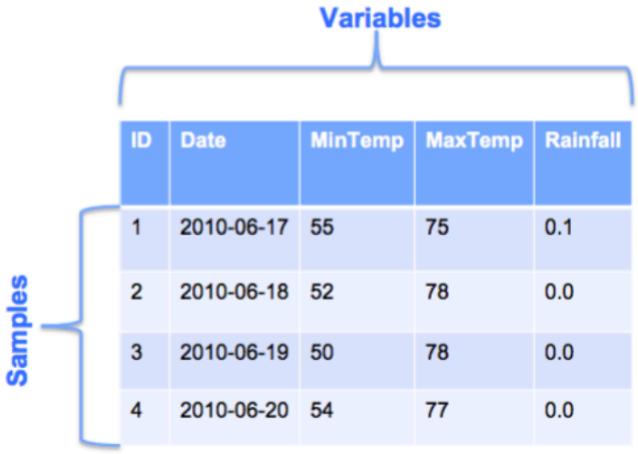
# Samenvatting – Data Exploration

**Steps of Data Exploration and Preparation**

- Terminology
- Normal Distribution
- Garabage in garbage out
- Why Data Exploration
- Step in Data Exploration

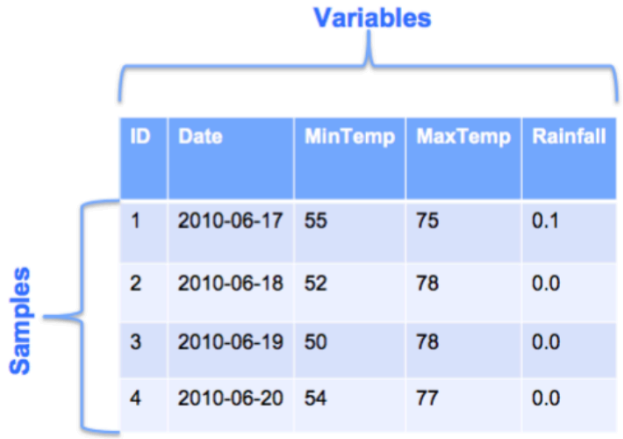
**Terminology**

- **Samples**
  - The set of instances/examples/observations/records/row




| ID | Date | MinTemp | MaxTemp | Rainfall |
|---|---|---|---|---|
| 1 | 2010-06-17 | 55 | 75 | 0.1 |
| 2 | 2010-06-18 | 52 | 78 | 0.0 |
| 3 | 2010-06-19 | 50 | 78 | 0.0 |
| 4 | 2010-06-20 | 54 | 77 | 0.0 |

- **Variable**
  - Characteristic/Feature/Attribute/Column/Dimension of samples




| ID | Date | MinTemp | MaxTemp | Rainfall |
|---|---|---|---|---|
| 1 | 2010-06-17 | 55 | 75 | 0.1 |
| 2 | 2010-06-18 | 52 | 78 | 0.0 |
| 3 | 2010-06-19 | 50 | 78 | 0.0 |
| 4 | 2010-06-20 | 54 | 77 | 0.0 |

**Variable Identification**

- **Variable:**
  - Numeric/Quantitative:
    - Can be used to compute or sort.
    - Integer: -1, 0, 1, 2
    - Float: -1.9, 0.05, 3.14
  - Categorical/Qualitative/Nominal
    - Label: Red, Blue, Green
    - Boolean: True, False

**Terminology**

- **Datatype:**
  - Domain of a variable (data, integer, float, text, boolean)

