FLIP ROBO

Housing Project: Price Prediction

Submitted by:

Akriti Kakkar

ACKNOWLEDGMENT

I have referred to data trained study material and python documentation for this model development.

INTRODUCTION

- Business Problem Framing
  - **Housing price prediction is a regression based problem statement to:**
    1. **Which variables are important to predict the price of variable?**
    2. **How do these variables describe the price of the house?**

    Data Science is an evolving and new topic industry-wide. Real Estate is one of the beneficiaries of this development. Solution to these problem statements and its dominance in real estate sector can reap in a lot of profits by segmenting customer needs in a few minutes and making personalized recommendations possible in the click of a button.

- Conceptual Background of the Domain Problem
  1. A US BASED HOUSING COMPANY, SURPRISING HOUSING, WANTS TO EXPAND ITS BUSINESS TO AUSTRALIA. IT HAS PROVIDED 80 FEATURES FOR 1460 HOUSES AND WANTS A PREDICTIVE MODEL TO PREDICT PRICE OF HOUSES. SO THAT THEY CAN BUY LOW PRICED PROPERTIES AND SELL HIGH PRICED PROPERTIES. IN ADDITION, IT WANTS A COMPLETE MARKETING MIX MODELLING TO UNDERSTAND HOW DIFFERENT FACTORS IMPACT PRICES.
  2. OBJECTIVE: To help the companies increase their overall revenue, profits, improving their

marketing strategies and focusing on changing trends in house sales and purchases.

- Review of Literature

MS Zoning Analysis	**Street Analysis**	**Alley Analysis**	**Lot Shape Analysis**
1. Residential Low Density + Residential Medium Density = 93.53%	1. 2 kinds of streets are observed in the entire dataset: 1) Paved and 2) Gravel	1. All the properties have access to alley.	1. Regular+ Slightly Irregular Property shape=95.8% and total sale price=203125930
2. Least sale price is covered by Commercial ,0.3% of total, followed by Residential High Density covering 1% sale price	2. 99.8% sale price is covered by houses with paved streets and 0.229% sale price is covered by houses with Gravel streets.	2. Of the two allet type: paved and gravel, paved is slightly more costly than gravel.	2. Sale price of regular shape compared to sale price of irregular for 57.9% is 23182.56 (122794111.55999999.122770929)
3. 5.14% sale price is covered by floating village residential.		3. Total sale price of paved alley type is 1.04% of total sale price of gravel alley type.	3. Moderately Irregular + Irregular houses = 4.17% and total sale price is 7482488+1356725=8839213
Land Contour Analysis	**Utilities Analysis**	**LotConfig Analysis**	**Land Slope Analysis**
1. 89.60% sale price is covered by houses that have levelled land contour with sale price = 189,824,495.	1. 100% houses provide all public utilities (Electricity, Gas, Water And Septic Tank)	1. Lot configuration inside: total sale price is 149,878,675 which is 70.71% of total sale price.	Gentle slope houses cover the most sale price.
2. Remaining Proportion of houses is occupied by: 1) Bnk=Banked - Quick and significant rise from street grade to building 2) HLS=Hillside - Significant slope from side to side 3)Low=Depression	2. Total sale price for 100% houses and houses with all public utilities is 211,965,143.	2. Lot configuration corner: total sale price is 40,498,935 which is 19.11% of total sale price.	Moderate slope houses occupy only a fraction of total sale price followed by severe slope which occupies less than 1% sale price.
3. HLS+Bnk+Low=4.25+3.45+2.75=10.45% with sale price= 9,000,691+7,308,157+5,831,800= 22,140,648		Lot configuration: Cul-de-sac + Frontage on 2 sides of property + Frontage on 3 sides of property = 10.18%	Trend: Lesser the slope, higher the price.
Neighborhood Analysis	**Condition 1 Analysis**	**Condition 2 Analysis**	**Bldg Type Analysis**
1. Highest sale price is found in North Ames neighborhood.	1. Normal condition houses were traded the most with maximum portion of sale price, 87.6%.	1. All the houses have more than 1 nearby spots.	1. Most popular type of dwelling is Single-family Detached. With 981 appearances, it occupies 85.9% sale price.
2. Lowest sale price is found in Bluestem neighborhood.	2. Houses near artery street and feedr street are next major occupants, in terms of total sale price.	2. 98.9% sale price is occupied by Normal proximity to all the spots.	2. Two-family Conversion; originally built as one-family dwelling occupies 7.9% sale price.
Trend: higher sale prices for developed areas and lower sale prices for villages.	3. Houses near railroad occupy minor portion in total sale price.	3. Trend: Normal proximoity houses are traded more and hence occupy major portion of the sale price.	3. 6.2% sale price is occupied cumulatively by Duplex, Townhouse End Unit and Townhouse Inside Unit

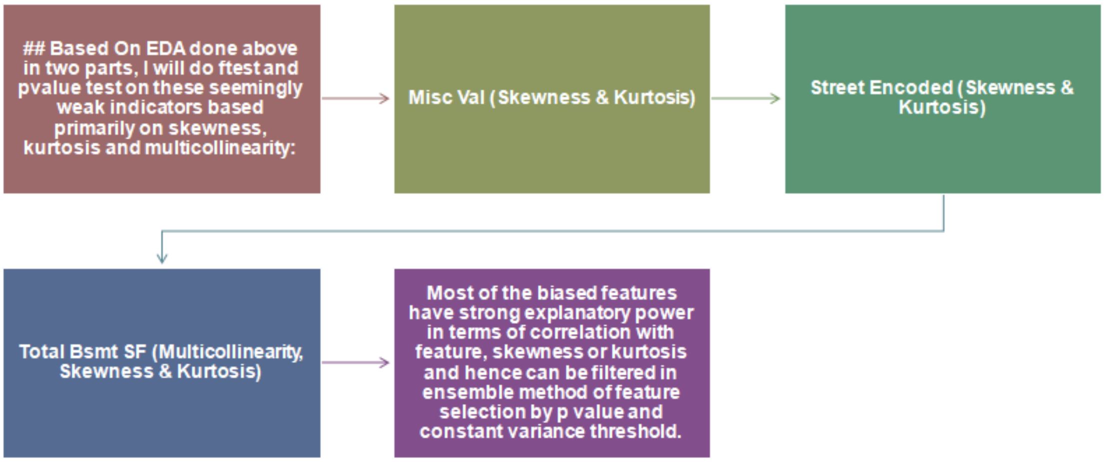
- We will understand the data by drawing anomalies and understanding their buying habits:
- 1. There are 43 categorical columns, containing housing data, example, 1 storey housing, 2 storey housing, etcetera. Normal sale condition is dominant, appearing, 80.91% times.
- 2. Most of the lot shape are regular.
- 3. Paved streets are the most popular.
- 4. Most of the zoning classification of sales are Residential Low Density.
- Q&A Answered
- How many houses are being surveyed: 1168
- How many sale condition are observed: 6
- What is the most popular fence quality: Minimum Privacy

EDA Steps Involve:

- HEAD VIEW OF DATA
- TAIL VIEW OF DATA
- SAMPLE VIEW OF DATA
- GROUPBY EXPLORATION
- DESCRIPTIVE STATISTICS
- SCATTER PLOTS
- CORRELATION ANALYSIS
- BOX PLOTS EXPLORATION

- DESCRIPTIVE STATISTICS
- DISTRIBUTION PLOTS

• Conclusion:



i. **Based on above analysis:**
ii. **1. There are many outliers in the data.**
iii. **2. Strong multicolliearity features are important for prediction because there f test and p value are acceptable. This means that the amount of multicollineaity is insignificant and removing the feature will impact the model much.**

iv. **3. Extereme leptokurtic and right skewed features are also relatively significant based on f test and p test.**
v. **Hence, as a solution, feature scaling will do a better job in explaining the dependent variable than removing whole columns.**
vi. **After passing through vif test to remove multicollinearity, only 33 features seem to be low bias with seemingly strong explanatory power.**
vii. **As part of data handling, I have closely analyzed features with high outliers (by analyzing box plots, dist plots, variable plot and scatter plots).**
viii. **I have removed features with multicollinearity by analyzing correlation, correlation heatmaps and variance inflation factor.**
ix. **I have done ANOVA testing, wherever, applicable to weigh importance against bias.Hence, the model can be expected to be low variance and low bias model**

- Motivation for the Problem Undertaken
  i. OBJECTIVE: To help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

1. **Data Extraction: Using read_ csv function of pandas library to read the data in tabulated format and analyze it.**
2. **Data Cleaning for missing values detection and its handiling.**
3. **Feature Engineering for encoding object format data and deriving more features.**
4. **EDA For data visualization and biasness detection:**

- HEAD VIEW OF DATA
- TAIL VIEW OF DATA
- SAMPLE VIEW OF DATA
- GROUPBY EXPLORATION
- DESCRIPTIVE STATISTICS
- SCATTER PLOTS
- CORRELATION ANALYSIS
- BOX PLOTS EXPLORATION
- DESCRIPTIVE STATISTICS
- DISTRIBUTION PLOTS

5. **VIF Test for multicollinearity reduction.**
6. **Power Transformation for standard scaling and outliers transformation.**
7. **Data PreProcessing for data transformation, scaling and vectorization.**
8. **Feature Selection (Ensemble Methods): ANOVA Test, p value, ftest, constant threshold filter to classify features based on relevance and biasness and select the most relevant features.**
9. **Model Development, Evaluation And Selection (Ensemble Methods and Grid Search CV) to do best hyper parameter tuning and develop low bias and low variance with right fit and minimal difference between test metrics and train metrics.**

- Data Sources and their formats

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

- Data Preprocessing Done
  - **Data Pre Processing**
    - **POWER TRANSFORM**
    - **MIN MAX SCALING**
    - **VECTORIZATION**
    - **TRAIN TEST SPLIT**

  The data was further used for feature selection.

  Assumption made:

    - **Acceptable Skewness Range Is +/-0.65**
    - **Acceptable VIF Score Is Than 6**
    - **Acceptable P Value Is Less Than 0.05**
    - **Variance Threshold Is 0.01**
- Data Inputs- Logic- Output Relationships
- **Data Inputs include, Index(['LandSlope_encoded', 'OpenPorchSF', 'Fireplaces', 'LotShape_encoded', 'HeatingQC_encoded', 'BsmtFinSF1_pct_change', 'MasVnrArea', 'WoodDeckSF', 'GarageFinish_encoded_pct_change', 'HalfBath'], dtype='object')**

Data Input Type: float; Min Max Scaling in the range of 0 to 1.

