FLIP ROBO

HOUSING PROJECT

Submitted by:

RAUSHAN KUMAR

ACKNOWLEDGMENT

I would like to express my special thanks to my mentor Khusboo Garg who gave me her support and assistance throughout the project. I am thankful to Flip Robo Technologies and Data Trained institute to give me guidelines and knowledge to complete this project.

Links and websites that I preferred:
https://towardsdatascience.com/oversampling-and-undersampling-5e2bbaf56dcf

https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

https://career-resource-center.udacity.com/portfolio/data-science-reports#:~:text=A%20data%20science%20report%20is,the%20legitimacy%20of%20your%20process.

https://towardsdatascience.com/top-3-methods-for-handling-skewed-data-1334e0debf45

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

INTRODUCTION

- ## Business Problem Framing

  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.

  The company is looking at prospective properties to buy houses to enter the market.

  This Project 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.

  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.

- ## Conceptual Background of the Domain Problem

  It is helpful for those who have Domain knowledge of real estate and property. It will help in analysing the data as there are 81 columns in the dataset the domain knowledge and experience can help to understand the correlated things and changing trends over the period of time.

- ## Review of Literature

  Affordable housing is, unfortunately, a topic that's largely misunderstood by the public. Myths and misconceptions about affordable housing developments are based on fear around negative stereotypes, property values, and the change it brings to neighbourhoods—all of which are common arguments in opposition of a new affordable housing community

  Real estate investment plays a significant role in home appreciation.

The process in which investors procure a property initiates a chain of events that stimulates the economy. Every step in rehabbing and/or flipping a house involves the exchange of money in return for goods and services.

- ## Motivation for the Problem Undertaken

A positive externality occurs when a third party benefits from the production or consumption of a good. In many cases, building the right kind of housing can have benefits to the rest of society. Therefore, the social benefit of good quality housing can be greater than the private benefit that property developers gain.

The model will be a good way for the management to understand the pricing dynamics of a new market.

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem

Data consist of 1163 rows and 81 columns, because of the large number of columns it is difficult to analyse easily also to find significant columns that are helpful to predict the price of the house.

EDA part takes more time for this problem also as per the domain we can not eliminate any observation for its odd value. Also, many columns have many null values.

- ## Data Sources and their formats

The sample data is provided by the US-based housing company named Surprise Housing 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 in csv format. First I have read the csv file and converted it into a data frame. This is a regular regression problem statement.

```
In [1]: import pandas as pd
```

```
In [19]: import warnings
         warnings.filterwarnings('ignore')
```

```
In [27]: df  = pd.read_csv('train.csv')
```

```
In [16]: pd.set_option('display.max_columns',None)
```

```
In [17]: df.head()
```

Out[17]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2
0	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	NPkVill	Norm	Norm
1	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	Inside	Mod	NAmes	Norm	Norm
2	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	CulDSac	Gtl	NoRidge	Norm	Norm
3	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	NWAmes	Norm	Norm
4	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	FR2	Gtl	NWAmes	Norm	Norm

- **Data Pre Processing**

  Pre-processing Pipe Line is an important step towards the data modelling, To make data ready for prediction.
  Following Steps I followed for Pre Processing:

- **Handling the null values**

  I have dropped two columns PoolQC and MiscFeature because of 93% and 99% of null values.

```
In [30]: #PoolQC has 99.4% null values
         df1.drop('PoolQC',axis=1,inplace=True)
         df1.drop('MiscFeature',axis=1,inplace=True)
         #MiscFeature has 96.2% null values
```