Variables

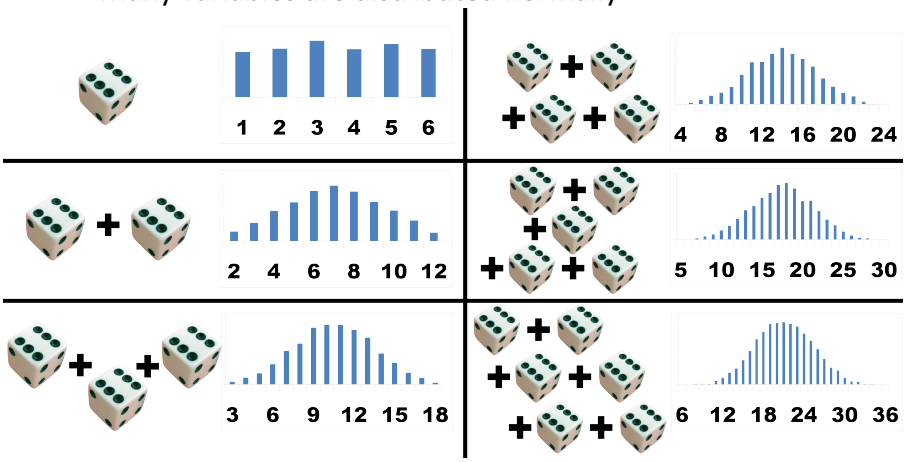
Samples

| ID | Date | MinTemp | MaxTemp | Rainfall |
|---|---|---|---|---|
| 1 | 2010-06-17 | 55 | 75 | 0.1 |
| 2 | 2010-06-18 | 52 | 78 | 0.0 |
| 3 | 2010-06-19 | 50 | 78 | 0.0 |
| 4 | 2010-06-20 | 54 | 77 | 0.0 |

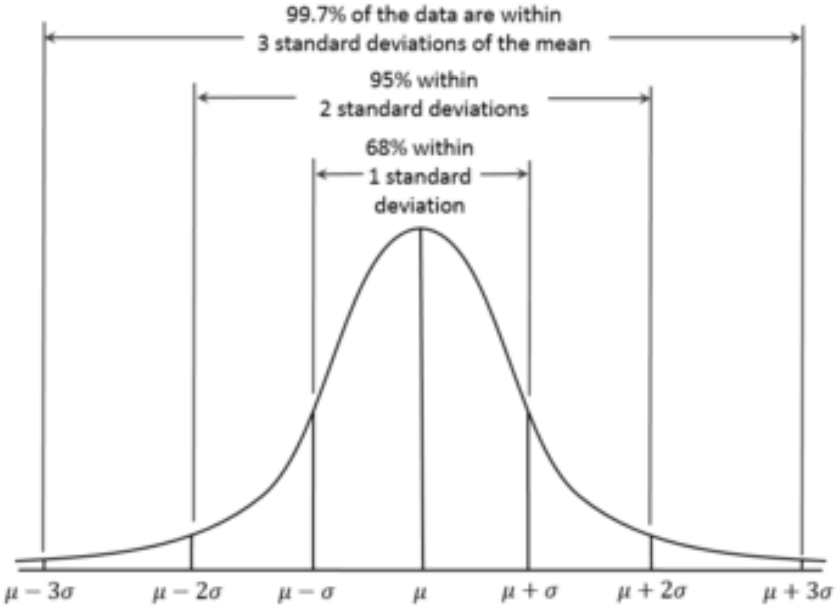
- **Model:**
  - Simplification of the real world that helps us understand/predict/tasks

**Normal distribution**

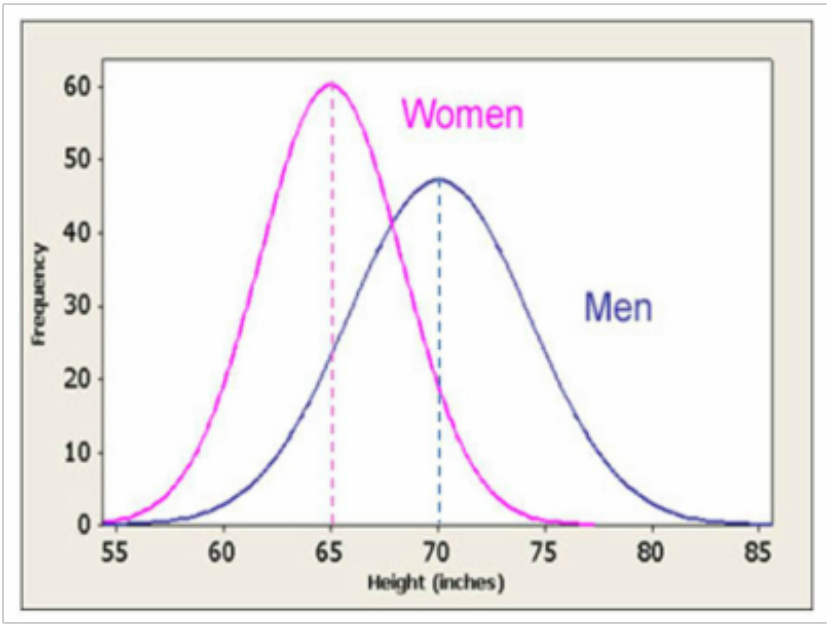
- Many variables are distributed normally



- Height dfference between men and women




Are men taller than women?




