FLIP ROBO

HOUSING: PRICE PREDICTION

Submitted by:

SAYAN PAUL

ACKNOWLEDGMENT

I would like to thank Surprise Housing for providing with the dataset and our MSE , MS Khusboo Garg for her constant help during the project.

Also I would like to thank websites like stackoverflow, geeksforgeeks and sklearn documentary for helping me when I got stuck somewhere.

INTRODUCTION

- ## Business Problem Framing

  We were 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.

- ## Review of Literature

  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 called ‘Surprise Housing’.

- ## Motivation for the Problem Undertaken
  1. Developing an understanding of the real estate market.
  2. Improving my machine learning skills.

Analytical Problem Framing

- Data Sources and their formats

  We were provided the training data from Flip_Robo Technologies in csv format.
  Test data was also provided by the same in csv format.

- Data Preprocessing Done

  1. Dropping columns having very high number of null values.
  2. Replacing other NaN values with mean or mode depending on the data type.

Column having NaN values	**Replacement of NaN values.**
LotFrontage	mean
MasVnrType GarageYrBlt	median
MasVnrArea GarageCond GarageQual GarageFinish GarageType BsmtQual BsmtCond BsmtExposure	mean

  3. Label encoding
  4. Skewness revomal using power transformer.
  5. Outlier removal
  6. Droppping of columns having very low correlation with target.

7. Multi collinearity removal.

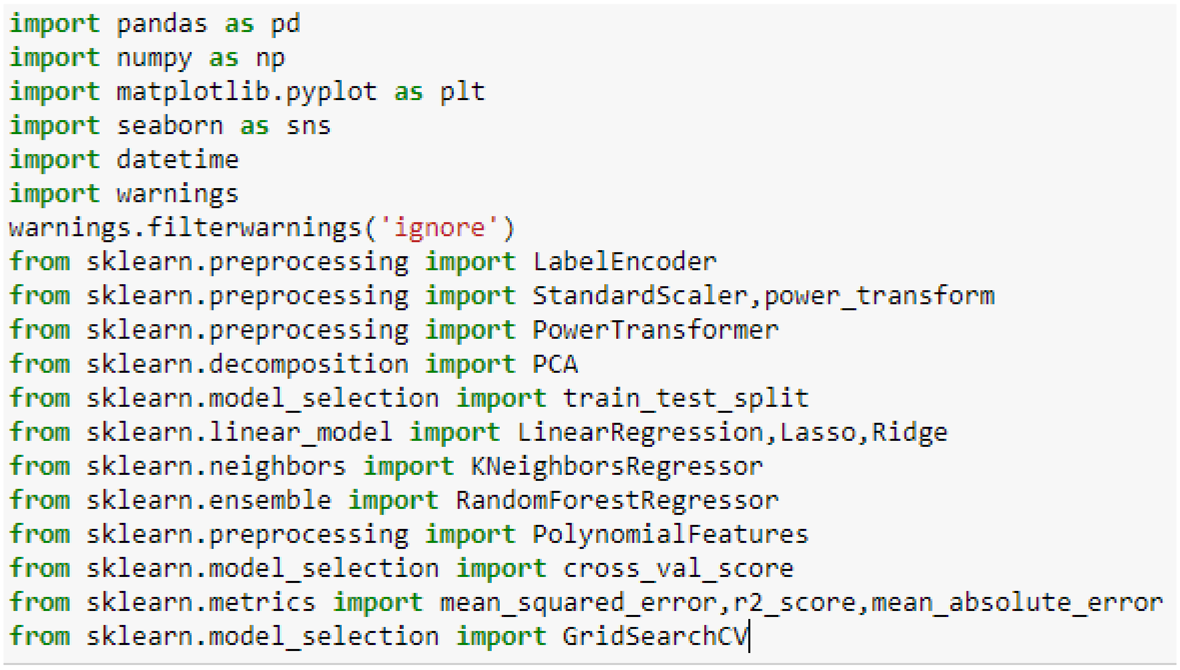
- <u>Data Inputs- Logic- Output Relationships</u>

  Random forest Regressor was used here because it was giving the best results.

- <u>State the set of assumptions (if any) related to the problem under consideration</u>

  1. During outlier removal, we kept the absolute value of z less than 3.5 instead of 3.0 to avoid huge loss of data.
  2. We have created two datasets for out analysis→df_rev and df_irev. df_rev doesnot contain the columns which has low correlation with the target. df_irev is created after viewing the visualizations and only the absolutely insensible columns are removed.
  3. df_rev is mainly used for linear,lasso ,ridge regression. While df_irev is used for KNNeighbors and RandomForest Regression.

