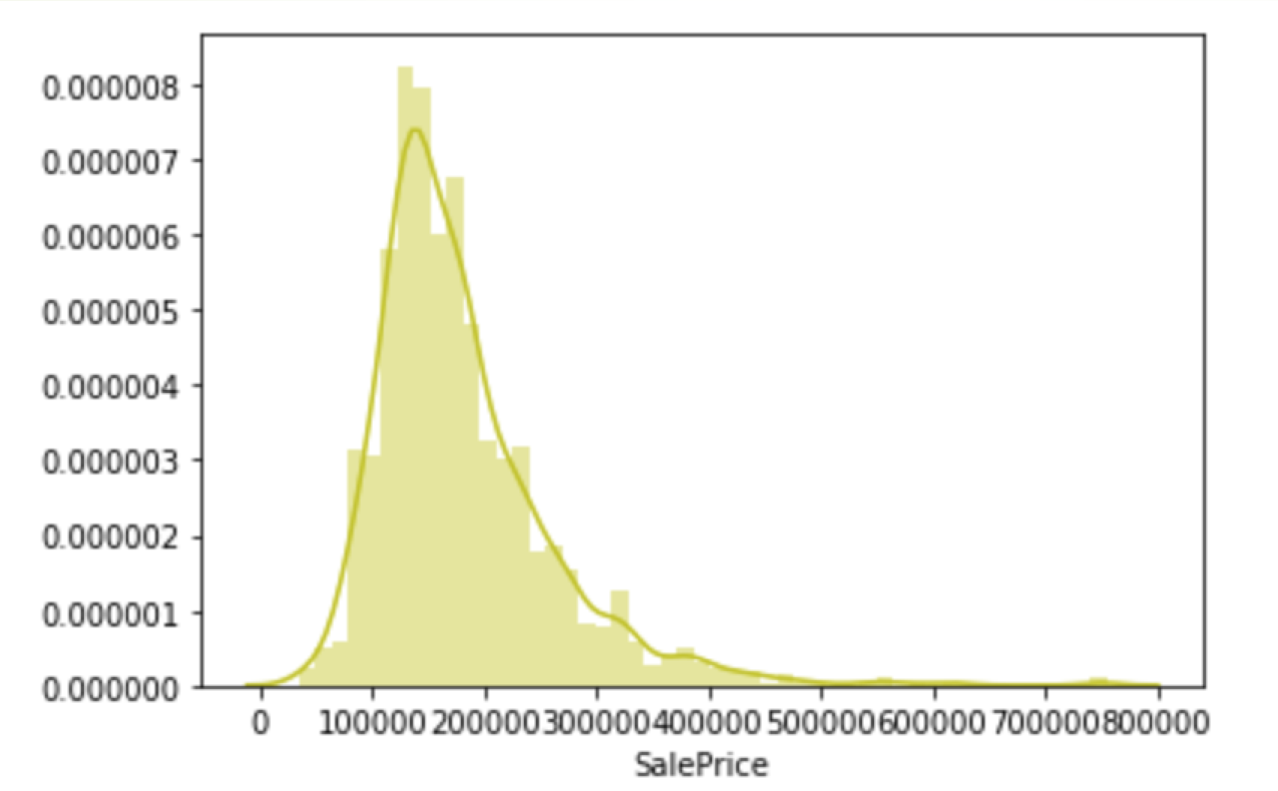
House price prediction

組員：許軒瑋、胡凱欣、陳胤銓、楊捷安
Group: 9

Data

- Data downloaded from **Kaggle** : ***House Prices: Advanced Regression Techniques***
  - 1460 training data, 1459 testing data
  - 79 attributes
- View of **Sale Price statistics** in training data