- Because they are normally distributed, we can use proven statistics (student T-test) to answer that!
- However, when data is not normally distributed the result may be invalid.

Not all data is normally distributed; we need to check so we know which instrument to use
Statistics do not flag when something is wrong; you make the difference between valid and invalid results.

**Garbage in → Garbage Out**

- Inconsistencies in data:
    - Incorrect data formats
    - Lower/Uppercase
    - Whitespaces in codes
    - Decimals converted to whole numbers

- Faults in reading data:
    - Wrong column headers
    - Comments interpreted as data
    - Weakly structured data (HTML)

**Why Data Exploration?**

- To understand the characteristics of the data
- To prepare the data for analysis

- Precedes analytical treatment of data
    - Remove problems before analysis
    - Investigate potential directions:
        - Would a very simple model work?

**Data Exploration**

**Steps of Data Exploration**

1. Check Data edges
2. Variable Identification
3. Univariate Analysis
4. Bi-variate Analysis
5. Missing Values
6. Outliers
7. Variable transformation
8. Variable creation

Steps 2 to 8 are called Feature Engineering.

1. **Check Data edges**
    - **a.** Check number of rows
    - **b.** Check number of columns
    - **c.** Check first few rows
    - **d.** Check last few rows
    - **e.** Is the formatting ok?
    - **f.** Are the values within realm of reality?

```
df.head() # first rows
```

| | Id | MSSubClass | MSZoning | LotFrontage | LotArea | Street | Alley | LotShape | Lanc |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 60 | RL | 65.0 | 8450 | Pave | NaN | Reg | Lvl |
| **1** | 2 | 20 | RL | 80.0 | 9600 | Pave | NaN | Reg | Lvl |
| **2** | 3 | 60 | RL | 68.0 | 11250 | Pave | NaN | IR1 | Lvl |
| **3** | 4 | 70 | RL | 60.0 | 9550 | Pave | NaN | IR1 | Lvl |
| **4** | 5 | 60 | RL | 84.0 | 14260 | Pave | NaN | IR1 | Lvl |

2. **Codebook**
    - **a.** A codebook describes the contents, structure and layout of a data collection

    **Per set:**
    - Where did the data come from?
    - How did they collect the data (sampling, response rate)
    - Technical info about files (how many, size, format)

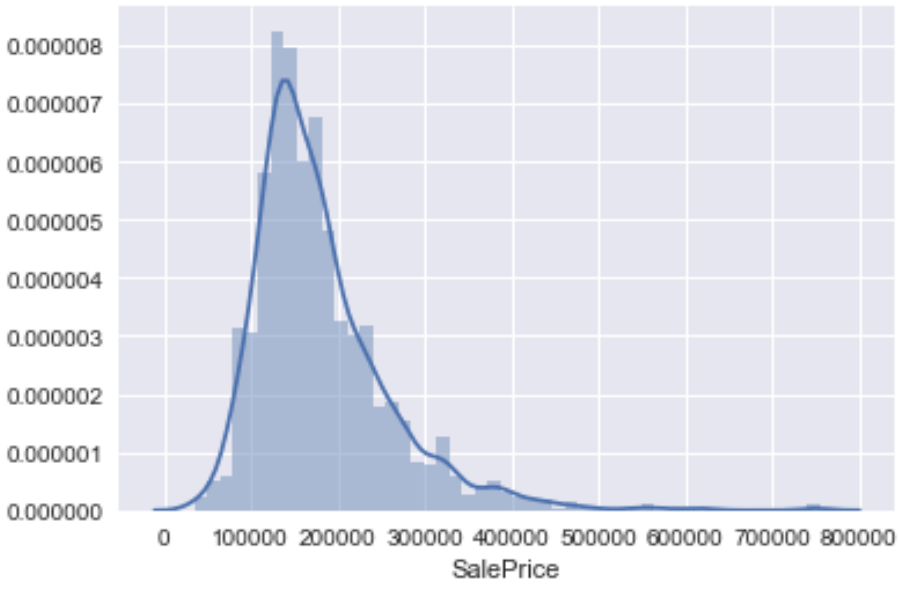
**Per variable:**

- Position
- Name
- Label (clearly describing meaning)
- Values (classes, range or an example)
- Datatype
- Numerical/Categorical
- Predictor/Target Variable
- Summary statistics

