House Price Prediction

Advanced Regression technique

Steps involved are:

1. Data Understanding
2. Data Analysis and Cleaning
3. Data Preparation
4. Model Building and Evaluation
5. Question-Answer

Step 1: Data Understanding

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# ignoring warnings
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv('train.csv')
df.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoS
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	

5 rows × 81 columns

```
df.shape
```

(1460, 81)

Importing libraries and understanding data set

Step 2: Data Analysis and Cleaning

Dividing the data set into numeric_vars and non_numeric_vars and analysing it separately

Analysing numerical columns

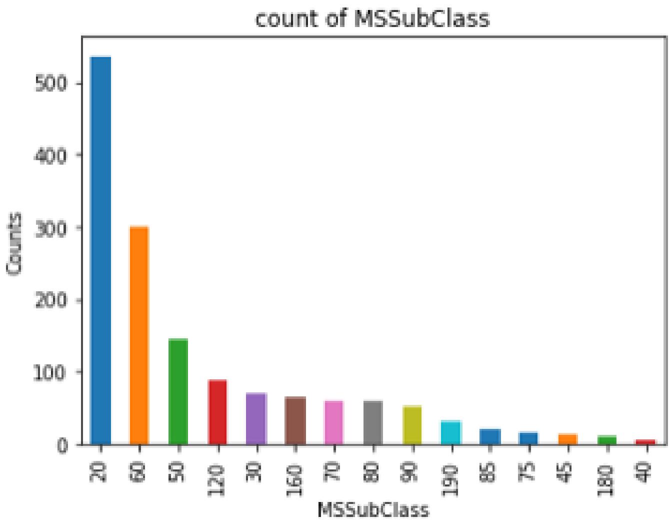
```
numeric_vars = df.select_dtypes(['float64','int64'])
numeric_vars.head()
```

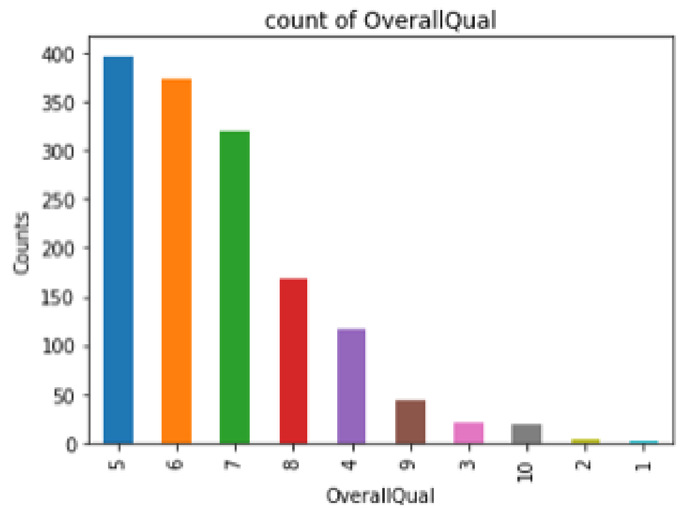
	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	WoodDeckSF	OpenPorchSF	Enc
0	1	60	65.0	8450	7	5	2003	2003	196.0	706	...	0	61	
1	2	20	80.0	9600	6	8	1976	1976	0.0	978	...	298	0	
2	3	60	68.0	11250	7	5	2001	2002	162.0	486	...	0	42	
3	4	70	60.0	9550	7	5	1915	1970	0.0	216	...	0	35	
4	5	60	84.0	14260	8	5	2000	2000	350.0	655	...	192	84	

5 rows × 38 columns

Numerical column has few categorical data, let's analyse them:

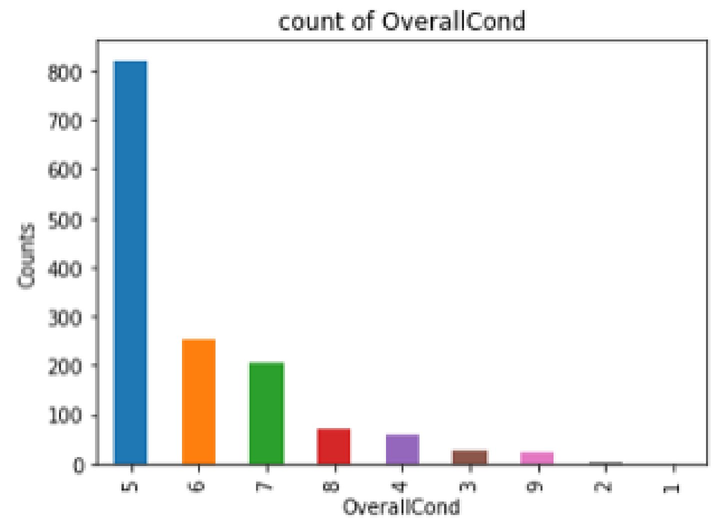
- MSSubClass
- OverallQual
- OverallCond



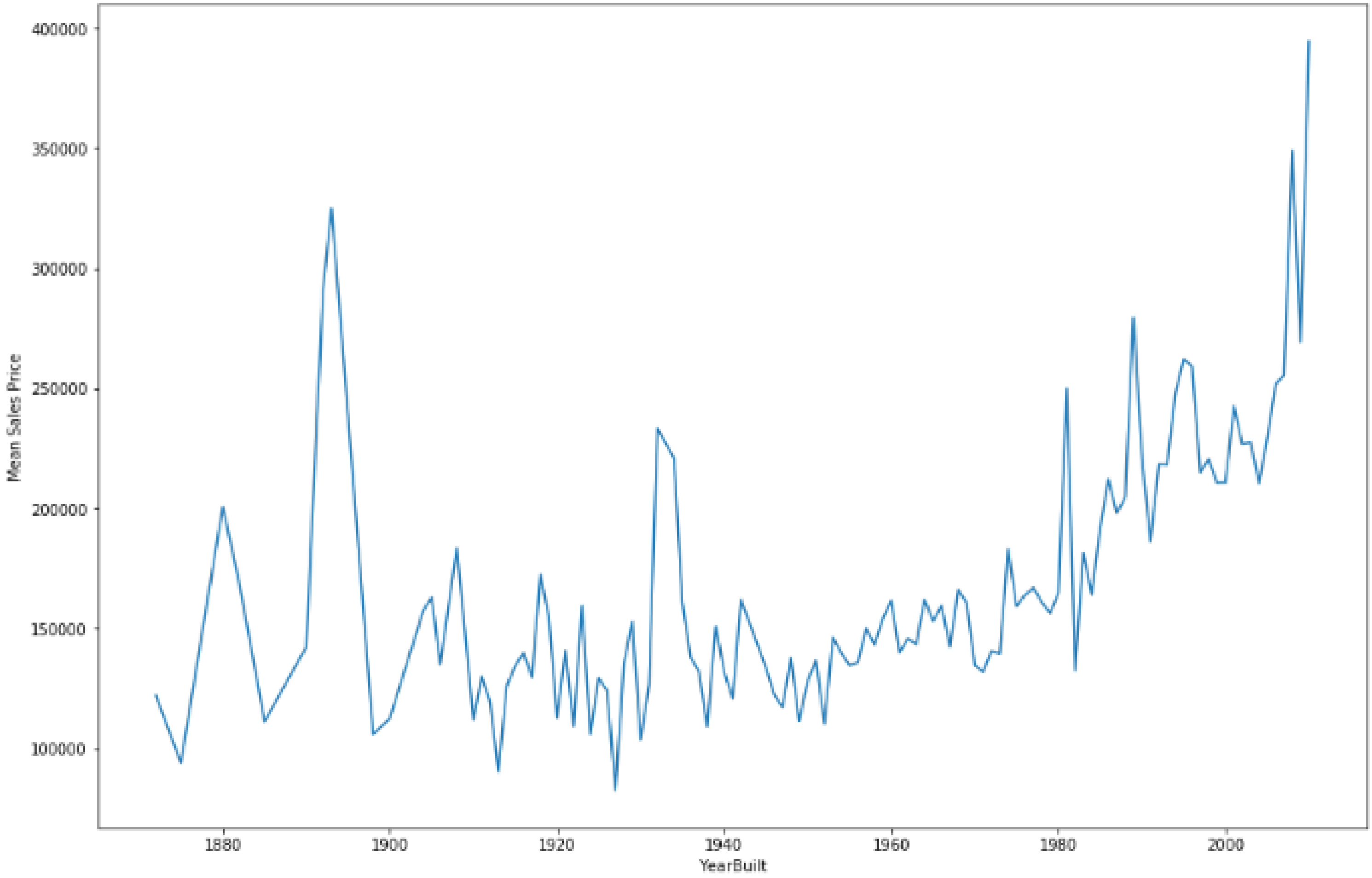


inferences:

1. counts of 20, 60 and 50 MSSubClass are more where
   - 20 = 1-STORY 1946 & NEWER ALL STYLES
   - 60 = 2-STORY 1946 & NEWER
   - 50 = 1-1/2 STORY FINISHED ALL AGES
2. mostly the houses have the overall quality between 5-7 where
   - 5 = Average
   - 6 = Above Average
   - 7 = Good
3. maximum houses have the overall condition 5
   - 5 = Average



Analysing Sale Price of the houses on the basis of the year it is built

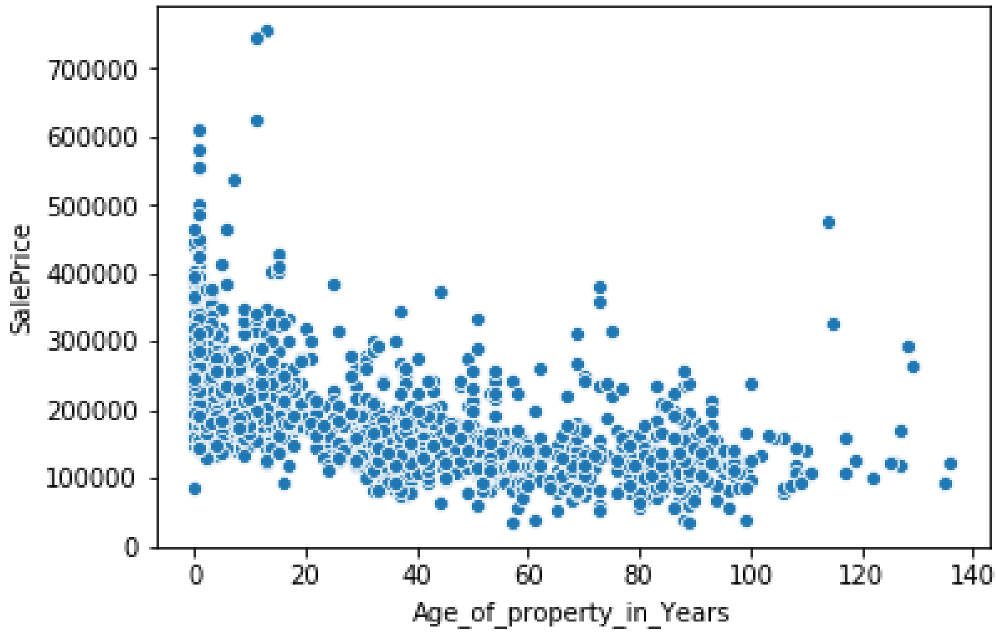


inferences:

- new houses have higher prices

creating column "Age_of_property_in_Years" and dropping YearBuilt, YrSold and MoSold columns

```
numeric_vars['Age_of_property_in_Years'] = numeric_vars['YrSold'] - numeric_vars['YearBuilt']
numeric_vars = numeric_vars.drop(['YearBuilt','YrSold','MoSold'],1)
numeric_vars.head()
```



inferences:

- as the age of the property increases, the price decreases

Finding the percentage of null values in numeric_vars data frame

```
Id                          0.00
MSSubClass                  0.00
LotFrontage                17.74
LotArea                     0.00
OverallQual                 0.00
OverallCond                 0.00
YearRemodAdd                0.00
MasVnrArea                  0.55
BsmtFinSF1                  0.00
BsmtFinSF2                  0.00
BsmtUnfSF                   0.00
TotalBsmtSF                 0.00
1stFlrSF                    0.00
2ndFlrSF                    0.00
LowQualFinSF                0.00
GrLivArea                   0.00
BsmtFullBath                0.00
BsmtHalfBath                0.00
FullBath                    0.00
HalfBath                    0.00
BedroomAbvGr                0.00
KitchenAbvGr                0.00
TotRmsAbvGrd                0.00
Fireplaces                  0.00
GarageYrBlt                 5.55
GarageCars                  0.00
GarageArea                  0.00
WoodDeckSF                  0.00
OpenPorchSF                 0.00
EnclosedPorch               0.00
3SsnPorch                   0.00
ScreenPorch                 0.00
PoolArea                    0.00
MiscVal                     0.00
SalePrice                   0.00
Age_of_property_in_Years    0.00
dtype: float64
```

