	-0.386787	0.0372
LotConfig	0.053927	0.076880	-0.027246	-0.192468	-0.152063	0.000153	0.047806	0.211395	-0.031496	1.000000	-0.007934	-0.0304
LandSlope	0.007152	-0.014930	-0.023952	0.046051	0.395410	-0.141572	-0.005436	-0.101187	-0.386787	-0.007934	1.000000	-0.1108

Checking the columns which are positively and negative correlated with the target columns



Check the data distribution among all the columns

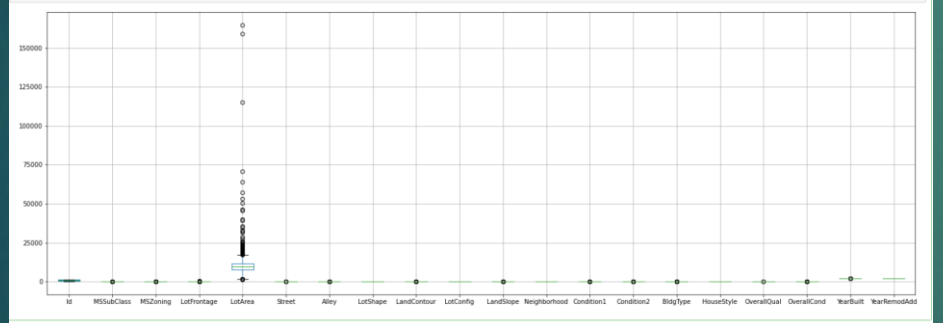


Outliers Check

There are 80 columns in dataset so it's not possible to plot each and every column separately or plot all together. so, we will print in 4 steps:

First set

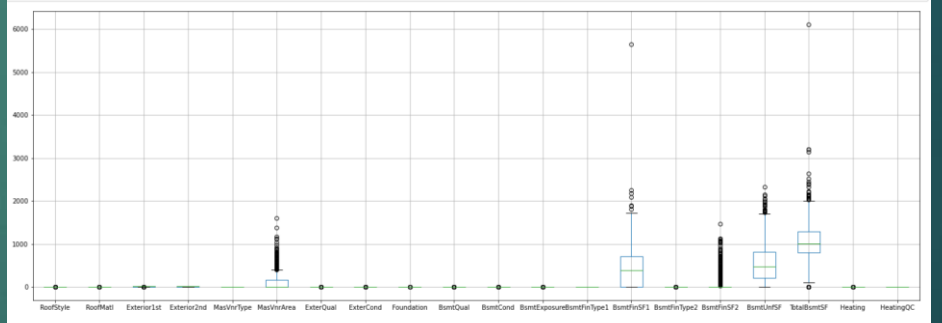
```
#Plotting Boxplots for first 20 columns
df.iloc[:,0:20].boxplot(figsize=[25,10])
plt.subplots_adjust(bottom=0.25)
plt.show()
```



There are outliers in lot area column.

Second set

```
#Plotting boxplot for next 20 columns
df.iloc[:,20:40].boxplot(figsize=[25,10])
plt.subplots_adjust(bottom=0.25)
plt.show()
```

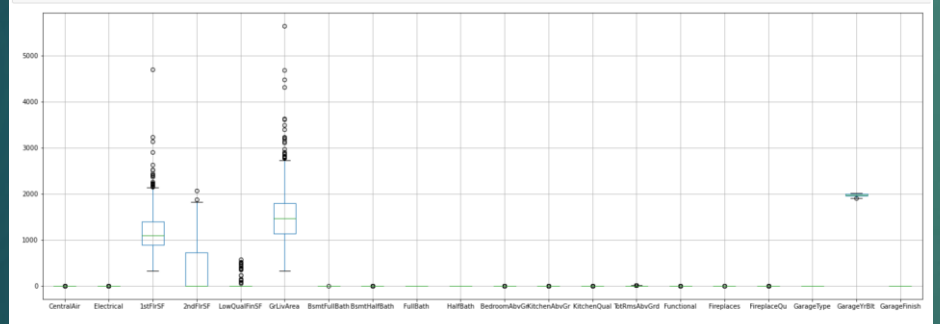


We can see outliers in MasVnrArea, BsmtFinSF1,BsmtFinSF2,BsmtUnfSF,TotalBsmtSF.