For treating the null values the columns that have more than 2% of the null value I have created new categories for them.And columns that have less than 2% I have replaced null values with the mode in the categorical columns and with the median for numerical columns.

```
In [33]: df1['FireplaceQu'] = np.where(df1['FireplaceQu'].isnull(),'Missing_Qu',df1['FireplaceQu'])

In [34]: df1['Fence'] = np.where(df1['Fence'].isnull(),'Missing_fence',df1['Fence'])

In [35]: df1['Alley'] = np.where(df1['Alley'].isnull(),'Missing_alley',df1['Alley'])

In [36]: df1['GarageQual'] = np.where(df1['GarageQual'].isnull(),'Missing_garagequal',df1['GarageQual'])

In [37]: df1['GarageType'] = np.where(df1['GarageType'].isnull(),'Missing_garagetype',df1['GarageType'])

In [38]: df1['GarageCond'] = np.where(df1['GarageCond'].isnull(),'Missing_garagecond',df1['GarageCond'])

In [39]: df1['GarageFinish'] = np.where(df1['GarageFinish'].isnull(),'Missing_garagefinish',df1['GarageFinish'])

In [40]: df1['BsmtFinType2'] = np.where(df1['BsmtFinType2'].isnull(),df1['BsmtFinType2'].mode(),df1['BsmtFinType2'])

In [41]: df1['BsmtCond'] = np.where(df1['BsmtCond'].isnull(),df1['BsmtCond'].mode(),df1['BsmtCond'])

In [42]: df1['BsmtExposure'] = np.where(df1['BsmtExposure'].isnull(),df1['BsmtExposure'].mode(),df1['BsmtExposure'])

In [43]: df1['BsmtFinType1'] = np.where(df1['BsmtFinType1'].isnull(),df1['BsmtFinType1'].mode(),df1['BsmtFinType1'])

In [44]: df1['MasVnrType'] = np.where(df1['MasVnrType'].isnull(),df1['MasVnrType'].mode(),df1['MasVnrType'])

In [45]: df1['BsmtQual'] = np.where(df1['BsmtQual'].isnull(),df1['BsmtQual'].mode(),df1['BsmtQual'])
```

- **Handling outliers and skewness**

  Considering the domain of the problem the outliers can be the real values and which leads to the skewness of the columns .By analysing the data in the columns, it is clear that the number of zeros is the main reason for the skewness. So for both issues, I have used power transformation.

  ```
  In [51]: from sklearn.preprocessing import power_transform
           df_new = pd.DataFrame(power_transform(df1),columns=df1.columns)

  In [52]: df_new.head(10)
  ```

  Out[52]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	LotConfig	LandSlope	Neighborhood	Condition1	Condition2
0	1.370435	-0.162456	0.057324	-1.213954	0.058621	-0.004343	-1.366794	0.341434	0.617281	-0.238775	0.198357	0.053284	0.0038
1	-1.167999	-0.162456	1.125853	1.100521	0.058621	-0.004343	-1.366794	0.341434	0.617281	4.188040	0.037086	0.053284	0.0038
2	0.490047	-0.162456	1.007841	0.158048	0.058621	-0.004343	-1.366794	0.341434	-1.482445	-0.238775	0.512775	0.053284	0.0038
3	-1.167999	-0.162456	1.502999	0.496002	0.058621	-0.004343	-1.366794	0.341434	0.617281	-0.238775	0.356843	0.053284	0.0038
4	-1.167999	-0.162456	0.057324	1.196626	0.058621	-0.004343	-1.366794	0.341434	-1.025661	-0.238775	0.356843	0.053284	0.0038
5	0.490047	-0.162456	-0.543121	0.855555	0.058621	-0.004343	-1.366794	0.341434	0.617281	-0.238775	-0.642095	0.053284	0.0038
6	-1.167999	-0.162456	0.057324	0.424957	0.058621	-0.004343	-1.366794	0.341434	0.617281	-0.238775	1.114534	0.053284	0.0038
7	-1.167999	-0.162456	0.846644	0.717859	0.058621	-0.004343	0.753907	0.341434	-1.725008	-0.238775	1.114534	0.053284	0.0038
8	-1.167999	-0.162456	0.057324	0.001967	0.058621	-0.004343	0.753907	0.341434	-1.725008	-0.238775	-0.822623	-1.390093	0.0038
9	0.237618	-0.162456	0.509784	-0.152859	0.058621	-0.004343	0.753907	0.341434	0.617281	-0.238775	0.037086	0.053284	0.0038

- **Encoding**

  For the encoding of all the categorical columns, I used Ordinal Encoder.

  ```
  In [49]: from sklearn.preprocessing import OrdinalEncoder

  In [50]: ord_en = OrdinalEncoder()
           for i in df1.columns:
               if(df1[i].dtypes=='O'):
                   df1[i] = ord_en.fit_transform(df1[i].values.reshape(-1,1))
  ```