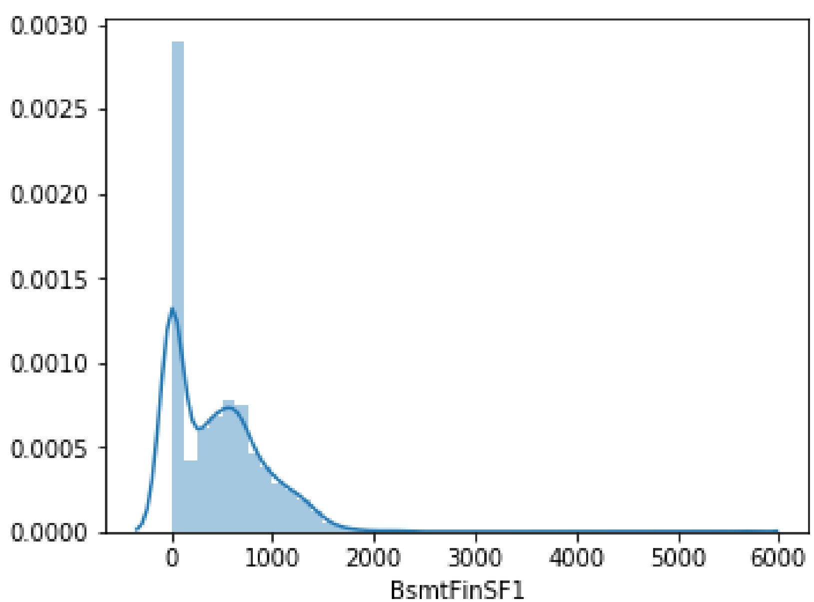
Imputing the missing values with their mean

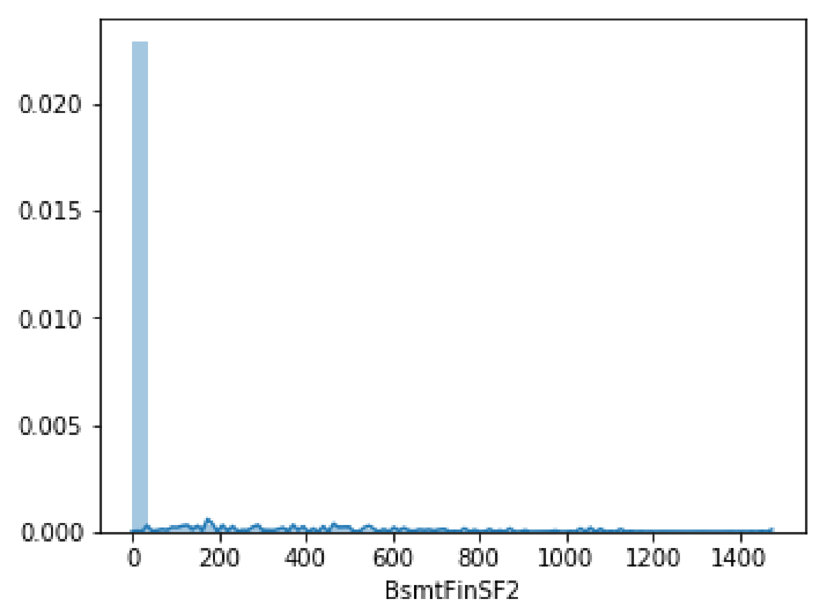
```
numeric_vars['LotFrontage'] = numeric_vars['LotFrontage'].fillna(numeric_vars['LotFrontage'].mean())
numeric_vars['MasVnrArea'] = numeric_vars['MasVnrArea'].fillna(numeric_vars['MasVnrArea'].mean())
```

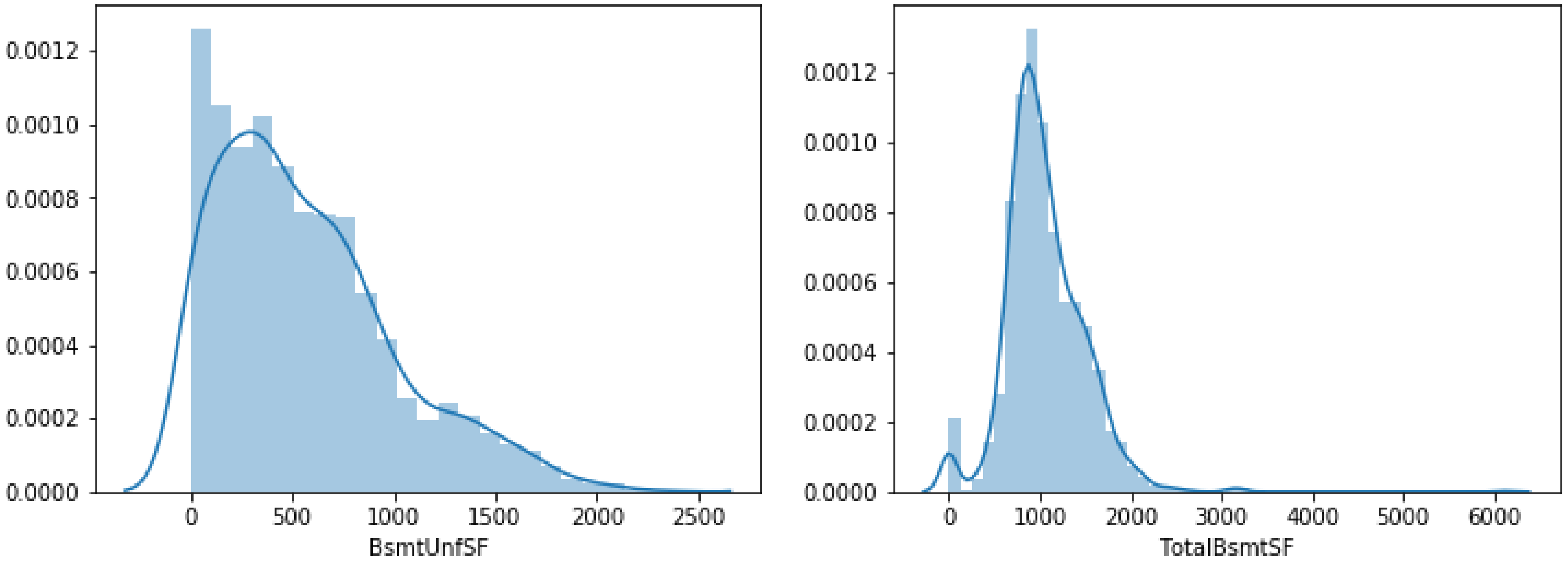
deleting unnecessary/redundant columns ('OverallCond','GarageYrBlt','YearRemodAdd')

```
numeric_vars = numeric_vars.drop(['OverallCond','GarageYrBlt','YearRemodAdd'], axis = 1)
numeric_vars.head()
```

analysing basement area



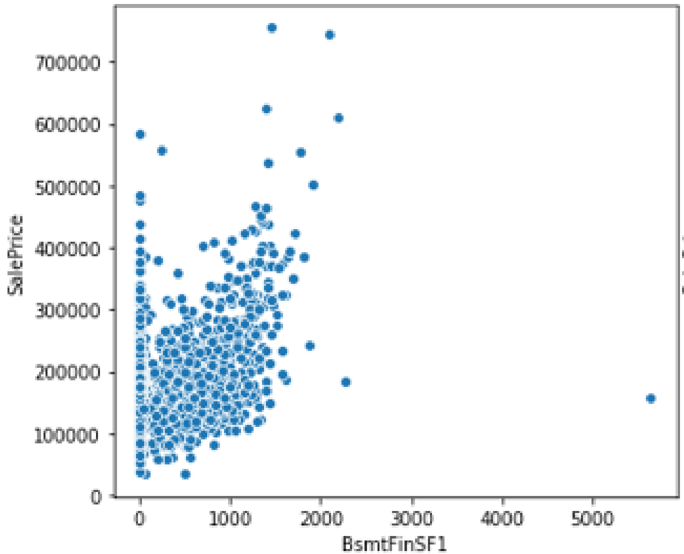


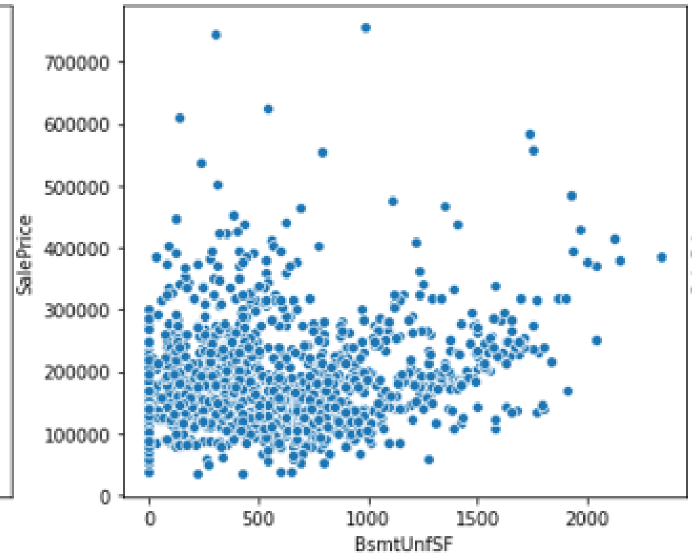


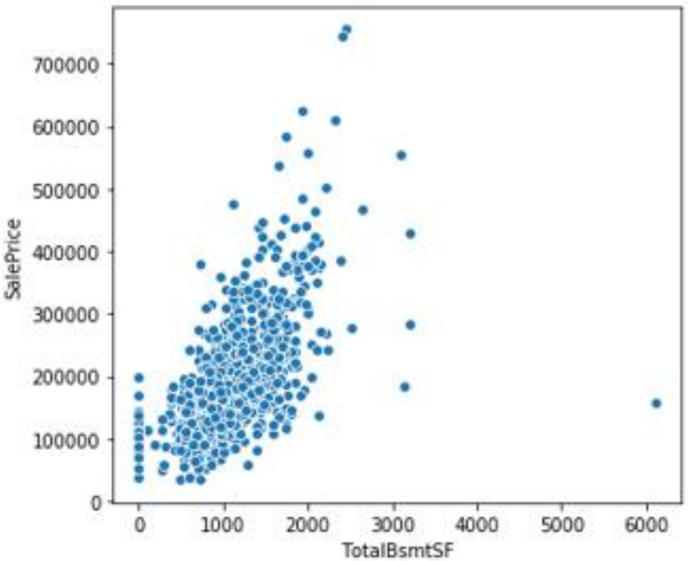
Since "BsmtFinSF2" shows no variance therefore dropping this column

```
numeric_vars = numeric_vars.drop(['BsmtFinSF2'], axis = 1)
```

analysing how basement area is affecting the sale price of the houses



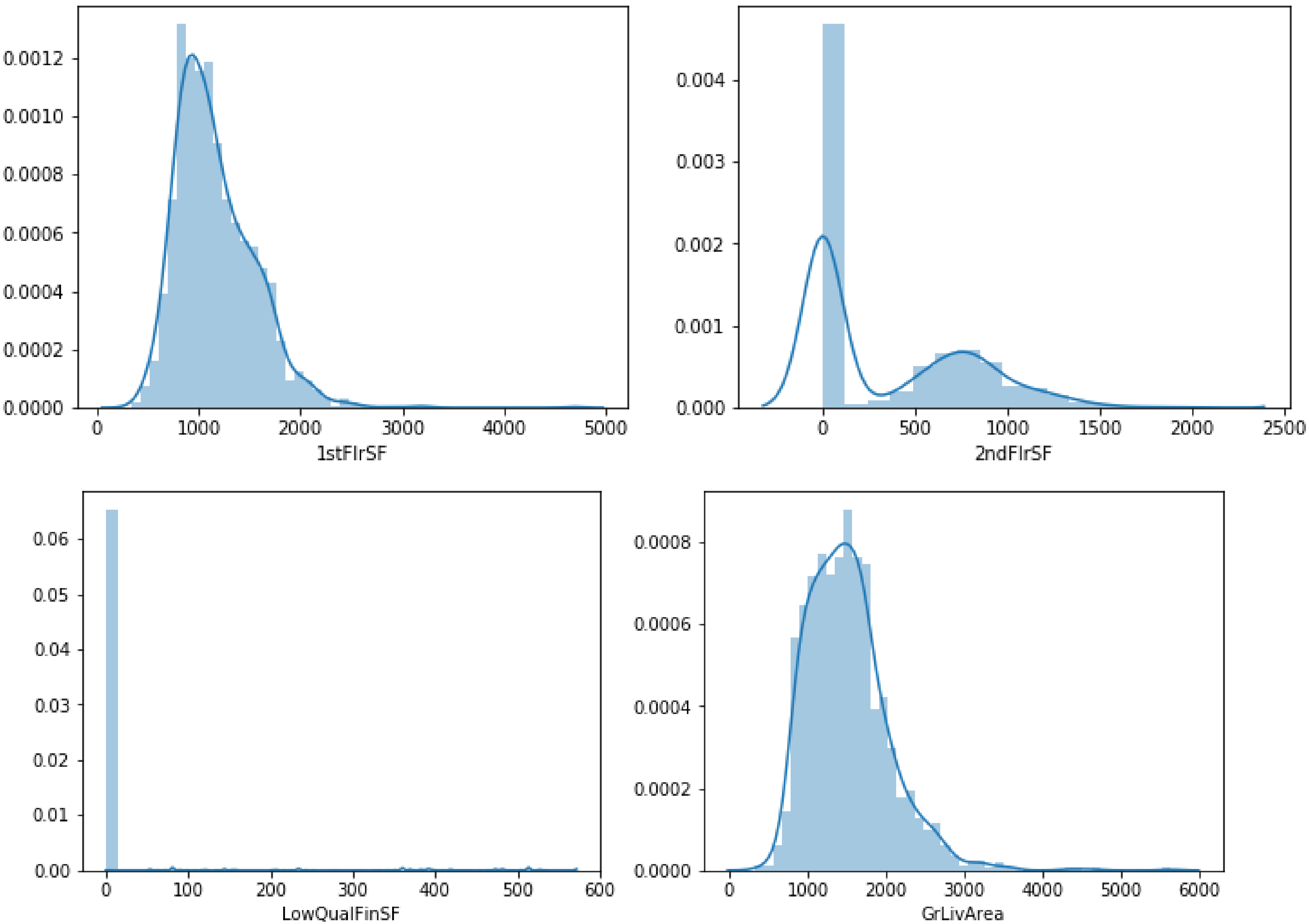




inferences:

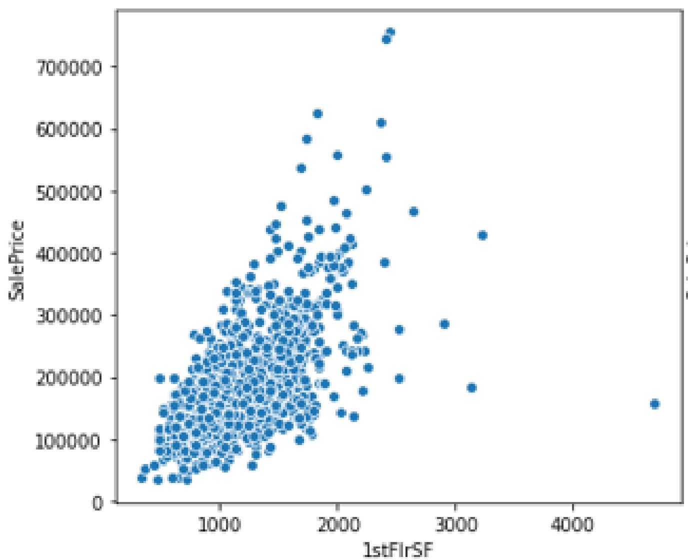
- Basement area does not really affecting the sale price of the houses
- As mostly the basement area of the houses are between 0 to 2000 range

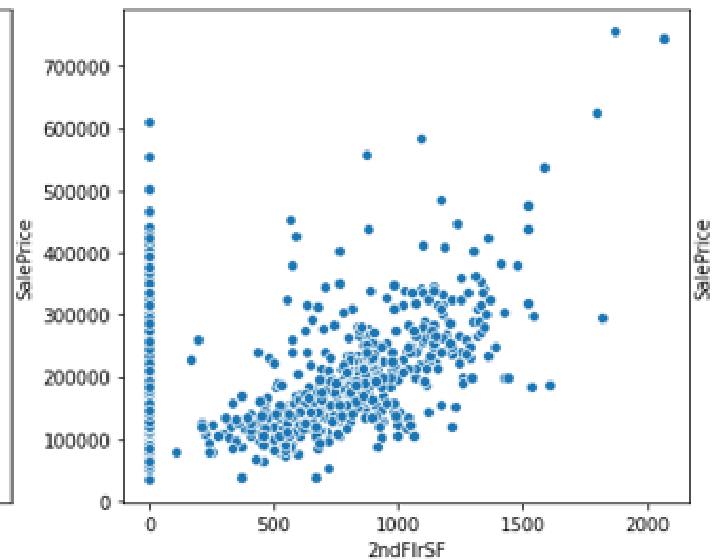
Analysing few continuous variables

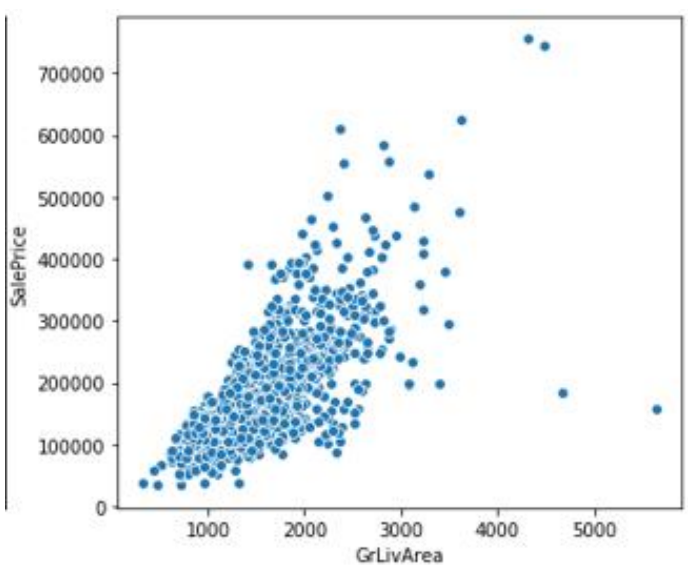


Dropping the column "LowQualFinSF" as it has no value to the data set

Analysing these variables on the basis of Sales price





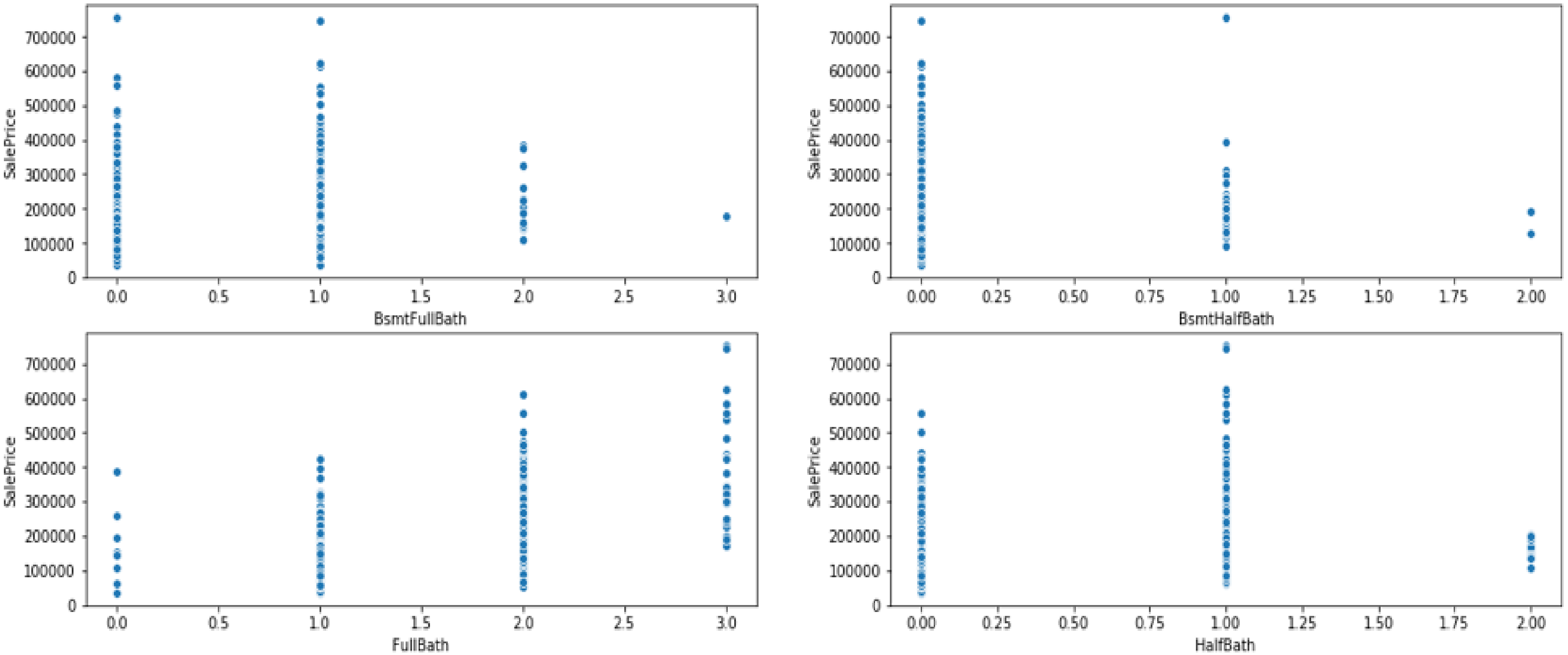


inferences:

- Sale price and the above three variables are linearly related

Analysing the columns with the sale price having following meanings:

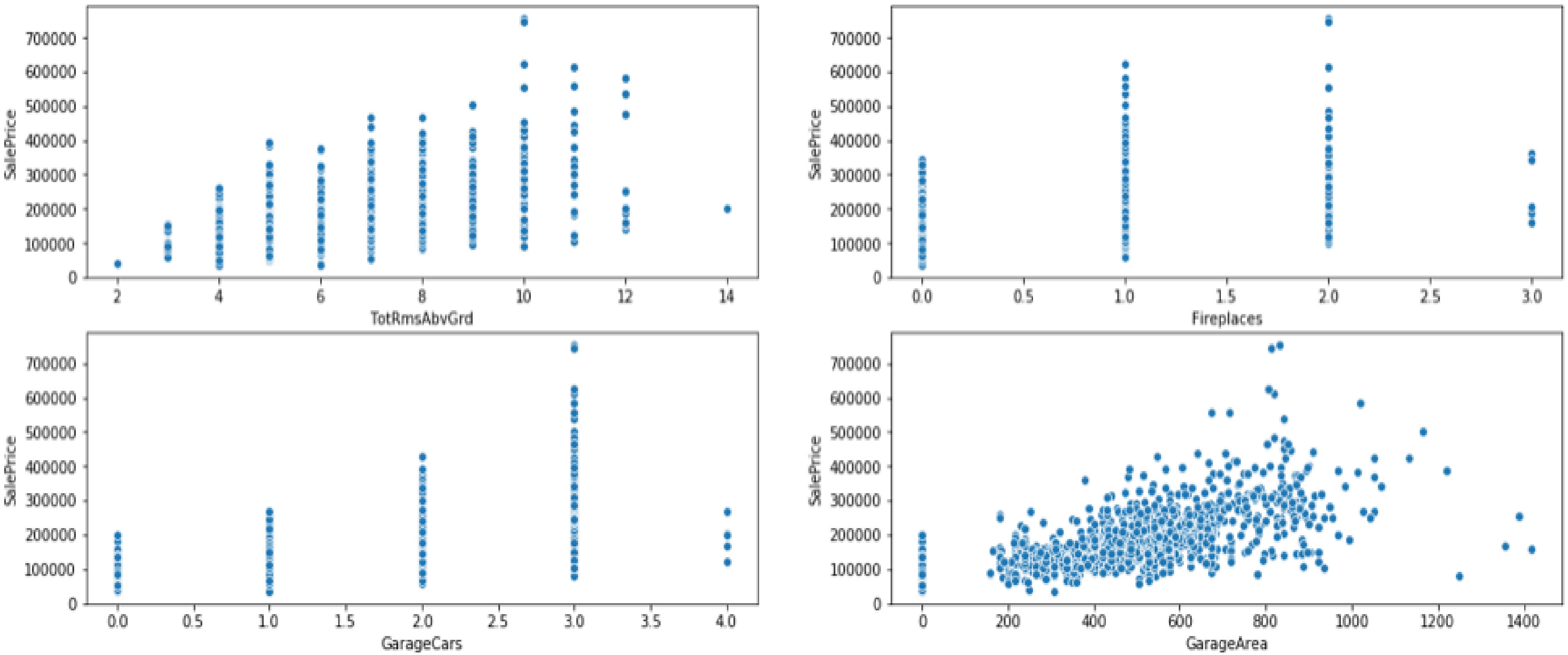
- BsmtFullBath: Basement full bathrooms
- BsmtHalfBath: Basement half bathrooms
- FullBath: Full bathrooms above grade
- HalfBath: Half baths above grade



Dropping columns "BsmtFullBath" and "BsmtHalfBath" as it is adding less value to the data set

Analysing the columns with the sale price having following meanings:

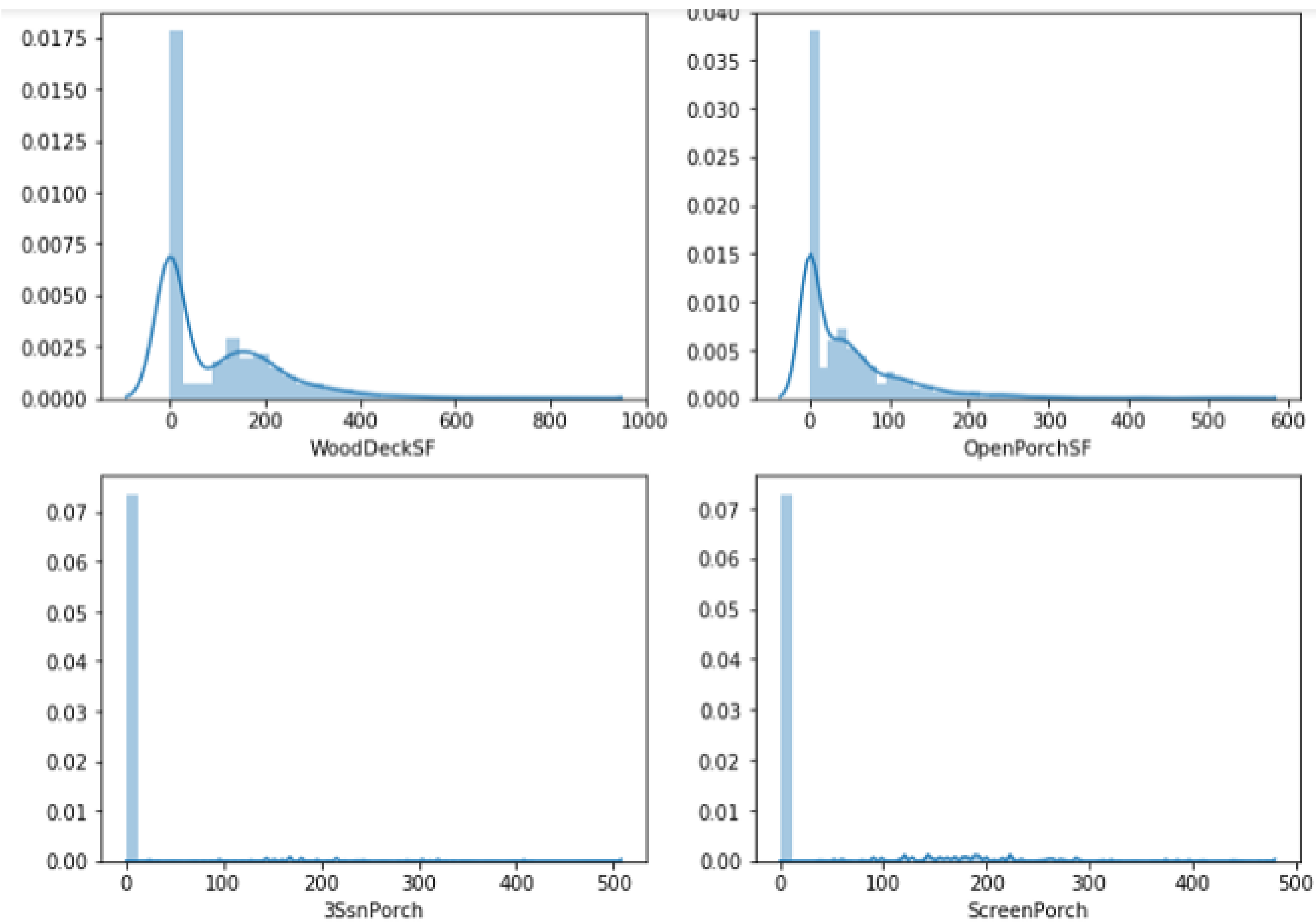
- TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
- Fireplaces: Number of fireplaces
- GarageCars: Size of garage in car capacity
- GarageArea: Size of garage in square feet

