```
# Core
import pandas as pd
import numpy as np

# Data Visualisation
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```
House_Data=pd.read_csv('train.csv')
```

```
#target varriable
target = House_Data.SalePrice
```

```
House_Data.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	E
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	Norm	Norm	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	Norm	Norm	

5 rows × 81 columns

```
House_Data.shape
```

```
(1460, 81)
```

```
# check how many missings we have per feature
missings = House_Data.isnull().sum().to_frame()

# give column the name nMissings
missings.columns = ['nMissings']

# make new columns that gives information about missing percentage
missings['percMissing'] = missings['nMissings']/1460

# sort values by nMissings values
ordered_missings = missings.sort_values(by = ['nMissings'], ascending=False)
ordered_missings
```

	nMissings	percMissing
PoolQC	1453	0.995205
MiscFeature	1406	0.963014
Alley	1369	0.937671
Fence	1179	0.807534
FireplaceQu	690	0.472603
...	...	...
ExterQual	0	0.000000
Exterior2nd	0	0.000000
Exterior1st	0	0.000000
RoofMatl	0	0.000000
SalePrice	0	0.000000

81 rows × 2 columns

```
# List of numerical attributes
House_Data.select_dtypes(exclude=['object']).columns
```

```
Index(['Id', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual',
       'OverallCond', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1',
       'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd',
       'Fireplaces', 'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF',
       'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea',
       'MiscVal', 'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

```
len(House_Data.select_dtypes(exclude='object').columns)
```

```
38
```

```
#Description of all numerical varriables
House_Data.select_dtypes(exclude=['object']).describe().round(decimals=2)
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF
count	1460.00	1460.0	1201.00	1460.00	1460.00	1460.00	1460.00	1460.00	1452.00	1460.00	1460.00	1460.00	1460.00
mean	730.50	56.9	70.05	10516.83	6.10	5.58	1971.27	1984.87	103.69	443.64	46.55	567.24	1057.43
std	421.61	42.3	24.28	9981.26	1.38	1.11	30.20	20.65	181.07	456.10	161.32	441.87	438.71
min	1.00	20.0	21.00	1300.00	1.00	1.00	1872.00	1950.00	0.00	0.00	0.00	0.00	0.00
25%	365.75	20.0	59.00	7553.50	5.00	5.00	1954.00	1967.00	0.00	0.00	0.00	223.00	795.75
50%	730.50	50.0	69.00	9478.50	6.00	5.00	1973.00	1994.00	0.00	383.50	0.00	477.50	991.50
75%	1095.25	70.0	80.00	11601.50	7.00	6.00	2000.00	2004.00	166.00	712.25	0.00	808.00	1298.25

```
#Categorical columns within the dataset
House_Data.select_dtypes(include=['object']).columns
```

```
Index(['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'],
      dtype='object')
```

```
len(House_Data.select_dtypes(include='object').columns)
```

```
43
```

```
House_Data.select_dtypes(include=['object']).describe()
```

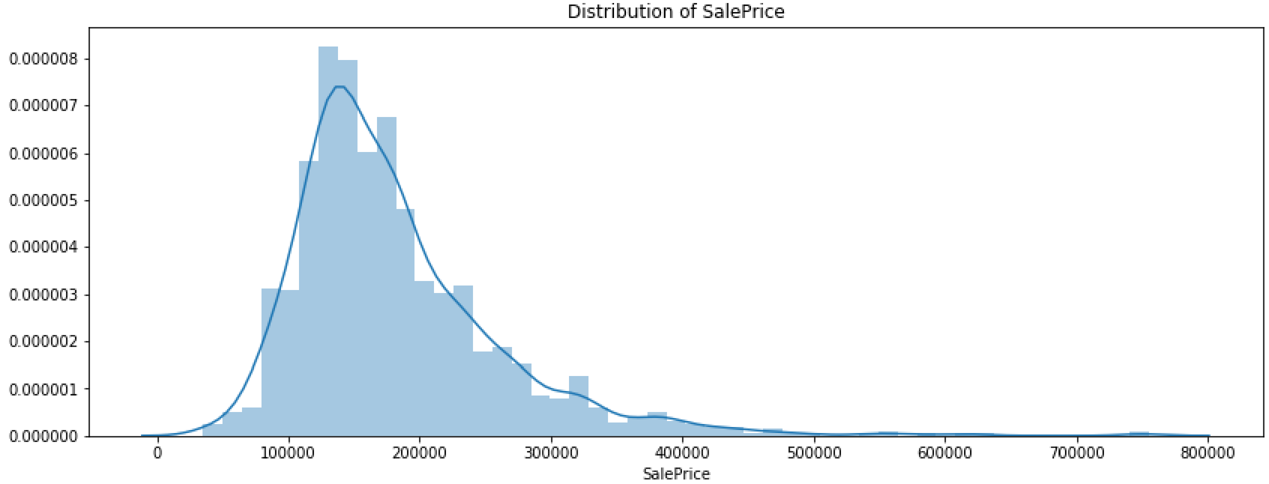
	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	HouseStyle	RoofStyle	R
count	1460	1460	91	1460	1460	1460	1460	1460	1460	1460	1460	1460	1460	1460	
unique	5	2	2	4	4	2	5	3	25	9	8	5	8	6	
top	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	Gtl	NAmes	Norm	Norm	1Fam	1Story	Gable	
freq	1151	1454	50	925	1311	1459	1052	1382	225	1260	1445	1220	726	1141	