Data Inputs- Logic- Output Relationships

To analysis the relation between input and output first I have separated the numerical, categorical, and temporal columns.

```python
In [6]: numerical_feats = [i for i in df.columns if df[i].dtypes!='O']
        print(len(numerical_feats))
        print(numerical_feats)

38
['Id', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFi
nSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalf
Bath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars', 'Gara
geArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold',
'SalePrice']
```

```python
In [7]: categorical_feats = [i for i in df.columns if df[i].dtypes=='O']
        print(len(categorical_feats))
        print(categorical_feats)

43
['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1',
'Condition2', 'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'ExterQual', 'Exte
rCond', 'Foundation', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating', 'HeatingQC', 'CentralA
ir', 'Electrical', 'KitchenQual', 'Functional', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond', 'Paved
Drive', 'PoolQC', 'Fence', 'MiscFeature', 'SaleType', 'SaleCondition']
```

Hardware and Software Requirements and Tools Used

Tool:
Jupyter NoteBook 6.1.4:
- Web-based interactive computing notebook Environment.
  Software Requirement:
- The client environment may be Windows, macOS, or Linux.
  Hardware Requirement:
- CPU: 2 x 64-bit, 2.8 GHz, 8.00 GT/s CPUs or better.
- Memory: minimum RAM size of 32 GB, or 16 GB RAM with 1600 MHz DDR3 installed, for a typical installation with 50 regular users.
  Libraries:

- Pandas: For reading CSV file, Converting dataset into a data frame, handling date datatype, and more.
- Seaborn and matplotlib: For EDA and Visualization.
- sklearn: For preprocessing, scaling, and modeling of data.

Model/s Development and Evaluation

• Identification of possible problem-solving approaches (methods)

To prepare the data for data modelling it needs to be clean and scale.The size of this dataset is 1163 rows and 81 columns.

The number of columns is more therefore to handle this number of columns it is important to analyse the significance of the column.Two columns Utilities and Id have all the unique values and PoolQC and MiscFeatures have almost null values therefore I have deleted these four columns. Moreover, many columns have many zeros observations which increase the skewness of the columns also taking the domain of the problem into the consideration it is possible that the values may vary for each column. So we can not remove the outliers also.

• Testing of Identified Approaches (Algorithms)

Feature Selection Algorithms:
RFE , SelectFromModel.
Some Standard Regression algorithms:
Linear Regression, Decision Tree Regressor, Support Vector Regressor, Kneigbors Regressor.
Ensemble Techniques:
Random Forest Regressor, AdaBoost Regressor, Gradient BoostRegressor
Hyper Parametric Tuning of Gradient Boost Regressor

- ## **Run and Evaluate selected models**

As mentioned above I have applied 7 to 8 algorithms to predict the label. But I selected Gradient Boost Regressor as the best model on the basis of the difference between cross-validation score and r2 score and mean square root value and the plot between the predicted value and actual values.

```
In [63]: grad =GradientBoostingRegressor(n_estimators=300,min_samples_split=150,max_depth=4)
         grad.fit(x_train,y_train)
         pr = grad.predict(x_test)
         r2_score(pr,y_test)
```

Out[63]: 0.8721102348159874

```
In [109]: cross_val.mean()
```

Out[109]: 0.871814342606005

```
In [110]: print('MSE of final Model',mean_squared_error(pr,y_test))
          print('RMSE of final Model',np.sqrt(mean_squared_error(pr,y_test)))
```

MSE of final Model 0.10468290378930978
RMSE of final Model 0.3235473748762456

```
In [50]: sns.regplot(x=y_test, y=pr, data=df1)
```

Out[50]: <AxesSubplot:xlabel='SalePrice'>
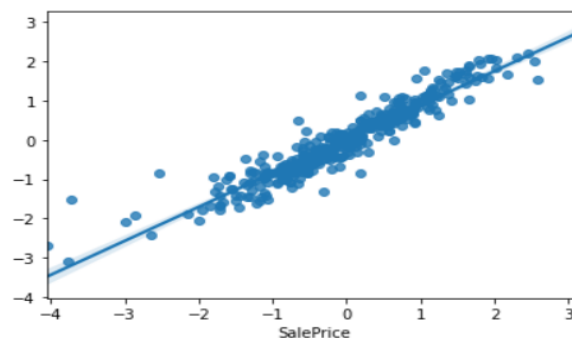
- **Key Metrics for success in solving a problem under consideration**

Key Metrics that I have used in this project are r2 score, Mean Square Error(MSE), Root Mean Square Error(RMSE), and for checking the fitting of the model I have used cross-validation score
R Square measures how much variability in the dependent variable can be explained by the model.
Mean Square Error is an absolute measure of the goodness for the fit.
Cross-validation is a statistical method used to estimate the skill of machine learning models.
Also, I have used the regplot to analyze the best fitted line of the selected model.