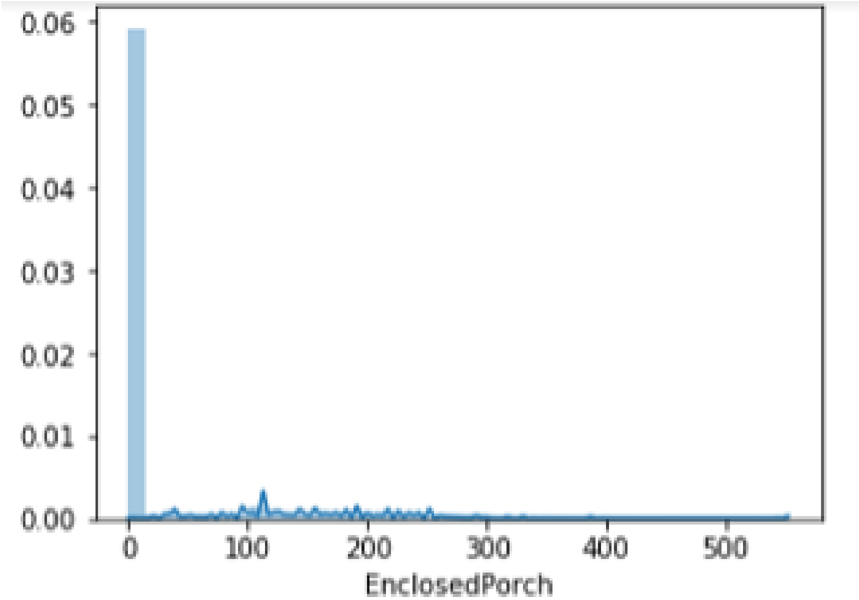


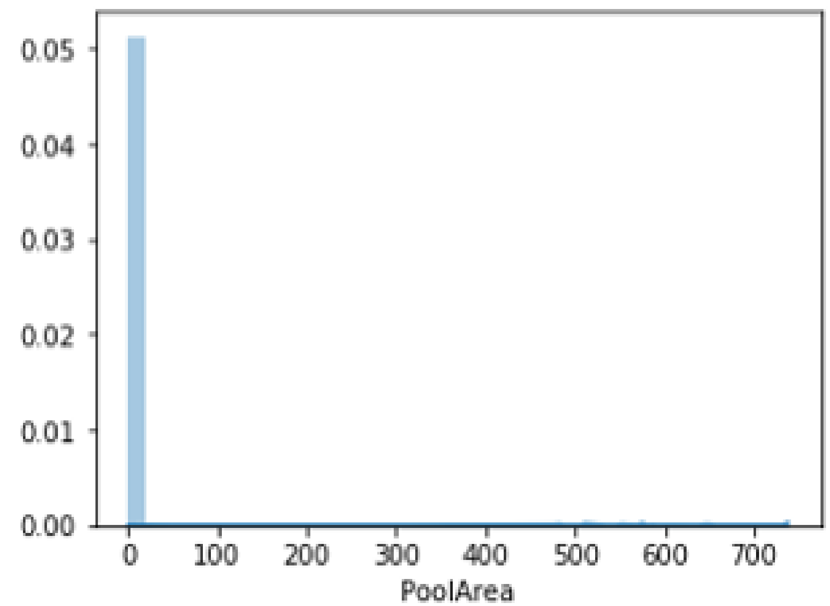
Dropping "Fireplaces" and "GarageCars"

Analysing columns which has following meanings:

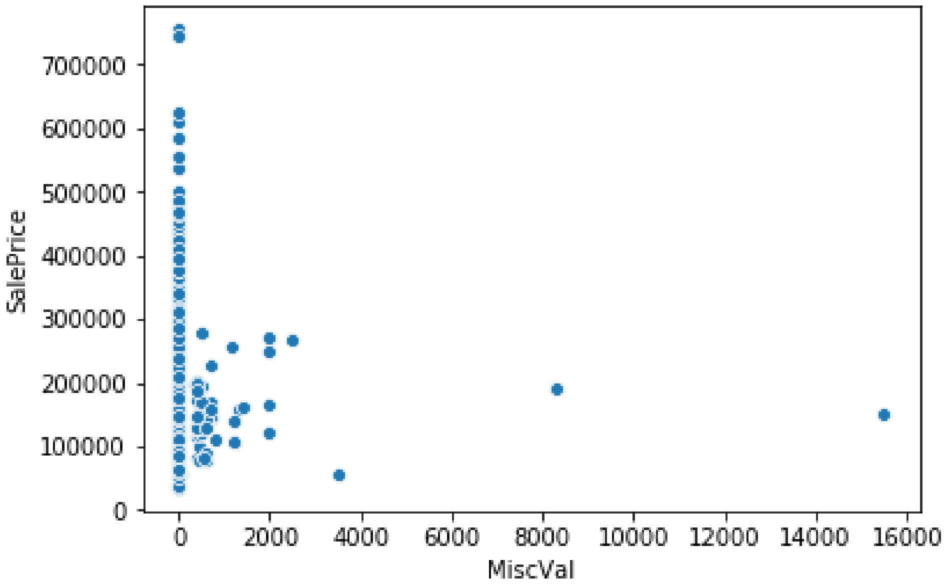
- WoodDeckSF: Wood deck area in square feet
- OpenPorchSF: Open porch area in square feet
- EnclosedPorch: Enclosed porch area in square feet
- 3SsnPorch: Three season porch area in square feet
- ScreenPorch: Screen porch area in square feet
- PoolArea: Pool area in square feet







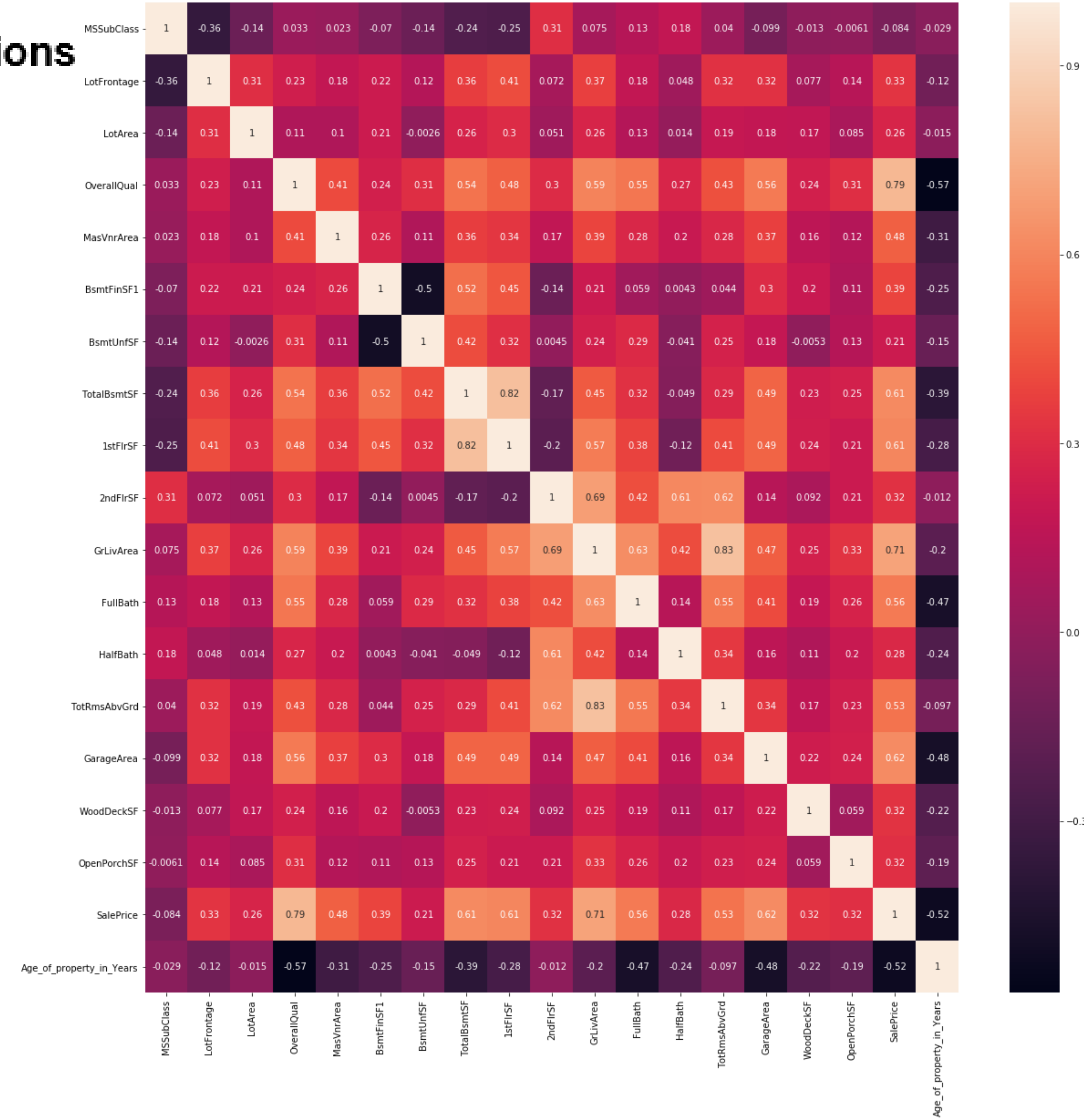
Dropping (EnclosedPorch, 3SsnPorch, ScreenPorch and PoolArea) because of their low variance



Dropping "MiscVal", "Id" and few redundant variables like "BedroomAbvGr" and "KitchenAbvGr"

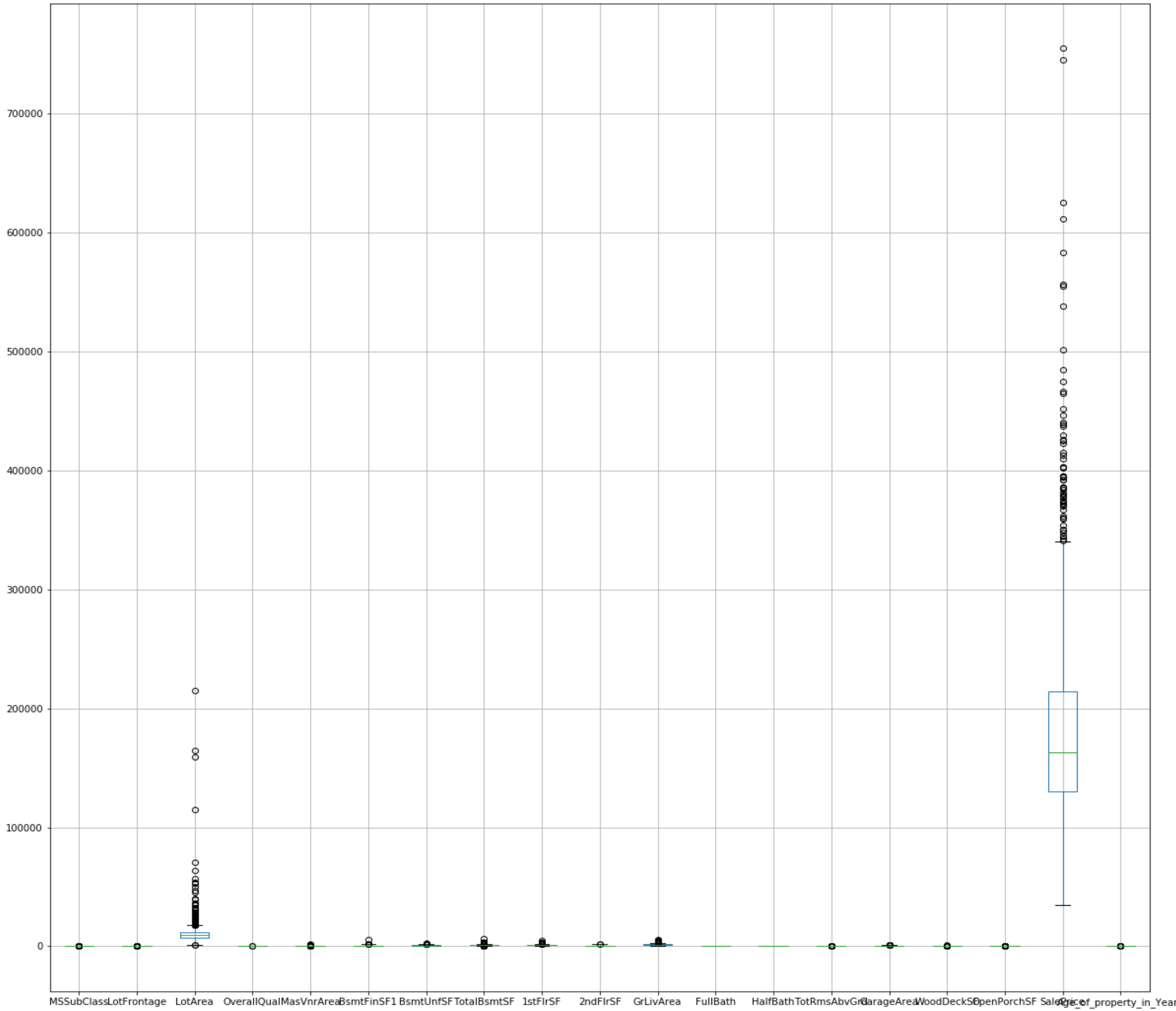
Checking correlations

There seem to be no variables which are highly correlated



Checking outliers

LotArea has few outliers which will not affect our analysis therefore we can keep it and move ahead with that