- <u>Hardware and Software Requirements and Tools Used</u>



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler,power_transform
from sklearn.preprocessing import PowerTransformer
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression,Lasso,Ridge
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error,r2_score,mean_absolute_error
from sklearn.model_selection import GridSearchCV
```

Model/s Development and Evaluation

- **Identification of possible problem-solving approaches (methods)**

  The algorithms tried for solving this problem are:-

  1. Linear Regression
  2. Ridge Regression
  3. Lasso Regression
  4. KNeighbors Regression
  5. Random Forest Regression

- **Testing of Identified Approaches (Algorithms)**

Hyperparameter tuning was performed on the above above algorithms and the best results are as shown below.

Algorithms	**Best r2_score**
Linear Regression	0.9183461118175837
Ridge Regression	0.9013675441601351
Lasso Regression	0.9013903564592171
KNN Regression	0.9041579347151206
Random Forest Regression	0.927325610260045

- **Run and Evaluate selected models**

```python
#final LinearRegression model
#testing with df_rev
x=df_irev.drop(columns=['SalePrice'],axis=1)
y=df_irev['SalePrice'].values
pca_reg=PCA(n_components=63)
xpca=pca_reg.fit_transform(x)
xpca=pd.DataFrame(xpca)

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=12)
lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
print('r2_score is:',r2_score(y_test,y_pred))
print('root_mean_sqaured_error:',mean_squared_error(y_test,y_pred)**0.5)
print('mean_absolute_error:',mean_absolute_error(y_test,y_pred))
```

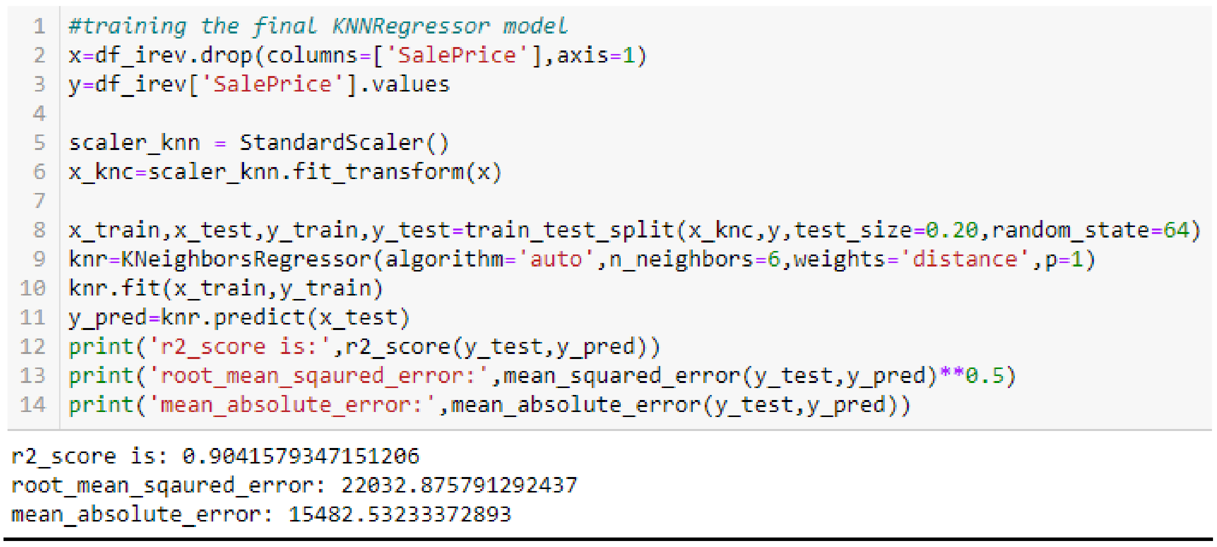
```
r2_score is: 0.9183461118175837
root_mean_sqaured_error: 21779.904026224704
mean_absolute_error: 15966.113854410862
```

```python
#training the final Ridge regression model
x=df_rev.drop(columns=['SalePrice'],axis=1)
y=df_rev['SalePrice'].values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=27)

r=Ridge(alpha= 1, solver= 'auto')
r.fit(x_train,y_train)
y_pred=r.predict(x_test)
print('r2_score is:',r2_score(y_test,y_pred))
print('root_mean_sqaured_error:',mean_squared_error(y_test,y_pred)**0.5)
print('mean_absolute_error:',mean_absolute_error(y_test,y_pred))
```