```
#investigate the target varriable(House Prices)
print(House_Data['SalePrice'].describe())
print("median  ",House_Data['SalePrice'].median())
print("Number of missings:", House_Data['SalePrice'].isna().sum())
```

```
count      1460.000000
mean     180921.195890
std       79442.502883
min       34900.000000
25%      129975.000000
50%      163000.000000
75%      214000.000000
max      755000.000000
Name: SalePrice, dtype: float64
median   163000.0
```

The avarage saleprice for which a house was sold was 180.921 dollars, with a minimum of 34.900 and a maximum of 755.000 dollars. The median saleprice was 163.000 dollars.

```
# visualizing the distribution of target varriable(house prices)
target = House_Data.SalePrice
plt.figure(figsize=(14, 5))
sns.distplot(target)
plt.title('Distribution of SalePrice')
plt.show()
```



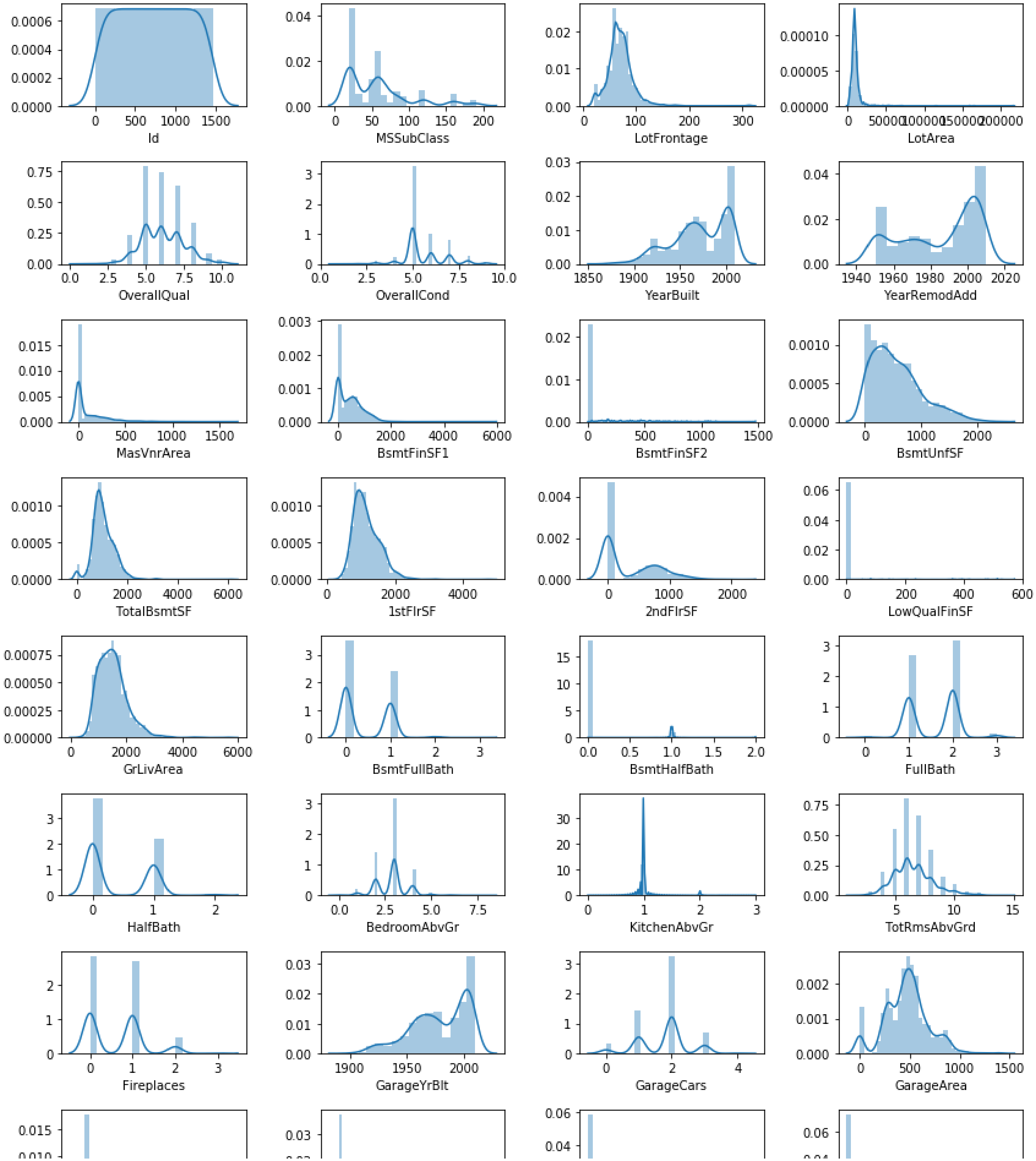