Analysing non-numerical columns

```
non_numeric_vars = df.select_dtypes(exclude = ['float64','int64'])
non_numeric_vars.head()
```

Finding missing value if any

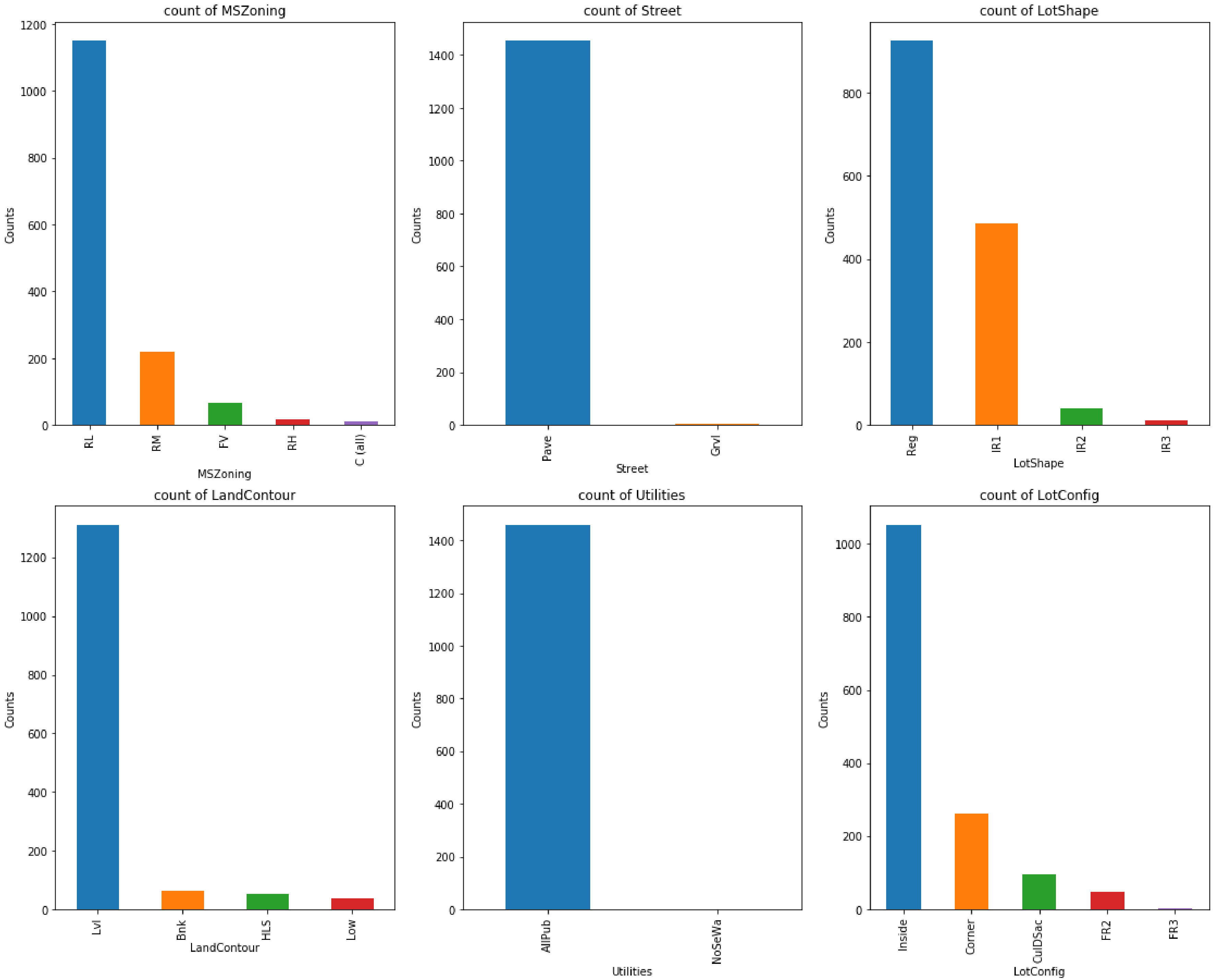
```
MSZoning          0.00
Street            0.00
Alley            93.77
LotShape          0.00
LandContour       0.00
Utilities         0.00
LotConfig         0.00
LandSlope         0.00
Neighborhood      0.00
Condition1        0.00
Condition2        0.00
BldgType          0.00
HouseStyle        0.00
RoofStyle         0.00
RoofMatl          0.00
Exterior1st       0.00
Exterior2nd       0.00
MasVnrType        0.55
ExterQual         0.00
ExterCond         0.00
Foundation        0.00
BsmtQual          2.53
BsmtCond          2.53
BsmtExposure      2.60
BsmtFinType1      2.53
BsmtFinType2      2.60
Heating           0.00
HeatingQC         0.00
CentralAir        0.00
Electrical        0.07
KitchenQual       0.00
Functional        0.00
FireplaceQu      47.26
GarageType        5.55
GarageFinish      5.55
GarageQual        5.55
GarageCond        5.55
PavedDrive        0.00
PoolQC           99.52
Fence            80.75
MiscFeature      96.30
```

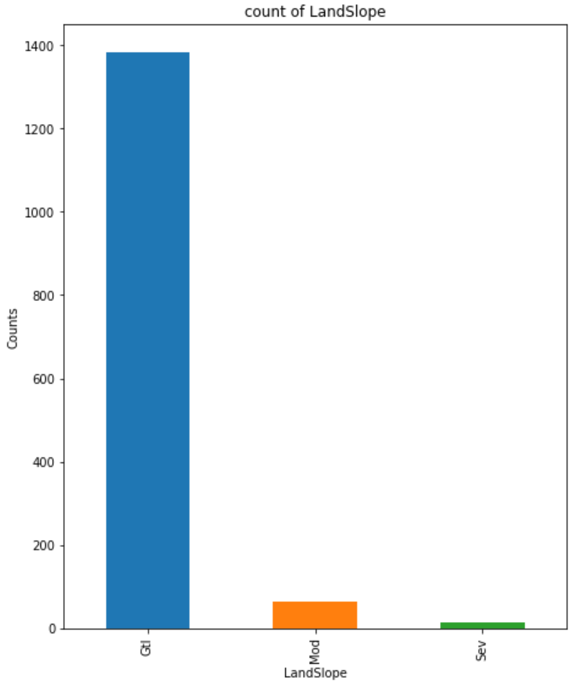
Deleting columns having more than 45% missing data

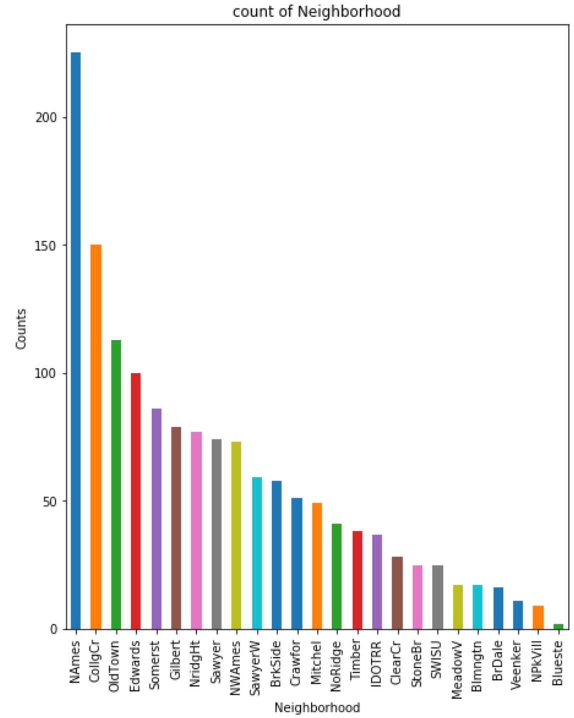
```
non_numeric_vars = non_numeric_vars.drop(['Alley','FireplaceQu','PoolQC','Fence','MiscFeature'], axis = 1)
```

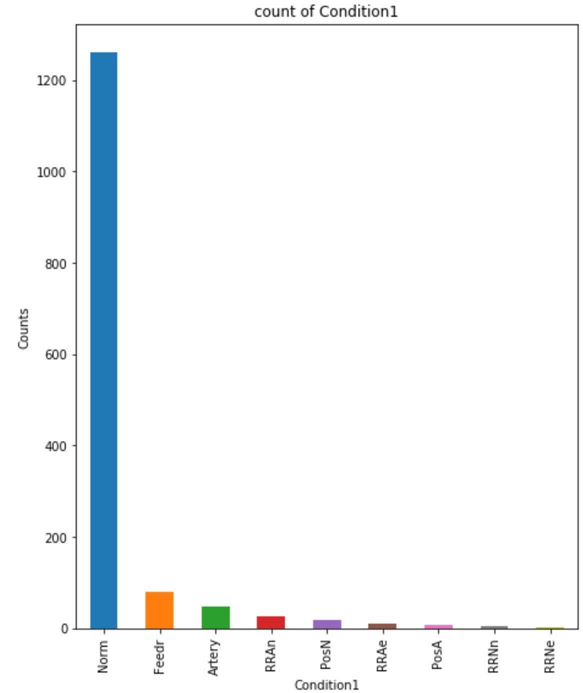
Imputing NaN values with their mode

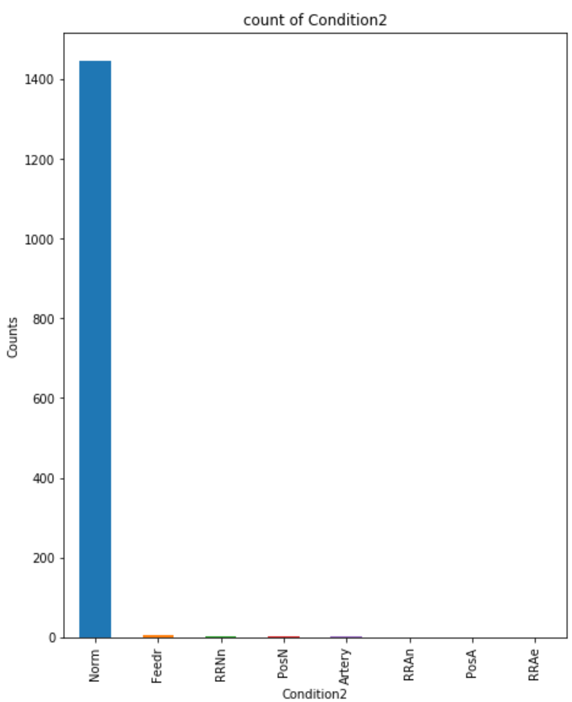
```
temp_vars = ['MasVnrType','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2','Electrical','GarageType',
             'GarageFinish','GarageQual','GarageCond']

for var in temp_vars:
    non_numeric_vars[var] = non_numeric_vars[var].fillna(non_numeric_vars[var].mode()[0])
```