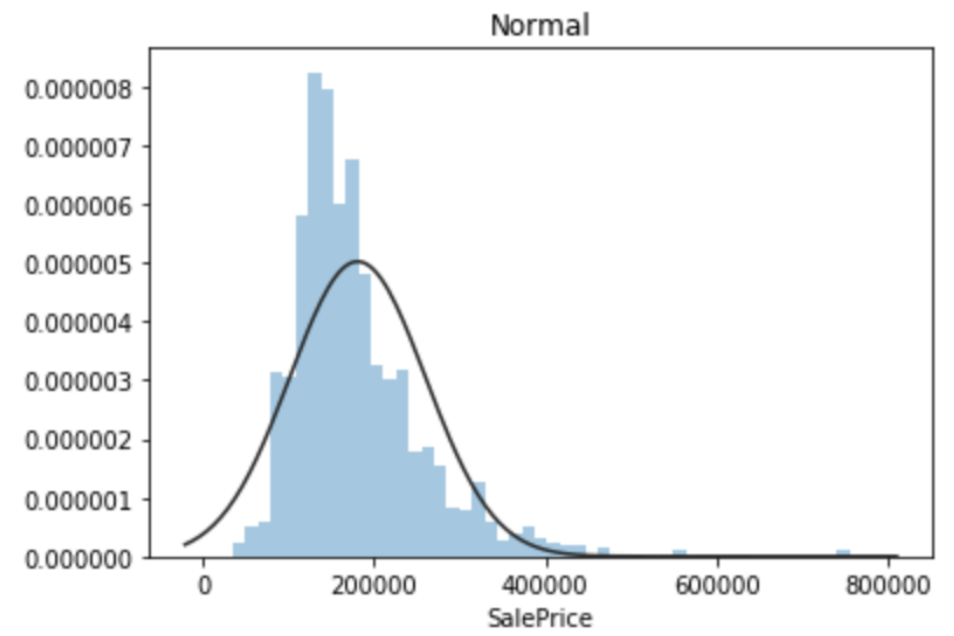


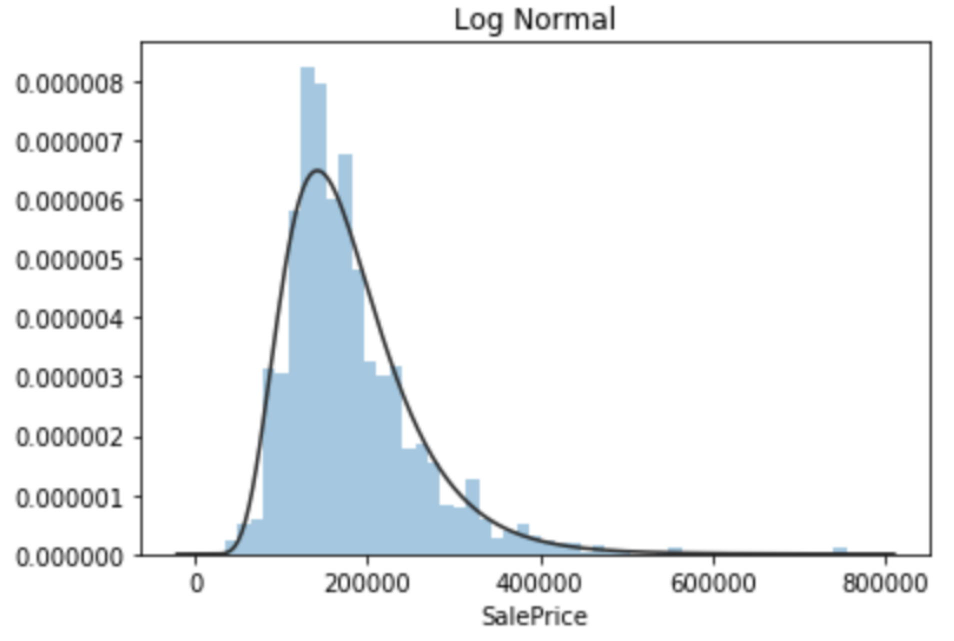
Skewness: 1.88288
Kurtosis: 6.53268

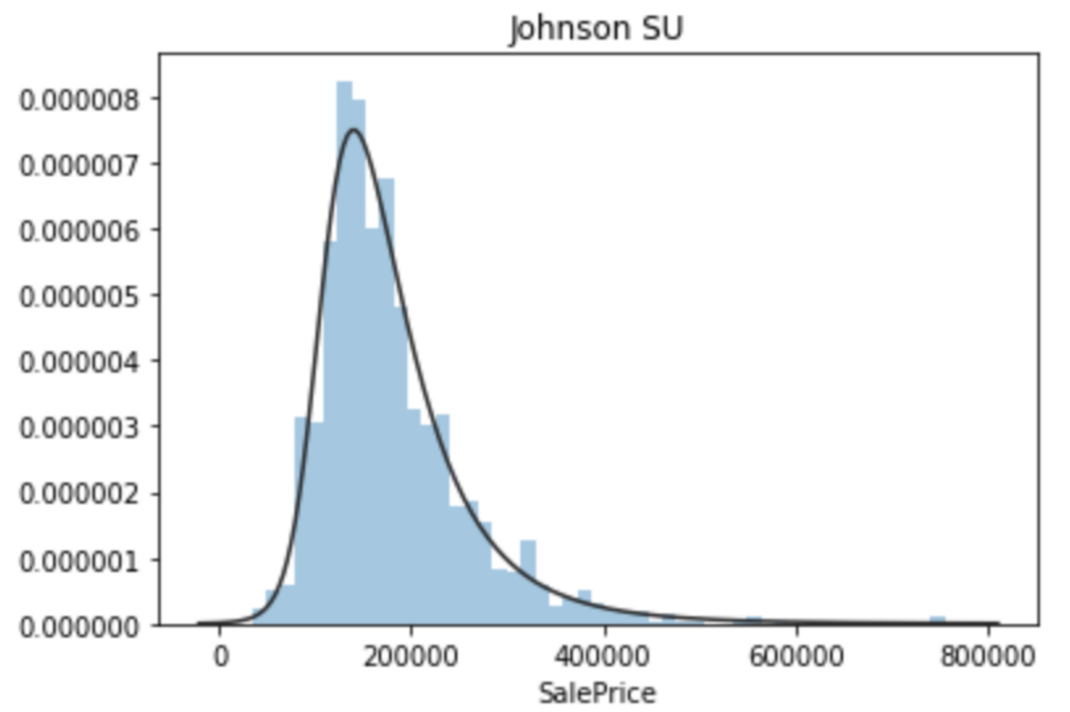
Analysis **:**

- ***Deviate from the normal distribution.***
- ***Have appreciable positive skewness***