```
#skewness and kurtosis
print("Skewness: %f" % House_Data['SalePrice'].skew())
print("Kurtosis: %f" % House_Data['SalePrice'].kurt())
```

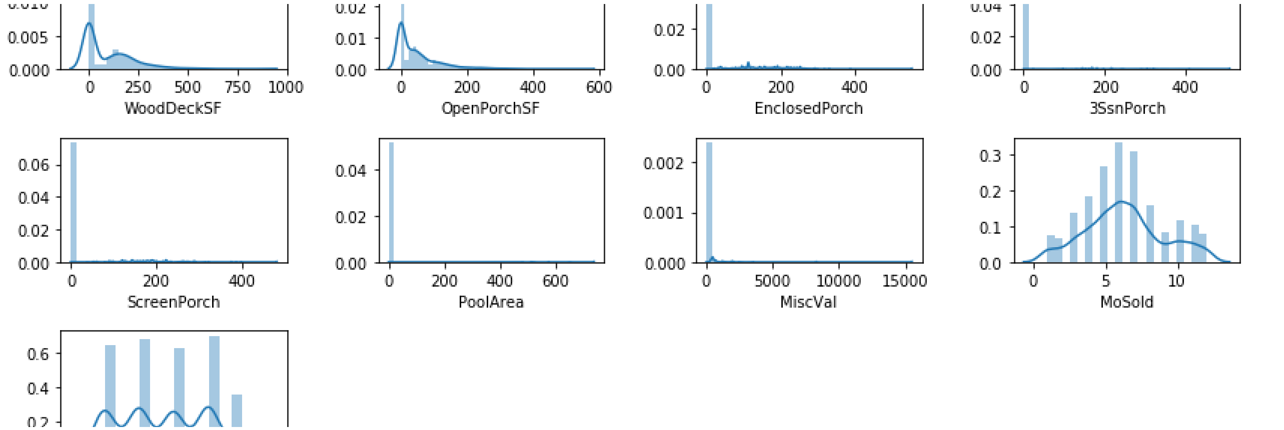
```
Skewness: 1.882876
Kurtosis: 6.536282
```

The plot above shows that the SalePrice has a right skewed distribution and a high kurtosis which means most of the data falls to the higher range values.However When the log is taken we get a normal distribution. Since a lot of machine learning models assume that they are fed normally distributed data, I think we should give them normally distributed data.

Considering the distributions pof other features in questions.features like BsmtUnfSf,GrlVArea,BsmtSFinSF1,1StFlrSf data most lie to the positive side of the distribution.YearBlt and GarageYearBlt are left skewed this makes sence since most garage will be built the same time the house is built.some features like LotFrontage,LotArea,MasVnrArea,GrlvArea shows posibility of outliers this could be confirmed using boxplot or scatterplots.Also features like MSSubClass,OverallCond,BsmtFullBath,BsmtHalfBath,FullBath,HalfBath,FirePlaces,GarageCars.i suspects the individual features values of the mention features are not linear with the target SalesPrice.The above feature could be divided into buckets(binning)to learn something different about housing Price values for each bucket individual buckets.example we ould check how group/categories(buckets) of BsmtFullBath,FullBath or both affects Salesprice.So those features will automatically threated as categorical varriables. We could generally view correlations of the dataset in threee categories. 1.Rightly skewed features 2.left skewed features 3.Bucketing(binning)