- **Visualizations**

1. Sale Price with Street

```
In [15]: sns.catplot(x='Street',y='SalePrice',kind='boxen',data=df)

Out[15]: <seaborn.axisgrid.FacetGrid at 0x20a7005ba60>
```
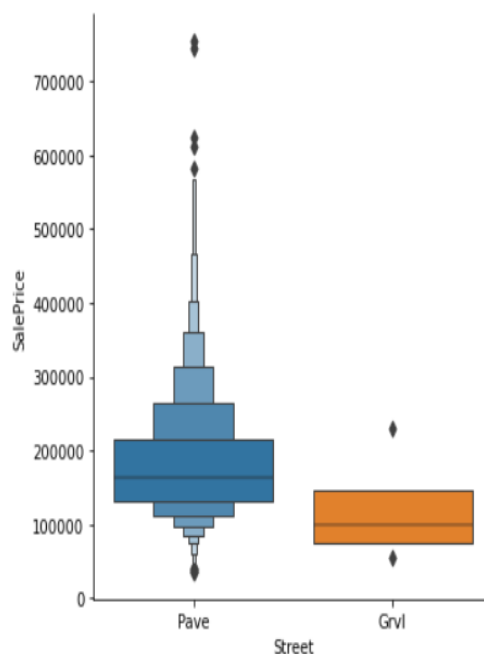
2. Sale Price with Foundation

```
In [18]: sns.catplot(x='Foundation',y='SalePrice',kind='boxen',data=df)
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x20a704fa430>
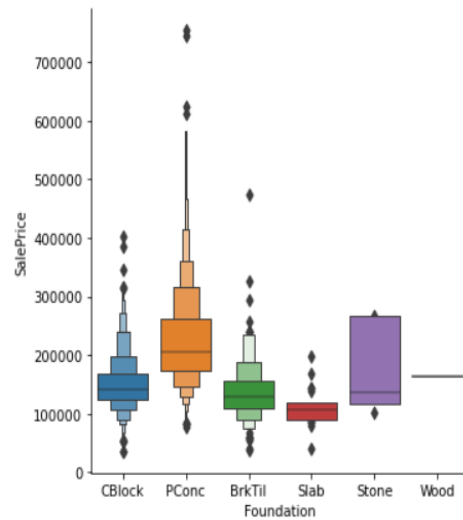
3. Sale Price with BsmtQual

```
In [22]: print(df['BsmtQual'].value_counts())
         sns.catplot(x='BsmtQual',y='SalePrice',kind='boxen',data=df)
```

```
TA    517
Gd    498
Ex     94
Fa     29
Name: BsmtQual, dtype: int64
```
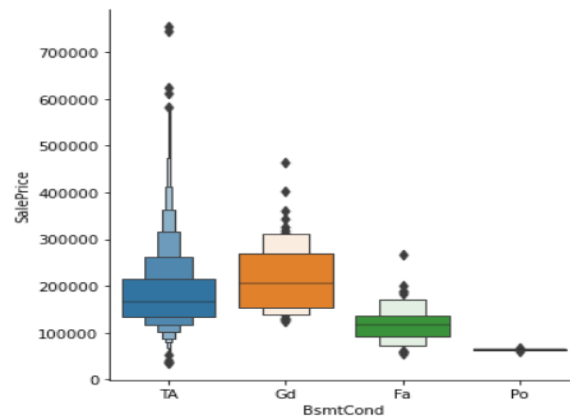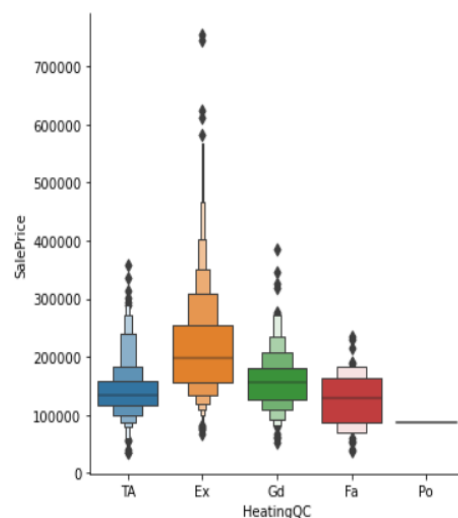
Out[22]: <seaborn.axisgrid.FacetGrid at 0x17a9d2d7070>
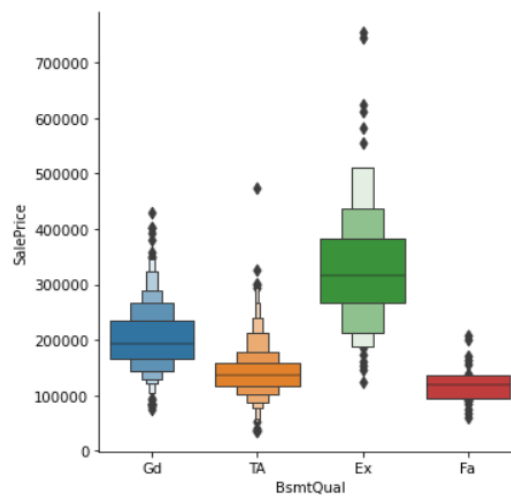
4. Sale Price with BsmtCond