```
r2_score is: 0.9013675441601351
root_mean_sqaured_error: 23115.716735518836
mean_absolute_error: 18527.42855509859
```

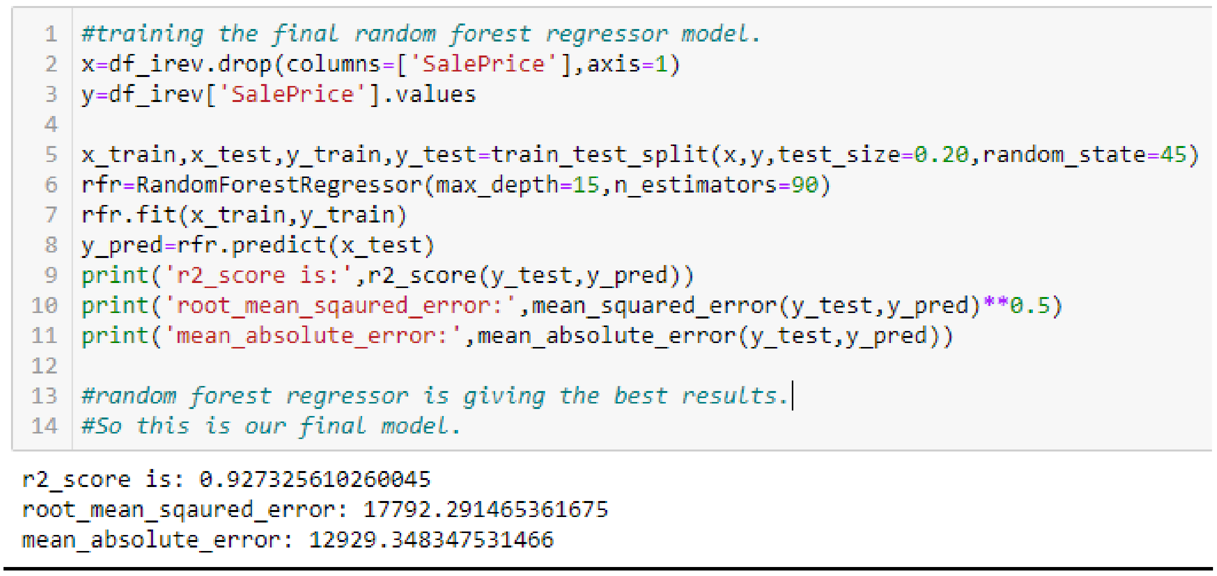
```python
#training the final Lasso regression model
x=df_rev.drop(columns=['SalePrice'],axis=1)
y=df_rev['SalePrice'].values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=27)

l=Lasso(alpha= 1, max_iter=1300,selection='random')
l.fit(x_train,y_train)
y_pred=l.predict(x_test)
print('r2_score is:',r2_score(y_test,y_pred))
print('root_mean_sqaured_error:',mean_squared_error(y_test,y_pred)**0.5)
print('mean_absolute_error:',mean_absolute_error(y_test,y_pred))
```

```
r2_score is: 0.9013903564592171
root_mean_sqaured_error: 23113.043410935774
mean_absolute_error: 18530.60519239076
```



```python
#training the final KNNRegressor model
x=df_irev.drop(columns=['SalePrice'],axis=1)
y=df_irev['SalePrice'].values

scaler_knn = StandardScaler()
x_knc=scaler_knn.fit_transform(x)

x_train,x_test,y_train,y_test=train_test_split(x_knc,y,test_size=0.20,random_state=64)
knr=KNeighborsRegressor(algorithm='auto',n_neighbors=6,weights='distance',p=1)
knr.fit(x_train,y_train)
y_pred=knr.predict(x_test)
print('r2_score is:',r2_score(y_test,y_pred))
print('root_mean_sqaured_error:',mean_squared_error(y_test,y_pred)**0.5)
print('mean_absolute_error:',mean_absolute_error(y_test,y_pred))
```

```
r2_score is: 0.9041579347151206
root_mean_sqaured_error: 22032.875791292437
mean_absolute_error: 15482.53233372893
```

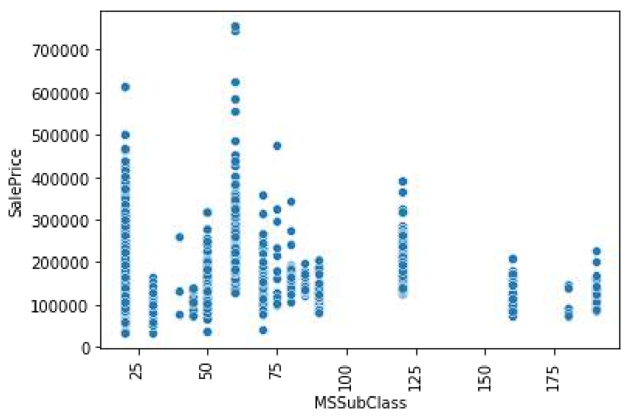