Data - Sale Price

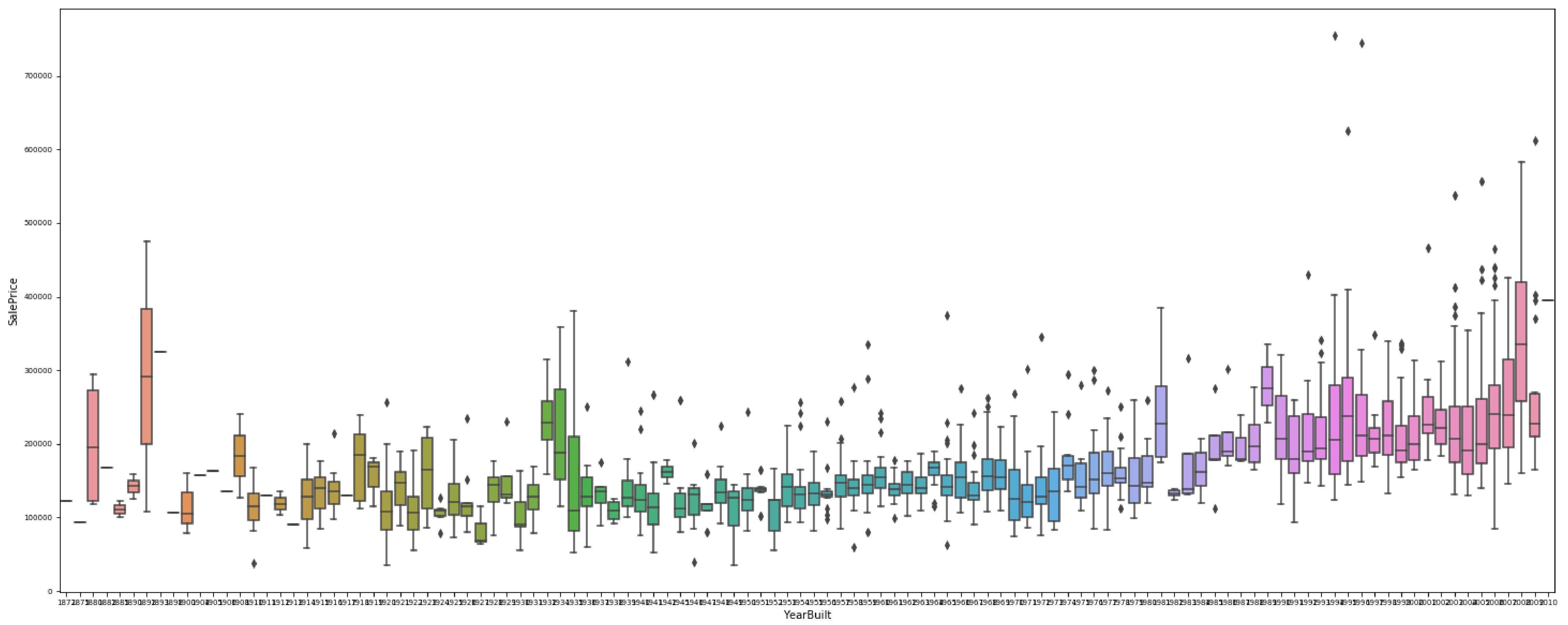






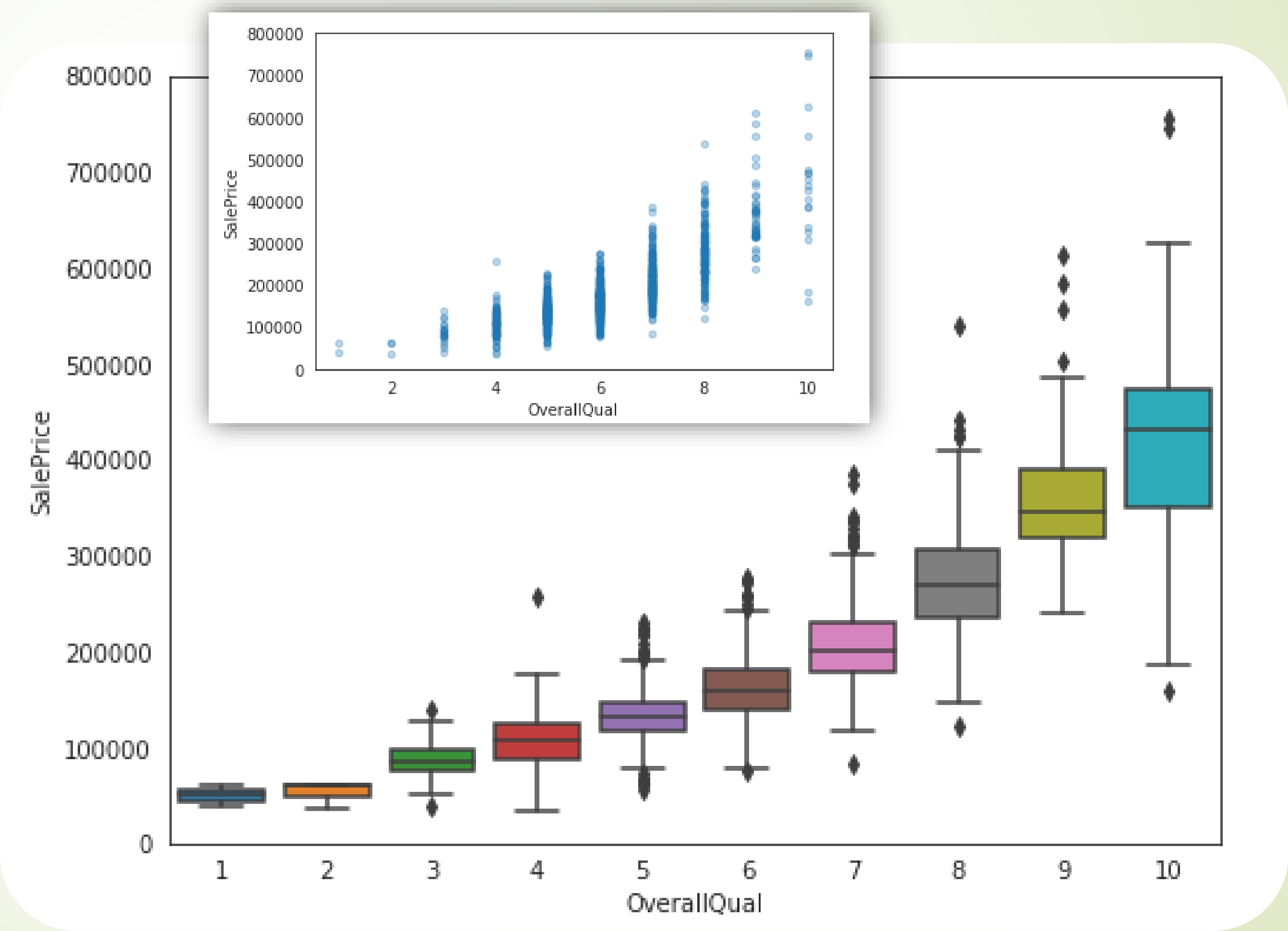
Johnson's SU has been used successfully to model **asset returns for portfolio management**

SalePrice v.s YearBuilt

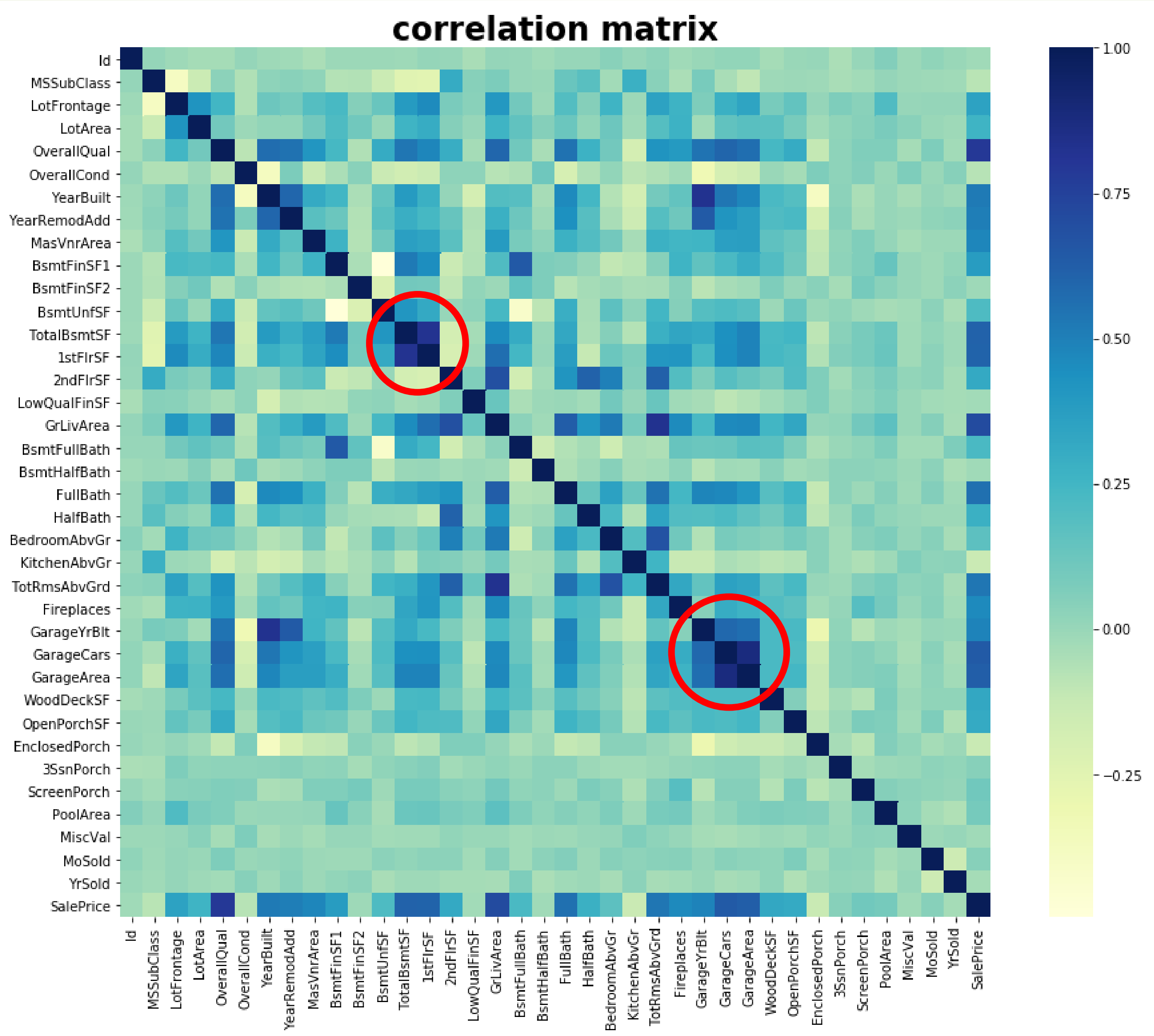


SalePrice v.s OverallQual

- OverallQual : Rates the overall condition of the house
  - 10 Very Excellent
  - 9 Excellent
  - 8 Very Good
  - 7 Good
  - 6 Above Average
  - 5 Average
  - 4 Below Average
  - 3 Fair
  - 2 Poor
  - 1 Very Poor