```
In [21]: print(df['BsmtCond'].value_counts())
         sns.catplot(x='BsmtCond',y='SalePrice',kind='boxen',data=df)

         TA    1041
         Gd      56
         Fa      39
         Po       2
         Name: BsmtCond, dtype: int64

Out[21]: <seaborn.axisgrid.FacetGrid at 0x17a9d343df0>
```


5. Sale Price with HeatingQC

```
In [20]: print(df['HeatingQC'].value_counts())
         sns.catplot(x='HeatingQC',y='SalePrice',kind='boxen',data=df)

         Ex    585
         TA    352
         Gd    192
         Fa     38
         Po      1
         Name: HeatingQC, dtype: int64

Out[20]: <seaborn.axisgrid.FacetGrid at 0x17a9d2c0b50>
```

6. Sale Price with Exterior

```
In [59]: plt.figure(figsize=(14,4))
         sns.stripplot(x='Exterior2nd',y='SalePrice',data=df)
```

```
Out[59]: <AxesSubplot:xlabel='Exterior2nd', ylabel='SalePrice'>
```
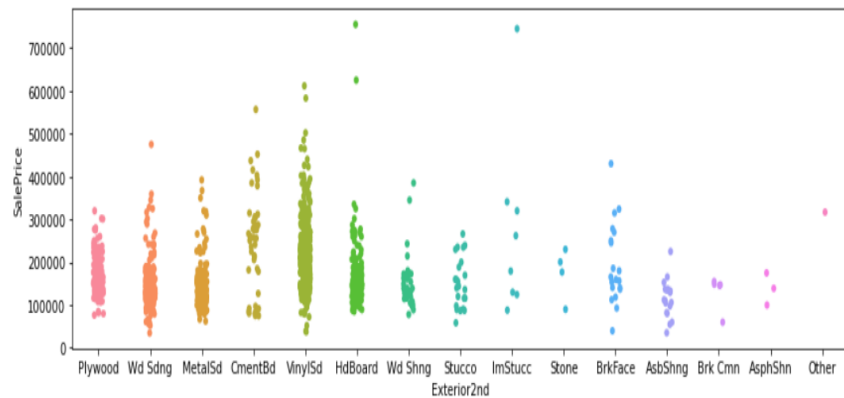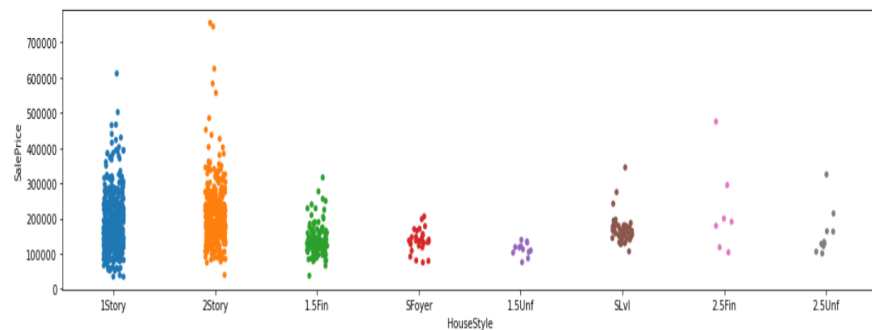

7. Sale Price with House Style