```python
#training the final random forest regressor model.
x=df_irev.drop(columns=['SalePrice'],axis=1)
y=df_irev['SalePrice'].values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=45)
rfr=RandomForestRegressor(max_depth=15,n_estimators=90)
rfr.fit(x_train,y_train)
y_pred=rfr.predict(x_test)
print('r2_score is:',r2_score(y_test,y_pred))
print('root_mean_sqaured_error:',mean_squared_error(y_test,y_pred)**0.5)
print('mean_absolute_error:',mean_absolute_error(y_test,y_pred))

#random forest regressor is giving the best results.
#So this is our final model.
```

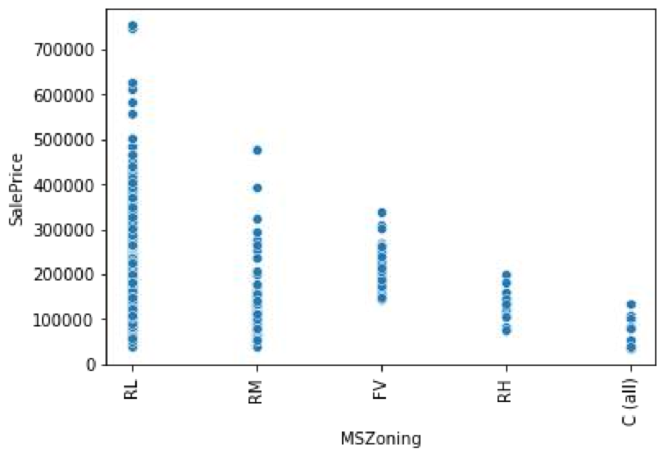
```
r2_score is: 0.927325610260045
root_mean_sqaured_error: 17792.291465361675