Impact On Output: Lot Shape_encoded; HeatingQC_encoded and GarageFinish_encoded_pct_change are negatively correlated with labeland others are positively correlated with label.

'LandSlope_encoded': 0.015484795080526005**,**

'OpenPorchSF': 0.33949955918549074**,**

'Fireplaces': 0.459610550802869**,**

'LotShape_encoded': -0.24817105697155653**,**
'HeatingQC_encoded': -0.4066035594011184**,**

'BsmtFinSF1_pct_change': 0.03927441750012172**,**

'MasVnrArea': 0.46120570017748197**,**
'WoodDeckSF': 0.31544416227339683**,**

'GarageFinish_encoded_pct_change': -0.23583840095816014**,**

'HalfBath': 0.29559237431380403

- State the set of assumptions (if any) related to the problem under consideration
  - Acceptable Skewness Range Is +/-0.65
  - Acceptable VIF Score Is Than 6
  - Acceptable P Value Is Less Than 0.05
  - Variance Threshold Is 0.01

- Hardware and Software Requirements and Tools Used

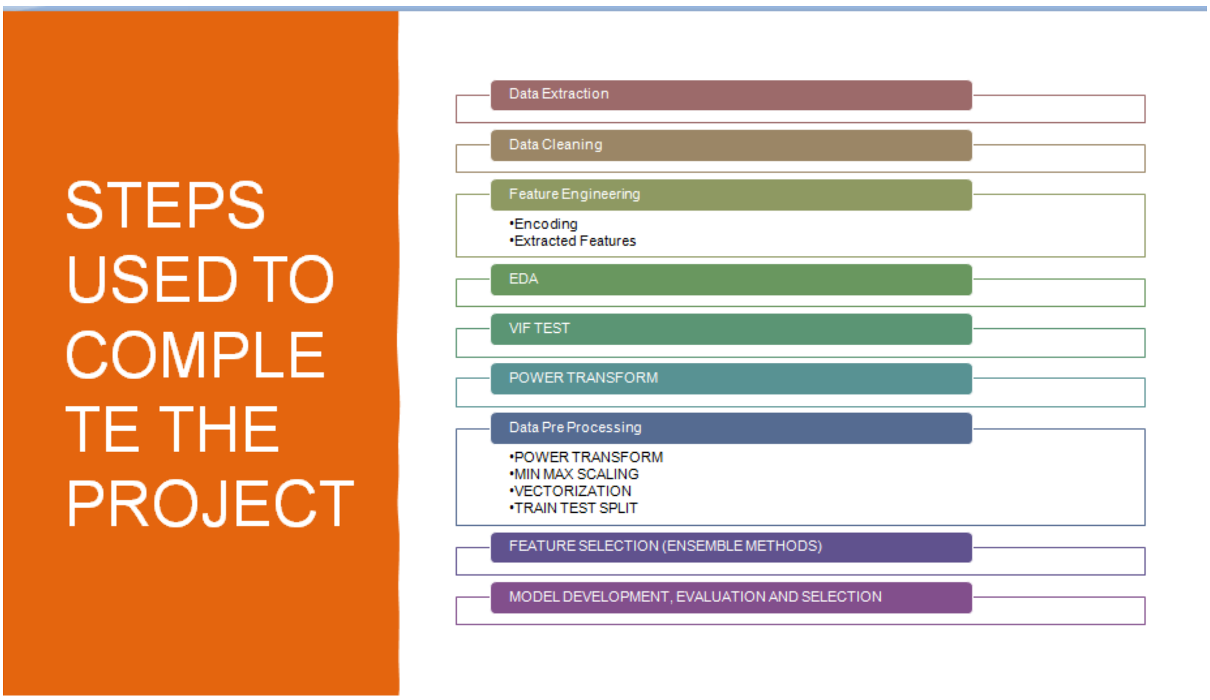
  Installation Of Anaconda Library.

  Required Installations:

  - **Pandas (Within environment)**
  - **Numpy (Within environment)**
  - **Seaborn (Within environment)**
  - **Matplotlib (Within environment)**
  - **Cufflinks**
  - **Plotly Express**
  - **Sklearn**

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)



-

- **Total Models = 7**
- **Selection Reasoning Of Models 1 to 7**
- **The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator.**
- **Two families of ensemble methods are usually distinguished:**
- **In averaging methods, the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced.**
- **Examples: Bagging methods, Forests of randomized trees, etcetera**
- **By contrast, in boosting methods, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensemble.**
- **Examples: AdaBoost, Gradient Tree Boosting,etcetera**
- **The use case assigned revolves around uneven label points hence there is high probality of not achieving a good fit. Hence, I have tried different ensembele techniques that can lower variance and bias and help achieve good scores. (Just as a**

  reminder, I have already applied variance threshold of 0.01 to ensure that risk of models is low).
  - **The theories in the above two cells explain why I have chosen Model 1, Model 2, Model 3, Model 4, Model 5, Model 6 and Model7**
  -

- Testing of Identified Approaches (Algorithms)

  Total Models = 7

- **Model 1: Random Forest Regressor With Grid Search CV Hyper Parameter Tuning**
- **Model 2: Random Forest Regressor With Default Hyper Parameter Tuning**
- **Model 3: Ada Boost Regressor And Random Forest Regressor With Grid Search CV Hyper Parameter Tuning and Ada Boost Boosting**
- **Model 4: Extra Trees Regressor With Grid Search CV Hyper Parameter Tuning**
- **Model 5: Linear Regression With Intuitional Hyper Parameter Tuning**
- **Model 6: Huber Regressor With Default Hyper Parameter Tuning**
- **Model 7: Ada Boost Regressor With Huber Regressor As Base Estimator**

- Run and Evaluate selected models
- **Model 1: Random Forest Regressor With Grid Search CV Hyper Parameter Tuning**
  - MSE Train: 472351089.3783724

- RMSE Train: 21733.6395796556
- MSE Test: 2941145189.730843
- RMSE Train: 54232.32605864184
- ## Random Forest Regressor With Grid Search CV Hyper Parameter Tuning

  In [109…

```
from sklearn.ensemble import RandomForestRegressor as el
from sklearn.model_selection import cross_val_score
from sklearn.datasets import load_boston
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostRegressor
from sklearn.metrics import mean_squared_error, make_scorer, r2_score
import matplotlib.pyplot as plt
import pickle
rfc2=el()
params = {
 'n_estimators': [50, 100],
 'criterion' : ['squared_error','absolute_error', 'poisson'],
 'bootstrap': [True],
 'oob_score':[True],
 'n_jobs':[-1],
 'warm_start':[True]
}
score2 = make_scorer(mean_squared_error)
gridsearch=GridSearchCV(rfc2, params, cv=5, return_train_score=True)
gridsearch.fit(x_train, y_train)
best_parameters=gridsearch.best_params_
best_score=gridsearch.best_score_
best_estim=gridsearch.best_estimator_
best_estim.fit(x_train,y_train)
y_pred2=best_estim.predict(x_train)
mse2 = mean_squared_error(y_train, y_pred2)
model3_train_score=best_estim.score(x_train, y_train)
model3_test_score=best_estim.score(x_test, y_test)
rmse2=np.sqrt(mse2)
ytst_pred2=best_estim.predict(x_test)
out_of_sample_pred=best_estim.predict(X_scaled_frame1)
mse_test2=mean_squared_error(y_test,ytst_pred2)
rmse_test2=np.sqrt(mse_test2)
model3_prediction_train=pd.DataFrame({'y_train':y_train, 'yhat':y_pred2})
model3_prediction_test=pd.DataFrame({'y_test':y_test, 'yhat':ytst_pred2})
model3_prediction_out_of_sample=pd.DataFrame({'Out Of Sample Prediction':out_of_sample_pred})
model3_prediction_out_of_sample
```

  Out[109…

	Out Of Sample Prediction
0	167711.90
1	132553.40
2	172036.47
3	196586.63

- **Model 2: Random Forest Regressor With Default Hyper Parameter Tuning**
  - mse_train: 445334449.60585994
    rmse_train 21102.948836735115
    mse_train: 2984787712.618337
    rmse_train 54633.21071123623

Random Forest Regressor With Default Hyper Parameter Tuning

In [172…

```
from sklearn.ensemble import RandomForestRegressor as el
from sklearn.model_selection import cross_val_score
rfc=el()
rfc.fit(x_train, y_train)
print(rfc.score(x_train, y_train))
print(rfc.score(x_test, y_test))
scores=cross_val_score(rfc, x, y, cv=5)
cv_scores=scores.mean()
print(scores.std())
print(cv_scores)
train_pred=rfc.predict(x_train)
test_pred=rfc.predict(x_test)
out_of_sample_pred=rfc.predict(X_scaled_frame1)
```

```
0.9337808846529457
0.41904855163269217
0.052705267719067674
0.5402278493502337
```

In [173…

```
import pickle
model3_save=pickle.dumps(rfc)
```

In [174…

```
ytr_pred=rfc.predict(x_train)
yts_pred=rfc.predict(x_test)
out_of_sample_pred=rfc.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', rfc.score(x_train, y_train))
print('Test score: ', rfc.score(x_test, y_test))
print('mse_train: ', mse)
print('rmse_train', np.sqrt(mse))
mse_test=mean_squared_error(yts_pred,y_test)
print('mse_train: ', mse_test)
print('rmse_train', np.sqrt(mse_test))
```

```
Train Score:  0.9337808846529457
Test score:  0.41904855163269217
mse_train:  445334449.60585994
rmse_train 21102.948836735115
mse_train:  2984787712.618337
rmse_train 54633.21071123623
```

In [175…

```
out_of_sample_pred=pd.DataFrame({'Out Of Sample Prediction':out_of_sample_pred})
out_of_sample_pred
```

- Out[175… **Out Of Sample Prediction**

- **Model 3: Ada Boost Regressor And Random Forest Regressor With Grid Search CV Hyper Parameter Tuning and Ada Boost Boosting**
    - mse_train 2261749348.614542
      rmse_train 47557.85264932114
      mse_test 2950461165.4071236
      rmse_test 54318.147661781724

```
In [117…
print(gridsearch.best_params_)
print(gridsearch.best_score_)
```

```
{'learning_rate': 0.1, 'loss': 'linear', 'n_estimators': 100}
0.5116466613115307
```

```
In [118…
best_estim=gridsearch.best_estimator_
print(best_estim)
```

```
AdaBoostRegressor(learning_rate=0.1, n_estimators=100)
```

```
In [119…
best_estim.fit(x_train,y_train)

ytr_pred=best_estim.predict(x_train)
ytst_pred=best_estim.predict(x_test)
out_of_sample_pred=best_estim.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', best_estim.score(x_train, y_train))
print('Test score: ', best_estim.score(x_test, y_test))
```

```
Train Score:  0.6636886251791545
Test score:  0.4257297830098843
```

```
In [120…
ytst_pred=best_estim.predict(x_test)
print('mse_train', mse)
print('rmse_train', np.sqrt(mse))
mse_test = mean_squared_error(ytst_pred,y_test)
print('mse_test', mse_test)
print('rmse_test', np.sqrt(mse_test))
```

```
mse_train 2261749348.614542
rmse_train 47557.85264932114
mse_test 2950461165.4071236
rmse_test 54318.147661781724
```

```
In [121…
out_of_sample_pred=pd.DataFrame({'Out Of Sample Prediction': out_of_sample_pred})
out_of_sample_pred
```

