Data mining and Adv. Statistical Modeling

Linear Regression

Prepared By :- Karan Singal

Under the guidance of :-Gitimoni Saikia

Objective

By using Machine learning model

Can we predict house prices ??

Ref:- https://www. kaggle.com/house-price-prediction/data

Prediction of Housing price

Steps to follow for Machine learning Idea

- Gathering and Exploring the data.
- Data Preparation
- Splitting the data
- Initializing the Model and Parameters
- Training and Cross-Validation
- Evaluation

• Gathering and Exploring the data.

```
In [1]: import pandas as pd

In [2]: import numpy as np

In [3]: import seaborn as sns

In [4]: import matplotlib as mpl

In [5]: import matplotlib.pyplot as plt

In [6]: import scipy.stats as stats
```

```
df= pd.read_csv('C:\\Desktop\\Project\\DM\\train.csv')

df.shape
df.head()
df.describe()
```

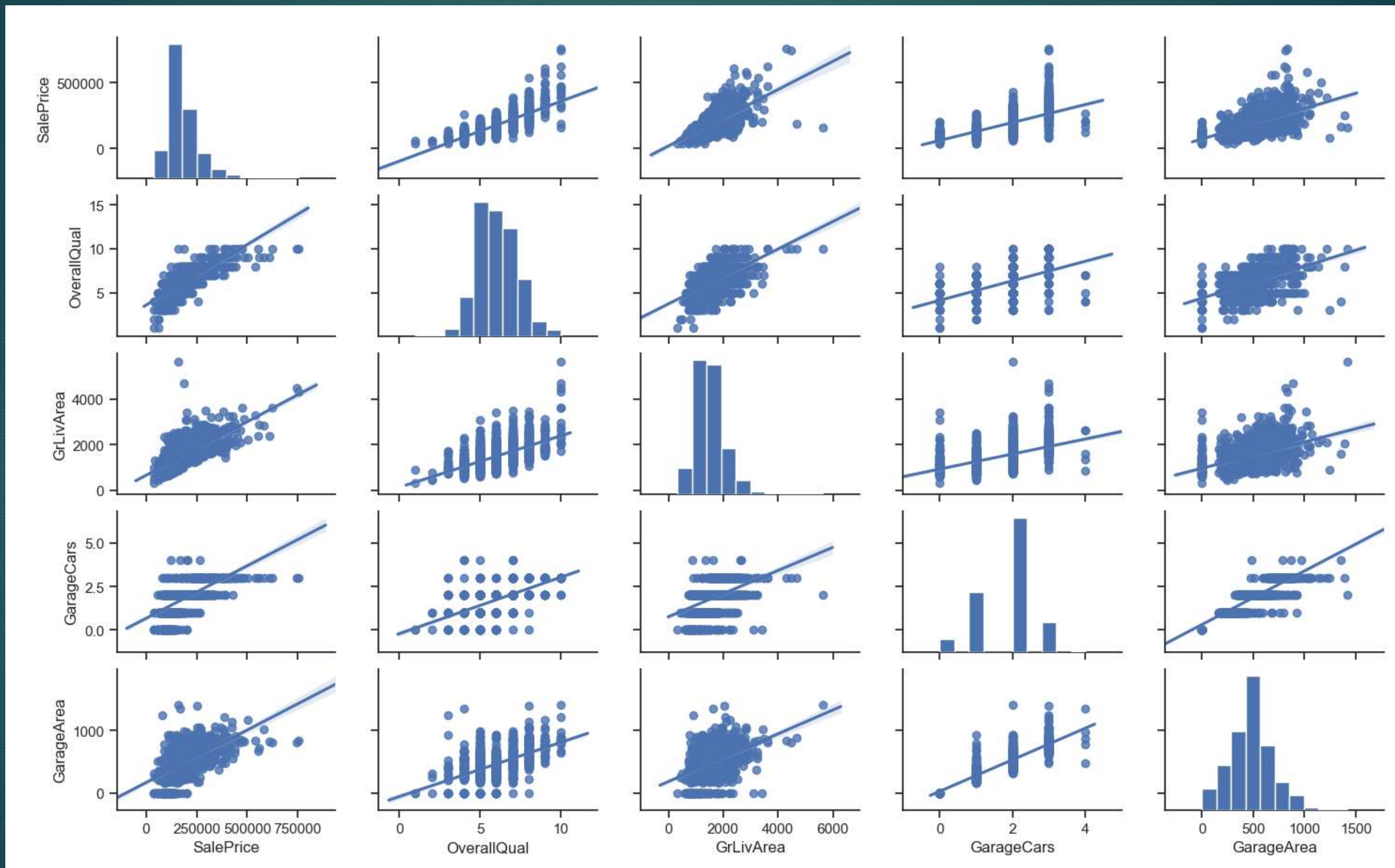
```
df.shape
(1460, 81)
```

	PoolArea	MiscVal	MoSold	YrSold	SalePrice
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000
mean	2.758904	43.489041	6.321918	2007.815753	180921.195890
std	40.177307	496.123024	2.703626	1.328095	79442.502883
min	0.000000	0.000000	1.000000	2006.000000	34900.000000
25%	0.000000	0.000000	5.000000	2007.000000	129975.000000
50%	0.000000	0.000000	6.000000	2008.000000	163000.000000
75%	0.000000	0.000000	8.000000	2009.000000	214000.000000
max	738.000000	15500.000000	12.000000	2010.000000	755000.000000