mean_absolute_error: 12929.348347531466
```

- <u>Key Metrics for success in solving problem under consideration</u>
    1. r2_score
    2. root_mean_squared_error
    3. mean_absolute_error

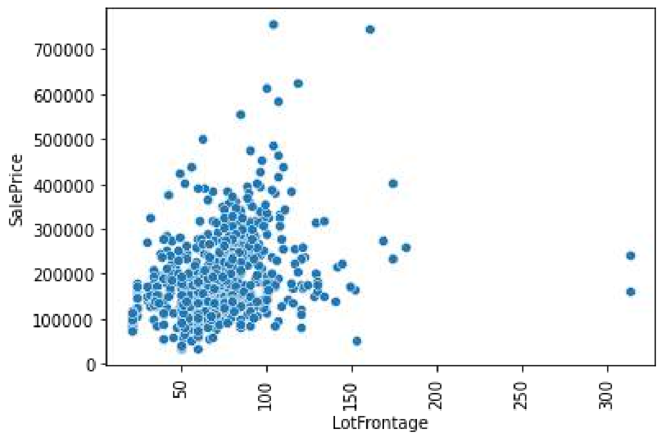
- <u>Visualizations</u>



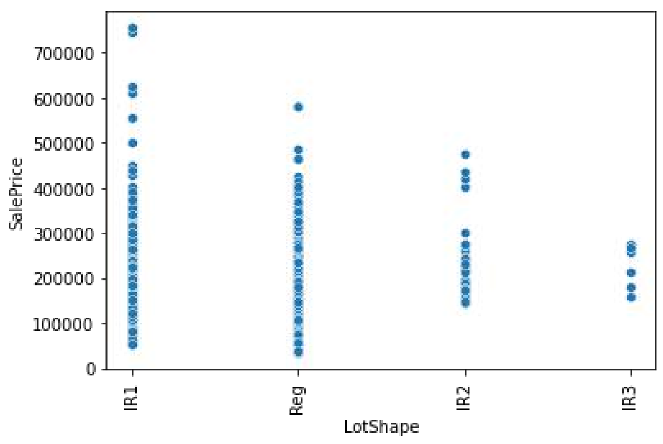
STORY 1946 & NEWER ALL STYLES has higher price
STORY 1946 & NEWER has higher price.



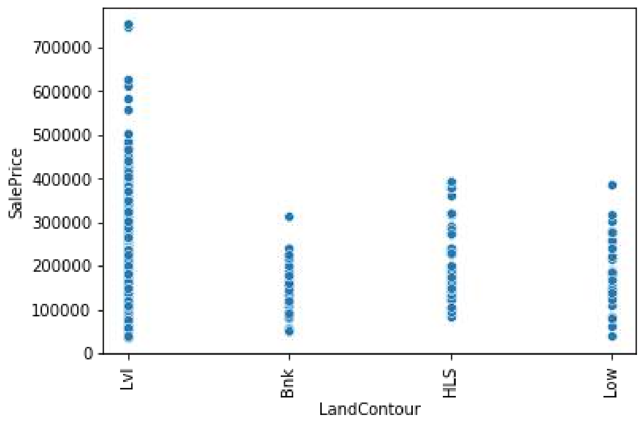
Residential Low Density houses can be sold at higher prices.
Commercial(all) are generally sold at very low price.(Also present in very low numbers)



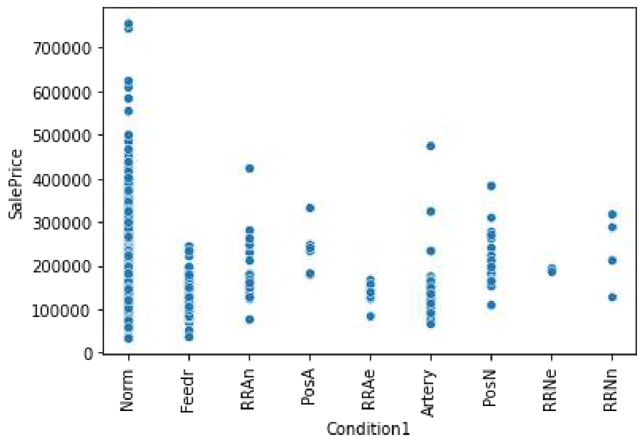
Price increases with LotFrontage