Remaining section of Outliers Check

Third set

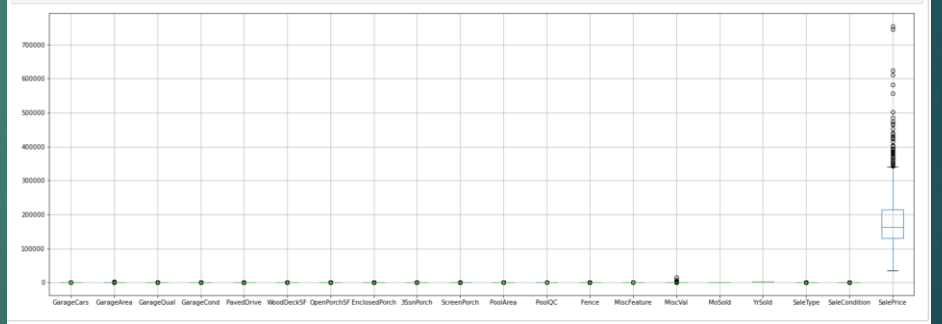
```
#Plotting boxplot for next 20 columns
df.iloc[:,40:60].boxplot(figsize=[25,10])
plt.subplots_adjust(bottom=0.25)
plt.show()
```



We can see outliers in 1stFlrSF, LowQualFinSF,GrLivArea,GarageYrBuilt.

Fourth set

```
#Plotting boxplot for rest all the columns
df.iloc[:,60:80].boxplot(figsize=[25,10])
plt.subplots_adjust(bottom=0.25)
plt.show()
```



We can see one or two values outer of whiskers but those are near to whiskers so these are not outliers.The outliers are not consider for target variables.

Checking Skewness:

Now here we are going too use power transform function to handle skewness in dataset

Before handling Skewness

Columns	Skewness	Columns	Skewness
Id	0.026526	CentralAir	-3.475188
MSSubClass	1.422019	Electrical	-3.104209
MSZoning	-1.796785	1stFlrSF	1.513707
LotFrontage	2.710383	2ndFlrSF	0.823479
LotArea	10.659285	LowQualFinSF	8.666142
Street	-17.021969	GrLivArea	1.449952
Alley	5.436187	BsmtFullBath	0.627106
LotShape	-0.603775	BsmtHalfBath	4.264403
LandContour	-3.125982	FullBath	0.057809
LotConfig	-1.118821	HalfBath	0.656492
LandSlope	4.812568	BedroomAbvGr	0.243855
Neighborhood	0.043735	KitchenAbvGr	4.365259
Condition1	3.008289	KitchenQual	-1.408106
Condition2	11.514458	TotRmsAbvGrd	0.644657
BldgType	2.318657	Functional	-3.999663
HouseStyle	0.285680	Fireplaces	0.671966
OverallQual	0.175082	FireplaceQu	0.753507
OverallCond	0.580714	GarageType	0.831142
YearBuilt	-0.579204	GarageYrBlt	-0.662934
YearRemodAdd	-0.495864	GarageFinish	-0.450190
RoofStyle	1.498560	GarageCars	-0.358556
RoofMatl	7.577352	GarageArea	0.189665
Exterior1st	-0.612816	GarageQual	-4.582386
Exterior2nd	-0.592349	GarageCond	-5.422472
MasVnrType	-0.104609	PavedDrive	-3.274035
MasVnrArea	2.834658	WoodDeckSF	1.504929
ExterQual	-1.810843	OpenPorchSF	2.410840
ExterCond	-2.516219	EnclosedPorch	3.043610
Foundation	-0.002761	3SsnPorch	9.770611
BsmtQual	-1.343781	ScreenPorch	4.105741
BsmtCond	-3.293554	PoolArea	13.243711
BsmtExposure	-1.166987	PoolQC	-19.401558
BsmtFinType1	-0.068901	Fence	-3.185107
BsmtFinSF1	1.871606	MiscFeature	-17.238424
BsmtFinType2	-3.615783	MiscVal	23.065943
BsmtFinSF2	4.365829	MoSold	0.220979
BsmtUnfSF	0.909057	YrSold	0.115765
TotalBsmtSF	1.744591	SaleType	-3.660513
Heating	10.103609	SaleCondition	-2.671829
HeatingQC	0.449933	SalePrice	1.953878