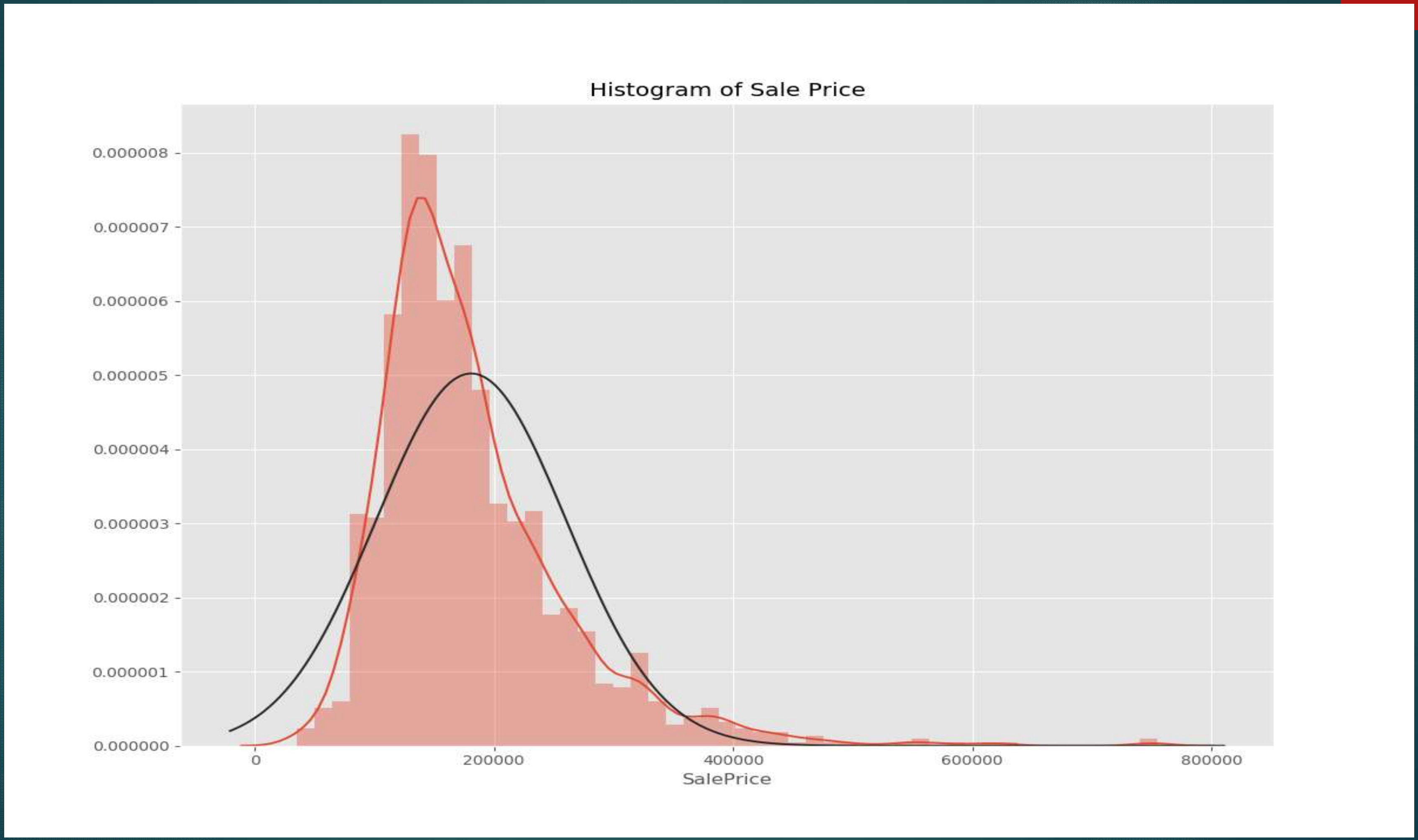
DATAFRAME

Index	SalePrice	OverallQual	GrLivArea	GarageCars	GarageArea
0	208500	7	1710	2	548
1	181500	6	1262	2	460
2	223500	7	1786	2	608
3	140000	7	1717	3	642
4	250000	8	2198	3	836
5	143000	5	1362	2	480