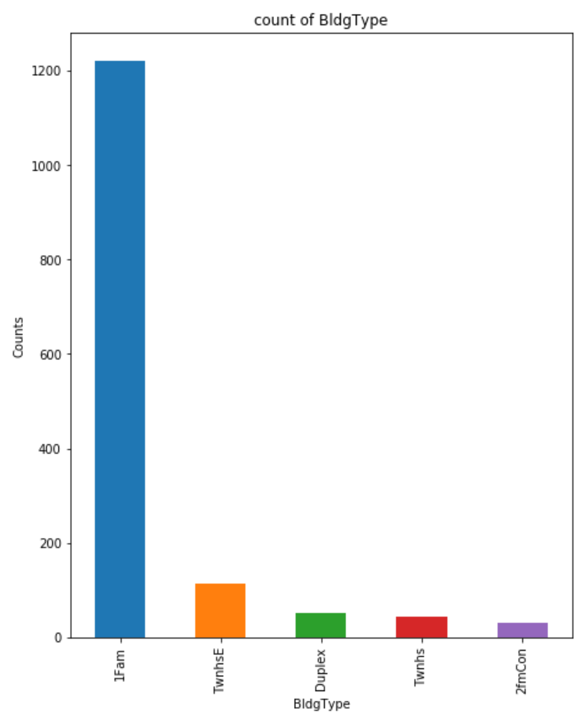

count of MSZoning
Counts
MSZoning
RL
RM
FV
RH
C (all)
count of Street
Counts
Street
Pave
Grvl
count of LotShape
Counts
LotShape
Reg
IR1
IR2
IR3
count of LandContour
Counts
LandContour
Lvl
Bnk
HLS
Low
count of Utilities
Counts
Utilities
AllPub
NoSeWa
count of LotConfig
Counts
LotConfig
Inside
Corner
CulDSac
FR2
FR3

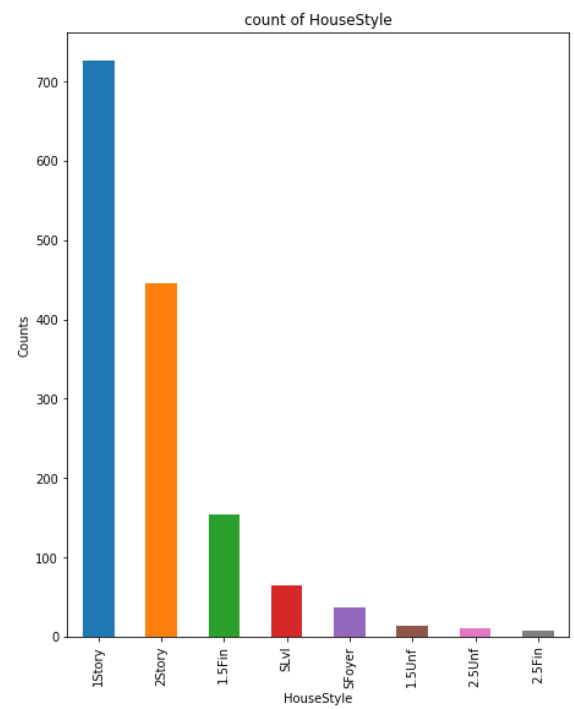

count of LandSlope
Counts
0
200
400
600
800
1000
1200
1400
Gtl
Mod
Sev
LandSlope

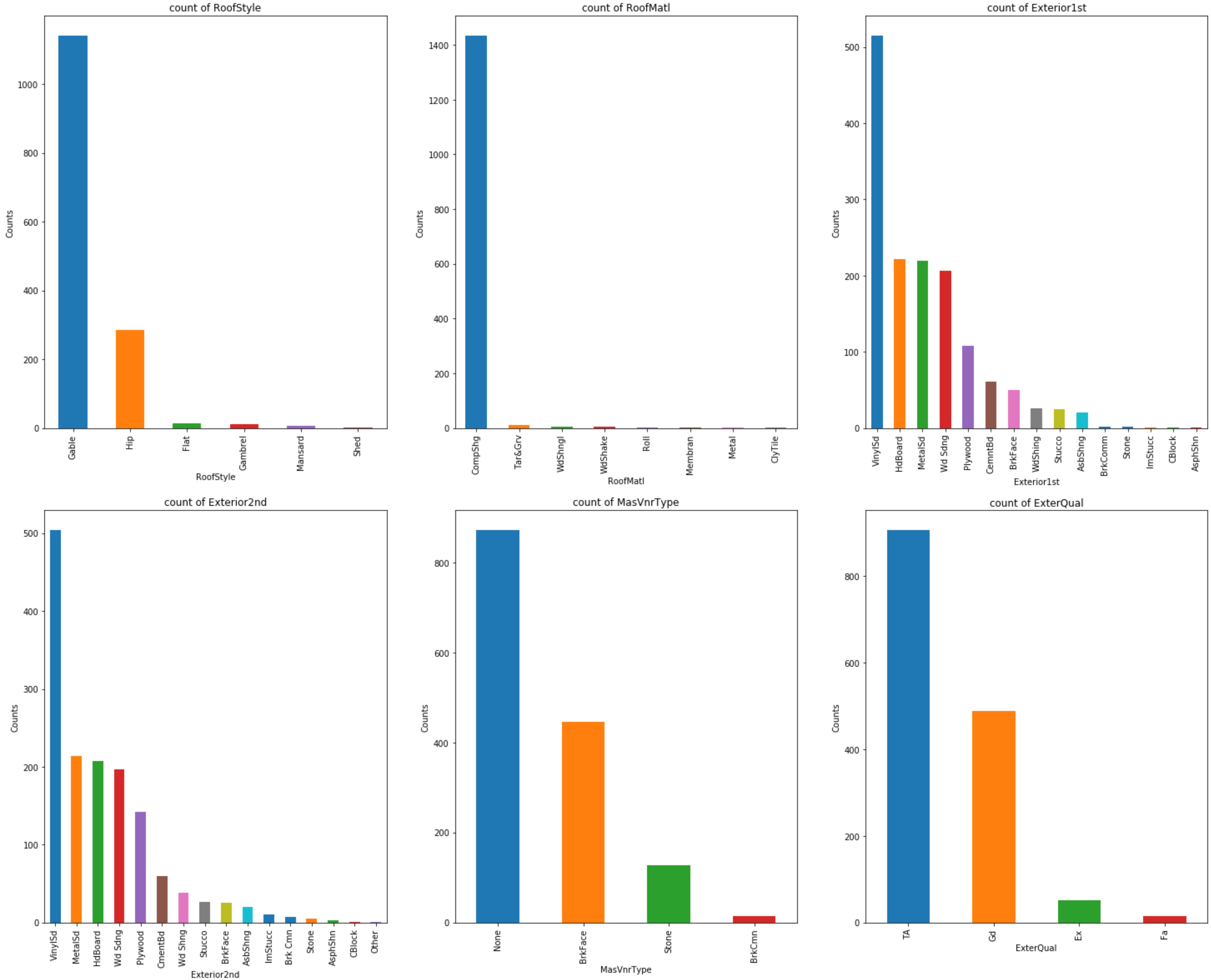

count of Neighborhood
Counts
0
50
100
150
200
NAmes
CollgCr
OldTown
Edwards
Somerst
Gilbert
NridgHt
Sawyer
NWAmes
SawyerW
BrkSide
Crawfor
Mitchel
NoRidge
Timber
IDOTRR
ClearCr
StoneBr
SWISU
MeadowV
Blmngtn
BrDale
Veenker
NPkVill
Blueste
Neighborhood

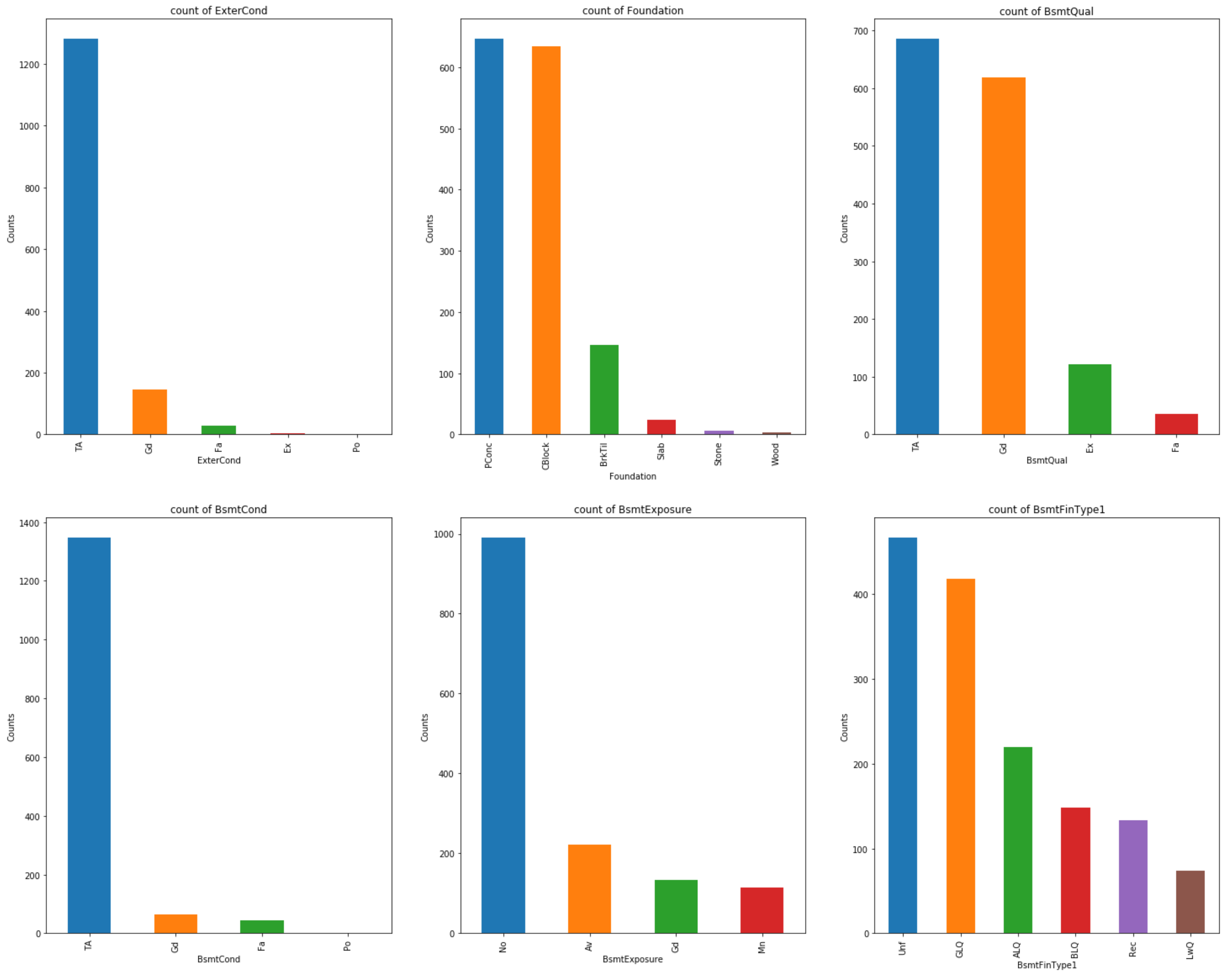

count of Condition1
Counts
0
200
400
600
800
1000
1200
Norm
Feedr
Artery
RRAn
PosN
RRAe
PosA
RRNn
RRNe
Condition1


count of Condition2
Counts
0
200
400
600
800
1000
1200
1400
Norm
Feedr
RRNn
PosN
Artery
RRAn
PosA
RRAe
Condition2

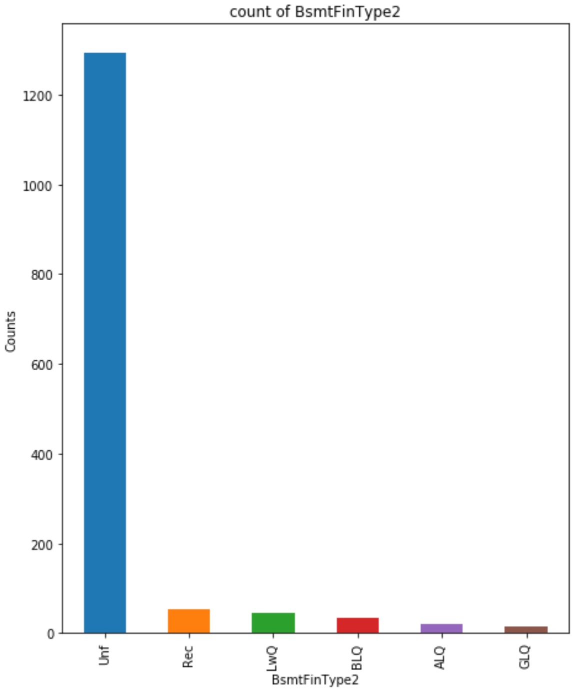

count of BldgType
Counts
0
200
400
600
800
1000
1200
1Fam
TwnhsE
Duplex
Twnhs
2fmCon
BldgType

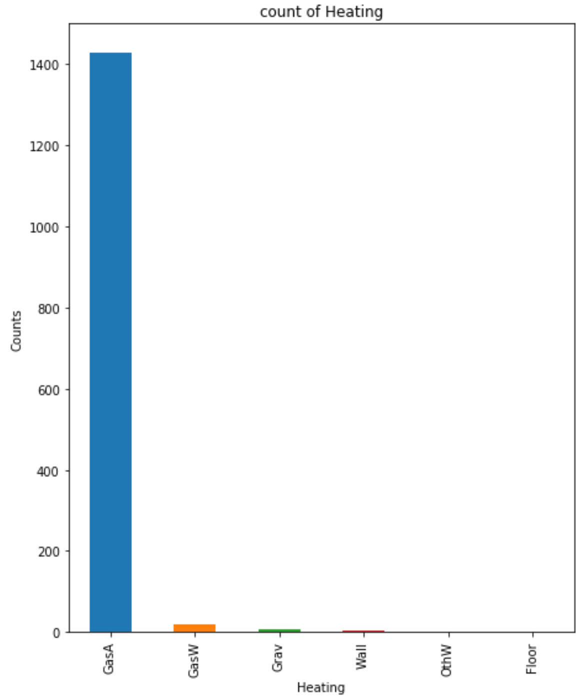

count of HouseStyle
Counts
0
100
200
300
400
500
600
700
1Story
2Story
1.5Fin
SLvl
SFoyer
1.5Unf
2.5Unf
2.5Fin
HouseStyle