| Name | Pos | Label | Values |
|---|---|---|---|
| Id | 1 | unique id of sample | 1-1459 |
| MSSubClass | 2 | The building class | 20,30,40,45,50,…,180 |
| MSZoning | 3 | The general zoning classification | C Commercial<br>RH Residential High Density<br>RL Residential Low Density<br>RP Residential Low Density Park<br>RM Residential Medium Density |
| LotFrontage | 4 | Linear feet of street connected to property | 21.0-313.0 |
| LotArea | 5 | Lot size in square feet | 1300-215245 |

3. **Univariate Analysis**

```
sns.distplot(df.SalePrice);
```

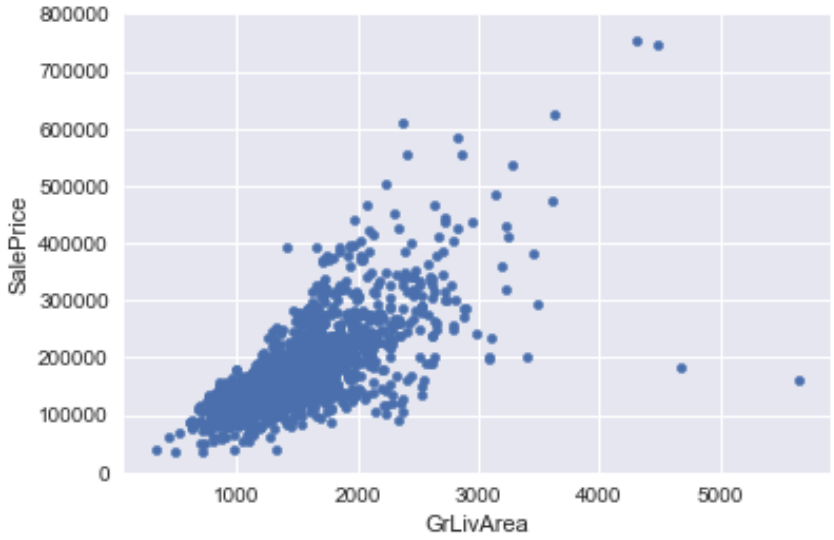



---

a. Not normal: left-skew, peaked

## 4. Bi-variate Analysis

```
#scatter plot grlivarea/saleprice
data = pd.concat([df.SalePrice, df.GrLivArea], axis=1)
data.plot.scatter(x='GrLivArea', y='SalePrice', ylim=(0,800000));
```