Pair plot

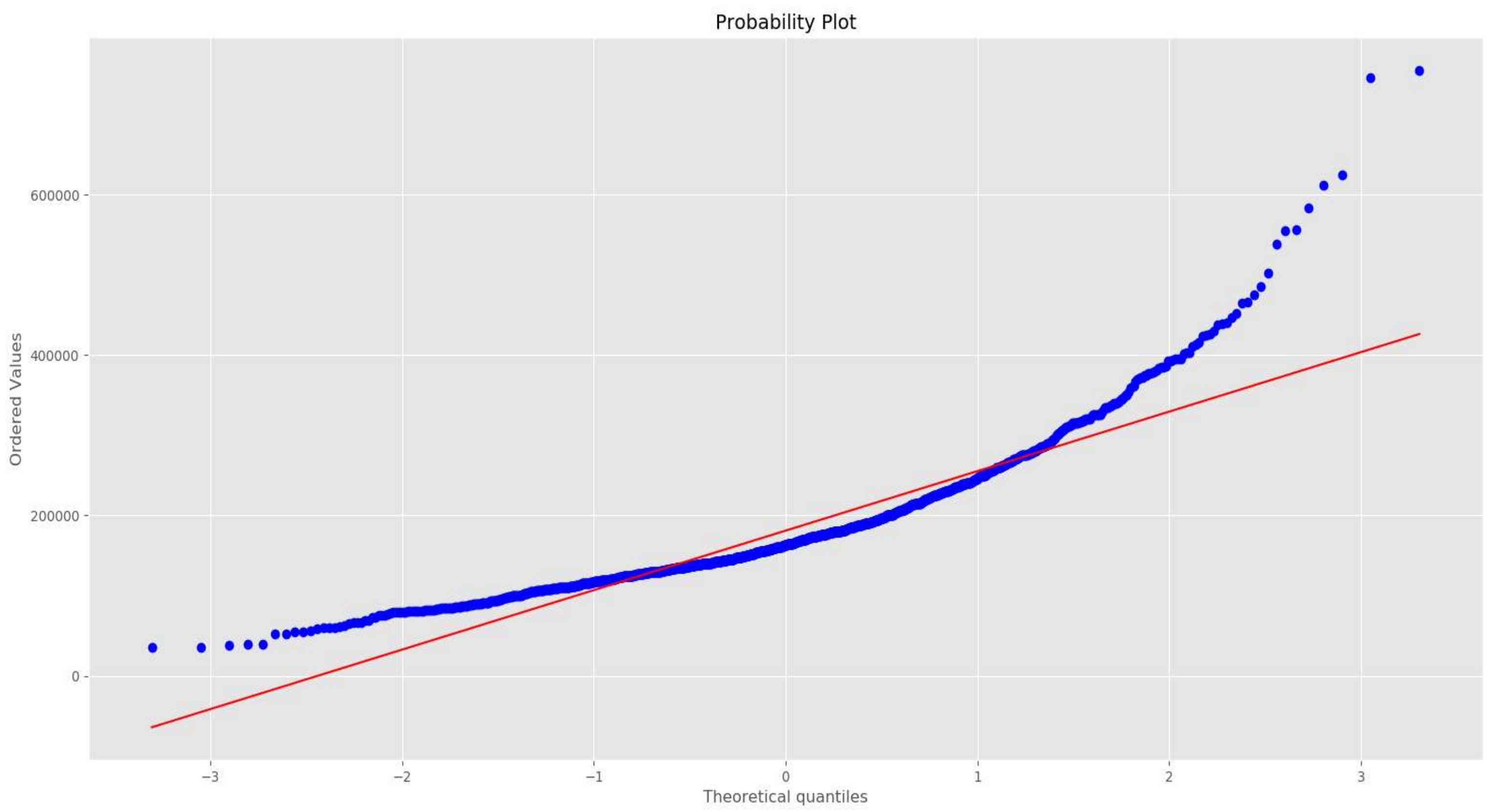

SalePrice
OverallQual
GrLivArea
GarageCars
GarageArea
500000
0
15
10
5
4000
2000
0
5.0
2.5
0.0
1000
0
0
250000 500000 750000
0
5
10
0
2000
4000
6000
0
2
4
0
500
1000
1500
SalePrice
OverallQual
GrLivArea
GarageCars
GarageArea

Histogram



Histogram of Sale Price
0.000008
0.000007
0.000006
0.000005
0.000004
0.000003
0.000002
0.000001
0.000000
0
200000
400000
600000
800000
SalePrice