Out[121…

	Out Of Sample Prediction
0	176530.899471
1	169006.301887
2	174773.425373
3	175076.518072
4	181736.116022

- **Model 4: Extra Trees Regressor With Grid Search CV Hyper Parameter Tuning**

  - mse_train: 1148887032.2666373
    rmse_train 33895.2361293831
    mse_train: 2682387627.887125
    rmse_train 51791.771816449036

Out[167…
```
GridSearchCV(cv=5, estimator=ExtraTreesRegressor(),
             param_grid={'bootstrap': [True],
                         'criterion': ['squared_error', 'absolute_error'],
                         'max_samples': [0.01, 0.05, 0.1, 0.5],
                         'n_estimators': [50, 100], 'n_jobs': [-1],
                         'oob_score': [True],
                         'random_state': [0, 10, 100, 1000, 10000, 100000,
                                          100000],
                         'warm_start': [True]},
             return_train_score=True)
```

In [168…
```
print(gridsearch.best_params_)
print(gridsearch.best_score_)
```

```
{'bootstrap': True, 'criterion': 'squared_error', 'max_samples': 0.5, 'n_estimators': 50, 'n_jobs': -1, 'oob_score': True, 'random_state': 10, 'war
m_start': True}
0.5871580679701349
```

In [169…
```
best_estim=gridsearch.best_estimator_
print(best_estim)
```

```
ExtraTreesRegressor(bootstrap=True, max_samples=0.5, n_estimators=50, n_jobs=-1,
                    oob_score=True, random_state=10, warm_start=True)
```

In [170…
```
best_estim.fit(x_train,y_train)

ytr_pred=best_estim.predict(x_train)
yts_pred=best_estim.predict(x_test)
out_of_sample_pred=best_estim.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', best_estim.score(x_train, y_train))
print('Test score: ', best_estim.score(x_test, y_test))
print('mse_train: ', mse)
print('rmse_train', np.sqrt(mse))
mse_test=mean_squared_error(yts_pred,y_test)
print('mse_train: ', mse_test)
print('rmse_train', np.sqrt(mse_test))
```

```
Train Score:  0.8291659605994284
Test score:  0.47790693089641645
mse_train:  1148887032.2666373
rmse_train 33895.2361293831
mse_train:  2682387627.887125
rmse_train 51791.771816449036
```

In [171…
```
out_of_sample_pred=pd.DataFrame({'Out Of Sample Prediction':out_of_sample_pred})
out_of_sample_pred
```

Out[171…

	Out Of Sample Prediction
0	188311.640000
1	130391.160000
2	156467.380000
3	173817.000000

- **Model 5: Linear Regression With Intuitional Hyper Parameter Tuning**
  - mse_train: 3643318304.1421824
    rmse_train 60359.906429203336
    mse_train: 3238713380.888534
    rmse_train 56909.694963938564

Linear Regression With Intuitional Hyper Parameter Tuning

In [128…
```
from sklearn.linear_model import LinearRegression
lm=LinearRegression(fit_intercept=True,n_jobs=-1, positive=True)
lm.fit(x_train, y_train)
```
Out[128…
```
LinearRegression(n_jobs=-1, positive=True)
```
In [129…
```
lm.score(x_train, y_train)
```
Out[129…
```
0.4582558900584751
```
In [130…
```
lm.score(x_test, y_test)
```
Out[130…
```
0.3696251088412448
```
In [131…
```
ytr_pred=lm.predict(x_train)
yts_pred=lm.predict(x_test)
out_of_sample_pred=lm.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', lm.score(x_train, y_train))
print('Test score: ', lm.score(x_test, y_test))
print('mse_train: ', mse)
print('rmse_train', np.sqrt(mse))
mse_test=mean_squared_error(yts_pred,y_test)
print('mse_train: ', mse_test)
print('rmse_train', np.sqrt(mse_test))
```
```
Train Score:  0.4582558900584751
Test score:  0.3696251088412448
mse_train:  3643318304.1421824
rmse_train 60359.906429203336
mse_train:  3238713380.888534
rmse_train 56909.694963938564
```
In [132…
```
out_of_sample_pred=pd.DataFrame({'Out Of Sample Prediction':out_of_sample_pred})
out_of_sample_pred
```
Out[132…

	Out Of Sample Prediction
0	155829.731072
1	134886.251425
2	167375.345526
3	143828.137000

- **Model 6: Huber Regressor With Default Hyper Parameter Tuning**
  - mse_train: 3449320674.0164285
    rmse_train 58730.91753085787
    mse_train: 2763713597.6159506
    rmse_train 52571.03382677528

Model 6: Huber Regressor With Default Hyper Parameter Tuning

In [163…
```
from sklearn.linear_model import HuberRegressor
lm=HuberRegressor()
lm.fit(x_train, y_train)
```
Out[163…
```
HuberRegressor()
```
In [164…
```
ypred=lm.predict(x_train)
ytst_pred=lm.predict(x_test)
pred=lm.predict(X_scaled_frame1)
pred
```
Out[164…
```
array([175070.47092235, 139649.30596878, 171871.83149429, ...,
       113415.09496655, 113415.09496655, 113415.09496655])
```
In [165…
```
ytr_pred=lm.predict(x_train)
yts_pred=lm.predict(x_test)
out_of_sample_pred=lm.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', lm.score(x_train, y_train))
print('Test score: ', lm.score(x_test, y_test))
print('mse_train: ', mse)
print('rmse_train', np.sqrt(mse))
mse_test=mean_squared_error(yts_pred,y_test)
print('mse_train: ', mse_test)
print('rmse_train', np.sqrt(mse_test))
```
```
Train Score:  0.4871024153109501
Test score:  0.46207785209657404
mse_train:  3449320674.0164285
rmse_train 58730.91753085787
mse_train:  2763713597.6159506
rmse_train 52571.03382677528
```
In [176…
```
model3_save=pickle.dumps(lm) #saving the best model
```

- **Model 7: Ada Boost Regressor With Huber Regressor As Base Estimator**
  - mse_train 2527254513.044728
    rmse_train 50271.806343563265
    mse_test 3015933702.827419
    rmse_test 54917.51726751145

Model 7: Ada Boost Regressor With Huber Regressor As Base Estimator

In [155…

```
abreg = AdaBoostRegressor()
params = {
 'n_estimators': [50, 100],
 'learning_rate' : [0.01, 0.05, 0.5],
 'loss' : ['linear', 'square', 'exponential']
 }
score = make_scorer(mean_squared_error)
print(score)
gridsearch=GridSearchCV(abreg, params, cv=5, return_train_score=True)
gridsearch.fit(x_train, y_train)
GridSearchCV(cv=5, error_score='raise', estimator=AdaBoostRegressor(base_estimator=lm, learning_rate=1.0,loss='linear', n_estimators=50, random_st
```

```
make_scorer(mean_squared_error)
```

Out[155…

```
GridSearchCV(cv=5, error_score='raise',
             estimator=AdaBoostRegressor(base_estimator=HuberRegressor()),
             n_jobs=1,
             param_grid={'learning_rate': [0.01, 0.05, 0.1, 0.5],
                         'loss': ['linear', 'square', 'exponential'],
                         'n_estimators': [50, 100]},
             return_train_score=True)
```

In [156…

```
best_estim=gridsearch.best_estimator_
print(best_estim)
```

```
AdaBoostRegressor(learning_rate=0.05, loss='exponential', n_estimators=100)
```

In [157…

```
best_estim.fit(x_train,y_train)

ytr_pred=best_estim.predict(x_train)
ytst_pred=best_estim.predict(x_test)
out_of_sample_pred=best_estim.predict(X_scaled_frame1)
mse = mean_squared_error(ytr_pred,y_train)
print('Train Score: ', best_estim.score(x_train, y_train))
print('Test score: ', best_estim.score(x_test, y_test))
```

```
Train Score:  0.6242092695528348
Test score:  0.4129863757378017
```

In [158…

```
ytst_pred=best_estim.predict(x_test)
print('mse_train', mse)
print('rmse_train', np.sqrt(mse))
mse_test = mean_squared_error(ytst_pred,y_test)
print('mse_test', mse_test)
print('rmse_test', np.sqrt(mse_test))
```

```
mse_train 2527254513.044728
rmse_train 50271.806343563265
mse_test 3015933702.827419
rmse_test 54917.51726751145
```

-

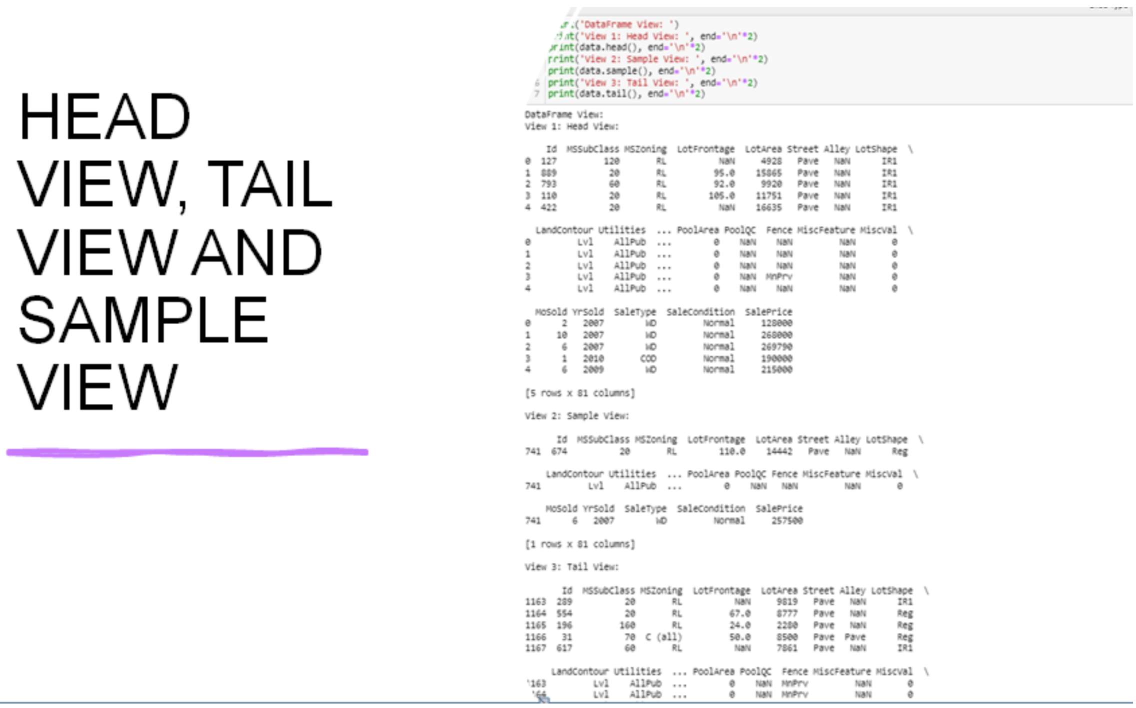
- Key Metrics for success in solving problem under consideration

1. Power Transform: To remove outliers from extremely spread out data.
2. VIF Scores: To reduce multicollinearity from a highly biased dataset.
3. Ensemble Methods: To remove over fitting in a complex dataset and finding maximum explanatory power .