Correlation matrix (heatmap style)



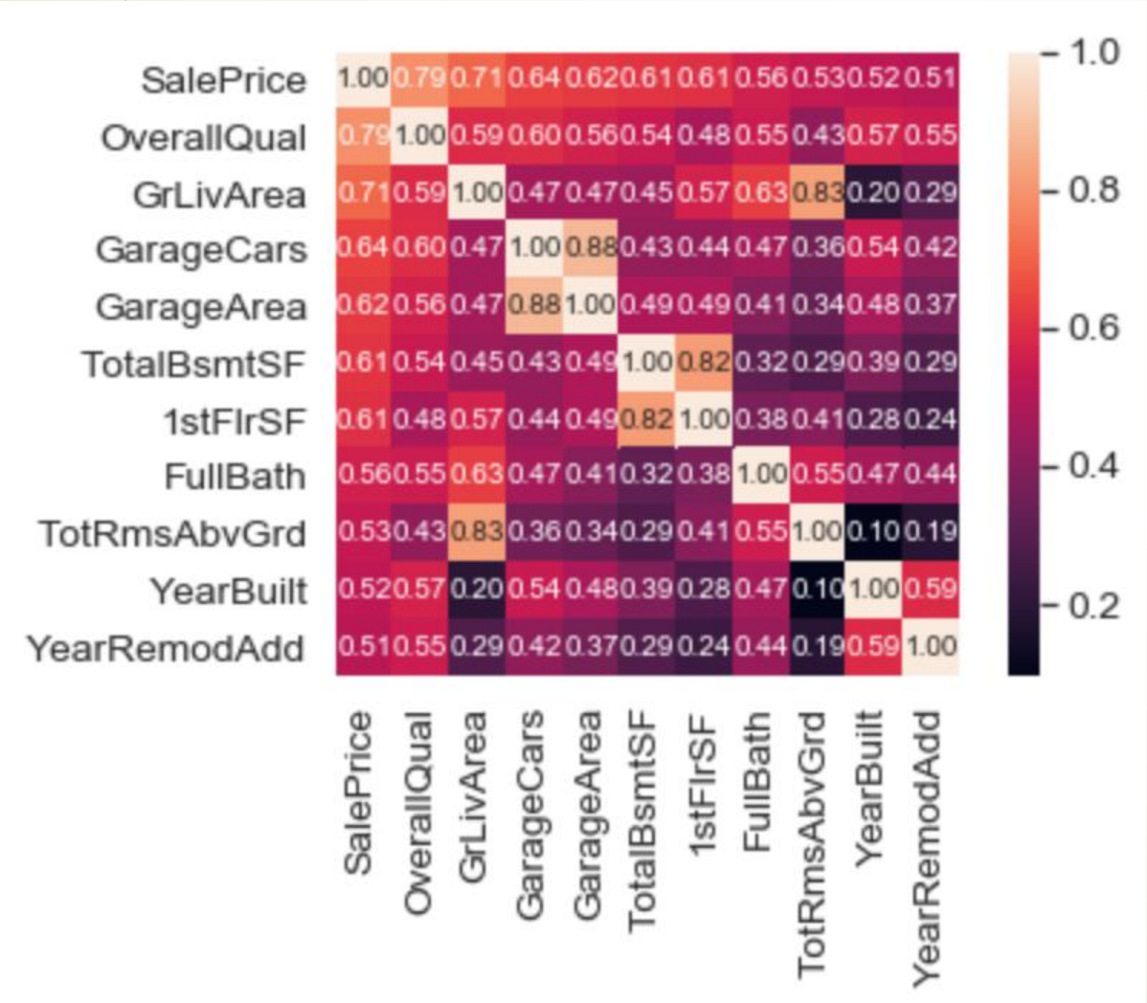
Data- Attributes

- **Correlation between Sale Price and Attributes**

	SalePrice
SalePrice	1
OverallQual	0.790982
GrLivArea	0.708624
GarageCars	0.640409
GarageArea	0.623431
TotalBsmtSF	0.613581
1stFlrSF	0.605852
FullBath	0.560664
TotRmsAbvGrd	0.533723
YearBuilt	0.522897
YearRemodAdd	0.507101
GarageYrBlt	0.486362
MasVnrArea	0.477493
Fireplaces	0.466929
BsmtFinSF1	0.38642
LotFrontage	0.351799
WoodDeckSF	0.324413
2ndFlrSF	0.319334
OpenPorchSF	0.315856
HalfBath	0.284108
LotArea	0.263843
BsmtFullBath	0.227122
BsmtUnfSF	0.214479
BedroomAbvGr	0.168213
KitchenAbvGr	0.135907
EnclosedPorch	0.128578
ScreenPorch	0.111447
PoolArea	0.0924035
MSSubClass	0.0842841
OverallCond	0.0778559
MoSold	0.0464322
3SsnPorch	0.0445837
YrSold	0.0289226
LowQualFinSF	0.0256061
Id	0.0219167
MiscVal	0.0211896
BsmtHalfBath	0.0168442
BsmtFinSF2	0.0113781