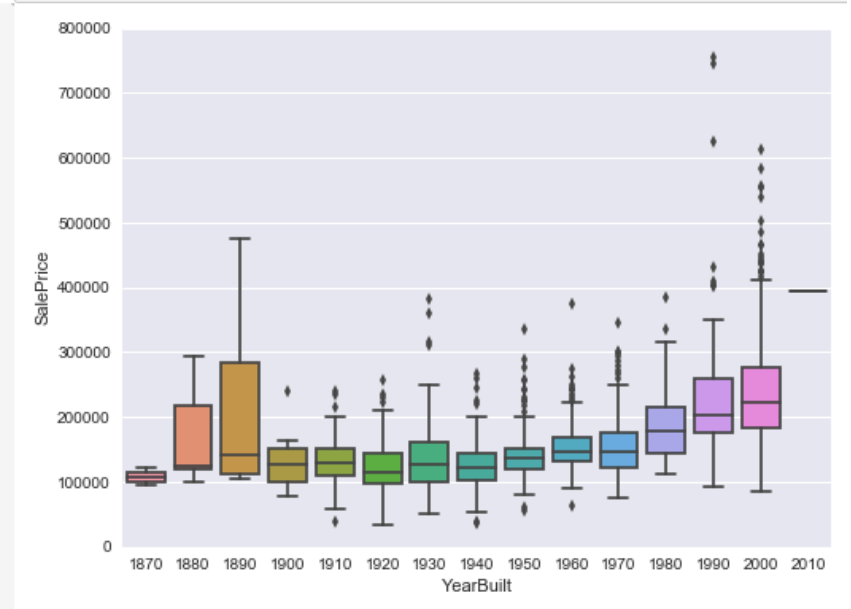



**a.** Related, but not clear of linear
**b.** Outliers

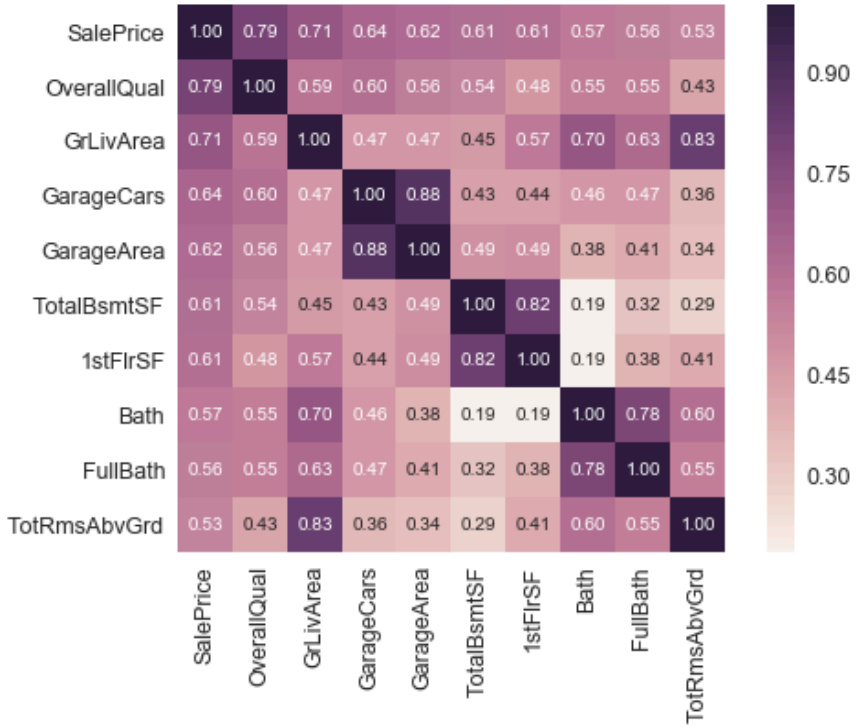
Categorical:

- Buckets
- Box n Whisker
- Weak Relation

```
import matplotlib.pyplot as plt
decades = df.YearBuilt.map(lambda x: x - x % 10)
data = pd.concat([df.SalePrice, decades], axis=1)
f, ax = plt.subplots(figsize=(8, 6))
fig = sns.boxplot(x='YearBuilt', y="SalePrice", data=data)
fig.axis(ymin=0, ymax=800000);
```



```
#saleprice correlation matrix
k = 10 #number of variables for heatmap
cols = corrmat.nlargest(k, 'SalePrice')['SalePrice'].index
cm = np.corrcoef(df[cols].values.T)
sns.set(font_scale=1.25)
hm = sns.heatmap(cm, cbar=True, annot=True, square=True, fmt='.2f', a
plt.show()
```



## 5. Missing Data

```
#missing data
total = df.isnull().sum().sort_values(ascending=False)
percent = (df.isnull().sum()/df.count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing_data.head(20)
```

| | Total | Percent |
|---|---|---|
| PoolQC | 1453 | 207.571429 |
| MiscFeature | 1406 | 26.037037 |
| Alley | 1369 | 15.043956 |
| Fence | 1179 | 4.195730 |
| FireplaceQu | 690 | 0.896104 |
| LotFrontage | 259 | 0.215654 |
| GarageCond | 81 | 0.058738 |
| GarageType | 81 | 0.058738 |
| GarageYrBlt | 81 | 0.058738 |
| GarageFinish | 81 | 0.058738 |
| GarageQual | 81 | 0.058738 |
| BsmtExposure | 38 | 0.026723 |
| BsmtFinType2 | 38 | 0.026723 |
| BsmtFinType1 | 37 | 0.026001 |
| BsmtCond | 37 | 0.026001 |
| BsmtQual | 37 | 0.026001 |
| MasVnrArea | 8 | 0.005510 |
| MasVnrType | 8 | 0.005510 |
| Electrical | 1 | 0.000685 |
| Utilities | 0 | 0.000000 |

NA misread, mean No Alley/Fence, etc.

