Feature Engineering on House Pricing - Kaggle Dataset

We are performing the below steps in Feature Engineering

1. Handling Missing Values
2. Temporal Variables
3. Handle Categorical Variables : remove rare labels
4. Standardize the values of the variable to the same range

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

# Display all the columns of the dataframs
pd.pandas.set_option('display.max_columns',None)
```

In [2]:
```
# Reading Dataset
dataset = pd.read_csv('train.csv')
dataset.head(2)
```

Out[2]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Conditic
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Nc
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Fe

In [3]:
```
# Always remember there are always be a chance of data lekage so need to split the data first and then apply feature eng
from sklearn.model_selection import train_test_split
X = dataset.iloc[:,1:81]
y = dataset.iloc[:,-1]
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.1)
```

In [4]:
```
X_train.shape,X_test.shape
```

Out[4]:
```
((1314, 80), (146, 80))
```

Missing Values

categorical feature with nan

In [5]:
```
# We can capture all the nan values
# First lets handle categorical feature which are missing

categorical_features_nan=[feature for feature in dataset.columns if dataset[feature].isnull().sum()>=1 and dataset[featu
for feature in categorical_features_nan:
    print("The feature is {} and missing values in {}% ".format(feature,np.round(dataset[feature].isnull().mean(),4)))
```

```
The feature is Alley and missing values in 0.9377%
The feature is MasVnrType and missing values in 0.0055%
The feature is BsmtQual and missing values in 0.0253%
The feature is BsmtCond and missing values in 0.0253%
The feature is BsmtExposure and missing values in 0.026%
The feature is BsmtFinType1 and missing values in 0.0253%
The feature is BsmtFinType2 and missing values in 0.026%
The feature is Electrical and missing values in 0.0007%
The feature is FireplaceQu and missing values in 0.4726%
The feature is GarageType and missing values in 0.0555%
The feature is GarageFinish and missing values in 0.0555%
The feature is GarageQual and missing values in 0.0555%
The feature is GarageCond and missing values in 0.0555%
The feature is PoolQC and missing values in 0.9952%
The feature is Fence and missing values in 0.8075%
The feature is MiscFeature and missing values in 0.963%
```

In [6]:
```
## Replacing categorical features missing value with new value
data = dataset.copy()
def replace_cat_features(dataset,categorical_features_nan):
    data[categorical_features_nan] = data[categorical_features_nan].fillna("Missing")
    return data

dataset = replace_cat_features(dataset,categorical_features_nan)
dataset[categorical_features_nan].isnull().sum()
```

Out[6]:
```
Alley           0
MasVnrType      0
BsmtQual        0
BsmtCond        0
BsmtExposure    0
BsmtFinType1    0
BsmtFinType2    0
Electrical      0
FireplaceQu     0
GarageType      0
GarageFinish    0
GarageQual      0
GarageCond      0
PoolQC          0
Fence           0
MiscFeature     0
dtype: int64
```

In [7]:
```
dataset.head()
```

Out[7]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condi
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR2	Gtl	Veenker	F
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	

Numerical Variables with nan

In [8]:
```
# Checking the numerical variables with missing values
numerical_with_nan = [feature for feature in dataset.columns if dataset[feature].isnull().sum()>=1 and dataset[feature].

# we will print the numerical nan variable and percentage of missing values

for feature in numerical_with_nan:
    print(" {}: {}% missing value".format(feature,np.around(dataset[feature].isnull().mean(),4)))
```

```
 LotFrontage: 0.1774% missing value
 MasVnrArea: 0.0055% missing value
 GarageYrBlt: 0.0555% missing value
```

In [9]:
```
# Replacing the numerical missing values

for feature in numerical_with_nan:
    ## since there are outliers we are goign to replace wwith median
    median_values=dataset[feature].median()
    ## create a new feature to capture nan value
    dataset[feature+'nan']=np.where(dataset[feature].isnull(),1,0)
    dataset[feature].fillna(median_values,inplace=True)

dataset[numerical_with_nan].isnull().sum()
```

Out[9]:
```
LotFrontage    0
MasVnrArea     0
GarageYrBlt    0
dtype: int64
```

In [10]:
```
dataset.head(15)
```

Out[10]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Con
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR2	Gtl	Veenker	
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	
5	6	50	RL	85.0	14115	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	Mitchel	
6	7	20	RL	75.0	10084	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	Somerst	
7	8	60	RL	69.0	10382	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	NWAmes	
8	9	50	RM	51.0	6120	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	OldTown	
9	10	190	RL	50.0	7420	Pave	Missing	Reg	Lvl	AllPub	Corner	Gtl	BrkSide	
10	11	20	RL	70.0	11200	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	Sawyer	
11	12	60	RL	85.0	11924	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	NridgHt	
12	13	20	RL	69.0	12968	Pave	Missing	IR2	Lvl	AllPub	Inside	Gtl	Sawyer	
13	14	20	RL	91.0	10652	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	
14	15	20	RL	69.0	10920	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	NAmes	

Temporal Variables

In [11]:
```
# Temporal Variables (Date Time Variables)
for feature in ['YearBuilt','YearRemodAdd','GarageYrBlt']:
    dataset[feature] = dataset['YrSold']-dataset[feature]
```

In [12]:
```
dataset.head(2)
```

Out[12]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condi
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR2	Gtl	Veenker	F

Numerical Variables

In [13]:
```
# we found some skewed feature in numerical feature, So we need to remove skewness
num_features = ['LotFrontage', 'LotArea','1stFlrSF','GrLivArea','SalePrice']

for feature in num_features:
    dataset[feature]=np.log(dataset[feature])
```

In [14]:
```
dataset.head(2)
```

Out[14]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Cond
0	1	60	RL	4.174387	9.041922	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	
1	2	20	RL	4.382027	9.169518	Pave	Missing	Reg	Lvl	AllPub	FR2	Gtl	Veenker	

Handling Rare Categorical Feature

we will remove categorical variable that are present less than 1% of the observations

In [15]:
```
categorical_features = [ feature for feature in dataset.columns if dataset [ feature].dtypes == 'O']
```

In [16]:
```
categorical_features
```

Out[16]:
```
['MSZoning',
 'Street',
 'Alley',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinType2',
 'Heating',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 'KitchenQual',
 'Functional',
 'FireplaceQu',
 'GarageType',
 'GarageFinish',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'PoolQC',
```