```
#Distributions of attributes
num_attributes = House_Data.select_dtypes(exclude='object').drop('SalePrice', axis=1).copy()

fig = plt.figure(figsize=(12,18))
for i in range(len(num_attributes.columns)):
    fig.add_subplot(10,4,i+1)
    sns.distplot(num_attributes.iloc[:,i].dropna())
    plt.xlabel(num_attributes.columns[i])

plt.tight_layout()
plt.show()
```

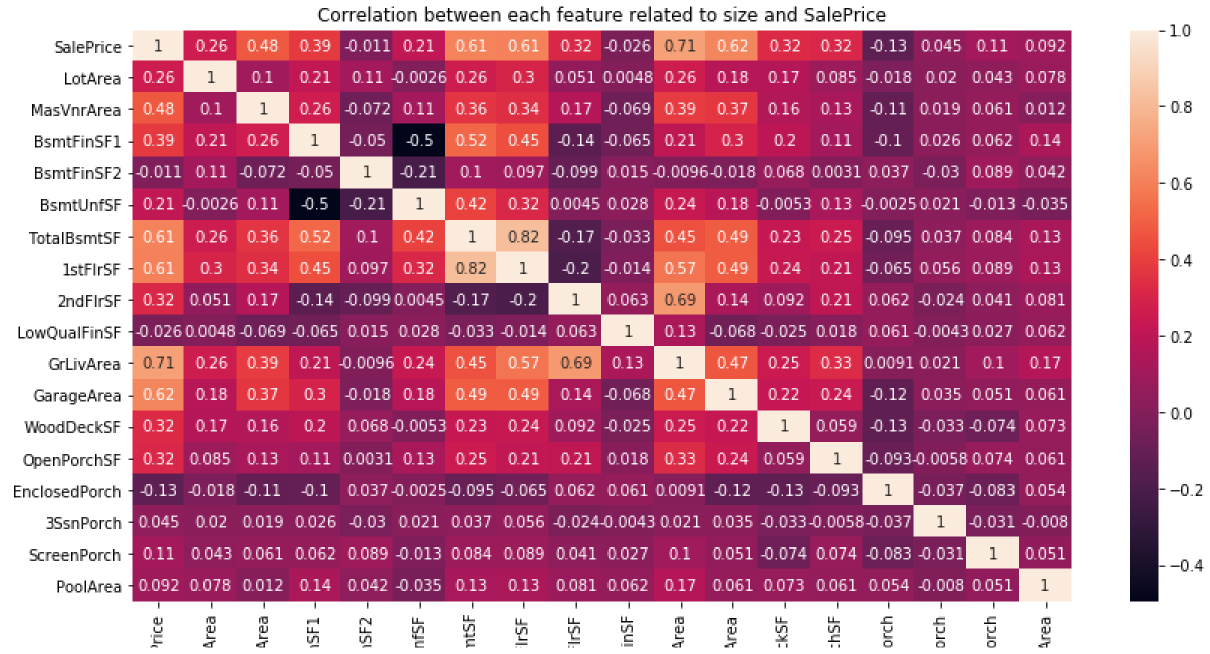

Id
MSSubClass
LotFrontage
LotArea
OverallQual
OverallCond
YearBuilt
YearRemodAdd
MasVnrArea
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageYrBlt
GarageCars
GarageArea



Lets check how numeric features are related to SalePrice. We start with the features that represent size. First we calculate the correlation coefficients between all these size related features and SalePrice, next these values are visualized in a heatmap.

```
# How are numeric features related to SalePrice
# select target variable SalePrice and features related to size
num_cols = ['SalePrice', 'LotArea', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF',
      'LowQualFinSF', 'GrLivArea', 'GarageArea', 'WoodDeckSF',
      'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea']

# get the correlation coefficients between these features.
num_corr =House_Data[num_cols].corr()


# Set the width and height of the figure
plt.figure(figsize=(14,7))

# Add title
plt.title("Correlation between each feature related to size and SalePrice")

# Heatmap showing average arrival delay for each airline by month
sns.heatmap(data=num_corr, annot=True)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fca9409ca58>




```
#Sort correlation with sale price in order and display
num_corr.sort_values(['SalePrice'], ascending=False, inplace=True)
print(num_corr.SalePrice)
```