- Visualizations

EDA Steps Involve:

- HEAD VIEW OF DATA
- TAIL VIEW OF DATA
- SAMPLE VIEW OF DATA
- GROUPBY EXPLORATION
- DESCRIPTIVE STATISTICS
- SCATTER PLOTS
- CORRELATION ANALYSIS
- BOX PLOTS EXPLORATION
- DESCRIPTIVE STATISTICS
- DISTRIBUTION PLOTS




```
print('DataFrame View: ')
print('View 1: Head View: ', end='\n'*2)
print(data.head(), end='\n'*2)
print('View 2: Sample View: ', end='\n'*2)
print(data.sample(), end='\n'*2)
print('View 3: Tail View: ', end='\n'*2)
print(data.tail(), end='\n'*2)

DataFrame View:
View 1: Head View:

    Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0  127         120       RL          NaN     4928   Pave   NaN      IR1
1  889          20       RL         95.0    15865   Pave   NaN      IR1
2  793          60       RL         92.0     9920   Pave   NaN      IR1
3  110          20       RL        105.0    11751   Pave   NaN      IR1
4  422          20       RL          NaN    16635   Pave   NaN      IR1

  LandContour Utilities  ... PoolArea PoolQC  Fence MiscFeature MiscVal  \
0         Lvl    AllPub  ...        0    NaN    NaN         NaN       0
1         Lvl    AllPub  ...        0    NaN    NaN         NaN       0
2         Lvl    AllPub  ...        0    NaN    NaN         NaN       0
3         Lvl    AllPub  ...        0    NaN  MnPrv         NaN       0
4         Lvl    AllPub  ...        0    NaN    NaN         NaN       0

  MoSold YrSold  SaleType  SaleCondition  SalePrice
0      2   2007        WD         Normal     128000
1     10   2007        WD         Normal     268000
2      6   2007        WD         Normal     269790
3      1   2010       COD         Normal     190000
4      6   2009        WD         Normal     215000

[5 rows x 81 columns]

View 2: Sample View:

      Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
741  674          20       RL        110.0    14442   Pave   NaN      Reg

    LandContour Utilities  ... PoolArea PoolQC Fence MiscFeature MiscVal  \
741         Lvl    AllPub  ...        0    NaN   NaN         NaN       0

    MoSold YrSold  SaleType  SaleCondition  SalePrice
741      6   2007        WD         Normal     257500

[1 rows x 81 columns]

View 3: Tail View:

       Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
1163  289          20       RL          NaN     9819   Pave   NaN      IR1
1164  554          20       RL         67.0     8777   Pave   NaN      Reg
1165  196         160       RL         24.0     2280   Pave   NaN      Reg
1166   31          70  C (all)         50.0     8500   Pave  Pave      Reg
1167  617          60       RL          NaN     7861   Pave   NaN      IR1

     LandContour Utilities  ... PoolArea PoolQC  Fence MiscFeature MiscVal  \
1163         Lvl    AllPub  ...        0    NaN  MnPrv         NaN       0
1164         Lvl    AllPub  ...        0    NaN  MnPrv         NaN       0
```

- **# View 1: Analysis**

- **1. Data alongside represents first five houses of 1168 houses in train dataset.**
- **2. Data seems complex and incomplete due to 81 columns and many missing values.**
- **3. Upper limit of sale price of first five houses is 268000 and lower limit is 128000**
- **4. All the five houses have been sold in normal condition with mode lying in the month of junefor the years 2007 and 2009 respectively.**
- **5. Only one house provides fencing facility and all the five houses provide pub utility**
- **6. Street type is pavement for all the five houses**
- **7. Saletype for 1 house is court office deed and the other four houses is warranty deed conventional.**
- **8. There are three varities of housing seen in mssubclass with 3 dwellings identified as 20 : 1 story 1946 and newer all styles , 1 dwelling identified as 60 : 2 story 1946 and newer and 1 dwelling identified as 120 : 1 story planned unit developement 1946 & newer.**
- **9. The general shape of all the five properties is IR1 slightly irregular and the flatness of all the property ois LV1 which is near flat/level.**
- **10. The identified general zoning classification is RL : Residential Low Density for all the five properties.**

- # View 2: Analysis
- 1. It shows a random house at index 741.It is 1 storey 1846 and newer all styles, regular shape house that provides all public utilities and is sold by warranty deed for 257500.
- # View 3: Analysis
- 1. It represents last 5 rows. In which all are warranty deeds, sold in normal condition.

MS Zoning Analysis	Street Analysis	Alley Analysis	Lot Shape Analysis
1. Residential Low Density + Residential Medium Density = 93.53%	1. 2 kinds of streets are observed in the entire dataset: 1) Paved and 2) Gravel	1. All the properties have access to alley.	1. Regular+Slightly Irregular Property shape=95.8% and total sale price=203125930
2. Least sale price is covered by Commercial ,0.3% of total, followed by Residential High Density covering 1% sale price	2. 99.8% sale price is covered by houses with paved streets and 0.229% sale price is covered by houses with Gravel streets.	2. Of the two allet type: paved and gravel, paved is slightly more costly than gravel.	2. Sale price of regular shape compared to sale price of irregular for 57.9% is 23182.56 (122794111.5599999-122770929)
3. 5.14% sale price is covered by floating village residential.		3. Total sale price of paved alley type is 1.04% of total sale price of gravel alley type.	3. Moderately Irregular + Irregular houses = 4.17% and total sale price is 7482488+1356725=8839213
Land Contour Analysis	Utilities Analysis	LotConfig Analysis	Land Slope Analysis
1. 89.60% sale price is covered by houses that have levelled land contour with sale price = 189,824,495.	1. 100% houses provide all public utilities (Electricity, Gas, Water And Septic Tank)	1. Lot configuration inside: total sale price is 149,878,675 which is 70.71% of total sale price.	Gentle slope houses cover the most sale price.
2. Remaining Proportion of houses is occupied by: 1) Bnk=Banked - Quick and significant rise from street grade to building 2) HLS=Hillside - Significant slope from side to side 3)Low=Depression	2. Total sale price for 100% houses and houses with all public utilities is 211,965,143.	2. Lot configuration corner: total sale price is 40,498,935 which is 19.11% of total sale price.	Moderate slope houses occupy only a fraction of total sale price followed by severe slope which occupies less than 1% sale price.
3. HLS+Bnk+Low=4.25+3.45+2.75=10.45% with sale price= 9,000,691+7,308,157+5,831,800= 22,140,648		Lot configuration: Cul-de-sac + Frontage on 2 sides of property + Frontage on 3 sides of property = 10.18%	Trend: Lesser the slope, higher the price.
Neighborhood Analysis	Condition 1 Analysis	Condition 2 Analysis	Bldg Type Analysis
1. Highest sale price is found in North Ames neighborhood.	1. Normal condition houses were traded the most with maximum portion of sale price, 87.6%.	1. All the houses have more than 1 nearby spots.	1. Most popular type of dwelling is Single-family Detached. With 981 appearances, it occupies 85.9% sale price.
2. Lowest sale price is found in Bluestem neighborhood.	2. Houses near artery street and feedr street are next major occupants, in terms of total sale price.	2. 98.9% sale price is occupied by Normal proximity to all the spots.	2. Two-family Conversion; originally built as one-family dwelling occupies 7.9% sale price.
Trend: higher sale prices for developed areas and lower sale prices for villages.	3. Houses near railroad occupy minor portion in total sale price.	3. Trend: Normal proximoity houses are traded more and hence occupy major portion of the sale price.	3. 6.2% sale price is occupied cumulatively by Duplex, Townhouse End Unit and Townhouse Inside Unit

In [15]: Slide Typ

```
object_description=data.select_dtypes(object).describe()
object_description=object_description.rename(index={'count':'count of values', 'unique':'unique values', 'top
dict1=object_description.iloc[0,:]-object_description.iloc[3,:]
object_description1=pd.concat([object_description, pd.DataFrame(dict1).T])
object_description1=object_description1.rename(index={0:'count of remaining values'})
object_description1
```

Out[15]:

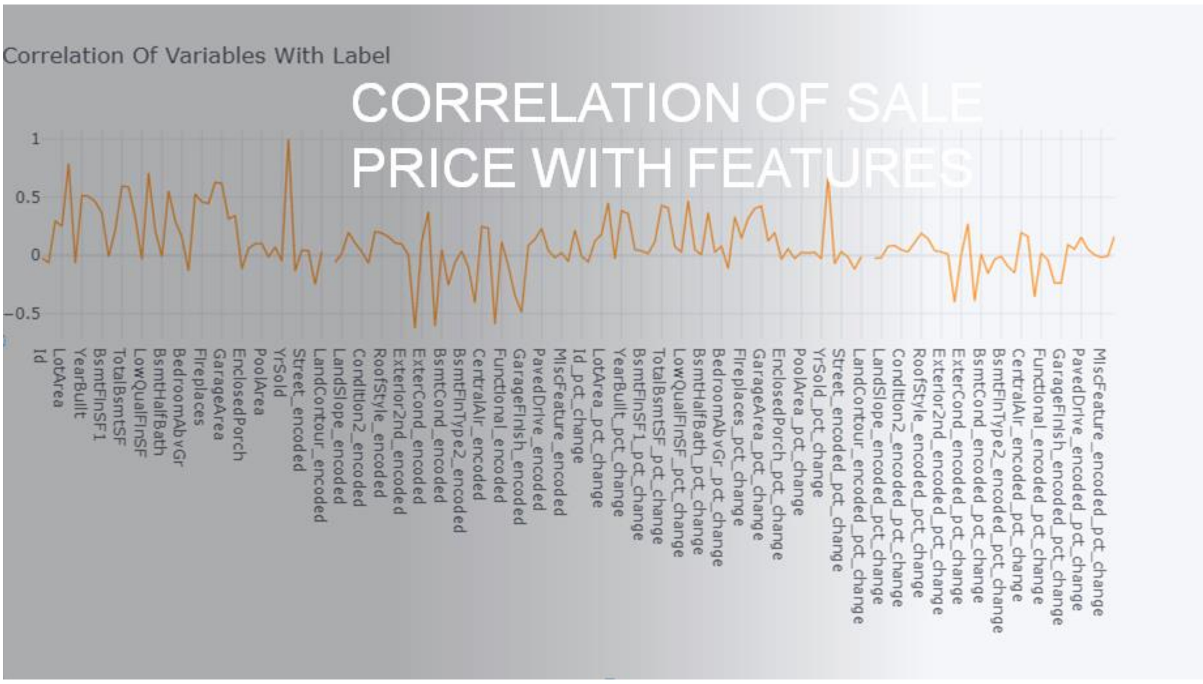
	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	...	GarageType	Gar
count of values	1168	1168	1168	1168	1168	1168	1168	1168	1168	1168	...	1168	
unique values	5	2	2	4	4	1	5	3	25	9	...	6	
most frequent value	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	Gtl	NAmes	Norm	...	Attchd	
frequency of most frequent value	928	1164	584	740	1046	1168	842	1105	182	1005	...	727	
count of remaining values	240	4	584	428	122	0	326	63	986	163	...	441	

5 rows × 43 columns

Descriptive statistics

- **We will understand the data by drawing anomalies and understanding their buying habits:**
- **1. There are 43 categorical columns, containing housing data, example, 1 storey housing, 2 storey housing, etcetera. Normal sale condition is dominant, appearing, 80.91% times.**
- **2. Most of the lot shape are regular.**
- **3. Paved streets are the most popular.**
- **4. Most of the zoning classification of sales are Residential Low Density.**
- **Q&A Answered**
- **How many houses are being surveyed: 1168**
- **How many sale condition are observed: 6**
- **What is the most popular fence quality: Minimum Privacy**
- **# Anomalies Detected**

- ## Highest sale price is 755000 and is obtained for a property built in 1994 in 2 storey 1946 and newer.
- ## Maximum lot area is 164660 and lot area of maximum sale price property is 21535.
- ## Overall quality of this house Very Excellent and overall Condition is above average.
- ## Most significant sale price pct change is 6.5.
- # Q&A Answered
- ## What is the range of Open Porch SF: 0 to 600.
- ## What is the range of Garage Cars: 0 to 4.
- ## What is the range of Total Price Above Ground: 0 to 15.
- ## What is the range of Half Bath: 0 to 2.



- # The above iplot represents correlation of features with label. Correlation of features with label is of high relevance. The stronger the relationship of label with axis, the more accurate the prediction. In the above line graph:

- ## 1. Ignoring sale price pct change, highest correlation with label is observed to be 78.92%, that is shared with Overall Quality. Minimum correlation is -0.4%, shared with BsmtHalfBath_pct_change.
- ## 2. Weak Positive To Strong Positive Relationship Is Found With 110 Features, including these:
- BsmtHalfBath_pct_change 0.004155282515700739
- ExterCond_encoded_pct_change 0.004999542764231268
- Fence_encoded_pct_change 0.006188192429984 86
- MasVnrType_encoded 0.006763415444002462
- BsmtCond_encoded_pct_change 0.011509937574556038
- MasVnrType_encoded_pct_change 0.011631168977197338
- LandSlope_encoded 0.015484795080526005
- BsmtFinSF2_pct_change 0.01694247841896264
- ## 3. Weak Negative to Strong Negative Relationship Is Found With Following Features:
- ExterQual_encoded -0.624820046916615
- BsmtQual_encoded -0.6074933757146362

- KitchenQual_encoded -0.5924675972943289
- GarageFinish_encoded -0.48745288300973294
- HeatingQC_encoded -0.4066035594011184
- ExterQual_encoded_pct_change -0.4037123724436022
- ## 4. Correlation Of Label with itself is of no relevance

- 1. Label has weak positive and strong positive relationship with these features:
- ['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', 'MoSold', 'SalePrice',

'Street_encoded', 'Alley_encoded',
'LandContour_encoded', 'LandSlope_encoded',
'Neighborhood_encoded', 'Condition1_encoded',
'Condition2_encoded', 'HouseStyle_encoded',
'RoofStyle_encoded', 'RoofMatl_encoded',
'Exterior1st_encoded', 'Exterior2nd_encoded',
'MasVnrType_encoded', 'ExterCond_encoded',
'Foundation_encoded', 'BsmtCond_encoded',
'BsmtFinType2_encoded', 'CentralAir_encoded',
'Electrical_encoded', 'Functional_encoded',
'GarageQual_encoded', 'GarageCond_encoded',
'PavedDrive_encoded', 'PoolQC_encoded',
'MiscFeature_encoded', 'SaleCondition_encoded',
'LotFrontage_pct_change', 'LotArea_pct_change',
'OverallQual_pct_change', 'YearBuilt_pct_change',
'YearRemodAdd_pct_change',
'MasVnrArea_pct_change', 'BsmtFinSF1_pct_change',
'BsmtFinSF2_pct_change', 'BsmtUnfSF_pct_change',
'TotalBsmtSF_pct_change', '1stFlrSF_pct_change',
'2ndFlrSF_pct_change', 'LowQualFinSF_pct_change',
'GrLivArea_pct_change', 'BsmtFullBath_pct_change',
'BsmtHalfBath_pct_change', 'FullBath_pct_change',
'HalfBath_pct_change', 'BedroomAbvGr_pct_change',
'TotRmsAbvGrd_pct_change', 'Fireplaces_pct_change',
'GarageYrBlt_pct_change', 'GarageCars_pct_change',
'GarageArea_pct_change', 'WoodDeckSF_pct_change',
'OpenPorchSF_pct_change', '3SsnPorch_pct_change',
'PoolArea_pct_change', 'MiscVal_pct_change',
'MoSold_pct_change', 'SalePrice_pct_change',

'Street_encoded_pct_change', 'Neighborhood_encoded_pct_change', 'Condition1_encoded_pct_change', 'Condition2_encoded_pct_change', 'BldgType_encoded_pct_change', 'HouseStyle_encoded_pct_change', 'RoofStyle_encoded_pct_change', 'RoofMatl_encoded_pct_change', 'Exterior1st_encoded_pct_change', 'Exterior2nd_encoded_pct_change', 'MasVnrType_encoded_pct_change', 'ExterCond_encoded_pct_change', 'Foundation_encoded_pct_change', 'BsmtCond_encoded_pct_change', 'CentralAir_encoded_pct_change', 'Electrical_encoded_pct_change', 'Functional_encoded_pct_change', 'GarageQual_encoded_pct_change', 'GarageCond_encoded_pct_change', 'PavedDrive_encoded_pct_change', 'PoolQC_encoded_pct_change', 'Fence_encoded_pct_change', 'SaleCondition_encoded_pct_change']

- **2. Label has weak negative to strong negative relationship with these features in this heatmap:**
- **['Id', 'MSSubClass', 'OverallCond', 'BsmtFinSF2', 'LowQualFinSF', 'BsmtHalfBath', 'KitchenAbvGr',**

'EnclosedPorch', 'MiscVal', 'YrSold',
'MSZoning_encoded', 'LotShape_encoded',
'LotConfig_encoded', 'BldgType_encoded',
'ExterQual_encoded', 'BsmtQual_encoded',
'BsmtExposure_encoded', 'BsmtFinType1_encoded',
'Heating_encoded', 'HeatingQC_encoded',
'KitchenQual_encoded', 'FireplaceQu_encoded',
'GarageType_encoded', 'GarageFinish_encoded',
'Fence_encoded', 'SaleType_encoded', 'Id_pct_change',
'MSSubClass_pct_change', 'OverallCond_pct_change',
'KitchenAbvGr_pct_change',
'EnclosedPorch_pct_change',
'ScreenPorch_pct_change', 'YrSold_pct_change',
'MSZoning_encoded_pct_change',
'Alley_encoded_pct_change',
'LotShape_encoded_pct_change',
'LandContour_encoded_pct_change',
'LotConfig_encoded_pct_change',
'LandSlope_encoded_pct_change',
'ExterQual_encoded_pct_change',
'BsmtQual_encoded_pct_change',
'BsmtExposure_encoded_pct_change',
'BsmtFinType1_encoded_pct_change',
'BsmtFinType2_encoded_pct_change',
'Heating_encoded_pct_change',
'HeatingQC_encoded_pct_change',
'KitchenQual_encoded_pct_change',
'FireplaceQu_encoded_pct_change',
'GarageType_encoded_pct_change',

**'GarageFinish_encoded_pct_change',
'MiscFeature_encoded_pct_change',
'SaleType_encoded_pct_change']**

- **3. Strong Multicollinearity is detecte in most of the features, including features like:**
- Overall Quality

  Half Bath, and many more.

  We will explore this deeply in vif section of data analysis.
- **4. Multicollinearity seems to be moderate among all other data points.**
- **Q&A Answered**
- 1. Most useful feature is: Overall Quallity.