No garage?

THE HAG

## 6. Outliers


```
#missing data
total = df.isnull().sum().sort_values(ascending=False)
percent = (df.isnull().sum()/df.count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing_data.head(20)
```

| | Total | Percent |
|---|---|---|
| PoolQC | 1453 | 207.571429 |
| MiscFeature | 1406 | 26.037037 |
| Alley | 1369 | 15.043956 |
| Fence | 1179 | 4.195730 |
| FireplaceQu | 690 | 0.896104 |
| LotFrontage | 259 | 0.215654 |
| GarageCond | 81 | 0.058738 |
| GarageType | 81 | 0.058738 |
| GarageYrBlt | 81 | 0.058738 |
| GarageFinish | 81 | 0.058738 |
| GarageQual | 81 | 0.058738 |
| BsmtExposure | 38 | 0.026723 |
| BsmtFinType2 | 38 | 0.026723 |
| BsmtFinType1 | 37 | 0.026001 |
| BsmtCond | 37 | 0.026001 |
| BsmtQual | 37 | 0.026001 |
| MasVnrArea | 8 | 0.005510 |
| MasVnrType | 8 | 0.005510 |
| Electrical | 1 | 0.000685 |
| Utilities | 0 | 0.000000 |

NA misread, mean No Alley/Fence, etc.

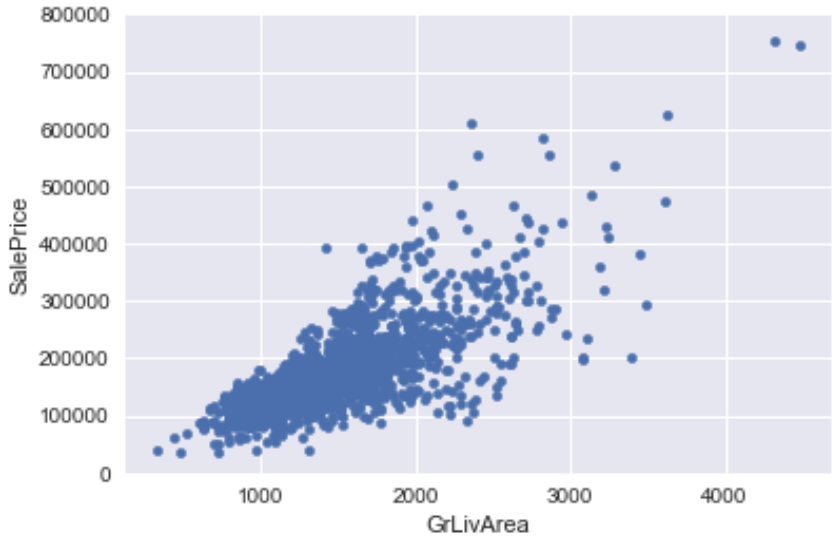
No garage?


THE HA

## 7. Transformation

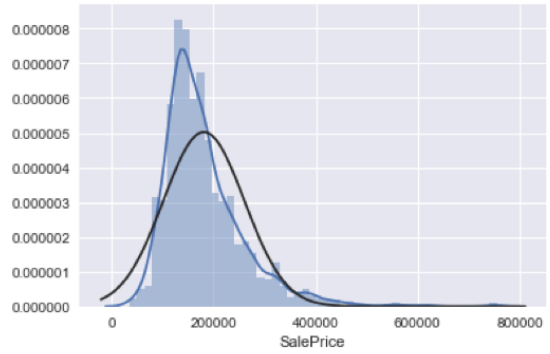
**a.** Normality and homoscedasticity

```
#scatter plot grlivarea/saleprice
data = pd.concat([df.SalePrice, df.GrLivArea], axis=1)
data.plot.scatter(x='GrLivArea', y='SalePrice', ylim=(0,800000));
```