Probability plot



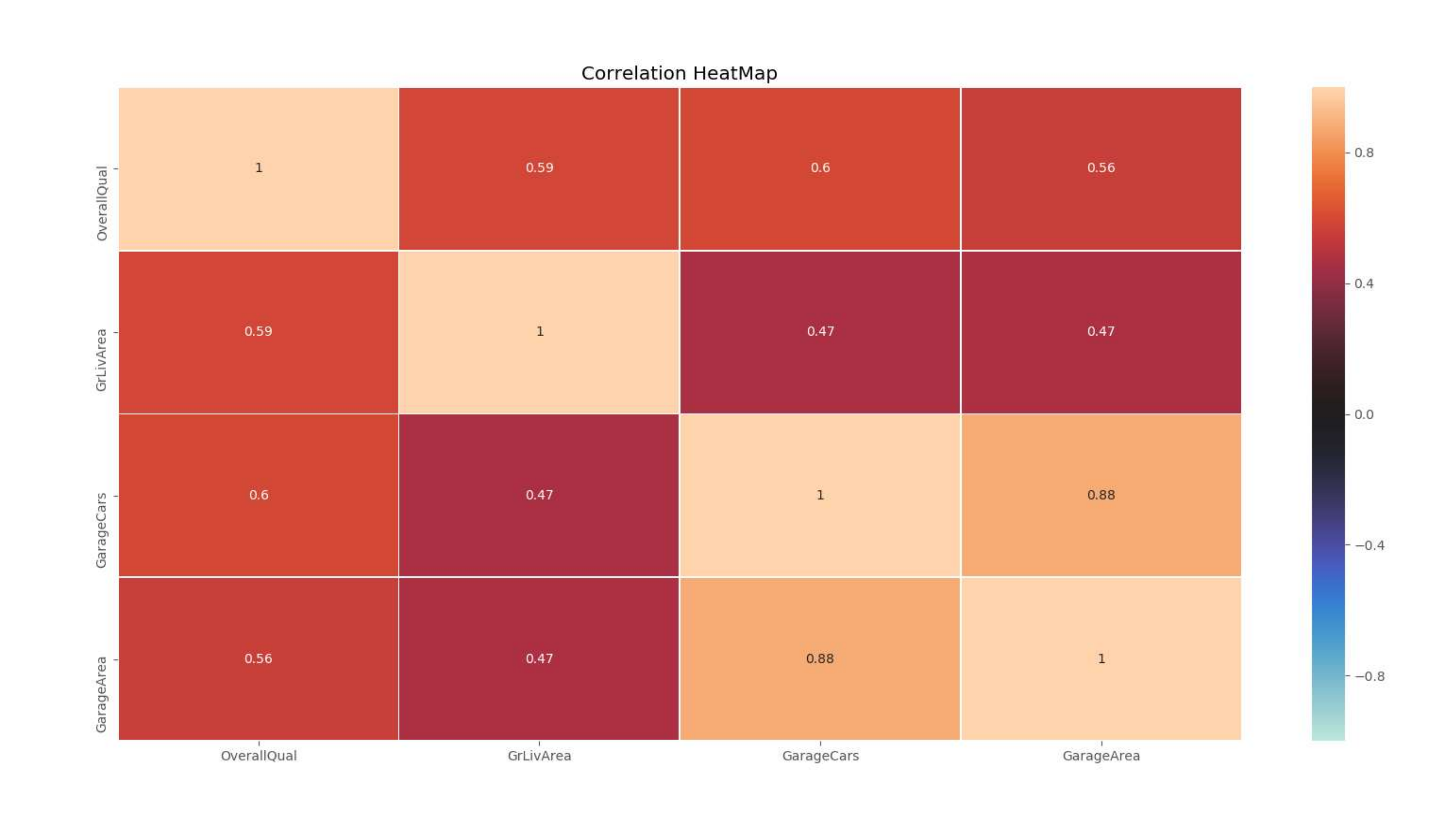
Data Preparation

```
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 5 columns):
SalePrice      1460 non-null int64
OverallQual    1460 non-null int64
GrLivArea      1460 non-null int64
GarageCars     1460 non-null int64
GarageArea     1460 non-null int64
dtypes: int64(5)
memory usage: 57.1 KB
```