  2. Features that can cause bias are: Overall Quality, Half Bath and many more.

  3. Explanatory Power of all variable: Weak to strong (due to weak, moderate and strong correlation with label).
- **Overall strong correlation among dataset. There seems some multicollinearity due to presence of Overall Quality, Half Bath, etcetera. We will do further eda before arriving at a conclusion to delete these columns.**

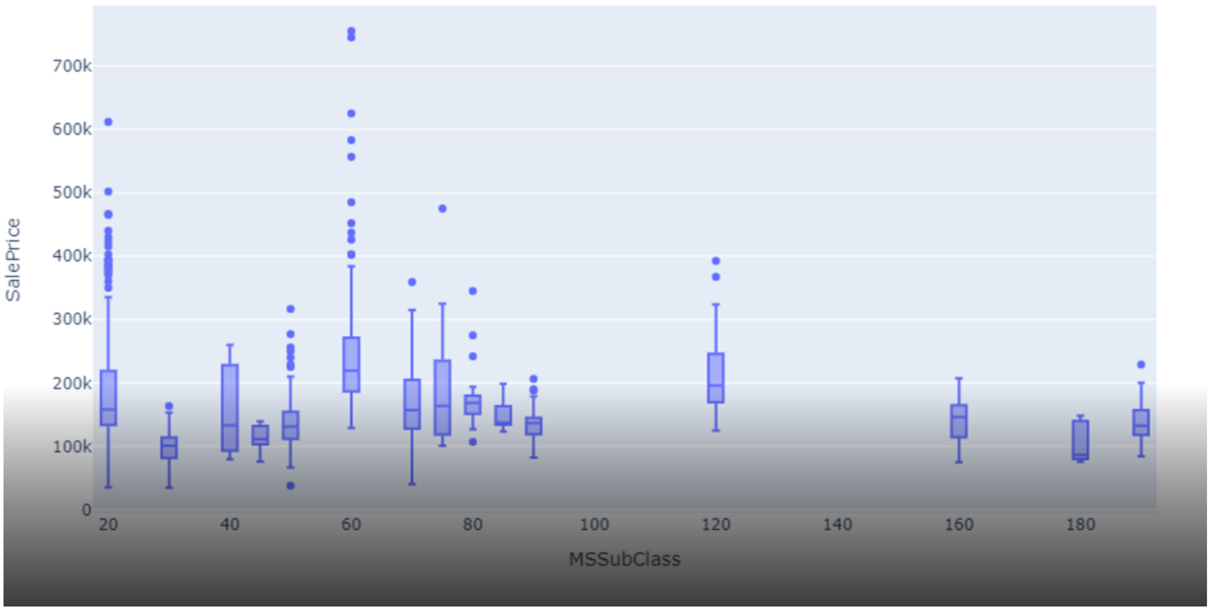
- **The most useful feature is Overall Quality, apart from this, there seems modest correlation of features with label.**
- **As a conclusion, we will do further cause and effect analysis based on skewness and distribution plots to conclude if we want to remove multicollinarity pairs:**
- i. Half Bath

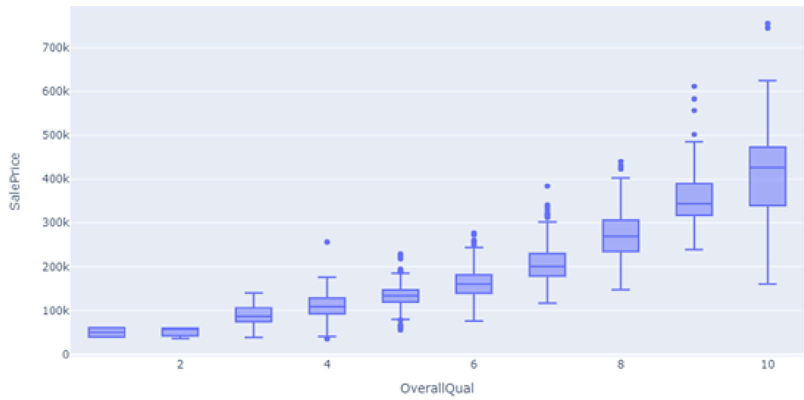
  ii. MasVnrArea

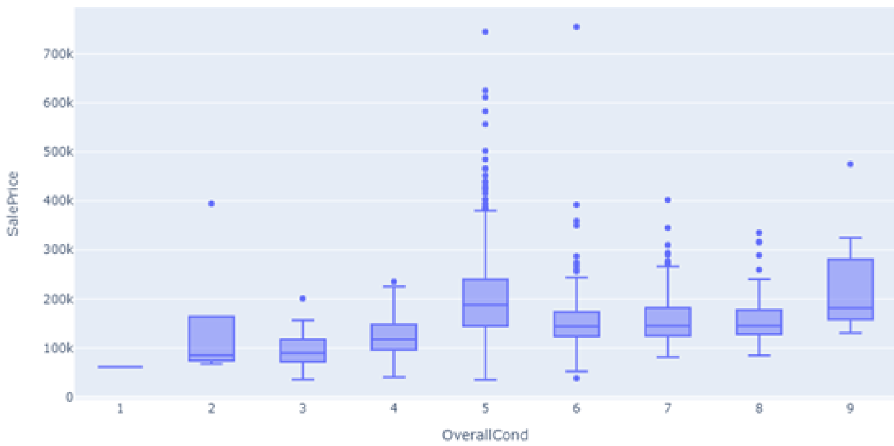
  iii. Total Bsmt SF, etcetera

  Box Plots (I have included few visualizations so that file can be uploaded in github repository and is not forbidden because of too large size, In jupyter notebook, all visualizations can be seen clearly).

VISUALIZATIONS







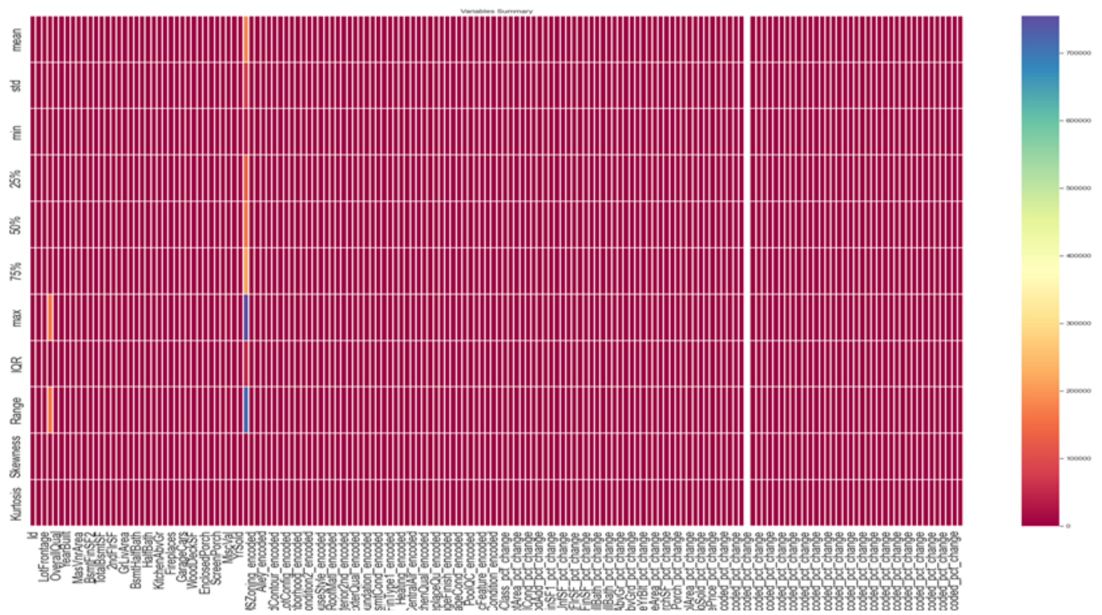
- 1. The data has many outliers.
- 2. To name a few:
- MS Sub Class
- Lot Frontage
- Lot Area
- **By studying box plots and correlation analysis, we have found that outliers are present in all the columns, hence, I am removing outliers by data points. We will do further eda with vif, dist plots and descriptive statistics to finally select a set of features.**