```
In [25]: print(df['HouseStyle'].value_counts())
         plt.figure(figsize=(18,4))
         sns.stripplot(x='HouseStyle',y='SalePrice',data=df)
```

```
1Story    578
2Story    361
1.5Fin    121
SLvl       47
SFoyer     32
1.5Unf     12
2.5Unf     10
2.5Fin      7
Name: HouseStyle, dtype: int64
```

```
Out[25]: <AxesSubplot:xlabel='HouseStyle', ylabel='SalePrice'>
```
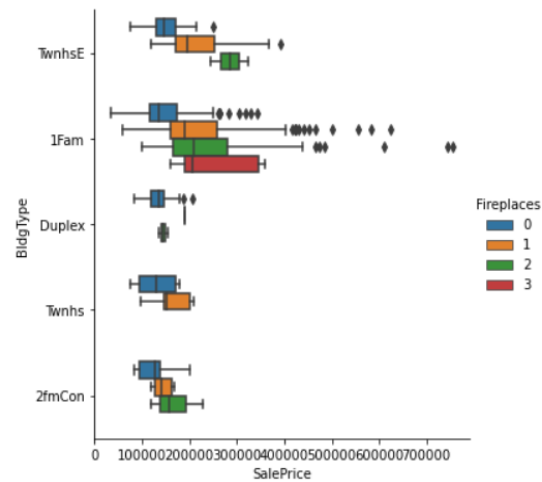
8. Sale Price with Building Type

```
In [30]: plt.figure(figsize=(14,4))
         sns.catplot(x='SalePrice',y='BldgType',hue='Fireplaces',kind='box',orient='h',data=df)

Out[30]: <seaborn.axisgrid.FacetGrid at 0x17a9edff0a0>

         <Figure size 1008x288 with 0 Axes>
```
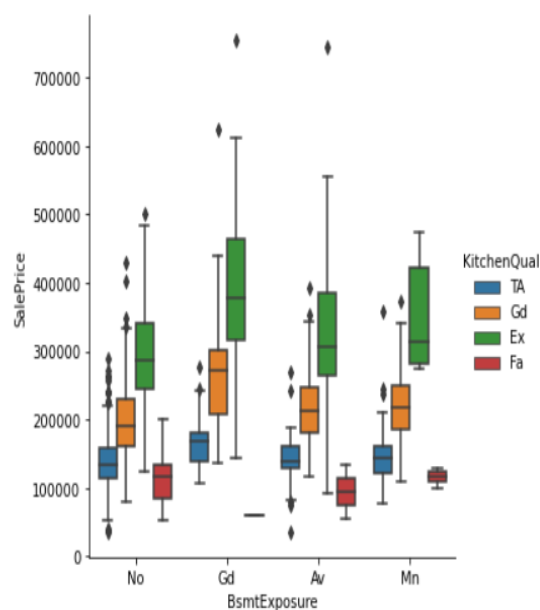

9. Sale Price with KitchQual

```
In [35]: sns.catplot(x='BsmtExposure',y='SalePrice',hue='KitchenQual',kind='box',data=df)

Out[35]: <seaborn.axisgrid.FacetGrid at 0x17aa0475c70>
```
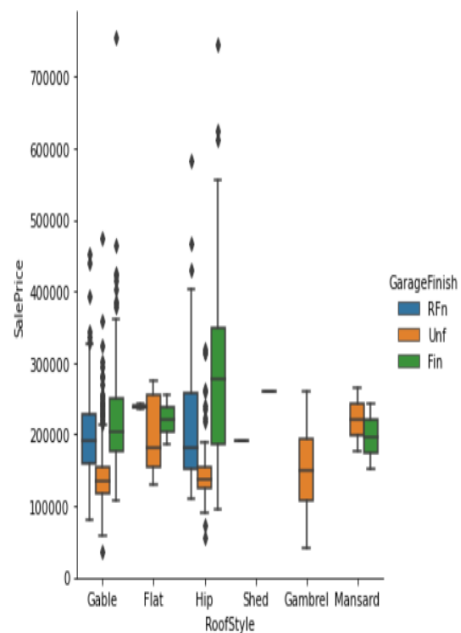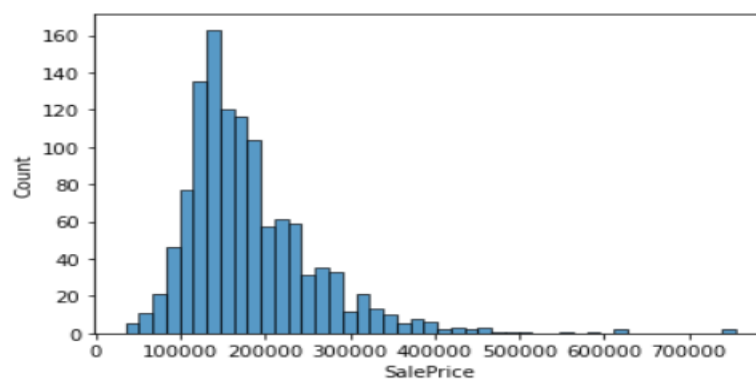
10. Sale Price with Roof Style