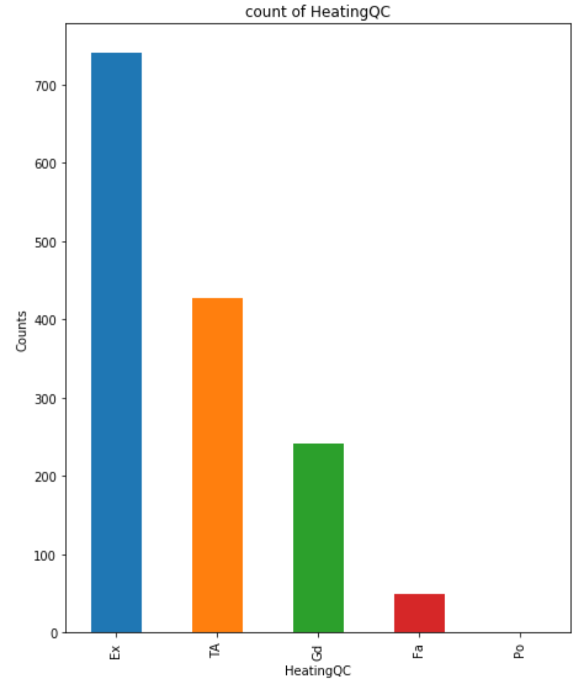

count of RoofStyle
Counts
0
200
400
600
800
1000
Gable
Hip
Flat
Gambrel
Mansard
Shed
RoofStyle
count of RoofMatl
Counts
0
200
400
600
800
1000
1200
1400
CompShg
Tar&Grv
WdShngl
WdShake
Roll
Membran
Metal
ClyTile
RoofMatl
count of Exterior1st
Counts
0
100
200
300
400
500
VinylSd
HdBoard
MetalSd
Wd Sdng
Plywood
CemntBd
BrkFace
WdShing
Stucco
AsbShng
BrkComm
Stone
ImStucc
CBlock
AsphShn
Exterior1st
count of Exterior2nd
Counts
0
100
200
300
400
500
VinylSd
MetalSd
HdBoard
Wd Sdng
Plywood
CmentBd
Wd Shng
Stucco
BrkFace
AsbShng
ImStucc
Brk Cmn
Stone
AsphShn
CBlock
Other
Exterior2nd
count of MasVnrType
Counts
0
200
400
600
800
None
BrkFace
Stone
BrkCmn
MasVnrType
count of ExterQual
Counts
0
200
400
600
800
TA
Gd
Ex
Fa
ExterQual

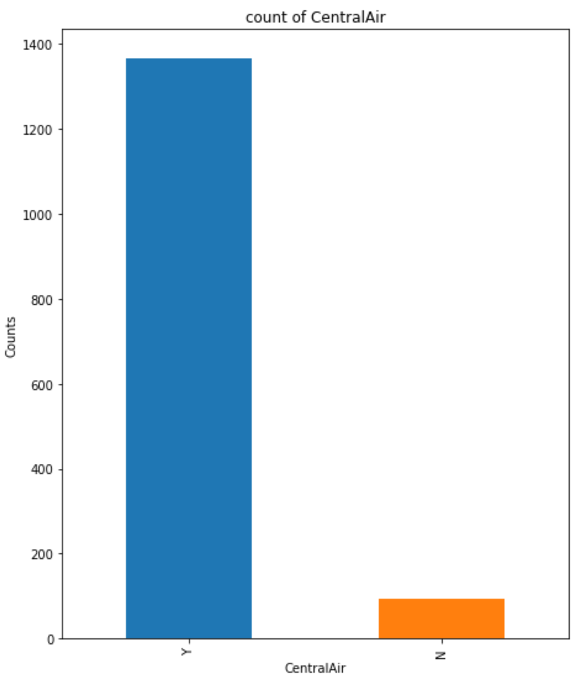


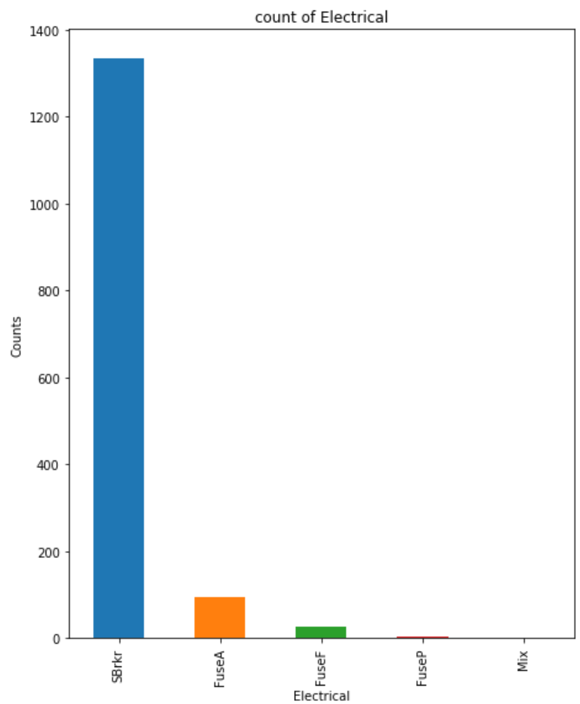
count of ExterCond
Counts
0
200
400
600
800
1000
1200
TA
Gd
Fa
Ex
Po
ExterCond
count of Foundation
Counts
0
100
200
300
400
500
600
PConc
CBlock
BrkTil
Slab
Stone
Wood
Foundation
count of BsmtQual
Counts
0
100
200
300
400
500
600
700
TA
Gd
Ex
Fa
BsmtQual
count of BsmtCond
Counts
0
200
400
600
800
1000
1200
1400
TA
Gd
Fa
Po
BsmtCond
count of BsmtExposure
Counts
0
200
400
600
800
1000
No
Av
Gd
Mn
BsmtExposure
count of BsmtFinType1
Counts
0
100
200
300
400
Unf
GLQ
ALQ
BLQ
Rec
LwQ
BsmtFinType1

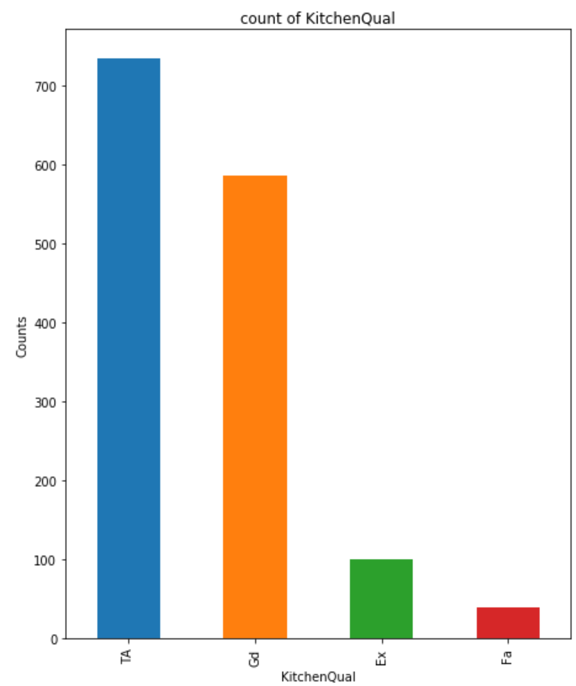

count of BsmtFinType2
Counts
0
200
400
600
800
1000
1200
Unf
Rec
LwQ
BLQ
ALQ
GLQ
BsmtFinType2

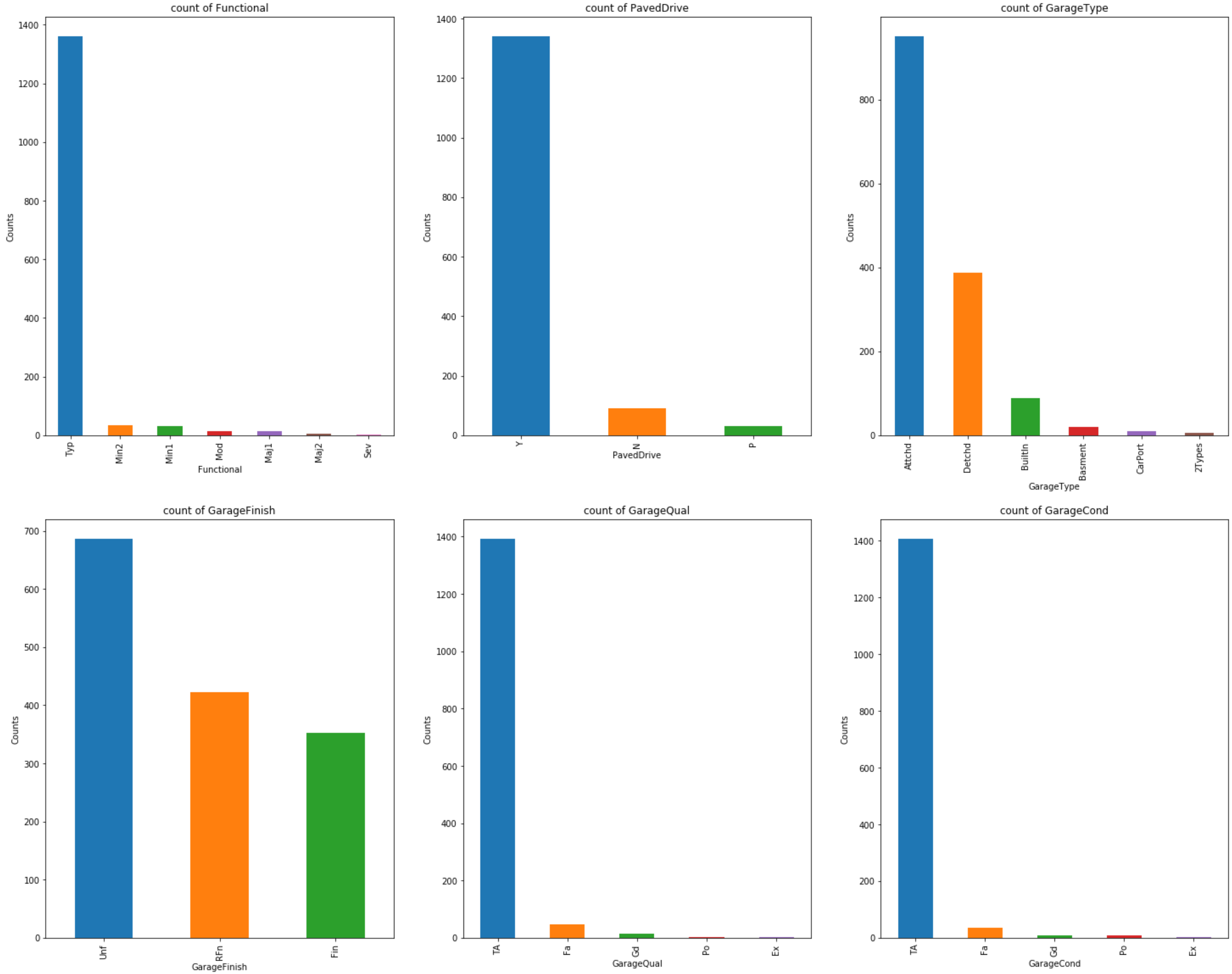

count of Heating
Counts
0
200
400
600
800
1000
1200
1400
GasA
GasW
Grav
Wall
OthW
Floor
Heating


count of HeatingQC
Counts
0
100
200
300
400
500
600
700
Ex
TA
Gd
Fa
Po
HeatingQC


count of CentralAir
Counts
0
200
400
600
800
1000
1200
1400
Y
N
CentralAir


count of Electrical
Counts
0
200
400
600
800
1000
1200
1400
SBrkr
FuseA
FuseF
FuseP
Mix
Electrical


count of KitchenQual
Counts
0
100
200
300
400
500
600
700
TA
Gd
Ex
Fa
KitchenQual



count of Functional
Counts
0
200
400
600
800
1000
1200
1400
Typ
Min2
Min1
Mod
Maj1
Maj2
Sev
Functional
count of PavedDrive
Counts
0
200
400
600
800
1000
1200
1400
Y
N
P
PavedDrive
count of GarageType
Counts
0
200
400
600
800
Attchd
Detchd
BuiltIn
Basment
CarPort
2Types
GarageType
count of GarageFinish
Counts
0
100
200
300
400
500
600
700
Unf
RFn
Fin
GarageFinish
count of GarageQual
Counts
0
200
400
600
800
1000
1200
1400
TA
Fa
Gd
Po
Ex
GarageQual
count of GarageCond
Counts
0
200
400
600
800
1000
1200
1400
TA
Fa
Gd
Po
Ex
GarageCond

inferences:

- MSzoning is mostly Residential Low Density(RL)
- Streets are generally Paved
- LotShape are Regular
- LandContour are nearly Flat or Leveled
- All Public Utilities are present
- Mostly it is Inside LotConfig

- Slopes are gentle
- Neighborhood are North Ames, College Creek and Old Town
- Conditions are generally normal
- Building type are mostly Single-family Detached
- House Styles are generally 1 story and 2 story