Slide Type

```
cription1=pd.concat([description, lst1_frame, lst3_frame, lst5, lst6])
cription1
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...
t	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	...
n	724.136130	56.767979	71.053082	10484.749144	6.104452	5.595890	1970.930651	1984.758562	102.080479	444.726027	...
d	416.159877	41.940650	24.998590	8957.442311	1.390153	1.124343	30.145255	20.785185	182.239859	462.664785	...
n	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1875.000000	1950.000000	0.000000	0.000000	...
%	360.500000	20.000000	60.000000	7621.500000	5.000000	5.000000	1954.000000	1966.000000	0.000000	0.000000	...
%	714.500000	50.000000	70.000000	9522.500000	6.000000	5.000000	1972.000000	1993.000000	0.000000	385.500000	...
%	1079.500000	70.000000	80.000000	11515.500000	7.000000	6.000000	2000.000000	2004.000000	160.000000	714.500000	...
x	1460.000000	190.000000	313.000000	164660.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000	...
R	365.000000	20.000000	10.000000	1993.000000	1.000000	1.000000	28.000000	11.000000	160.000000	329.000000	...
e	1459.000000	170.000000	292.000000	163360.000000	9.000000	8.000000	135.000000	60.000000	1600.000000	5644.000000	...
s	0.026526	1.422019	2.750805	10.659285	0.175082	0.580714	-0.579204	-0.495864	2.829214	1.871606	...
s	-1.185445	1.697882	22.744932	158.999786	0.137871	1.010681	-0.503733	-1.292204	11.353030	13.180303	...

162 columns



- **Mean = sum of values/count of values**
- **std = sqrt(((value - mean of distribution)**2 / number of values))**
- **3 quartile are measures of variance, calculated to spot the placeholder value, it returns index of the produced value. Step 1: sort the dataset**
  Step2:

i) Lower Quartile (Q1: 25% distribution) = ((number of values+1)/4)th Term
ii) Middle Quartile (Q2: 50% distribution) = ((number of values +1)/2)th Term
Also, know as median (central value).
iii) Upper Quartile (Q3: 75% distribution) = ¾(number of values + 1)th Term
iv) IQR = Upper Quartile - Lowe Quartile

- **Range = Maximum Value - Minimum Value**
- **Skewness = (sumation(value - mean of distribution)3)/((number of values - 1) * std3)**
- **Kurtosis = number of values * ((sumation(value - mean of distribution)4) / std4)**
- **1. For values that are scaled upto 1, mean is mostly around 0 and standard deviation is comparatively low. Hence, making the data much more acceptable by algorithms to process it more accurately.**
- **2. The entire dataset ranges from -1 till 755000.**
- **3. Skewness is 0 and within +/- 0.65 for:**
- ['Id', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
  'BsmtFullBath', 'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd',
  'GarageYrBlt', 'GarageCars', 'GarageArea', 'MoSold', 'YrSold',
  'Alley_encoded', 'LotShape_encoded', 'Utilities_encoded',

'Neighborhood_encoded', 'HouseStyle_encoded', 'Exterior1st_encoded',
'Exterior2nd_encoded', 'MasVnrType_encoded', 'Foundation_encoded',
'BsmtFinType1_encoded', 'HeatingQC_encoded', 'FireplaceQu_encoded',
'GarageFinish_encoded', 'PoolQC_encoded', 'Fence_encoded',
'YearBuilt_pct_change', 'YearRemodAdd_pct_change',
'2ndFlrSF_pct_change', 'Fireplaces_pct_change',
'GarageYrBlt_pct_change', 'YrSold_pct_change',
'Street_encoded_pct_change', 'Alley_encoded_pct_change',
'LotShape_encoded_pct_change', 'Utilities_encoded_pct_change',
'Foundation_encoded_pct_change', 'BsmtQual_encoded_pct_change',
'HeatingQC_encoded_pct_change', 'CentralAir_encoded_pct_change',
'KitchenQual_encoded_pct_change', 'GarageFinish_encoded_pct_change',
'PavedDrive_encoded_pct_change', 'PoolQC_encoded_pct_change',
'Fence_encoded_pct_change']

- **4. Acceptable skewness is +/- 0.65 and skewness for bell shaped curve should be 0.**

- ['Id', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
  'BsmtFullBath', 'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd',
  'GarageYrBlt', 'GarageCars', 'GarageArea', 'MoSold', 'YrSold',
  'Alley_encoded', 'LotShape_encoded', 'Utilities_encoded',
  'Neighborhood_encoded', 'HouseStyle_encoded', 'Exterior1st_encoded',
  'Exterior2nd_encoded', 'MasVnrType_encoded', 'Foundation_encoded',
  'BsmtFinType1_encoded', 'HeatingQC_encoded', 'FireplaceQu_encoded',
  'GarageFinish_encoded', 'PoolQC_encoded', 'Fence_encoded',
  'YearBuilt_pct_change', 'YearRemodAdd_pct_change',
  '2ndFlrSF_pct_change', 'Fireplaces_pct_change',
  'GarageYrBlt_pct_change', 'YrSold_pct_change',
  'Street_encoded_pct_change', 'Alley_encoded_pct_change',
  'LotShape_encoded_pct_change', 'Utilities_encoded_pct_change',
  'Foundation_encoded_pct_change', 'BsmtQual_encoded_pct_change',
  'HeatingQC_encoded_pct_change', 'CentralAir_encoded_pct_change',
  'KitchenQual_encoded_pct_change', 'GarageFinish_encoded_pct_change',

'PavedDrive_encoded_pct_change',
'PoolQC_encoded_pct_change',
'Fence_encoded_pct_change']

- **5. Kutosis is upto 3 for most dataset, indicating platykurtic curves:**
- ['Id', 'MSSubClass', 'OverallQual', 'OverallCond', 'YearBuilt',
'YearRemodAdd', 'BsmtUnfSF', '2ndFlrSF', 'BsmtFullBath', 'FullBath',
'HalfBath', 'BedroomAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt',
'GarageCars', 'GarageArea', 'WoodDeckSF', 'MoSold', 'YrSold',
'Alley_encoded', 'LotShape_encoded', 'Utilities_encoded',
'LotConfig_encoded', 'Neighborhood_encoded', 'HouseStyle_encoded',
'RoofStyle_encoded', 'Exterior1st_encoded', 'Exterior2nd_encoded',
'MasVnrType_encoded', 'Foundation_encoded', 'BsmtQual_encoded',
'BsmtExposure_encoded', 'BsmtFinType1_encoded', 'HeatingQC_encoded',
'KitchenQual_encoded', 'FireplaceQu_encoded', 'GarageType_encoded',
'GarageFinish_encoded', 'PoolQC_encoded', 'Fence_encoded',
'YearBuilt_pct_change', 'YearRemodAdd_pct_change',

'2ndFlrSF_pct_change', 'BsmtFullBath_pct_change', 'FullBath_pct_change',
'HalfBath_pct_change', 'TotRmsAbvGrd_pct_change',
'Fireplaces_pct_change', 'GarageYrBlt_pct_change',
'GarageCars_pct_change', '3SsnPorch_pct_change', 'YrSold_pct_change',
'Alley_encoded_pct_change', 'LotShape_encoded_pct_change',
'Utilities_encoded_pct_change', 'LotConfig_encoded_pct_change',
'BldgType_encoded_pct_change', 'HouseStyle_encoded_pct_change',
'RoofStyle_encoded_pct_change', 'MasVnrType_encoded_pct_change',
'Foundation_encoded_pct_change', 'BsmtQual_encoded_pct_change',
'BsmtExposure_encoded_pct_change', 'BsmtFinType1_encoded_pct_change',
'HeatingQC_encoded_pct_change', 'KitchenQual_encoded_pct_change',
'GarageType_encoded_pct_change', 'GarageFinish_encoded_pct_change',
'PoolQC_encoded_pct_change', 'Fence_encoded_pct_change']

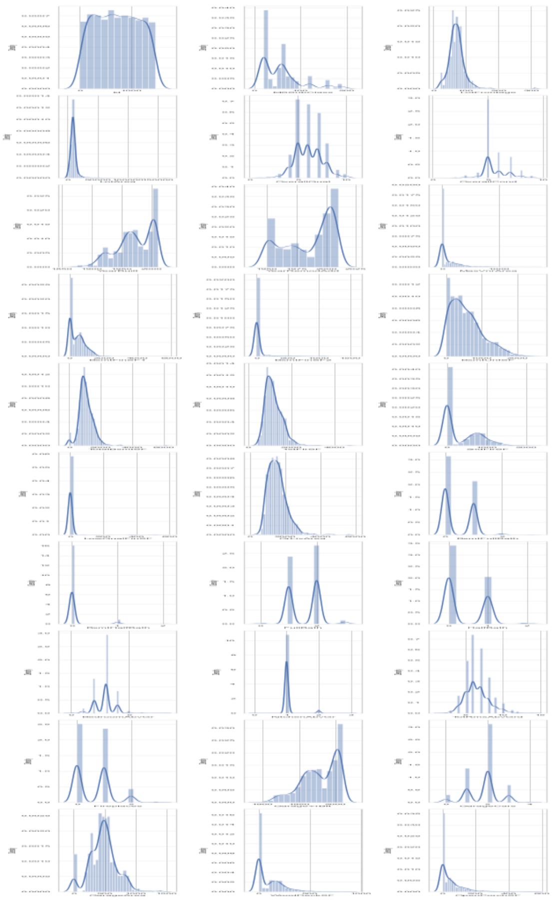
- **6. Kurtosis is greater than 3, indicating, leptokurtic curve:**
- ['LotFrontage', 'LotArea', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',

'TotalBsmtSF', '1stFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtHalfBath',
'KitchenAbvGr', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch',
'ScreenPorch', 'PoolArea', 'MiscVal', 'SalePrice', 'MSZoning_encoded',
'Street_encoded', 'LandContour_encoded', 'LandSlope_encoded',
'Condition1_encoded', 'Condition2_encoded', 'BldgType_encoded',
'RoofMatl_encoded', 'ExterQual_encoded', 'ExterCond_encoded',
'BsmtCond_encoded', 'BsmtFinType2_encoded', 'Heating_encoded',
'CentralAir_encoded', 'Electrical_encoded', 'Functional_encoded',
'GarageQual_encoded', 'GarageCond_encoded', 'PavedDrive_encoded',
'MiscFeature_encoded', 'SaleType_encoded', 'SaleCondition_encoded',
'Id_pct_change', 'MSSubClass_pct_change', 'LotFrontage_pct_change',
'LotArea_pct_change', 'OverallQual_pct_change',
'OverallCond_pct_change', 'MasVnrArea_pct_change',
'BsmtFinSF1_pct_change', 'BsmtFinSF2_pct_change',
'BsmtUnfSF_pct_change', 'TotalBsmtSF_pct_change', '1stFlrSF_pct_change',
'LowQualFinSF_pct_change', 'GrLivArea_pct_change',
'BsmtHalfBath_pct_change',

```
'BedroomAbvGr_pct_change',
  'KitchenAbvGr_pct_change',
'GarageArea_pct_change',
  'WoodDeckSF_pct_change',
'OpenPorchSF_pct_change',
  'EnclosedPorch_pct_change',
'ScreenPorch_pct_change',
  'PoolArea_pct_change', 'MiscVal_pct_change',
'MoSold_pct_change',
  'SalePrice_pct_change',
'MSZoning_encoded_pct_change',
  'Street_encoded_pct_change',
'LandContour_encoded_pct_change',
  'LandSlope_encoded_pct_change',
'Neighborhood_encoded_pct_change',
  'Condition1_encoded_pct_change',
'Condition2_encoded_pct_change',
  'RoofMatl_encoded_pct_change',
'Exterior1st_encoded_pct_change',
  'Exterior2nd_encoded_pct_change',
'ExterQual_encoded_pct_change',
  'ExterCond_encoded_pct_change',
'BsmtCond_encoded_pct_change',
  'BsmtFinType2_encoded_pct_change',
'Heating_encoded_pct_change',
  'CentralAir_encoded_pct_change',
'Electrical_encoded_pct_change',
  'Functional_encoded_pct_change',
'FireplaceQu_encoded_pct_change',
```

'GarageQual_encoded_pct_change',
'GarageCond_encoded_pct_change',
'PavedDrive_encoded_pct_change',
'MiscFeature_encoded_pct_change',
'SaleType_encoded_pct_change',
'SaleCondition_encoded_pct_change']

- **7. Kurtosis for bell shaped curve should be 3.**



Observations

1. Acceptable skewness is +/-0.65 and Right skewness for bell shaped curve is 0

- **2. Acceptable and Outliers Prone Left skewness is observed in many columns, including:**

- YearBuilt -0.579204
  YearRemodAdd -0.495864
  GarageYrBlt -0.645078
  GarageCars -0.358556
  MSZoning_encoded -1.796785
  Street_encoded -17.021969
  LotShape_encoded -0.603775
- **3. Acceptable And Outliers Prone Right Skewness is observed in most of the columns, including :**
- Id 0.026526032012241022
  MSSubClass 1.422018988135284
  LotFrontage 2.75080497659666
  LotArea 10.659284548299626
  OverallQual 0.1750824992845271

- Interpretation of the Results
- **## Based On EDA done above in two parts, I will do ftest and pvalue test on these seemingly weak indicators based primarily on skewness, kurtosis and multicollinearity:**
- **Misc Val (Skewness & Kurtosis)**
- **Street Encoded (Skewness & Kurtosis)**
- **Total Bsmt SF (Multicollinearity, Skewness & Kurtosis)**
- **Most of the biased features have strong explanatory power in terms of correlation with feature, skewness or kurtosis and hence can be filtered in ensemble method of feature selection by p value and constant variance threshold.**

  ANOVA Test On Selected Features

- **Ftest score should be greater than 1 and p value should be less than 0.05, to determine to keep these features for further analysis**

Feature 1: MiscVal

57]:

```
from scipy.stats import f_oneway
f,p=f_oneway(data['MiscVal'], data['SalePrice'])
f,p
```

ut[57]:

```
(6143.625978565872, 0.0)
```

Feature 2: Street_encoded

In [58]:

```
from scipy.stats import f_oneway
f,p=f_oneway(data['Street_encoded'], data['SalePrice'])
f,p
```

Out[58]:

```
(6147.053184417584, 0.0)
```

Feature 3: TotalBsmtSF

In [60]:

```
from scipy.stats import f_oneway
f,p=f_oneway(data['TotalBsmtSF'], data['SalePrice'])
f,p
```

]:

```
(6075.256588986826, 0.0)
```

- **Based on above analysis:**
- **1. There are many outliers in the data.**
- **2. Strong multicolliearity features are important for prediction because there f test and p value are acceptable. This means that the amount of multicollineaity is insignificant and removing the feature will impact the model much.**
- **3. Extereme leptokurtic and right skewed features are also relatively significant based on f test and p test.**

- **Hence, as a solution, feature scaling will do a better job in explaining the dependent variable than removing whole columns.**

Outliers Transformation

Slide Type

Outliers Transformation With Power Transform

In [67]:

Slide Type

```python
from sklearn.preprocessing import power_transform
x_array=power_transform(x, method='yeo-johnson')
x_frame=pd.DataFrame(x_array, columns=x.columns)
x_frame
```

Out[67]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	GarageType_encoded_
0	-1.544968	1.370435	1.032075	-1.213954	-0.052799	-0.496528	0.026859	-0.613054	-0.815710	-0.112649	...	
1	0.461941	-1.167999	1.032075	1.100521	1.345826	0.407009	-0.185817	-0.854247	-0.815710	0.394677	...	
2	0.254221	0.490047	0.926721	0.158048	0.656375	-0.496528	0.836680	0.495717	-0.815710	0.925416	...	
3	-1.610857	-1.167999	1.367391	0.496002	-0.052799	0.407009	0.063582	-0.569998	1.387482	0.797133	...	
4	-0.635571	-1.167999	-0.478610	1.196626	-0.052799	1.234321	0.063582	0.694762	1.150000	1.178115	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1163	-1.008279	-1.167999	-0.060479	0.137703	-0.785224	-0.496528	-0.287388	-0.964508	0.805682	0.530842	...	
1164	-0.299331	-1.167999	-0.060479	-0.084786	-1.545782	-0.496528	-0.835594	0.905817	-0.815710	-1.347679	...	
1165	-1.300337	1.700798	-2.662554	-2.681835	-0.052799	0.407009	0.026859	-0.613054	-0.815710	0.663505	...	
1166	-1.975680	0.696557	-0.889729	-0.148203	-1.545782	-1.498082	-1.530544	-1.478940	-0.815710	-1.347679	...	
1167	-0.147481	0.490047	-0.889729	-0.302416	-0.052799	-0.496528	1.113154	0.905817	-0.815710	0.539557	...	

1168 rows × 160 columns

Min Max Scaler Transformation

In [70]:

Slide Type

```python
import sklearn
from sklearn.preprocessing import MinMaxScaler
from statsmodels.stats.outliers_influence import variance_inflation_factor
import warnings
warnings.filterwarnings('ignore')
scaler=MinMaxScaler([0,1])
X_scaled=scaler.fit_transform(x_frame)
X_scaled_frame=pd.DataFrame(X_scaled, columns=x_frame.columns)
X_scaled_frame
```

Out[70]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	GarageType_encoded_p
0	0.173115	0.830391	0.456078	0.256316	0.596935	0.613289	0.607531	0.295731	0.000000	0.325076	...	
1	0.703401	0.000000	0.456078	0.492699	0.803551	0.722066	0.551199	0.213355	0.000000	0.458611	...	
2	0.648515	0.542392	0.443988	0.396442	0.701700	0.613289	0.822032	0.674416	0.000000	0.598309	...	
3	0.155705	0.000000	0.494555	0.430958	0.596935	0.722066	0.617258	0.310436	0.933715	0.564543	...	
4	0.413406	0.000000	0.282725	0.502514	0.596935	0.821667	0.617258	0.742397	0.833070	0.664296	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1163	0.314925	0.000000	0.330706	0.394364	0.488735	0.613289	0.524295	0.175697	0.687147	0.494452	...	
1164	0.502250	0.000000	0.330706	0.371640	0.376379	0.613289	0.379090	0.814480	0.000000	0.000000	...	
1165	0.237754	0.938461	0.032116	0.106397	0.596935	0.722066	0.607531	0.295731	0.000000	0.529370	...	
1166	0.059308	0.609947	0.235549	0.365163	0.376379	0.492711	0.195015	0.000000	0.000000	0.000000	...	
1167	0.542374	0.542392	0.235549	0.349413	0.596935	0.613289	0.895263	0.814480	0.000000	0.496745	...	

1168 rows × 160 columns

Variance Inflation Factor

After passing through vif test to remove multicollinearity, only 33 features seem to be low bias with seemingly strong explanatory power

Concluding Points:

- **As part of data handling, I have closely analyzed features with high outliers (by analyzing box plots, dist plots, variable plot and scatter plots).**
- **I have removed features with multicollinearity by analyzing correlation, correlation heatmaps and variance inflation factor.**
- **I have done ANOVA testing, wherever, applicable to weigh importance against bias.Hence, the model can be expected to be low variance and low bias model**

CONCLUSION

- Key Findings and Conclusions of the Study
- **Based on above analysis:**
- **1. There are many outliers in the data.**
- **2. Strong multicolliearity features are important for prediction because there f test and p value are acceptable. This means that the amount of multicollineaity is insignificant and removing the feature will impact the model much.**
- **3. Extereme leptokurtic and right skewed features are also relatively significant based on f test and p test.**
- **Hence, as a solution, feature scaling will do a better job in explaining the dependent variable than removing whole columns.**

Outliers Transformation

Slide Type

Outliers Transformation With Power Transform

In [67]:

Slide Type

```python
from sklearn.preprocessing import power_transform
x_array=power_transform(x, method='yeo-johnson')
x_frame=pd.DataFrame(x_array, columns=x.columns)
x_frame
```

Out[67]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	GarageType_encoded_
0	-1.544968	1.370435	1.032075	-1.213954	-0.052799	-0.496528	0.026859	-0.613054	-0.815710	-0.112649	...	
1	0.461941	-1.167999	1.032075	1.100521	1.345826	0.407009	-0.185817	-0.854247	-0.815710	0.394677	...	
2	0.254221	0.490047	0.926721	0.158048	0.656375	-0.496528	0.836680	0.495717	-0.815710	0.925416	...	
3	-1.610857	-1.167999	1.367391	0.496002	-0.052799	0.407009	0.063582	-0.569998	1.387482	0.797133	...	
4	-0.635571	-1.167999	-0.478610	1.196626	-0.052799	1.234321	0.063582	0.694762	1.150000	1.176115	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1163	-1.008279	-1.167999	-0.060479	0.137703	-0.785224	-0.496528	-0.287388	-0.964508	0.805682	0.530842	...	
1164	-0.299331	-1.167999	-0.060479	-0.084786	-1.545782	-0.496528	-0.835594	0.905817	-0.815710	-1.347679	...	
1165	-1.300337	1.700798	-2.662554	-2.681835	-0.052799	0.407009	0.026859	-0.613054	-0.815710	0.663505	...	
1166	-1.975680	0.696557	-0.889729	-0.148203	-1.545782	-1.498082	-1.530544	-1.478940	-0.815710	-1.347679	...	
1167	-0.147481	0.490047	-0.889729	-0.302416	-0.052799	-0.496528	1.113154	0.905817	-0.815710	0.539557	...	

1168 rows × 160 columns

Min Max Scaler Transformation

In [70]:

Slide Type

```python
import sklearn
from sklearn.preprocessing import MinMaxScaler
from statsmodels.stats.outliers_influence import variance_inflation_factor
import warnings
warnings.filterwarnings('ignore')
scaler=MinMaxScaler([0,1])
X_scaled=scaler.fit_transform(x_frame)
X_scaled_frame=pd.DataFrame(X_scaled, columns=x_frame.columns)
X_scaled_frame
```

Out[70]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	GarageType_encoded_p
0	0.173115	0.830391	0.456078	0.256316	0.596935	0.613289	0.607531	0.295731	0.000000	0.325076	...	
1	0.703401	0.000000	0.456078	0.492699	0.803551	0.722066	0.551199	0.213355	0.000000	0.458611	...	
2	0.648515	0.542392	0.443988	0.396442	0.701700	0.613289	0.822032	0.674416	0.000000	0.598309	...	
3	0.155705	0.000000	0.494555	0.430958	0.596935	0.722066	0.617258	0.310436	0.933715	0.564543	...	
4	0.413406	0.000000	0.282725	0.502514	0.596935	0.821667	0.617258	0.742397	0.833070	0.664296	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1163	0.314925	0.000000	0.330706	0.394364	0.488735	0.613289	0.524295	0.175697	0.687147	0.494452	...	
1164	0.502250	0.000000	0.330706	0.371640	0.376379	0.613289	0.379090	0.814480	0.000000	0.000000	...	
1165	0.237754	0.938461	0.032116	0.106397	0.596935	0.722066	0.607531	0.295731	0.000000	0.529370	...	
1166	0.059308	0.609947	0.235549	0.365163	0.376379	0.492711	0.195015	0.000000	0.000000	0.000000	...	
1167	0.542374	0.542392	0.235549	0.349413	0.596935	0.613289	0.895263	0.814480	0.000000	0.496745	...	

1168 rows × 160 columns

Variance Inflation Factor

After passing through vif test to remove multicollinearity, only 33 features seem to be low bias with seemingly strong explanatory power

Concluding Points:

- **As part of data handling, I have closely analyzed features with high outliers (by analyzing box plots, dist plots, variable plot and scatter plots).**
- **I have removed features with multicollinearity by analyzing correlation, correlation heatmaps and variance inflation factor.**
- **I have done ANOVA testing, wherever, applicable to weigh importance against bias.Hence, the model can be expected to be low variance and low bias model**

- Learning Outcomes of the Study in respect of Data Science

  Visualizations and data cleaning convert a whole complex and messy dataset into insightful and interesting representation, which make it easier to reach the core of the problem and solve it.

  The best model is Huber Regressor With Default Hyper Parameter tuning, the most challenging part in models development process was to reduce overfitting and that is why I have applied ensemble methods on base estimators... Huber Regressor provided the best framework to reduce overfitting.

- Limitations of this work and Scope for Future Work

  Further optimatization can be obtained by applying deep learning solutions. Since, it requires very high RAM capacity, it could not be displayed in jupyter notebook... I would like to update Google Colab Notebook for future projects, if acceptable... That can help me to submit a completely optimized model.