After handling Skewness

Columns	Skewness	Columns	Skewness
Id	-0.268486	CentralAir	-3.475188
MSSubClass	0.064007	Electrical	-3.006845
MSZoning	0.233113	1stFlrSF	-0.002391
LotFrontage	0.161368	2ndFlrSF	0.280208
LotArea	0.032509	LowQualFinSF	6.922843
Street	-17.021969	GrLivArea	-0.000054
Alley	5.436187	BsmtFullBath	0.365488
LotShape	-0.594207	BsmtHalfBath	3.954345
LandContour	-2.592303	FullBath	-0.045944
LotConfig	-1.030401	HalfBath	0.498003
LandSlope	3.954345	BedroomAbvGr	0.116498
Neighborhood	-0.146541	KitchenAbvGr	-2.370593
Condition1	0.225468	KitchenQual	-0.435558
Condition2	0.537277	TotRmsAbvGrd	0.002332
BldgType	1.857194	Functional	-3.343664
HouseStyle	-0.080331	Fireplaces	0.084950
OverallQual	0.021658	FireplaceQu	0.082653
OverallCond	0.048063	GarageType	0.222501
YearBuilt	-0.126641	GarageYrBlt	-0.132523
YearRemodAdd	-0.225131	GarageFinish	-0.335248
RoofStyle	-0.292233	GarageCars	-0.022970
RoofMatl	-6.314987	GarageArea	-0.320370
Exterior1st	-0.338023	GarageQual	-4.327379
Exterior2nd	-0.352793	GarageCond	-4.925781
MasVnrType	-0.016203	PavedDrive	-3.025809
MasVnrArea	0.416370	WoodDeckSF	0.113026
ExterQual	-0.605112	OpenPorchSF	-0.002749
ExterCond	-2.270791	EnclosedPorch	2.022616
Foundation	0.004296	3SsnPorch	7.087955
BsmtQual	-0.413999	ScreenPorch	3.067153
BsmtCond	-3.025865	PoolArea	12.817372
BsmtExposure	-0.914214	PoolQC	-17.021969
BsmtFinType1	-0.206639	Fence	1.116688
BsmtFinSF1	-0.404528	MiscFeature	9.291637
BsmtFinType2	-2.420885	MiscVal	4.991071
BsmtFinSF2	2.394737	MoSold	-0.035838
BsmtUnfSF	-0.284390	YrSold	0.112893
TotalBsmtSF	0.286779	SaleType	-2.067563
Heating	-4.541694	SaleCondition	-0.353292
HeatingQC	0.156511		

Model Building and Evaluation

These are modelling approach made to build an model :

- Linear
- Random Forest
- Decision Tree
- XGBoost
- k-nearest neighbors (KNN)

Hyper Parameter Tuning

THE HYPER PARAMETER TUNING IS CARRIED OUT FOR BOTH RANDOM FOREST AND XGBOOST.

BECAUSE BOTH PREDICT TEST VALUE IS SIMILAR I.E. 89.

Best Model

HYPER PARAMETER TUNING PERFORMANCE IS COMPARED FOR BOTH RANDOM FOREST AND XGBOOST HYPER PARAMETER TUNING I.E.,R2 SCORE = 86.79 AND 89.15 RESPECTIVELY.

FINALLY, XGBOOST HAS BETTER R2 SCORE.SO THIS IS OUR BEST MODEL FOR THESE DATASET.

Conclusion

- In this paper, we built several regression models to predict the price of some house given some of the house features. We evaluated and compared each model to determine the one with highest performance. We also looked at how some models rank the features according to their importance. In this paper, we followed the data science process starting with getting the data, then cleaning and pre-processing the data, followed by exploring the data and building models, then evaluating the results and communicating them with visualizations.

- As a recommendation, we advise to use this model (or a version of it trained with more recent data) by people who want to buy a house in the area covered by the dataset to have an idea about the actual price. The model can be used also with datasets that covered areas provided that they contain the same features. We also suggest that people take into consideration the features that were deemed as most important as seen in the previous section; this might help them estimate the house price better.

	0	1	2	3	4	5	6	7	8
Predicted	113854.754788	109242.095796	169449.620245	91794.050179	153953.428249	137420.862056	149271.096313	313596.405266	169880.949371
Original	113854.754788	109242.095796	169449.620245	91794.050179	153953.428249	137420.862056	149271.096313	313596.405266	169880.949371