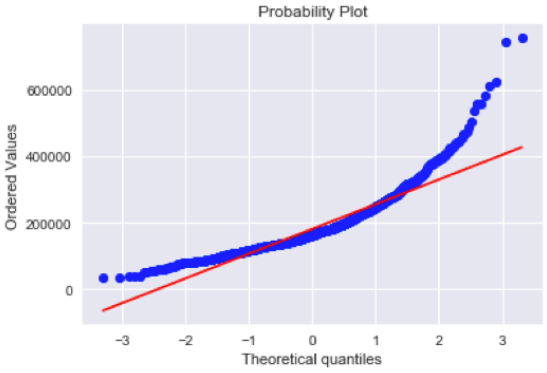



Analyze SalePrice

```
#histogram and normal probability plot
sns.distplot(df.SalePrice, fit=norm);
fig = plt.figure()
res = stats.probplot(df.SalePrice, plot=plt)
```




**More outliers?**




**Plot against normal distribution**

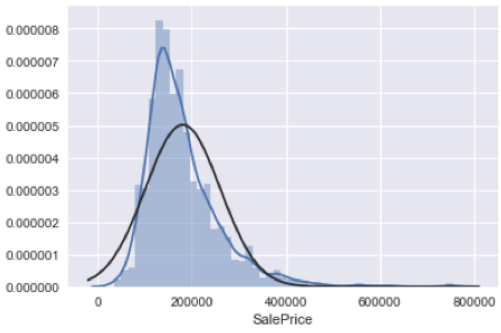
| Box-Cox | transformation of Y |
|---|---|
| -3 | $-Y^{-3}$ |
| -2 | $-Y^{-2}$ |
| -1 | $-Y^{-1} = -1/Y$ |
| -0.5 | $-Y^{-0.5} = -1/\sqrt{Y}$ |
| 0 | log(Y) |
| 0.5 | $Y^{0.5} = \sqrt{Y}$ |
| 1 | Y |
| 2 | $Y^2$ |
| 3 | $Y^3$ |

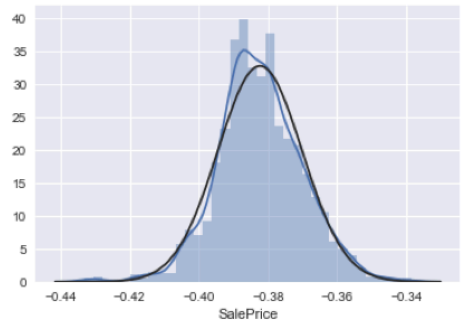
```
stats.boxcox(df.SalePrice)[1]
```

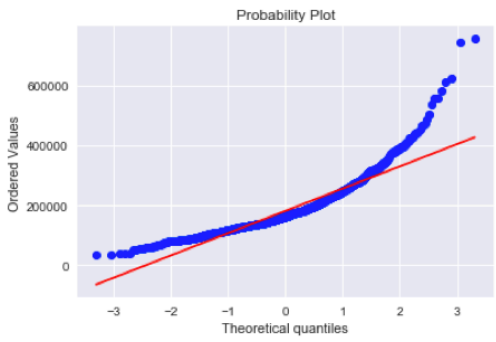
```
-0.077151566178292413
```

```
#applying power transformation
dfp = df.copy()
dfp.SalePrice = -dfp.SalePrice ** -0.07
```

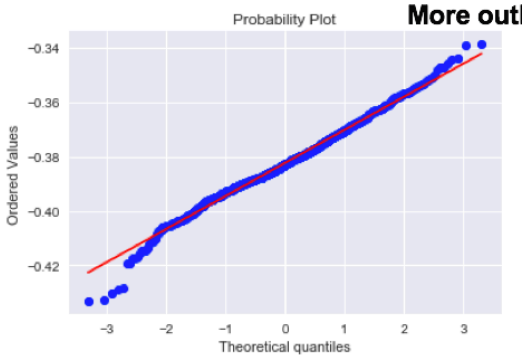
```
#applying log transformation
dfl = df.copy()
dfl.SalePrice = np.log(dfl.SalePrice)
```




More outliers?




More outliers?