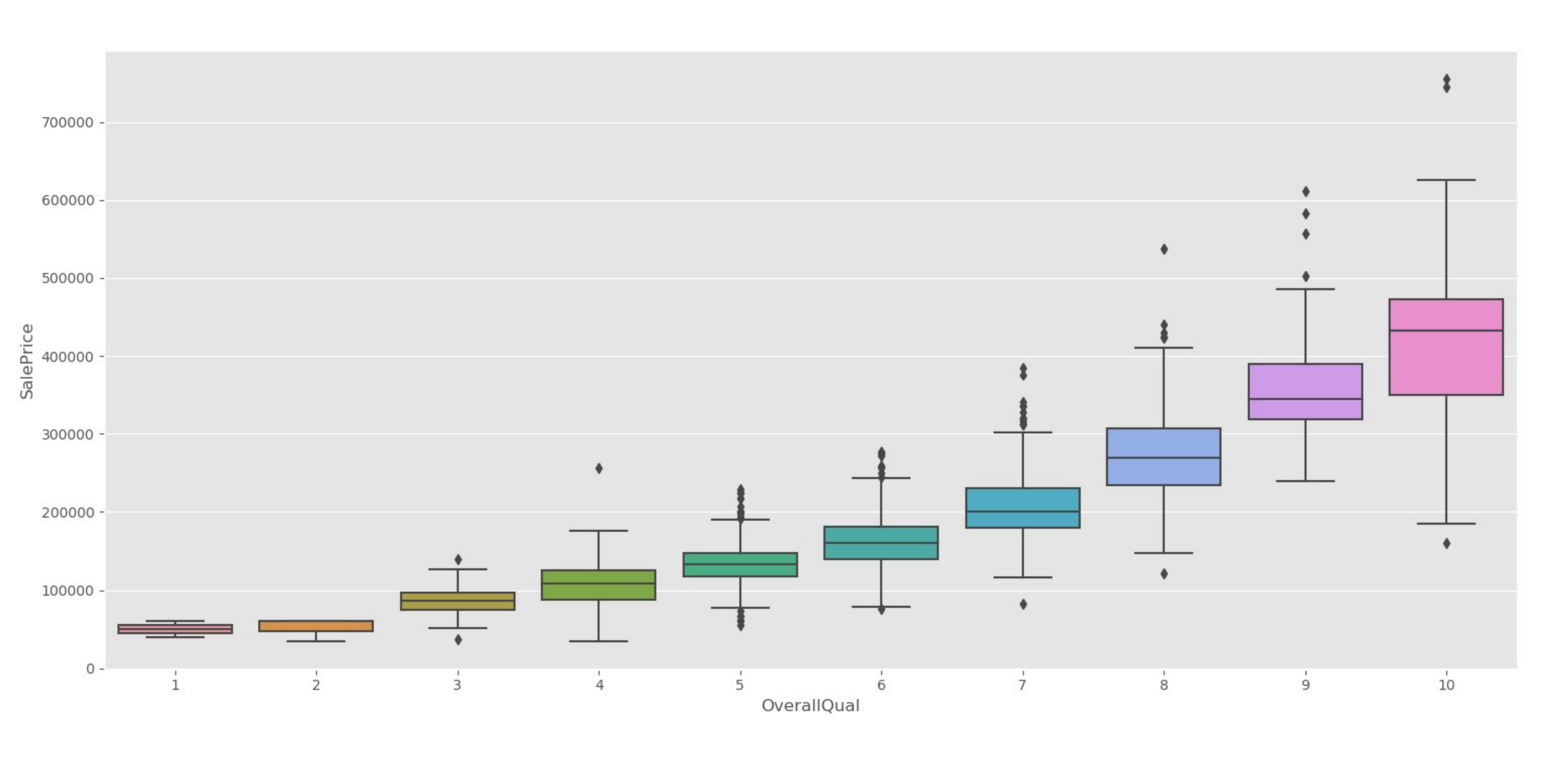
No null values

Reduced variables from 81 to 5

Correlation Heat Map



Box Plot



SalePrice
700000
600000
500000
400000
300000
200000
100000
0
1
2
3
4
5
6
7
8
9
10
OverallQual

Splitting the data (70:30)

```
In [72]: X_train.describe()
Out[72]:
```

	OverallQual	GrLivArea	GarageCars	GarageArea
count	1022.000000	1022.000000	1022.000000	1022.000000
mean	6.128180	1529.242661	1.783757	477.120352
std	1.371391	530.971805	0.730751	208.443296
min	1.000000	334.000000	0.000000	0.000000
25%	5.000000	1142.500000	1.000000	350.500000
50%	6.000000	1476.500000	2.000000	484.000000
75%	7.000000	1794.250000	2.000000	576.000000
max	10.000000	5642.000000	4.000000	1418.000000

```
In [84]: y_train.describe()
Out[84]:
```

	SalePrice
count	1022.000000
mean	181312.692759
std	77617.461005
min	34900.000000
25%	130000.000000
50%	165000.000000
75%	215000.000000
max	745000.000000

```
Training prediction variable contains : 1022 rows
Training independent variable contains : 1022 rows
Testing prediction variable contains : 438 rows
Testing independent variable contains : 438 rows
```

Training and Cross-Validation

	features	coeficients
0	OverallQual	26812.001964
1	GrLivArea	44.448933
2	GarageCars	16102.611108
3	GarageArea	29.393249