```
In [38]: sns.catplot(x='RoofStyle',y='SalePrice',hue='GarageFinish',kind='box',data=df)

Out[38]: <seaborn.axisgrid.FacetGrid at 0x17aa04eaa90>
```
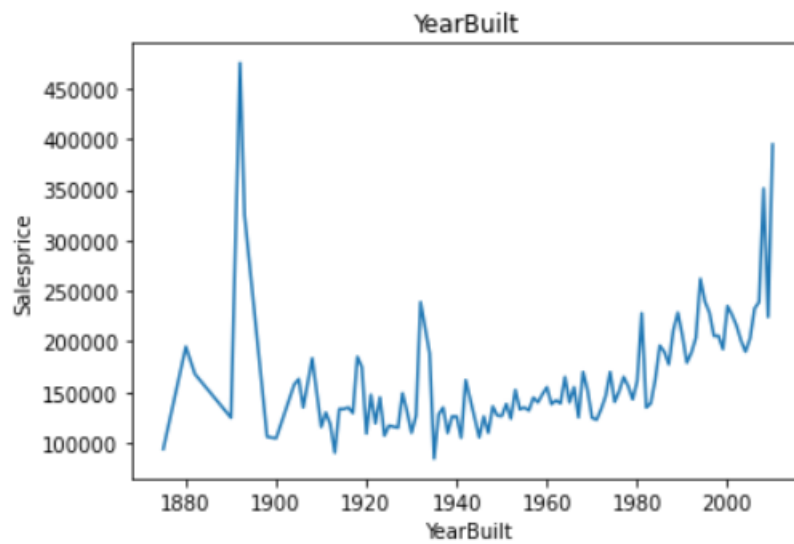

11.Sale Price
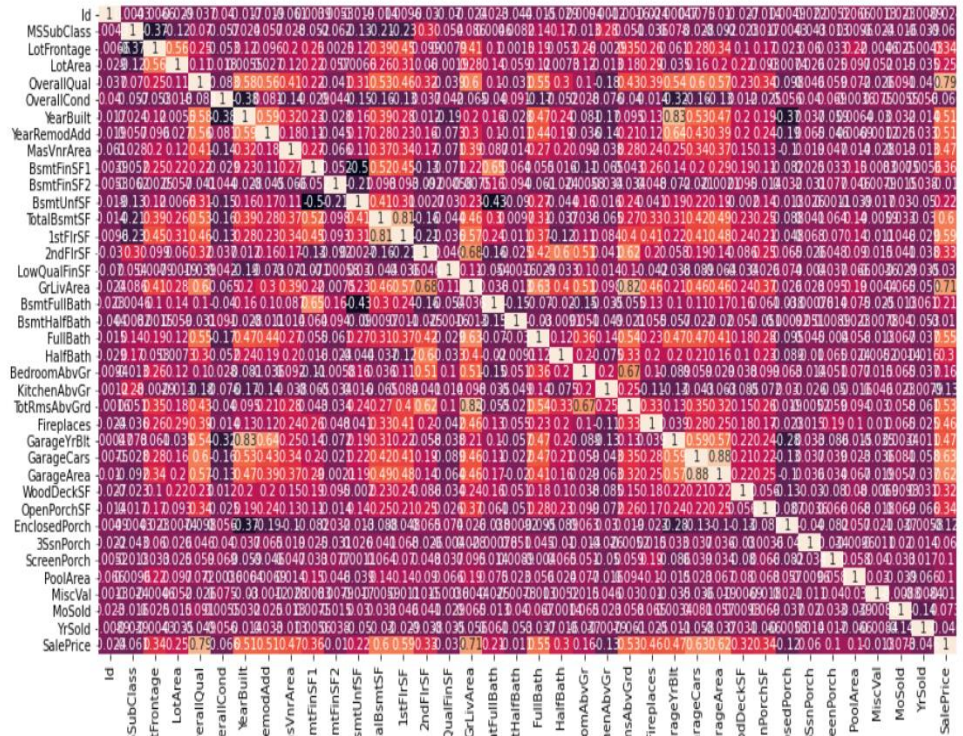
12.Sale Price with Year Sold

13.Sale Price with Year Built

14. Multivariate Analysis