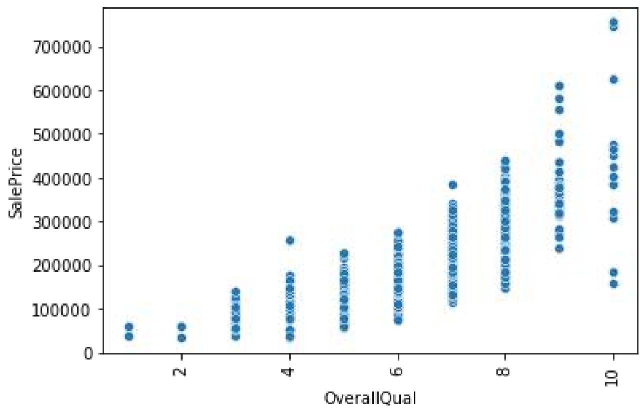


Highly Irregular LotShape has lower price.

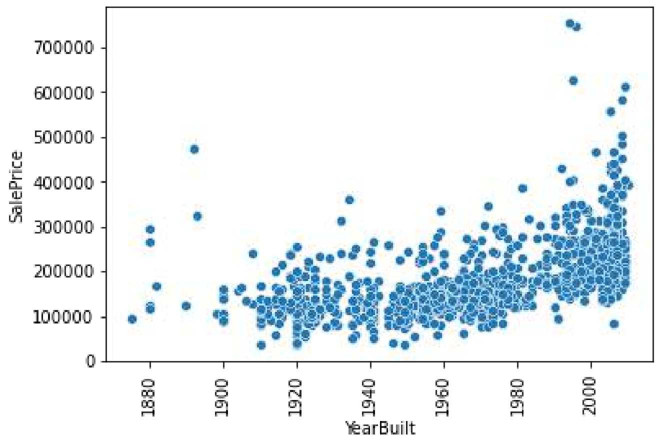


Level land is sold at higher price.

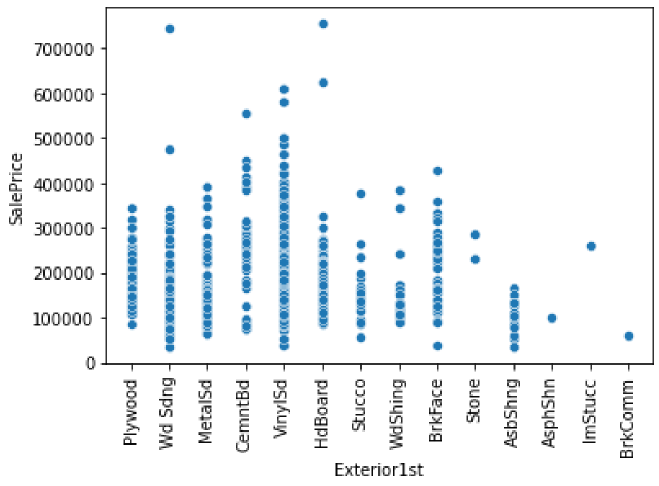


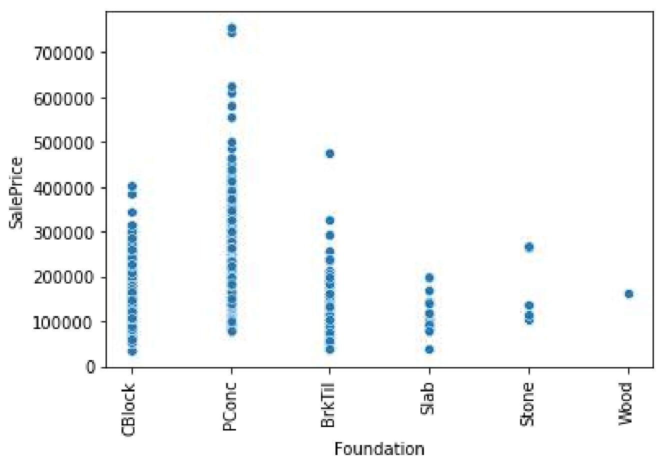


Price shows increase with overall quality.

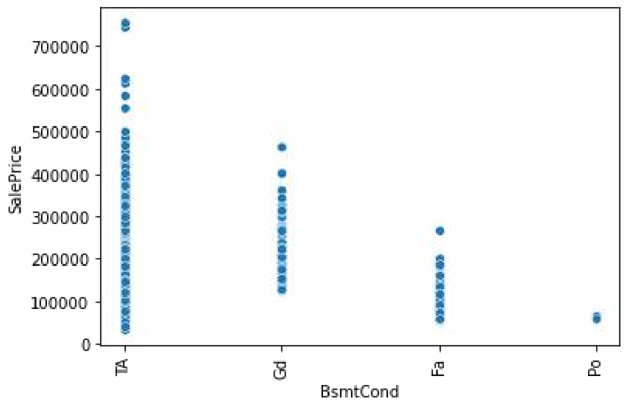


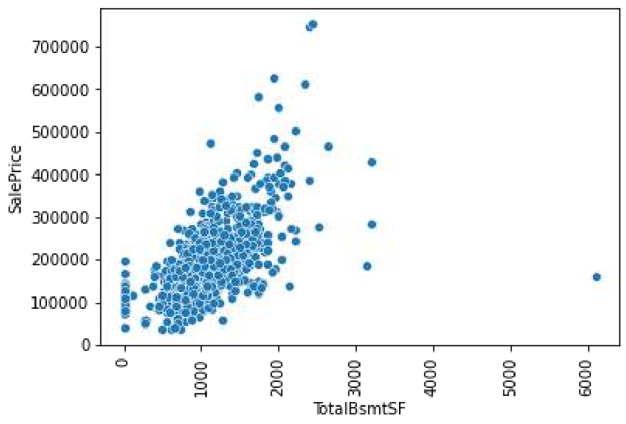
Older buildings as sold as lower price.



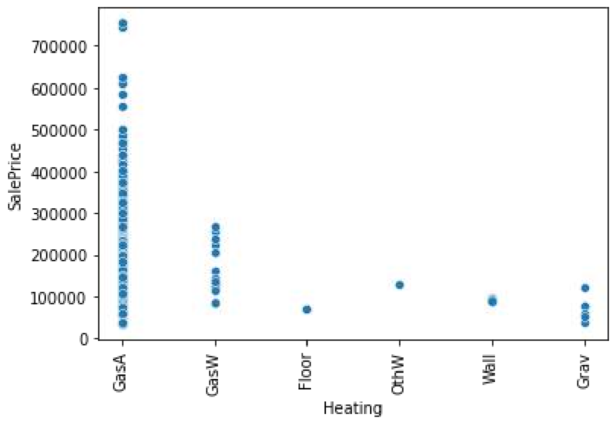


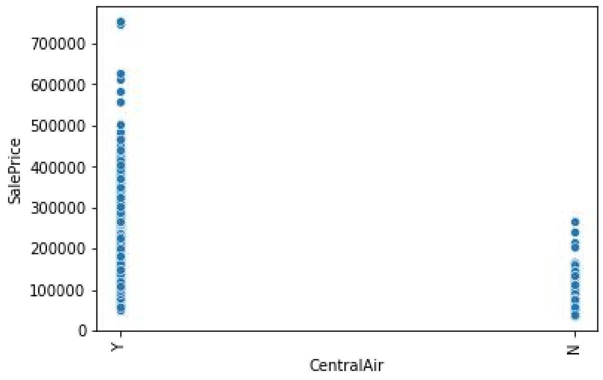
Poured concrete foundation buildings has higher price.



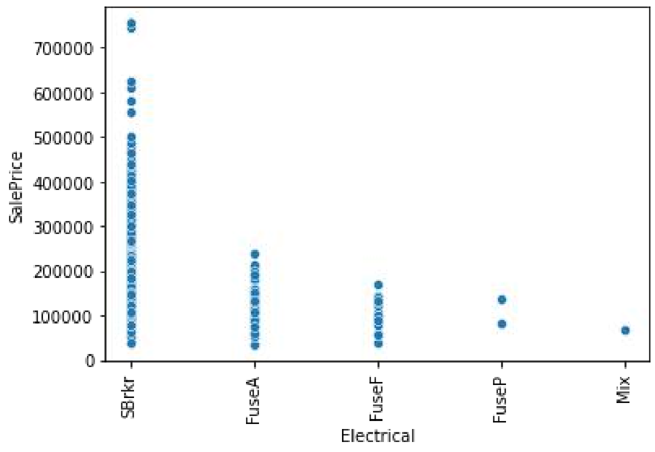


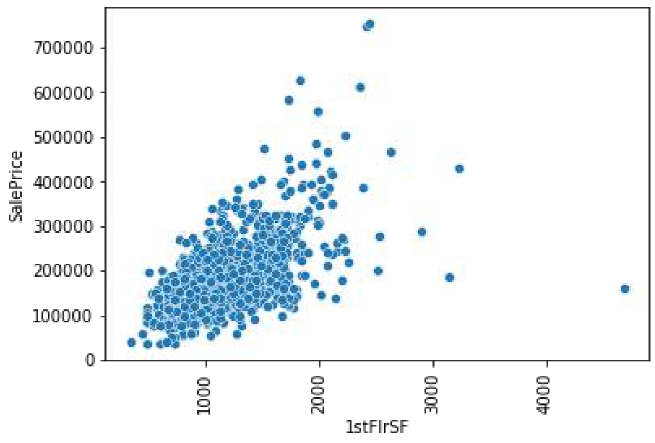
