HOUSING PROJECT

Submitted by:

AMANDEEP KAUR

ACKNOWLEDGMENT

I would like to express my special thanks of gratitude to Datatrained academy and fliprobo for their able guidance and support in completing my project.

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.

- Conceptual Background of the Domain Problem
- Basic knowledge of python
- Basic analytic skills

- Review of Literature
- In this project after studying and analysing the given data we have tried to build a model which can find price of a house depending on different various parameters.

- Motivation for the Problem Undertaken

  This project is to analyse, understand and predict the price of a house depending of different parameters through given data.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem
- Checking the shape and null values (if any)
- EDA process
- Removing outliers
- Training the data using different algorithms

- Data Sources and their formats
- Source: Provided by fliprobo
- Format: csv file

- Data Pre-processing Done
- Data Inputs- Logic- Output Relationships

- State the set of assumptions (if any) related to the problem under consideration
- No assumptions

- Hardware and Software Requirements and Tools Used
- import warnings
- warnings.simplefilter("ignore")
- import seaborn as sn
- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt.
- from sklearn.preprocessing import LabelEncoder
- from sklearn.preprocessing import StandardScaler
- from sklearn.model_selection import train_test_split
- from sklearn.linear_model import LinearRegression
- from sklearn.model_selection import cross_val_score
- from sklearn.metrics import r2_score

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
- Removing skewness and then training the data to give better accuracy
- Testing of Identified Approaches (Algorithms)
- from sklearn.preprocessing import LabelEncoder
- from sklearn.preprocessing import StandardScaler
- from sklearn.model_selection import train_test_split
- from sklearn.linear_model import LinearRegression
- from sklearn.model_selection import cross_val_score

- Run and Evaluate selected models

In [39]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [40]:
```
Scaledx=sc.fit_transform(x)
```

In [41]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
from sklearn.metrics import r2_score
```

In [42]:
```
best_rstate=0
accu=0
for i in range(30,200):
    x_train,x_test,y_train,y_test=train_test_split(Scaledx,y,test_size=25,random_state=i)
    mod=LinearRegression()
    mod.fit(x_train,y_train)
    y_pred=mod.predict(x_test)
    tempaccu=r2_score(y_test,y_pred)
    if tempaccu>accu:
        accu=tempaccu
        best_rstate=i
        print(f" Best accuracy {accu*100} found on random_state {best_rstate} ")
```

```
Best accuracy 90.26306299661766 found on random_state 30
Best accuracy 92.17865363654653 found on random_state 42
Best accuracy 93.93520745593848 found on random_state 58
Best accuracy 96.61603265011846 found on random_state 62
```

- Finding the best Model

- LinearRegression

In [48]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(Scaledx,y,test_size=25,random_state=62)
```

In [49]:
```
from sklearn.linear_model import LinearRegression
LR=LinearRegression()
LR.fit(x_train,y_train)
y_pred=LR.predict(x_test)
r2score=r2_score(y_test,y_pred)
cvscore=cross_val_score(LinearRegression(),x_train,y_train,cv=5).mean()
print(f"Accuracy={r2score*100},cross_val_score={cvscore*100} & difference ={(r2score*100)-(cvscore*100)}")
```

```
Accuracy=96.61603265011846,cross_val_score=77.28578134015767 & difference =19.330251309960786
```

- RandomForestRegressor

In [50]:
```
from sklearn.ensemble import RandomForestRegressor
RF=RandomForestRegressor()
RF.fit(x_train,y_train)
y_pred=RF.predict(x_test)
r2score=r2_score(y_test,y_pred)
cvscore=cross_val_score(RandomForestRegressor(),x_train,y_train,cv=5).mean()
print(f"Accuracy={r2score*100},cross_val_score={cvscore*100} & difference ={(r2score*100)-(cvscore*100)}")
```

```
Accuracy=96.82362480793189,cross_val_score=84.86245002319596 & difference =11.961174784735931
```

- AdaBoostRegressor

In [51]:
```
from sklearn.ensemble import AdaBoostRegressor
ADB=AdaBoostRegressor()
ADB.fit(x_train,y_train)
y_pred=ADB.predict(x_test)
r2score=r2_score(y_test,y_pred)
cvscore=cross_val_score(AdaBoostRegressor(),x_train,y_train,cv=5).mean()
print(f"Accuracy={r2score*100},cross_val_score={cvscore*100} & difference ={(r2score*100)-(cvscore*100)}")
```

```
Accuracy=89.36330594766324,cross_val_score=79.03632961082984 & difference =10.326976336833397
```