```
 'Fence',
 'MiscFeature',
 'SaleType',
 'SaleCondition']
```

In [17]:
```
for feature in categorical_features:
    temp = dataset.groupby(feature)['SalePrice'].count()/len(dataset)
    temp_df=temp[temp>.01].index
    dataset[feature]=np.where(dataset[feature].isin(temp_df),dataset[feature],'Rare_var')
```

In [18]:
```
dataset.head(10)
```

Out[18]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Conc
0	1	60	RL	4.174387	9.041922	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	
1	2	20	RL	4.382027	9.169518	Pave	Missing	Reg	Lvl	AllPub	FR2	Gtl	Rare_var	
2	3	60	RL	4.219508	9.328123	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	
3	4	70	RL	4.094345	9.164296	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	
4	5	60	RL	4.430817	9.565214	Pave	Missing	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	
5	6	50	RL	4.442651	9.554993	Pave	Missing	IR1	Lvl	AllPub	Inside	Gtl	Mitchel	
6	7	20	RL	4.317488	9.218705	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	Somerst	
7	8	60	RL	4.234107	9.247829	Pave	Missing	IR1	Lvl	AllPub	Corner	Gtl	NWAmes	
8	9	50	RM	3.931826	8.719317	Pave	Missing	Reg	Lvl	AllPub	Inside	Gtl	OldTown	
9	10	190	RL	3.912023	8.911934	Pave	Missing	Reg	Lvl	AllPub	Corner	Gtl	BrkSide	

In [19]:
```
for feature in categorical_features:
    labels_ordered=dataset.groupby([feature])['SalePrice'].mean().sort_values().index
    labels_ordered={k:i for i,k in enumerate(labels_ordered,0)}
    dataset[feature]=dataset[feature].map(labels_ordered)
```

In [20]:
```
scaling_feature=[feature for feature in dataset.columns if feature not in ['Id','SalePerice'] ]
len(scaling_feature)
```

Out[20]: 83

In [21]:
```
scaling_feature
```

Out[21]:
```
['MSSubClass',
 'MSZoning',
 'LotFrontage',
 'LotArea',
 'Street',
 'Alley',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'OverallQual',
 'OverallCond',
 'YearBuilt',
 'YearRemodAdd',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'MasVnrArea',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinSF1',
 'BsmtFinType2',
 'BsmtFinSF2',
 'BsmtUnfSF',
 'TotalBsmtSF',
 'Heating',
```

```
'HeatingQC',
'CentralAir',
'Electrical',
'1stFlrSF',
'2ndFlrSF',
'LowQualFinSF',
'GrLivArea',
'BsmtFullBath',
'BsmtHalfBath',
'FullBath',
'HalfBath',
'BedroomAbvGr',
'KitchenAbvGr',
'KitchenQual',
'TotRmsAbvGrd',
'Functional',
'Fireplaces',
'FireplaceQu',
'GarageType',
'GarageYrBlt',
'GarageFinish',
'GarageCars',
'GarageArea',
'GarageQual',
'GarageCond',
'PavedDrive',
'WoodDeckSF',
'OpenPorchSF',
'EnclosedPorch',
'3SsnPorch',
'ScreenPorch',
'PoolArea',
'PoolQC',
'Fence',
'MiscFeature',
'MiscVal',
'MoSold',
'YrSold',
'SaleType',
'SaleCondition',
'SalePrice',
'LotFrontagenan',
```

```
 'MasVnrAreanan',
 'GarageYrBltnan']
```

In [22]:
```
dataset.head(2)
```

Out[22]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Conditi
0	1	60	3	4.174387	9.041922	1	2	0	1	1	0	0	14	
1	2	20	3	4.382027	9.169518	1	2	0	1	1	2	0	11	

Feature Scaling

In [23]:
```
feature_scale = [feature for feature in dataset.columns if feature not in ['Id', 'SalePrice']]
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
print(scaler.fit(dataset[feature_scale]))
```

```
MinMaxScaler()
```

In [24]:
```
scaler.transform(dataset[feature_scale])
```

Out[24]:
```
array([[0.23529412, 0.75      , 0.41820812, ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.49506375, ..., 0.        , 0.        ,
        0.        ],
       [0.23529412, 0.75      , 0.434909  , ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.29411765, 0.75      , 0.42385922, ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.434909  , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.47117546, ..., 0.        , 0.        ,
        0.        ]])
```

In [25]:
```
# transform the train and test set, and add on the Id and SalePrice variables
data = pd.concat([dataset[['Id', 'SalePrice']].reset_index(drop=True),
                    pd.DataFrame(scaler.transform(dataset[feature_scale]), columns=feature_scale)],
                    axis=1)
```

In [26]:
```
data.head(2)
```

Out[26]:

	Id	SalePrice	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborho
0	1	12.247694	0.235294	0.75	0.418208	0.366344	1.0	1.0	0.0	0.333333	1.0	0.0	0.0	0.6363
1	2	12.109011	0.000000	0.75	0.495064	0.391317	1.0	1.0	0.0	0.333333	1.0	0.5	0.0	0.5000

In [27]:
```
data.to_csv('X_train.csv',index=False)
```