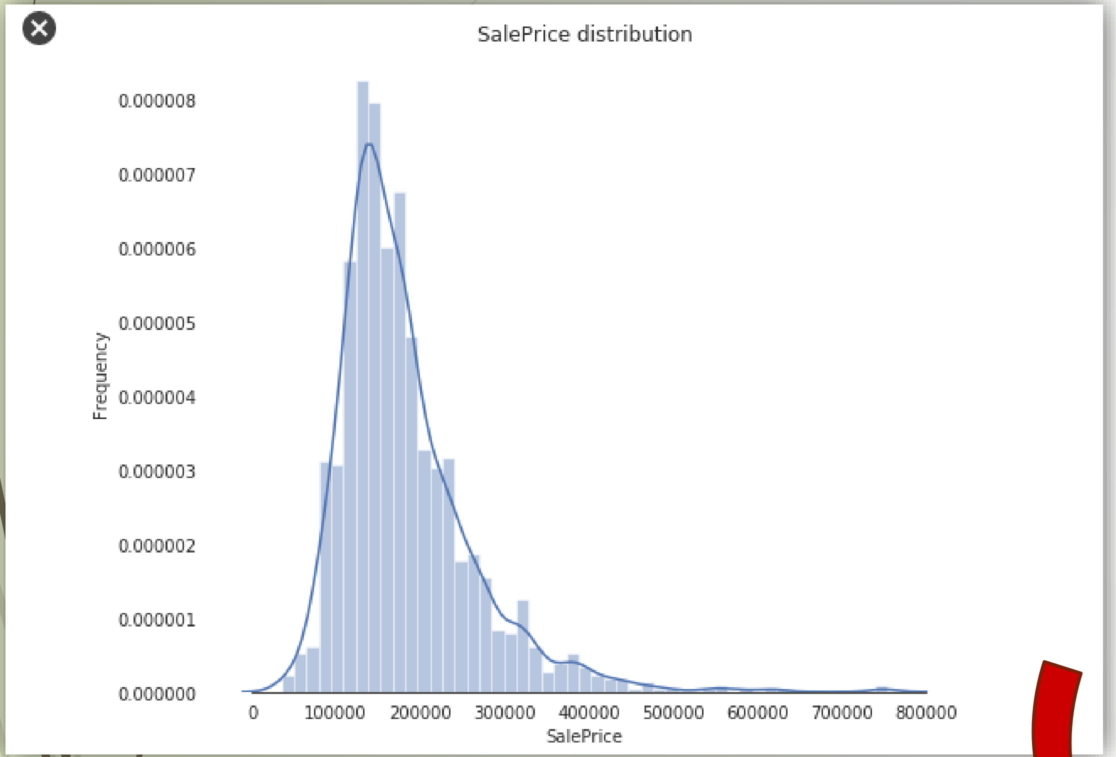
Data- Attributes

- Pick **Top 10 highest** correlation
- Look into correlation between attributes

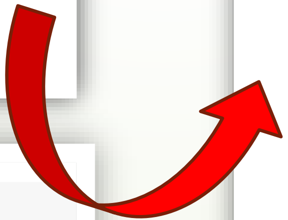


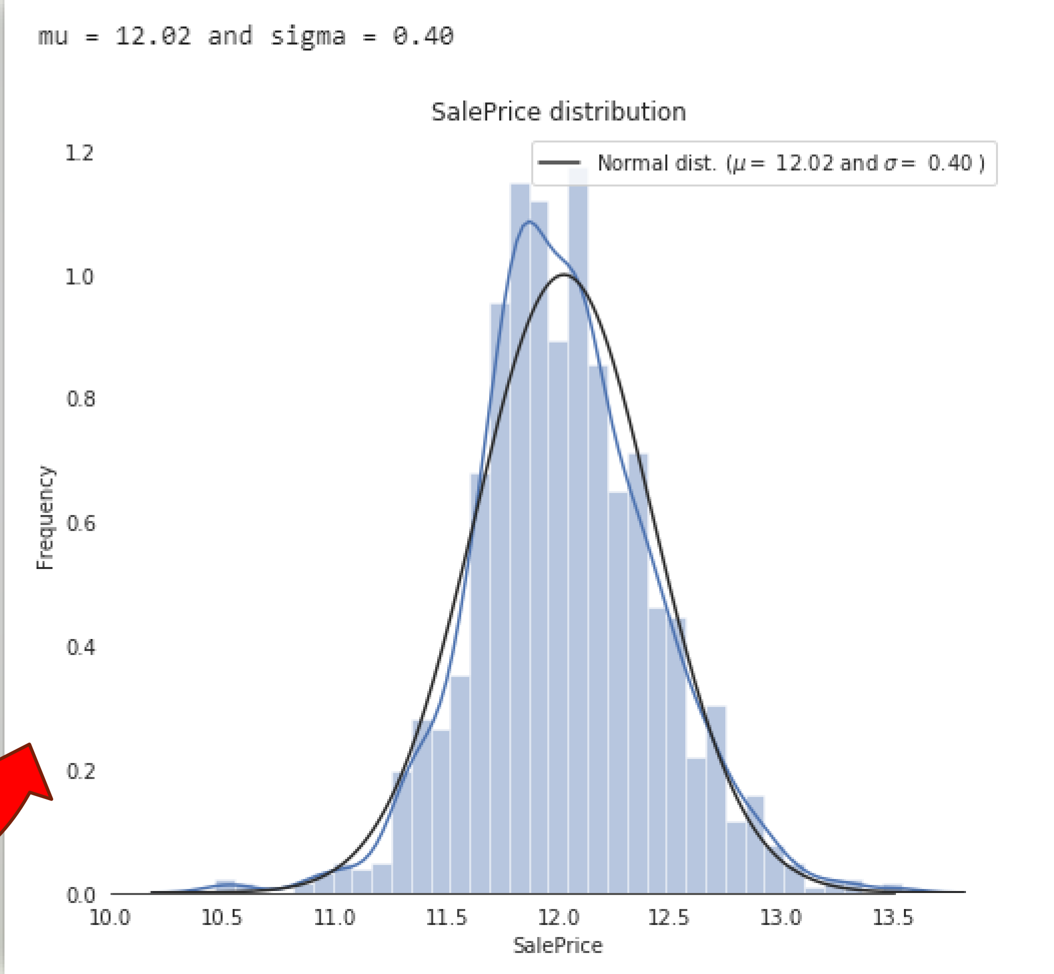
Data- Detect Outliers

Normalization - Fixing Skew Data

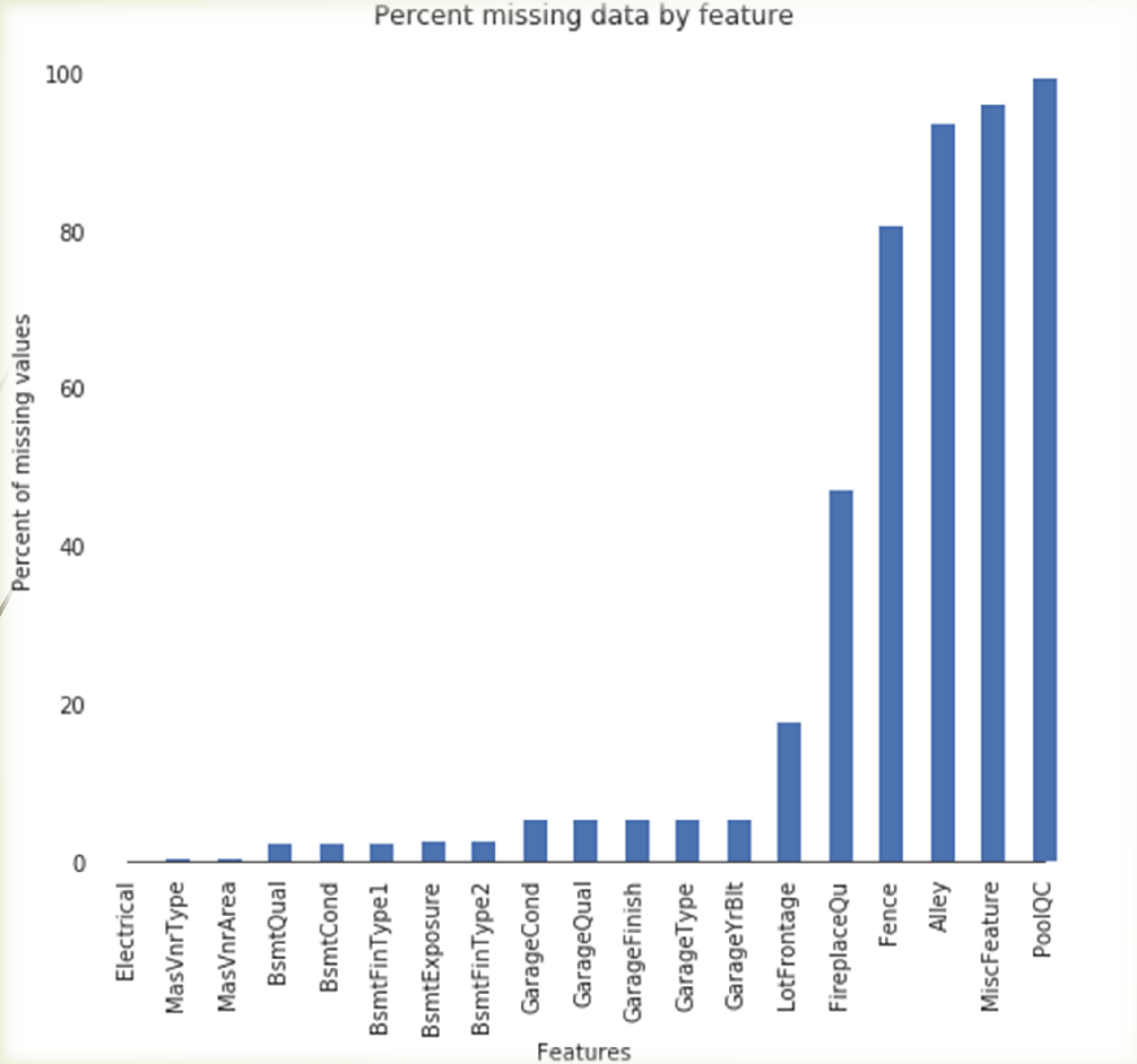



```
# log(1+x) transform
train["SalePrice"] = np.log1p(train["SalePrice"])
```