Intercept: -93716.555

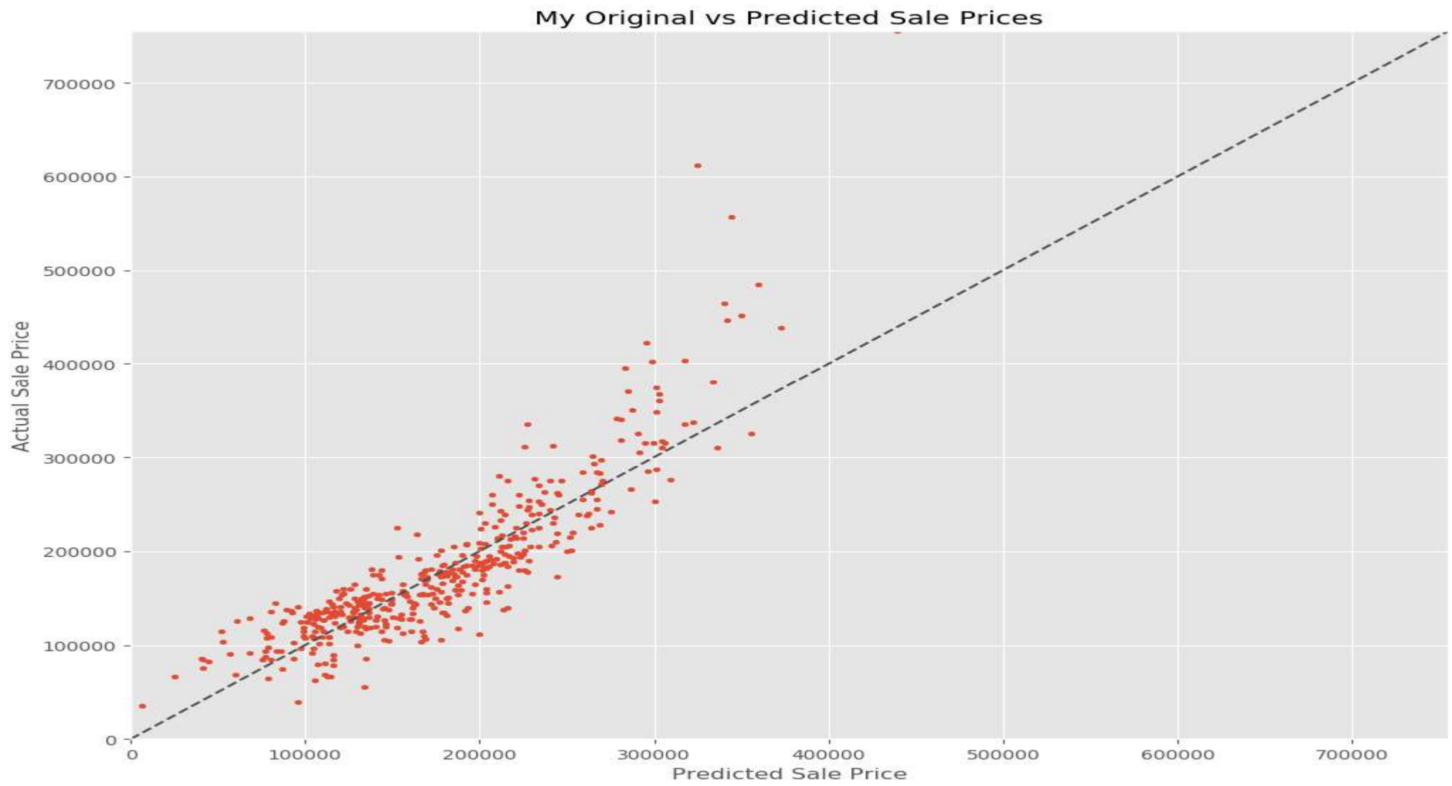
Training set score: 0.73

K-Fold Cross Validation

```
In [103]: cv_scores
Out[103]: array([0.78321963, 0.75020102, 0.74231776, 0.73157536, 0.66676168])

In [104]: cv_scores.mean()
Out[104]: 0.7348150907545123
```

Average cv Score is 0.73

Actual vs Predicted Sale Price



Adjusted R-squared & RMSE

Adjusted R-squared

```
Training set adj r2: 0.7267780468535634
```

```
Average 10-Fold CV adj r2: 0.7395639654485078
```

RMSE

```
Root Mean Squared Error of Training Set: 40471.77542233575
Root Mean Squared Error of Testing Set: 40511.41317382875
```

OLS Result

```
In [148]: print(sm_model.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:              SalePrice   R-squared:                       0.728
Model:                            OLS   Adj. R-squared:                  0.727
Method:                 Least Squares   F-statistic:                     680.0
Date:                Wed, 16 Oct 2019   Prob (F-statistic):         1.48e-285
Time:                        01:40:26   Log-Likelihood:                -12292.
No. Observations:                1022   AIC:                         2.459e+04
Df Residuals:                    1017   BIC:                         2.462e+04
Df Model:                           4
Covariance Type:            nonrobust
===============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
const       -9.372e+04   5844.549    -16.035      0.000   -1.05e+05   -8.22e+04
OverallQual  2.681e+04   1273.996     21.046      0.000    2.43e+04    2.93e+04
GrLivArea     44.4489      3.032     14.661      0.000      38.500      50.398
GarageCars    1.61e+04   3806.691      4.230      0.000    8632.743    2.36e+04
GarageArea    29.3932     13.079      2.247      0.025       3.727      55.059
==============================================================================
Omnibus:                      248.085   Durbin-Watson:                   2.052
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7009.502
Skew:                           0.443   Prob(JB):                         0.00
Kurtosis:                      15.799   Cond. No.                     7.88e+03
==============================================================================
```

Data mining and Adv. Statistical Modeling

Classification

Prepared By :- Karan Singal

Under the guidance of :-Gitimoni Saikia

Objective

Occupancy Detection Dataset

Occupancy detection sensors like environmental sensors, such as light, temperature, humidity and CO_2 sensors, can detect the change in the environment due to the presence of a human*

Ref:- https://en.wikipedia.org/wiki/Occupancy_sensor

- Gathering and Exploring the data.