- SGDRegressor

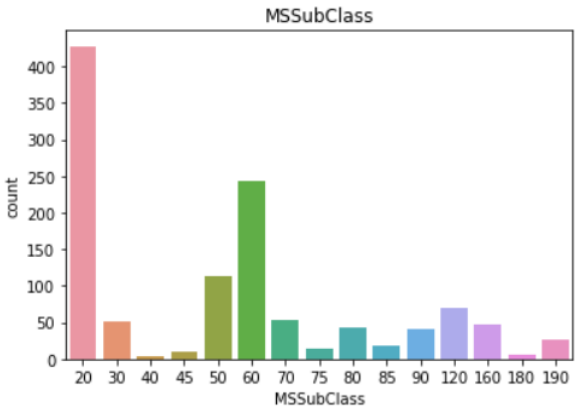
In [58]:
```
from sklearn.linear_model import SGDRegressor
SGD=SGDRegressor()
SGD.fit(x_train,y_train)
y_pred=SGD.predict(x_test)
r2score=r2_score(y_test,y_pred)
cvscore=cross_val_score(SGDRegressor(),x_train,y_train,cv=5).mean()
print(f"Accuracy={r2score*100},cross_val_score={cvscore*100} & difference ={(r2score*100)-(cvscore*100)}")
```

```
Accuracy=94.96194305162236,cross_val_score=77.3392509097628 & difference =17.62269214185956
```

RandomForestRegressor is performing best with accuracy 99.82 and cross validation score 84.86. i will choose RandomForestRegressor

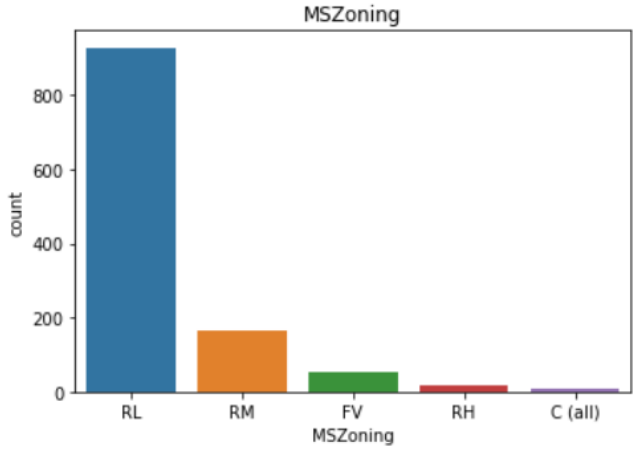
- Key Metrics for success in solving problem under consideration
- from sklearn.metrics import r2_score

- Visualizations



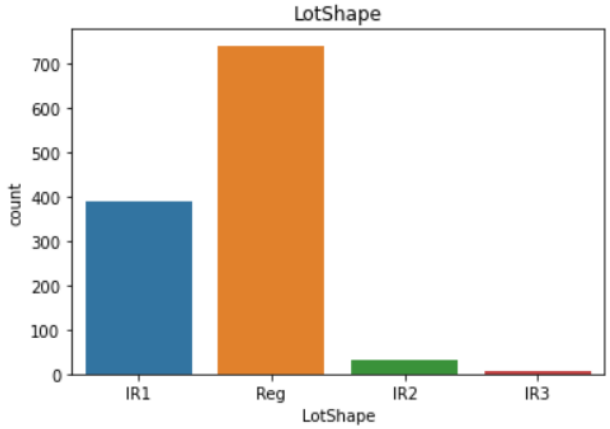
-highest type of dwelling involved in the sale is 1-STORY 1946 & NEWER ALL STYLES and the count is around 425

-lowest type of dwelling involved in the sale is 1-1/2 STORY - UNFINISHED ALL AGES and the count is around 5



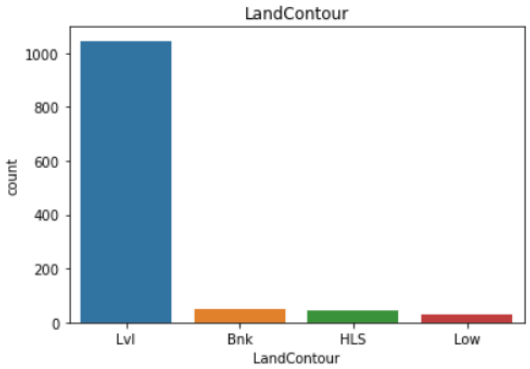
-The hieght zone sold is RL(Residential Low Density) with count around 750