Missing Data(pandas.isnull)



	Count	Percent(%)
PoolQC	2907	99.691358
MiscFeature	2811	96.399177
Alley	2718	93.209877
Fence	2345	80.418381
FireplaceQu	1420	48.696845
LotFrontage	485	16.632373
GarageCond	159	5.452675
GarageQual	159	5.452675
GarageYrBlt	159	5.452675
GarageFinish	159	5.452675
GarageType	157	5.384088
BsmtCond	82	2.812071
BsmtExposure	82	2.812071
BsmtQual	81	2.777778
BsmtFinType2	80	2.743484
BsmtFinType1	79	2.709191
MasVnrType	24	0.823045
MasVnrArea	23	0.788752
MSZoning	4	0.137174
BsmtHalfBath	2	0.068587

Filling Missing Values

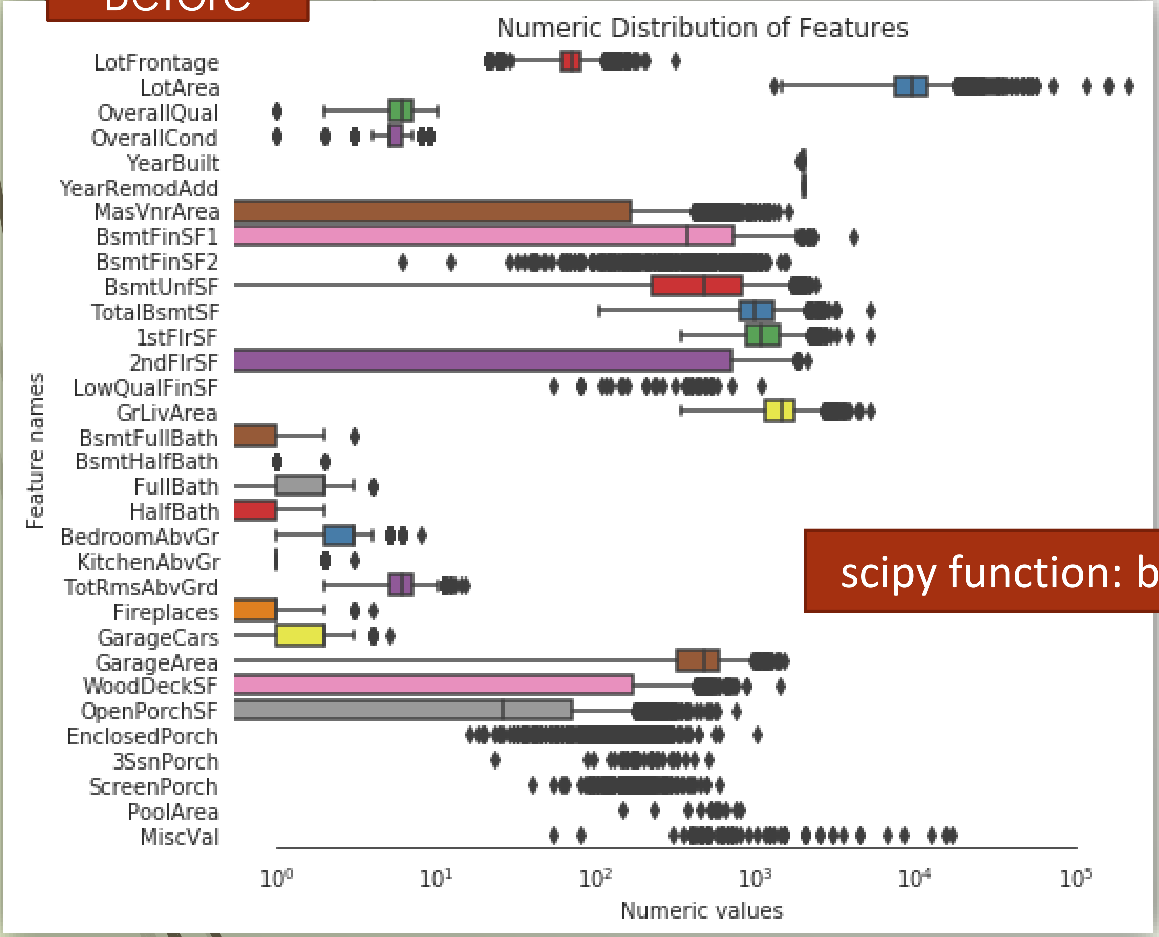
- Type1 ： NA -> "None“
  - Ex. **PoolQC**: no pool, **Fence**: no fence, **GarageQual**: no garage
  - (PoolQC：Pool quality(泳池品質)、Fence：Fence quality、GarageQual：Garage quality)
- Type2：NA -> 0
  - Ex. **TotalBsmtSF**: no basement so basement area = 0
  - (TotalBsmSF： Total square feet of basement area 地下室面積)

- Type3：typical ex.('Typ') values
  - Ex. **KitchenQual**： Kitchen quality(廚房品質)
    - Ex Excellent
    - Gd Good
    - TA Typical/Average
    - Fa Fair
    - Po Poor
- Type4：Using other features to help fill missing values
  - Ex. **LotFrontage**： Neighborhood
    - LotFrontage: Linear feet of street connected to property(房子鄰近的街道距離)
    - Neighborhood: Physical locations within Ames city limits(在Ames City的實際位置)