## 8. Variable Creation

- How can we use categorical values?

```
df.BldgType.unique()
```

```
array(['1Fam', '2fmCon', 'Duplex', 'TwnhsE', 'Twnhs'], dtype=object)
```

- Create separate variables for all values:
  - BldgType_1Fam = {0, 1}
  - BldgType_2fmCon = {0, 1}
  - BldgType_Duplex = {0, 1}

```
df = pd.get_dummies(df)
```

```
df.BldgType_1Fam
```

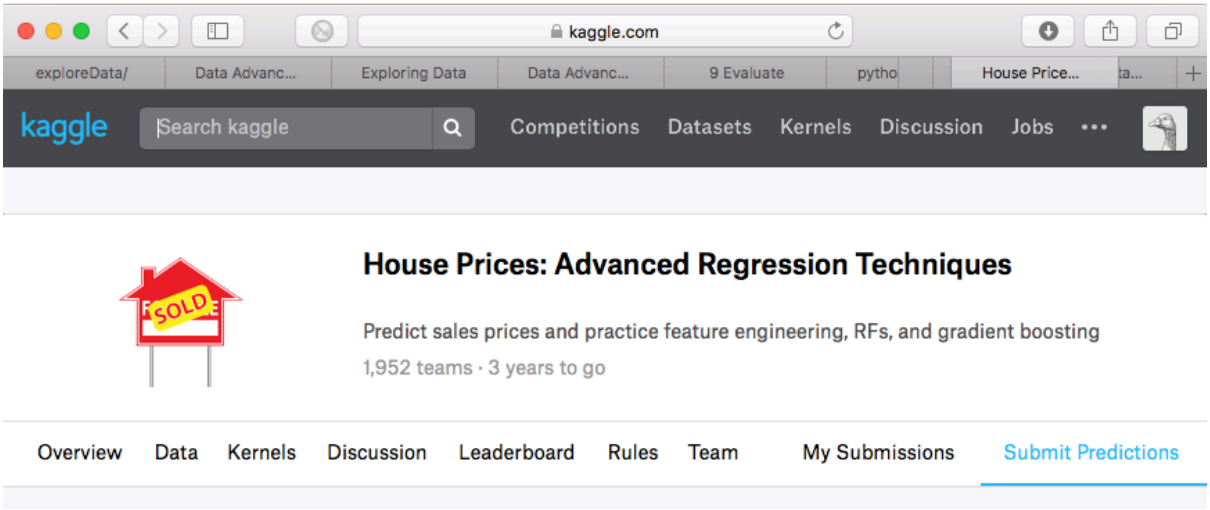
```
0     1
1     1
2     1
3     1
4     1
5     1
6     1
7     1
8     1
9     0
10    1
11    1
12    1
13    1
14    1
15    1
16    1
17    0
18    1
19    1
```

## 9. Evaluation

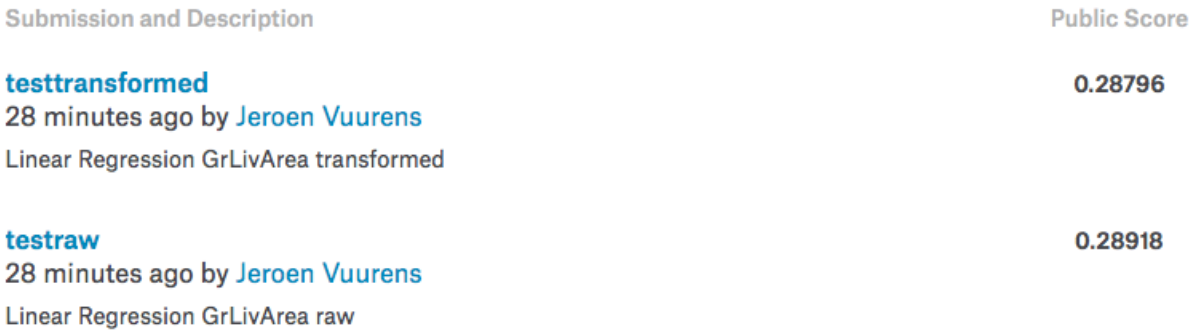
```
import statsmodels.formula.api as sm
result = sm.ols(formula="SalePrice ~ GrLivArea", data=df).fit()
```

```
df_test['SalePrice'] = result.predict(df_test)
```

- Average % from actual SalePrice on trainset
  - Original: 22.55%
  - Transformed: 21.83%
- But there is a better way to evaluate




- Learn model training data
- Evaluate test data (no cheating)




- Mean Root Squared Error (lower is better)
- Evaluate over separate test set is objective