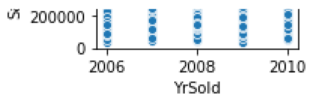
```
SalePrice        1.000000
GrLivArea        0.708624
GarageArea       0.623431
TotalBsmtSF      0.613581
1stFlrSF         0.605852
MasVnrArea       0.477493
BsmtFinSF1       0.386420
WoodDeckSF       0.324413
2ndFlrSF         0.319334
OpenPorchSF      0.315856
LotArea          0.263843
BsmtUnfSF        0.214479
ScreenPorch      0.111447
PoolArea         0.092404
3SsnPorch        0.044584
BsmtFinSF2      -0.011378
LowQualFinSF    -0.025606
EnclosedPorch   -0.128578
Name: SalePrice, dtype: float64
```

SalePrice shows strong correlations with the features GrLivArea (0.71), GarageArea (0.62), 1stFlSF (0.61), TotalBsmtSF (0.61). MasVnrArea (0.48) shows a moderate correlation with Saleprice . The remaining features show weak () or very weak (PoolArea, ScreenPorch, 3SsnPorch, EnclosedPorch, BsmtFinSF2) correlations with SalePrice.

Note that if a feature is not present a zero is given for the area from the data dictionary. Lets set all zero to missing!

```
f = plt.figure(figsize=(12,18))

for i in range(len(num_attributes.columns)):
    f.add_subplot(10,4, i+1)
    sns.scatterplot(num_attributes.iloc[:,i], target)

plt.tight_layout()
plt.show()
```