Fixing skewed features

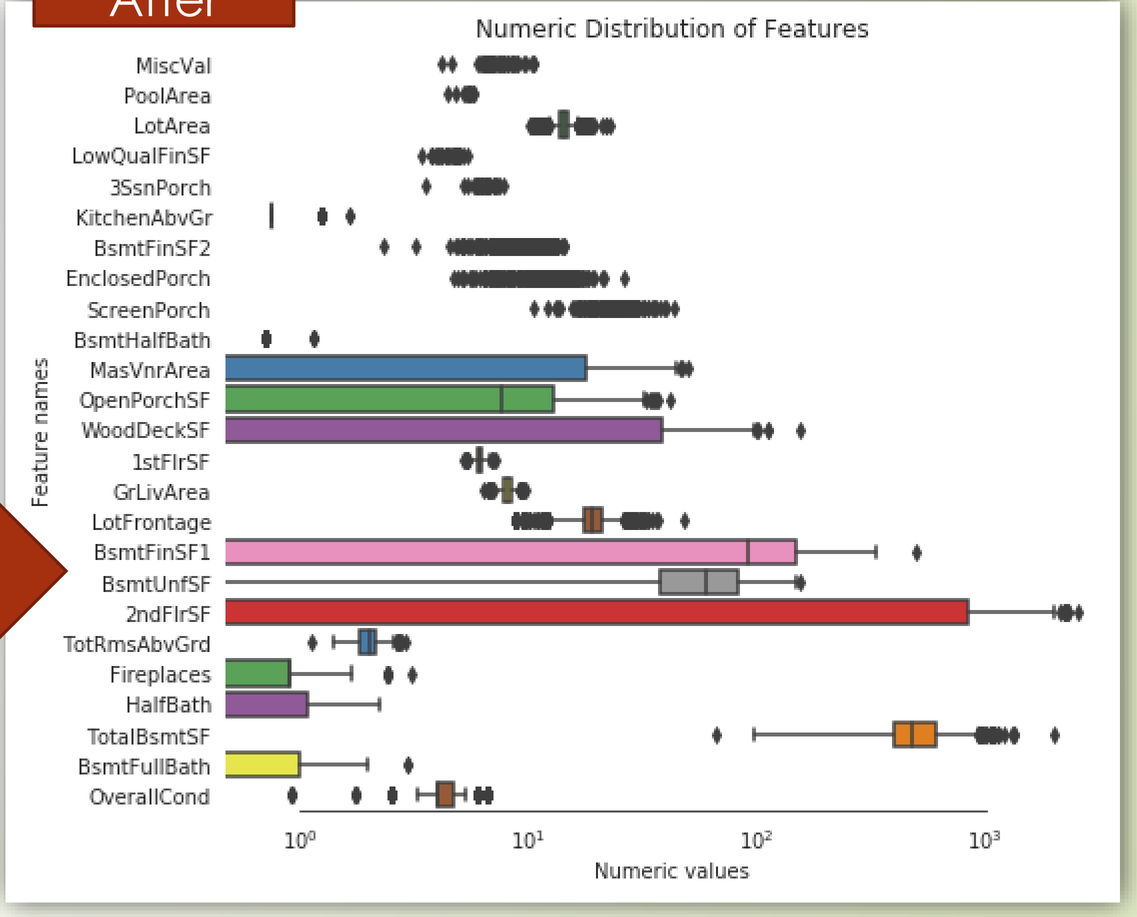
(Normalize skewed features)

Before



scipy function: boxcox1p

After



Create interesting features & Feature transformations

```
all_features['haspool'] = all_features['PoolArea'].apply(lambda x: 1 if x > 0 else 0)
all_features['has2ndfloor'] = all_features['2ndFlrSF'].apply(lambda x: 1 if x > 0 else 0)
all_features['hasgarage'] = all_features['GarageArea'].apply(lambda x: 1 if x > 0 else 0)
all_features['hasbsmt'] = all_features['TotalBsmtSF'].apply(lambda x: 1 if x > 0 else 0)
all_features['hasfireplace'] = all_features['Fireplaces'].apply(lambda x: 1 if x > 0 else 0)
```

```
def logs(res, ls):
    m = res.shape[1]
    for l in ls:
        res = res.assign(newcol=pd.Series(np.log(1.01+res[l])).values)
        res.columns.values[m] = l + '_log'
        m += 1
    return res

log_features = ['LotFrontage','LotArea','MasVnrArea','BsmtFinSF1','BsmtFinSF2','BsmtUnfSF',
                 'TotalBsmtSF','1stFlrSF','2ndFlrSF','LowQualFinSF','GrLivArea',
                 'BsmtFullBath','BsmtHalfBath','FullBath','HalfBath','BedroomAbvGr','KitchenAbvGr',
                 'TotRmsAbvGrd','Fireplaces','GarageCars','GarageArea','WoodDeckSF','OpenPorchSF',
                 'EnclosedPorch','3SsnPorch','ScreenPorch','PoolArea','MiscVal','YearRemodAdd','TotalSF']

all_features = logs(all_features, log_features)
```

Training a Model

RMSE : $\sqrt{\frac{\sum_{i=1}^{N}(y_i - \widehat{y_i})^2}{N}}$

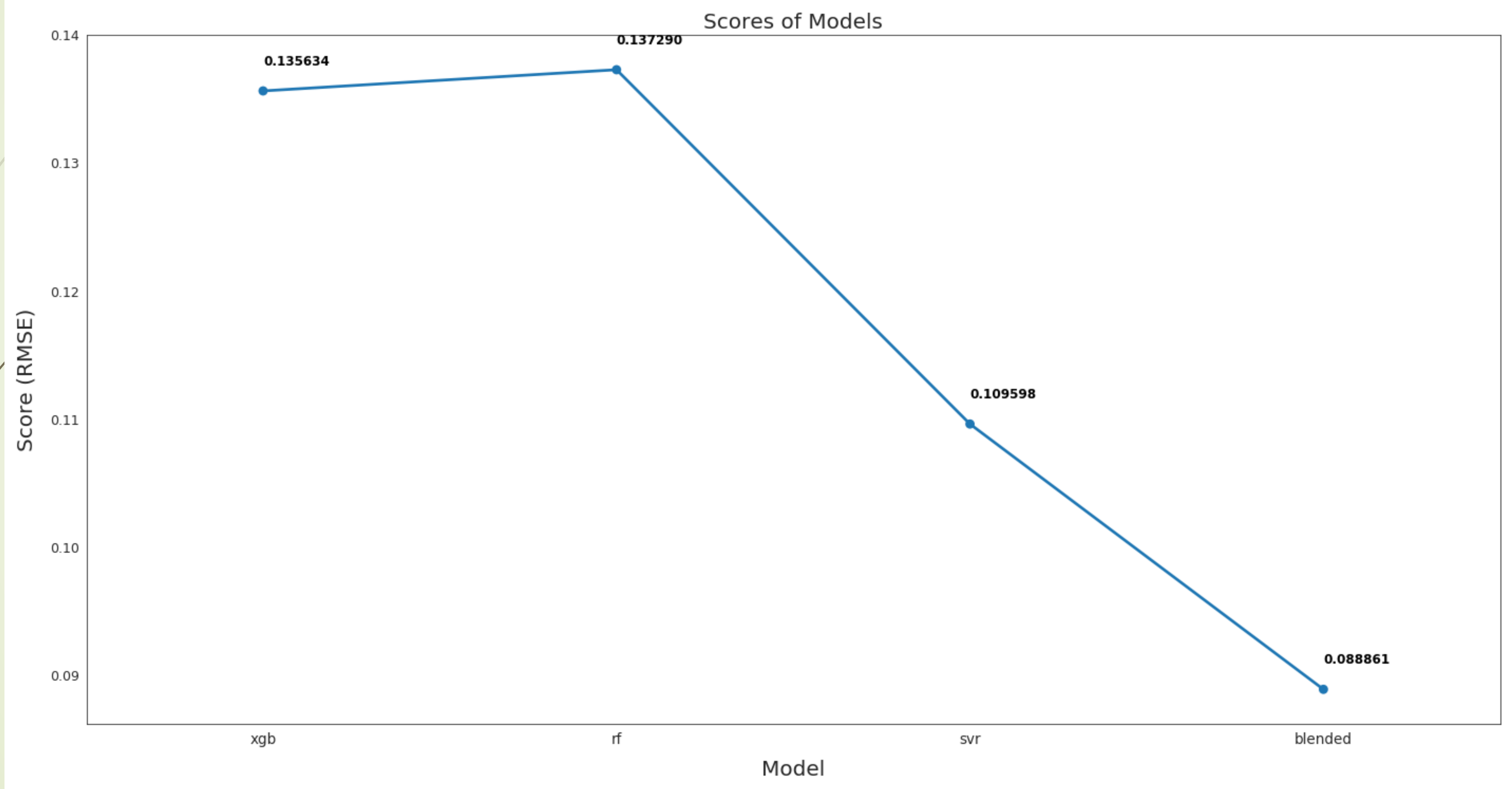
```
# Setup cross validation folds
kf = KFold(n_splits=12, random_state=42, shuffle=True)
```

```
# XGBoost Regressor
xgboost = XGBRegressor(learning_rate=0.01,
                       n_estimators=6000,
                       max_depth=4,
                       min_child_weight=0,
                       gamma=0.6,
                       subsample=0.7,
                       colsample_bytree=0.7,
                       objective='reg:squarederror',
                       nthread=-1,
                       scale_pos_weight=1,
                       seed=27,
                       reg_alpha=0.00006,
                       random_state=42)


# Random Forest
rf = RandomForestRegressor(n_estimators=1200,
                          max_depth=15,
                          min_samples_split=5,
                          min_samples_leaf=5,
                          max_features=None,
                          oob_score=True,
                          random_state=42)
# Support Vector regression
svr = make_pipeline(RobustScaler(), SVR(C= 20, epsilon= 0.008, gamma=0.0003))
```