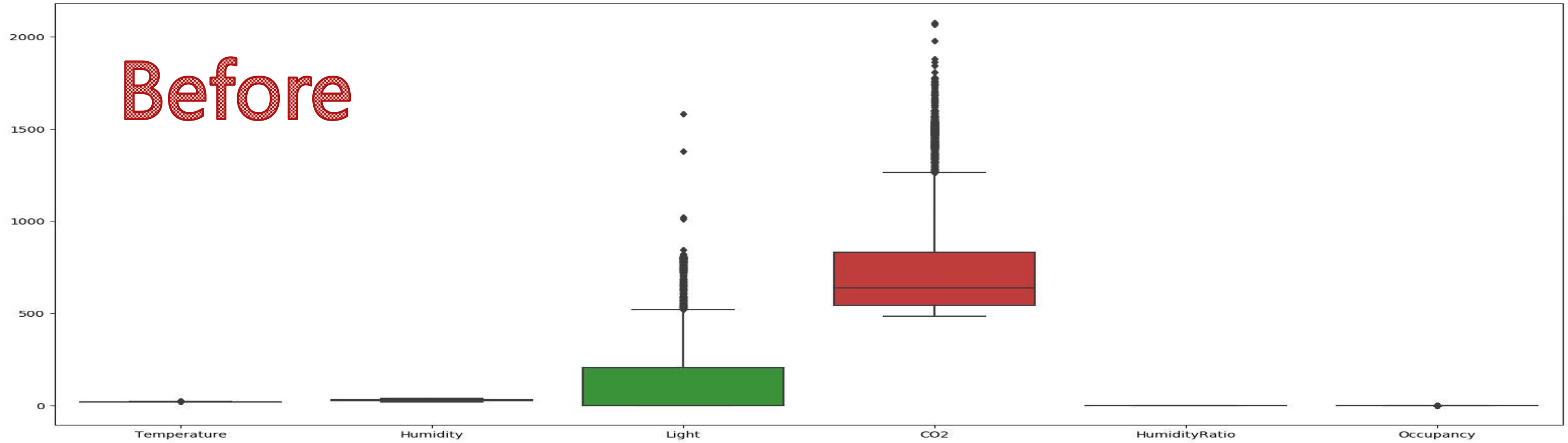
```
In [8]: import pandas as pd

In [9]: import os

In [10]: import numpy as np

In [11]: import seaborn as sbn
```

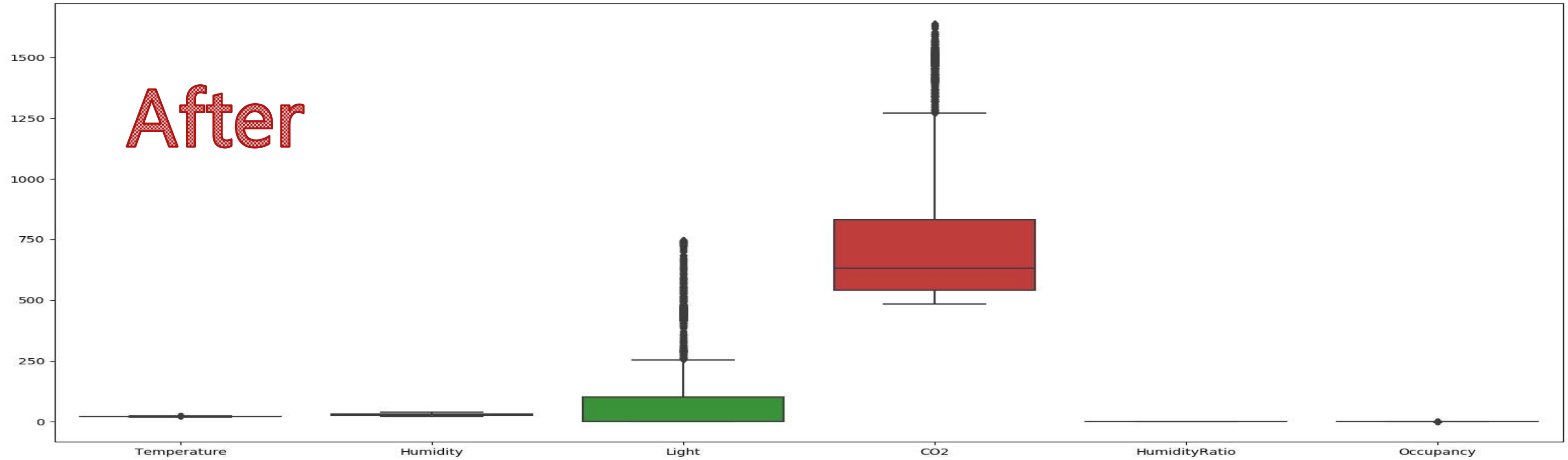
```
df= pd.read_csv('C:\\Desktop\\Project\\DM\\train.csv')

df.shape
df.head()
df.describe()
```