```
In [46]: plt.figure(figsize=(18,8))
         sns.heatmap(df.corr(),annot=True)

Out[46]: <AxesSubplot:>
```


• Observations:

- The range of the target value is 100000 to 500000
  and frequency of the sale price is between 100000 to 200000
    - Houses sold in the year 2007 were highest and lowest in the year 2010 which seems practically not possible.
    - Houses built in the year 1990 were sold at high prices.
    - On observing on the basis of different parameters :
    - Houses with Ex Basement Quality, PCond Foundation, Ex kitchen quality, 2nd story is expensive.
    - Excellent categories are of high prices.

- ## Interpretation of the Results

- All the independent columns (categorical and numerical) are imbalanced. In categorical columns, only one or two categories are of high frequency, and in numerical columns, zeros shrink the graphs.
- Categories that have excellent category show high prices.
- Looking at the temporal columns shows impractical records saying the prices of the houses reduced with the year sold.
- As the number of independent columns is more it is difficult to analyse each column. Also, the number of rows is less as compared to independent columns.
- Many columns have null values which I have treated using two techniques:
  creating new categories for more null values and replacing them with mode for less null values columns.
- For encoding the categorical variables I used an ordinal encoder.
  Also, the columns have outliers and skewness reason being the more number of zeros in numerical features for which I have used power transform.
- The full bath is highly correlated with the sale price and the year sold is least correlated.
- Following are the values of r2 scores for each algorithm:
  1.Linear Regression:
  r2 score: 0.89
  2.Support Vector Regressor:
  r2 score: 0.73
  3.Decision Tree Regressor:
  r2 score: 0.64
  4.Kneigbors Regressor:
  r2 score: 0.60
  4.Random Forest Regressor:
  r2 score: 0.85
  5.Ada Boost Regressor :
  r2 score: 0.80
  6.Gradient Boost Regressor
  r2 score: 0.89
  7.Grid Search CV:
  r2 score: 0.87

CONCLUSION

- ## Key Findings and Conclusions of the Study

  All the columns are imbalanced and the major drawback of the dataset is the number of columns.

  The key challenges of the problem were handling null values and skewed data .Also, some categorical columns have many categories from which we can not conclude results because of imbalance and null values. For numerical columns, more zeros are there.

- ## Learning Outcomes of the Study in respect of Data Science

  As the dataset contains 81 columns it is good practice to drop the column with no correlation with the target variable or with most null values, So I dropped PoolQC, misFeatures, Utilities, and Id.

  Because of 81 columns, I have tried two techniques for feature selection to eliminate the non-significant features.

  Some standard classification algorithms I have used are: Linear Regression, Support Vector regressor, Decision Tree Regressor, Kneigbors Regressor, and some ensemble techniques I have used are Random Forest Regressor and Ada Boost Regressor, Gradient Boost Regressor

  I have also used a cross-validation score to validate the overfitting of the model.

  EDA is time consuming as we need to do analysis between numerical and Numerical variable, categorical and numerical column and also the number Of columns are also a big factor.

  Predicting target variable after feature selection 2% score is less, therefore, I have not used that model because of the number of rows

  If for 1163 rows only it is affecting the accuracy for more rows it may not give a precise prediction.

- ## Limitations of this work and Scope for Future Work

  If the data can be balanced and all the values were there the model can easily predict the data. The number of dimensions is one of the limitations of the dataset. Also treating null values affect the precision of the model as

many columns have null values. EDA of this type of data took time because of the number of dimensions. If fewer null values could be there and it will be good for predicting the Sales Price.