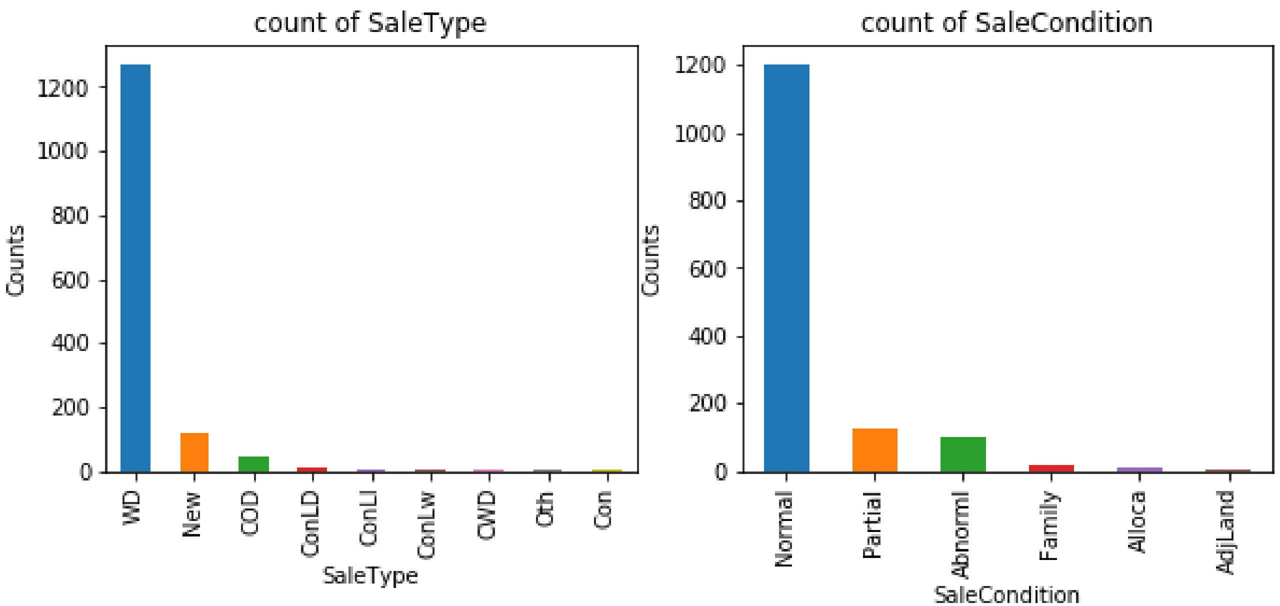
- Most of the Roof Style are Gable
- Exterior covering are mostly Vinyl Siding
- Exterior quality is average

- Exterior condition is average
- Type of Foundation are Poured Contrete and Cinder Block
- Basement condition are average
- There is no any basement exposure

- Basements are generally unfinished
- Heating is Gas forced warm air furnace type
- Heating quality and condition are excellent
- Central air conditioning is present
- Standard Circuit Breakers & Romex electrical system
- Kitchen quality is average

- Paved driveway is present
- Garage has average quality and are attached but unfinished

Dropping few redundant/unnecessay variables like "Functional" and "GarageQual"



Step 3: Data Preparation

Merging numerical and non-numerical columns

```
housing = numeric_vars.join(non_numeric_vars)
```

```
housing.columns
```

```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'MasVnrArea',
       'BsmtFinSF1', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF',
       'GrLivArea', 'FullBath', 'HalfBath', 'TotRmsAbvGrd', 'GarageArea',
       'WoodDeckSF', 'OpenPorchSF', 'SalePrice', 'Age_of_property_in_Years',
       'MSZoning', 'Street', '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',
       'GarageType', 'GarageFinish', 'GarageCond', 'PavedDrive', 'SaleType',
       'SaleCondition'],
      dtype='object')
```

Separating housing data set into independent (X) and dependent (y) variables

```
X = housing.drop('SalePrice',1)
X.head()
```

	MSSubClass	LotFrontage	LotArea	OverallQual	MasVnrArea	BsmtFinSF1	BsmtUnfSF	TotalBsmtSF	1stFlrSF	2ndFlrSF	...	HeatingQC	CentralAir	Electrica
0	60	65.0	8450	7	196.0	706	150	856	856	854	...	Ex	Y	SBrl
1	20	80.0	9600	6	0.0	978	284	1262	1262	0	...	Ex	Y	SBrl
2	60	68.0	11250	7	162.0	486	434	920	920	866	...	Ex	Y	SBrl
3	70	60.0	9550	7	0.0	216	540	756	961	756	...	Gd	Y	SBrl
4	60	84.0	14260	8	350.0	655	490	1145	1145	1053	...	Ex	Y	SBrl

5 rows × 54 columns

```
y = housing['SalePrice']
y.head()
```

```
0    208500
1    181500
2    223500
3    140000
4    250000
Name: SalePrice, dtype: int64
```

Creating dummy variables for categorical data

converting categorical variables (MSSubClass, OverallQual, FullBath, HalfBath, TotRmsAbvGrd) to object type for creating dummy variables

```
X[['MSSubClass','OverallQual','FullBath','HalfBath','TotRmsAbvGrd']] = X[['MSSubClass','OverallQual','FullBath','HalfBath','TotRm
```

```
# convert into dummies
housing_dummies = pd.get_dummies(X_categorical, drop_first=True)
housing_dummies.head()
```

	MSZoning_FV	MSZoning_RH	MSZoning_RL	MSZoning_RM	Street_Pave	LotShape_IR2	LotShape_IR3	LotShape_Reg	LandContour_HLS	LandContour_Low
0	0	0	1	0	1	0	0	1	0	0
1	0	0	1	0	1	0	0	1	0	0
2	0	0	1	0	1	0	0	0	0	0
3	0	0	1	0	1	0	0	0	0	0
4	0	0	1	0	1	0	0	0	0	0

5 rows × 225 columns

Deleting variables from "X" data frame having more than 80% correlation

```
['1stFlrSF',
 'MSZoning_RM',
 'Neighborhood_Somerst',
 'HouseStyle_2Story',
 'RoofStyle_Hip',
 'Exterior2nd_CBlock',
 'Exterior2nd_CmentBd',
 'Exterior2nd_HdBoard',
 'Exterior2nd_MetalSd',
 'Exterior2nd_VinylSd',
 'Exterior2nd_Wd Sdng',
 'MasVnrType_None',
 'ExterQual_TA',
 'ExterCond_TA',
 'BsmtQual_TA',
 'KitchenQual_TA',
 'GarageType_Detchd',
 'GarageCond_TA',
 'SaleCondition_Partial',
 'MSSubClass_45',
 'MSSubClass_80',
 'MSSubClass_90',
 'MSSubClass_190',
 'FullBath_2']
```

Variables to drop

Feature Scaling

```
# scaling the features
from sklearn.preprocessing import scale

# storing the column names of X dataframe in cols
cols = X.columns
X = pd.DataFrame(scale(X))
X.columns = cols
X.columns
```

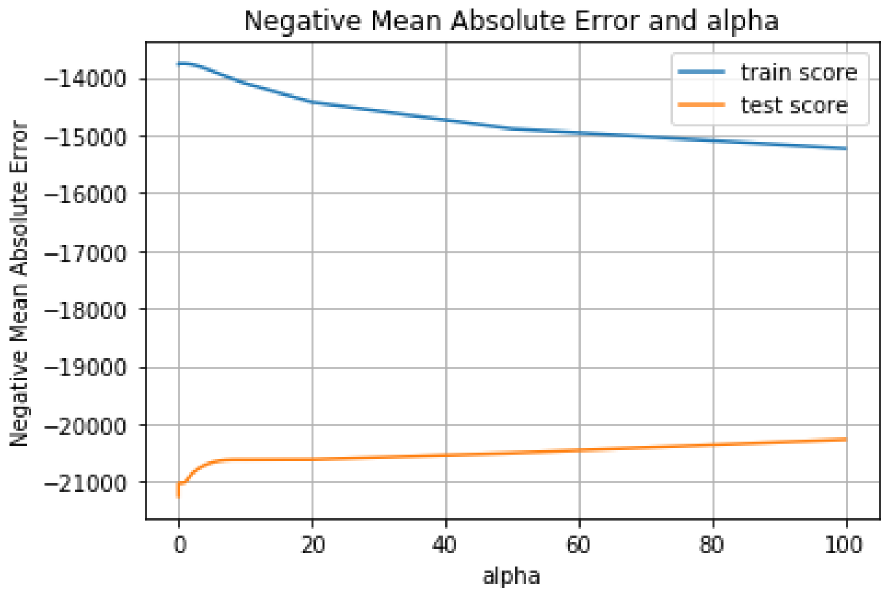
```
Index(['LotFrontage', 'LotArea', 'MasVnrArea', 'BsmtFinSF1', 'BsmtUnfSF',
       'TotalBsmtSF', '2ndFlrSF', 'GrLivArea', 'GarageArea', 'WoodDeckSF',
       ...
       'TotRmsAbvGrd_4', 'TotRmsAbvGrd_5', 'TotRmsAbvGrd_6', 'TotRmsAbvGrd_7',
       'TotRmsAbvGrd_8', 'TotRmsAbvGrd_9', 'TotRmsAbvGrd_10',
       'TotRmsAbvGrd_11', 'TotRmsAbvGrd_12', 'TotRmsAbvGrd_14'],
      dtype='object', length=214)
```

Splitting data set into train and test set

```
# split data set into test and train
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.8, test_size = 0.2, random_state = 100)
```

Step 4: Model Building and Evaluation

Ridge regression or L2 regularization



Finding optimum value of alpha for Ridge regression

choosing alpha = 20 as the optimum value, this is the place where train and test error are not much

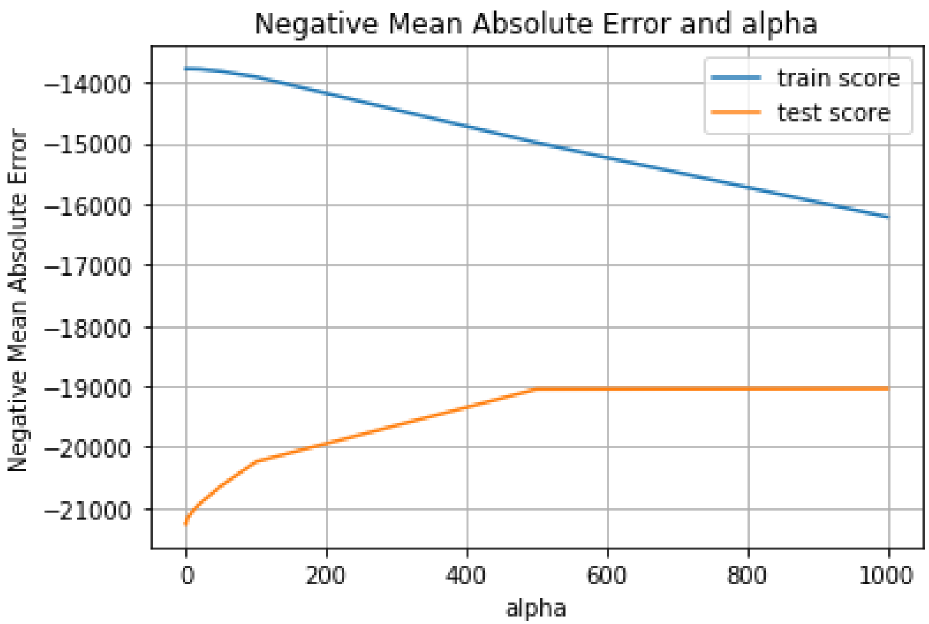
after alpha = 20, the "Negative Mean Absolute Error" is increasing gradually ¶

Metrics (r2_score)

```
# model with optimal alpha
# Ridge regression
from sklearn import metrics
lm = Ridge(alpha=20)
lm.fit(X_train, y_train)

# predict
y_train_pred = lm.predict(X_train)
print(metrics.r2_score(y_true=y_train, y_pred=y_train_pred))
y_test_pred = lm.predict(X_test)
print(metrics.r2_score(y_true=y_test, y_pred=y_test_pred))
```

```
0.9220215210903048
0.8276104632713882
```

Lasso Regression or L1 regularization



Finding optimum value of alpha for Lasso Regresion

choosing alpha = 500 as here the "Negative Mean Absolute Error" is minimum for the test train and also error is not increasing much for the train set

Step 5: Question-Answer

Question 1: Which variables are significant in predicting the price of a house?

According to Lasso regression:

some of the top variables help in predicting the price of the houses are:

1. Condition2
2. Age_of_property_in_Years
3. MSSubClass
4. BsmtFinType2
5. HeatingQC
6. RoofMatl
7. GrLivArea
8. OverallQual
9. SaleType
10. TotalBsmtSF

According to Ridge regression:

some of the top variables help in predicting the price of the houses are:

1. Condition2
2. Age_of_property_in_Years
3. MSSubClass
4. BsmtFinType2
5. HeatingQC
6. RoofMatl
7. GrLivArea
8. OverallQual
9. SaleType
10. TotalBsmtSF

Question 2: How well those variables describe the price of a house?

The coefficients involved for the variables are :

- Condition2_PosN = -17032.076
- Age_of_property_in_Years = -8914.617
- MSSubClass_160 = -3746.124
- BsmtFinType2_Unf = -2838.422
- HeatingQC_TA = -2834.721
- RoofMatl_CompShg = 78180.62
- GrLivArea = 28156.106
- OverallQual_9 = 13730.175
- SaleType_New = 8708.97
- TotalBsmtSF = 8044.272

Question 3: Determine the optimal value of lambda for ridge and lasso regression

For Ridge regression the optimal value of lambda is 20

For Lasso regression the optimal value of lambda is 500