-The lowest zone sold is RH(Residential High Density) with count around 20



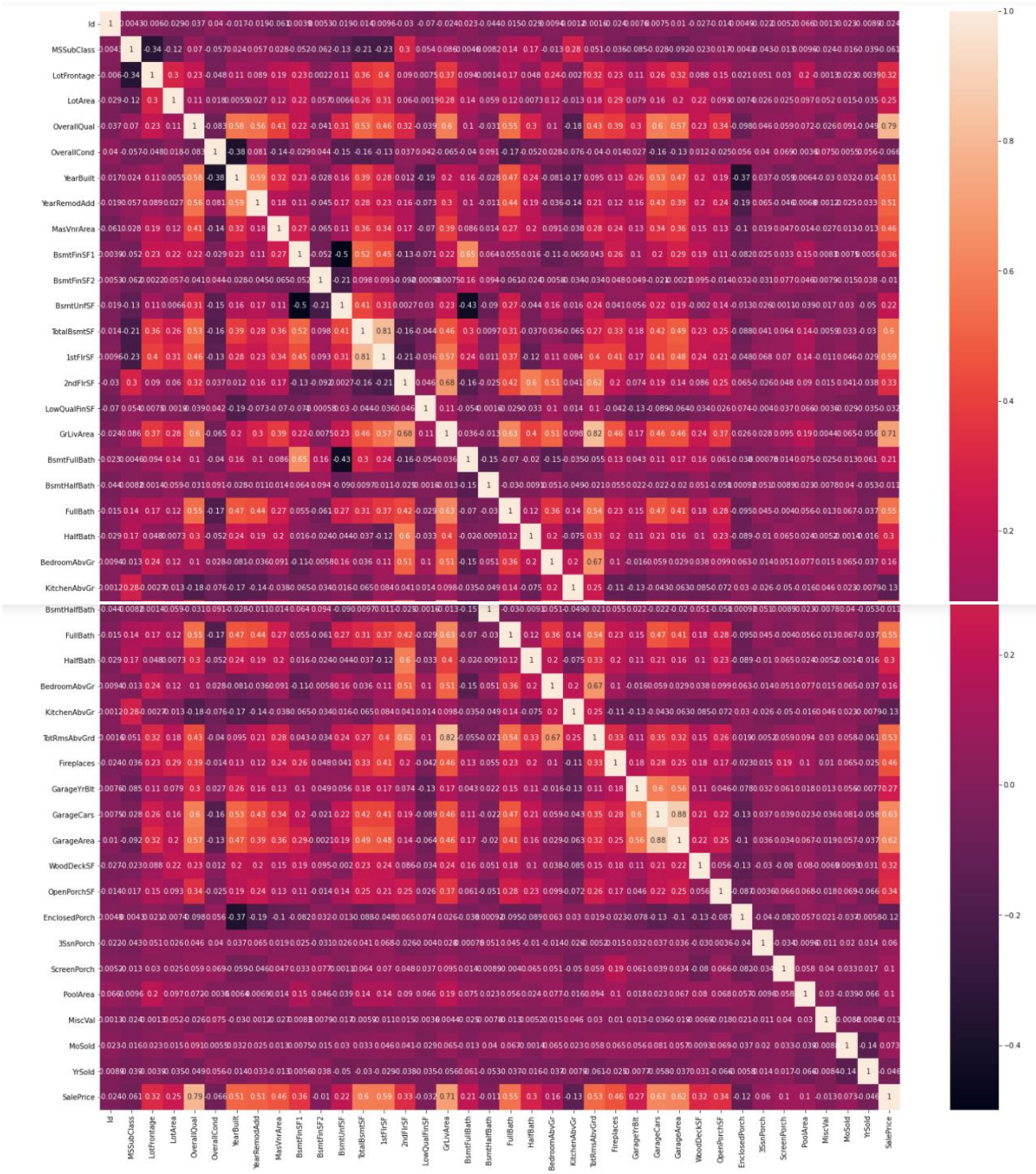
-Most of the property is of regular shape, with count around 725

-Most of the property is of Irregular shape, with count around 10



-Most of the property is Lvl(Near Flat/Level) with count around 1200

-Least amount of the property is Low(Depression) with count around 20



- # Interpretation of the Results

- From the heat map we can analyse that
- MSSubClass, LotFrontage, LotArea, OverallQual, YearBuilt, YearRemodAdd, MasVnrArea, BsmtFinSF1, BsmtUnfSF, TotalBsmtSF, 1stFlrSF, 2ndFlrSF, GrLivArea, BsmtFullBath, FullBath, HalfBath, BedroomAbvGr, TotRmsAbvGrd, Fireplaces, GarageYrBlt, GarageCars, GarageArea, WoodDeckSF, OpenPorchSF, 3SsnPorch, ScreenPorch, PoolArea and MoSold are positively correlated with Saleprice.

- OverallCond, BsmtFinSF2, LowQualFinSF, BsmtFullBath, KitchenAbvGr, EnclosedPorch, MiscVal and YrSold negatively correlated ieth saleprice.
- OverallQual has the highest correlation with sale price (79%)
- OverallCond has the least correlation with sale price

CONCLUSION

- Key Findings and Conclusions of the Study

- Learning Outcomes of the Study in respect of Data Science

  RandomForest Regressor works best for the given data with accuracy

- Limitations of this work and Scope for Future Work

  Project can be checked for more accuracy by reducing the skewness