Price is increasing with total basement finished surface area.

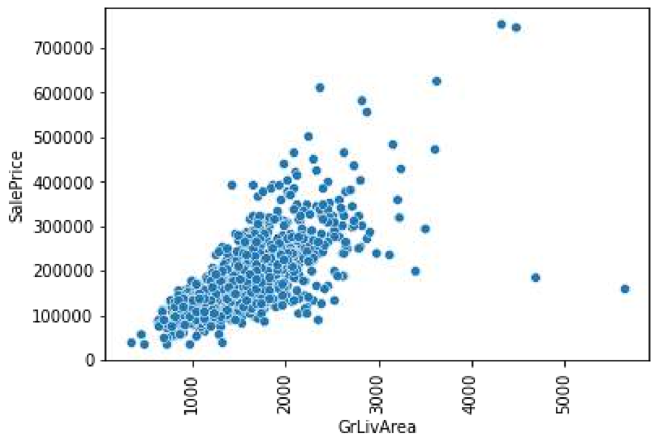


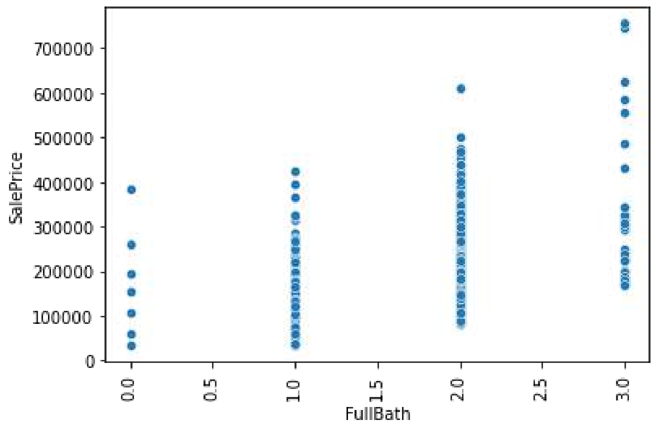


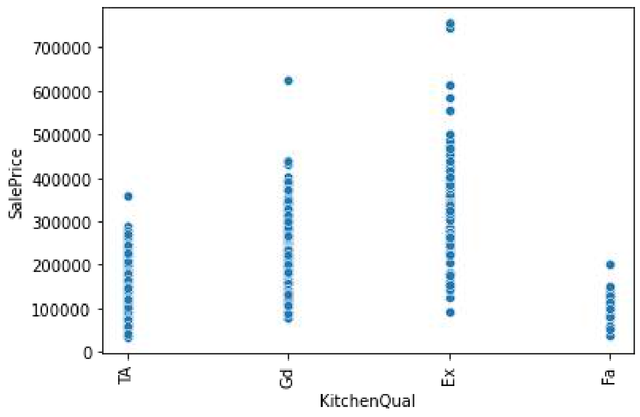
Buildings with central air conditioner is sold at higher price.

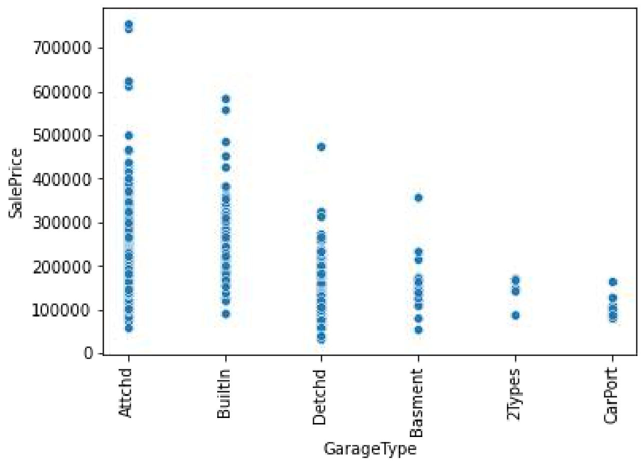


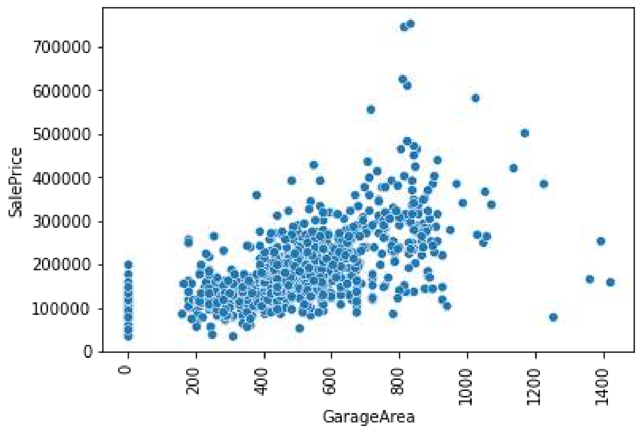


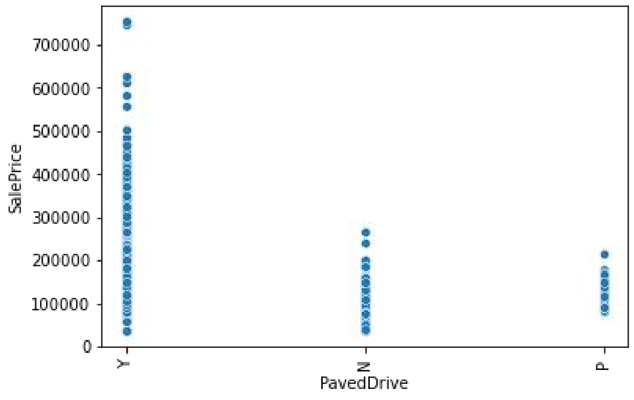

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
1000
2000
3000
4000
5000
GrLivArea

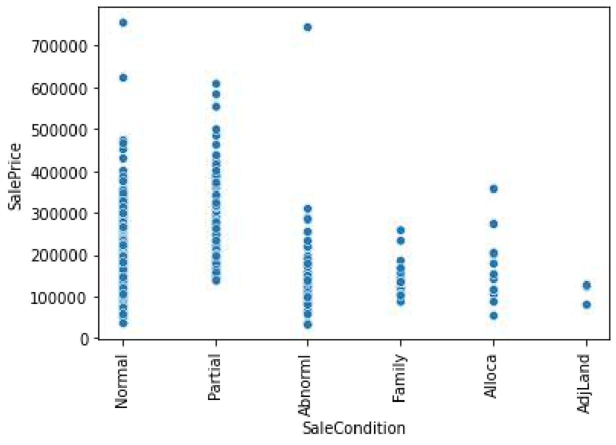

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
FullBath


SalePrice
0
100000
200000
300000
400000
500000
600000
700000
TA
Gd
Ex
Fa
KitchenQual


SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Attchd
BuiltIn
Detchd
Basment
2Types
CarPort
GarageType







- <u>Interpretation of the Results</u>

  OverallQual,GrLivArea,GarageArea,TotalBsmtSF,1stFlrSF,FullBath,YearBuilt etc has very high correlation with SalePrice.

<u>**CONCLUSION**</u>

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

  Dropping columns having low correlation with the target is not always the only way. Sometimes those correlation attributes are necessary to give good results with KNNRegressor and DecisionTree regressor, RandomForest Regressor.

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

  More hyper parameter tunings can be tried with the algorithms.