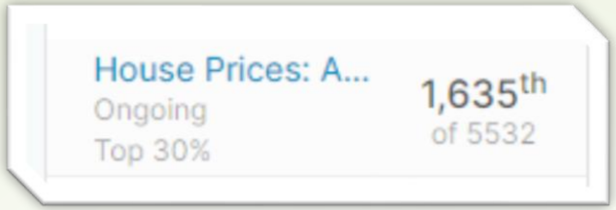
```
def Blending_pred(x):
    return ( 0.3 * xgb_model_full_data.predict(x) +
             0.2 * rf_model_full_data.predict(x) +
             0.5 * svr_model_full_data.predict(x) )
```

Ensemble Methods : Weight Average

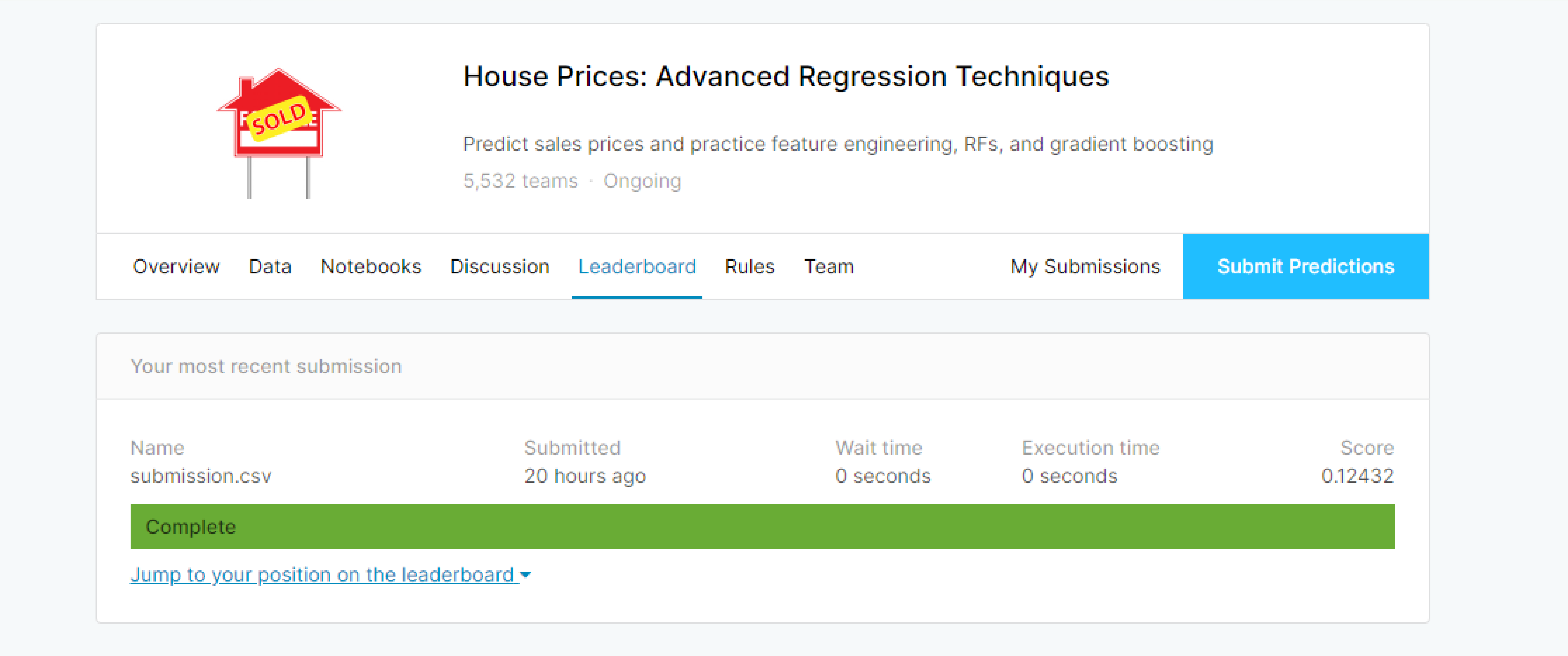


18

Result :



House Prices: A...
Ongoing
Top 30%
1,635th
of 5532



SOLD
House Prices: Advanced Regression Techniques
Predict sales prices and practice feature engineering, RFs, and gradient boosting
5,532 teams · Ongoing
Overview
Data
Notebooks
Discussion
Leaderboard
Rules
Team
My Submissions
Submit Predictions
Your most recent submission
Name
submission.csv
Submitted
20 hours ago
Wait time
0 seconds
Execution time
0 seconds
Score
0.12432
Complete
Jump to your position on the leaderboard

Future works

- Revised data attributes and create more interesting attributes with human intuition.
  - Swimming pool size, built year, Quality etc.
- Apply PCA to reduce high correlation attributes.