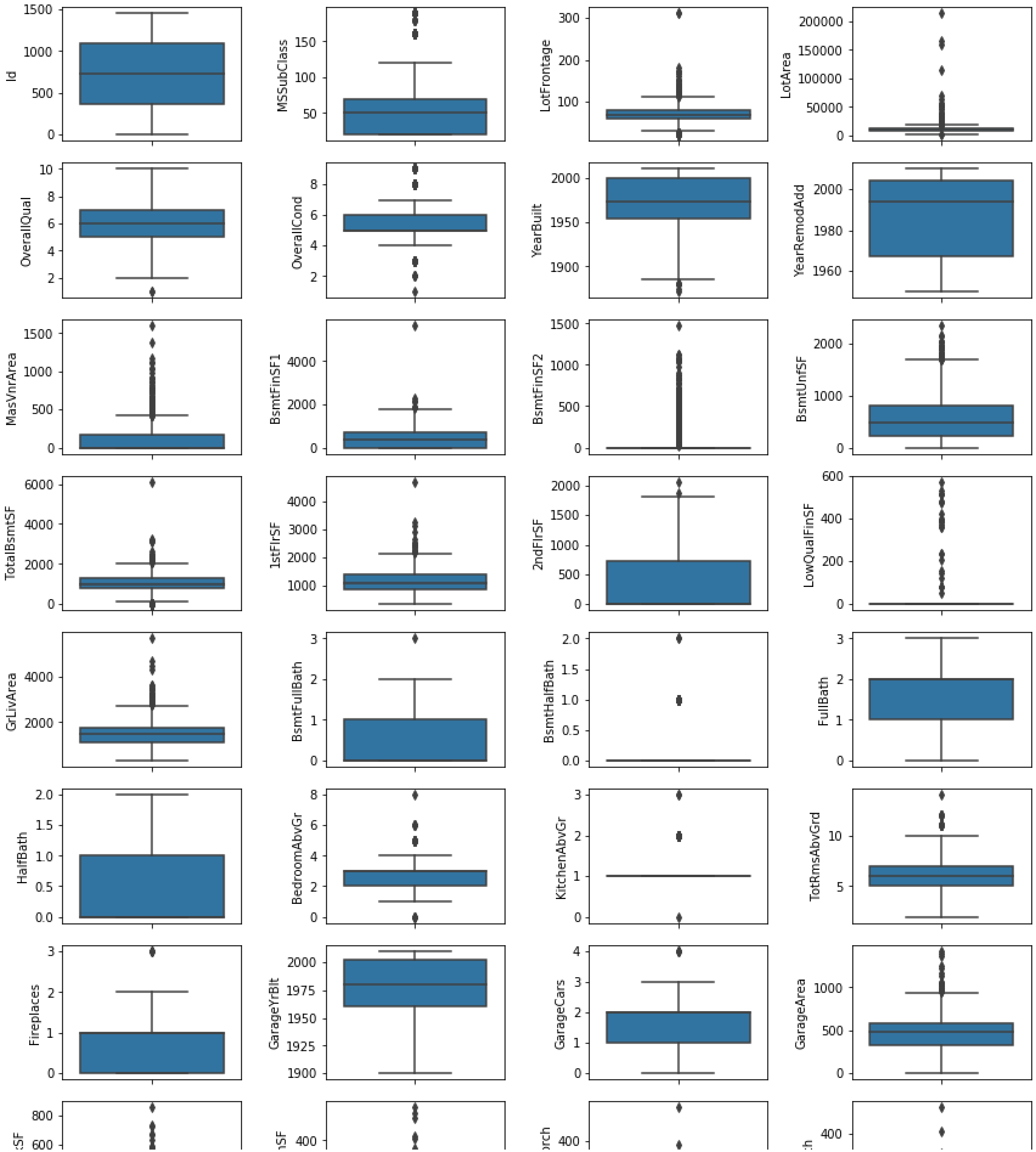

SalePrice
MasVnrArea
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageYrBlt
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold

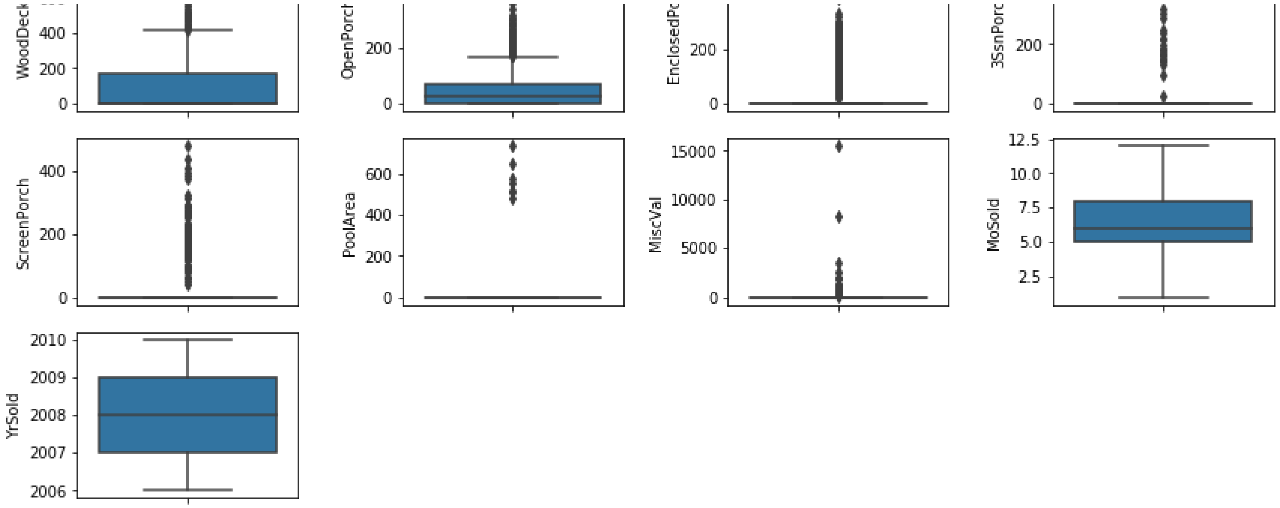


The scatterplot above shows outliers in majority of the numerical features. The BOXPLOT is a good way to see the outliers.

Notes for Data Cleaning & Preprocessing Based on a first viewing of the scatter plots against SalePrice, there appears to be: A few outliers on the LotFrontage (say, >200) and LotArea (>100000) data. BsmtFinSF1 (>4000) and TotalBsmtSF (>6000),1stFlrSF (>4000) GrLivArea (>4000 AND SalePrice <300000), LowQualFinSF (>550)

```
#Boxplot
#num_attributes =House_Data.select_dtypes(include='object').drop('SalePrice', axis=1).copy()



fig = plt.figure(figsize=(12, 18))

for i in range(len(num_attributes.columns)):
    fig.add_subplot(10, 4, i+1)
    sns.boxplot(y=num_attributes.iloc[:,i])

plt.tight_layout()
plt.show()
```


Id
MSSubClass
LotFrontage
LotArea
OverallQual
OverallCond
YearBuilt
YearRemodAdd
MasVnrArea
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageYrBlt
GarageCars
GarageArea



WoodDeck
OpenPorch
EnclosedPo
3SsnPorc
ScreenPorch
PoolArea
MiscVal
MoSold
YrSold