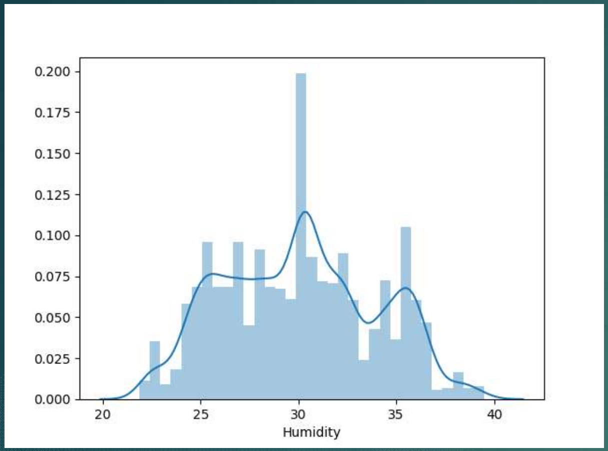
```
df.isnull().values.any()
False
```

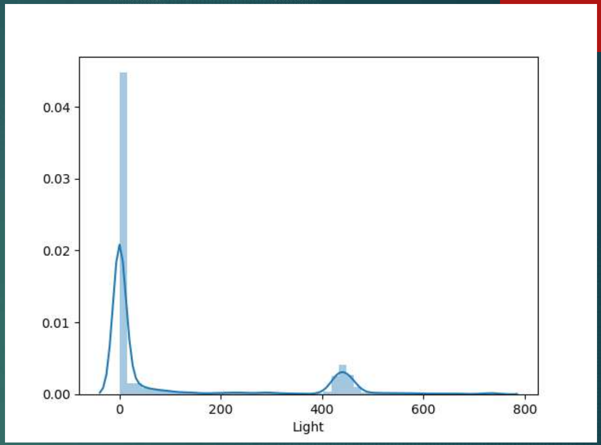
```
In [7]: df.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9752 entries, 1 to 9752
Data columns (total 7 columns):
date             9752 non-null object
Temperature      9752 non-null float64
Humidity         9752 non-null float64
Light            9752 non-null float64
CO2              9752 non-null float64
HumidityRatio    9752 non-null float64
Occupancy        9752 non-null int64
dtypes: float64(5), int64(1), object(1)
memory usage: 609.5+ KB
```

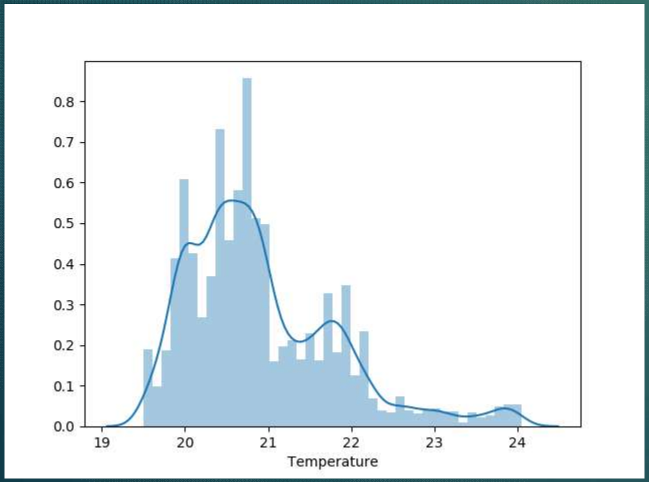
Index	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
1	2015-02-11 14:48:00	21.76	31.1333	437.333	1029.67	0.00502101	1
2	2015-02-11 14:49:00	21.79	31	437.333	1000	0.00500858	1
3	2015-02-11 14:50:00	21.7675	31.1225	434	1003.75	0.00502157	1
4	2015-02-11 14:51:00	21.7675	31.1225	439	1009.5	0.00502157	1
5	2015-02-11 14:51:59	21.79	31.1333	437.333	1005.67	0.0050303	1

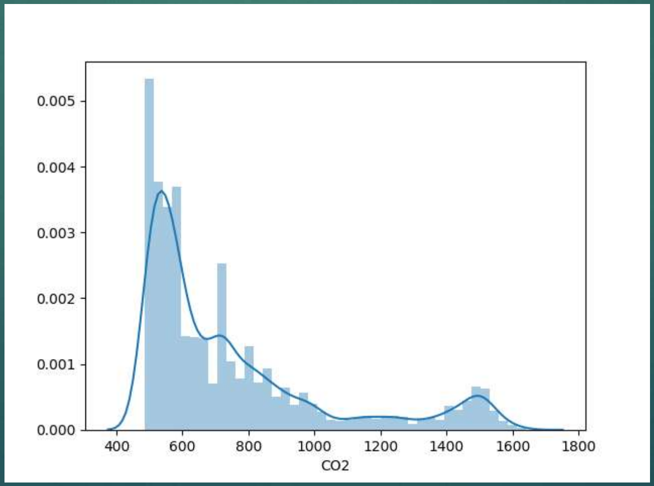

2000
1500
1000
500
0
Before
Temperature
Humidity
Light
CO2
HumidityRatio
Occupancy

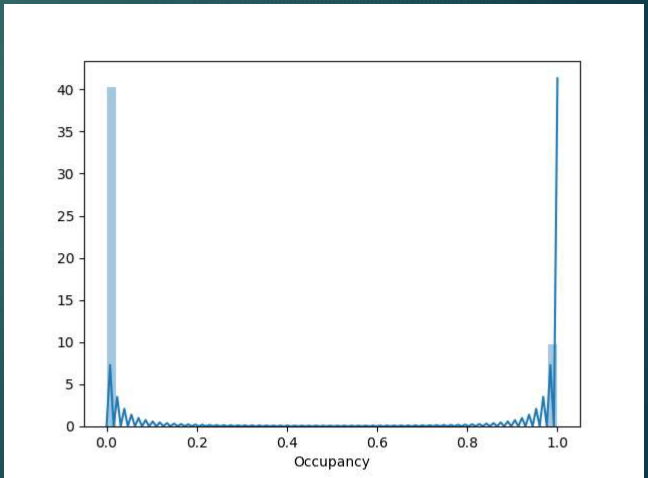

1500
1250
1000
750
500
250
0
After
Temperature
Humidity
Light
CO2
HumidityRatio
Occupancy


Humidity

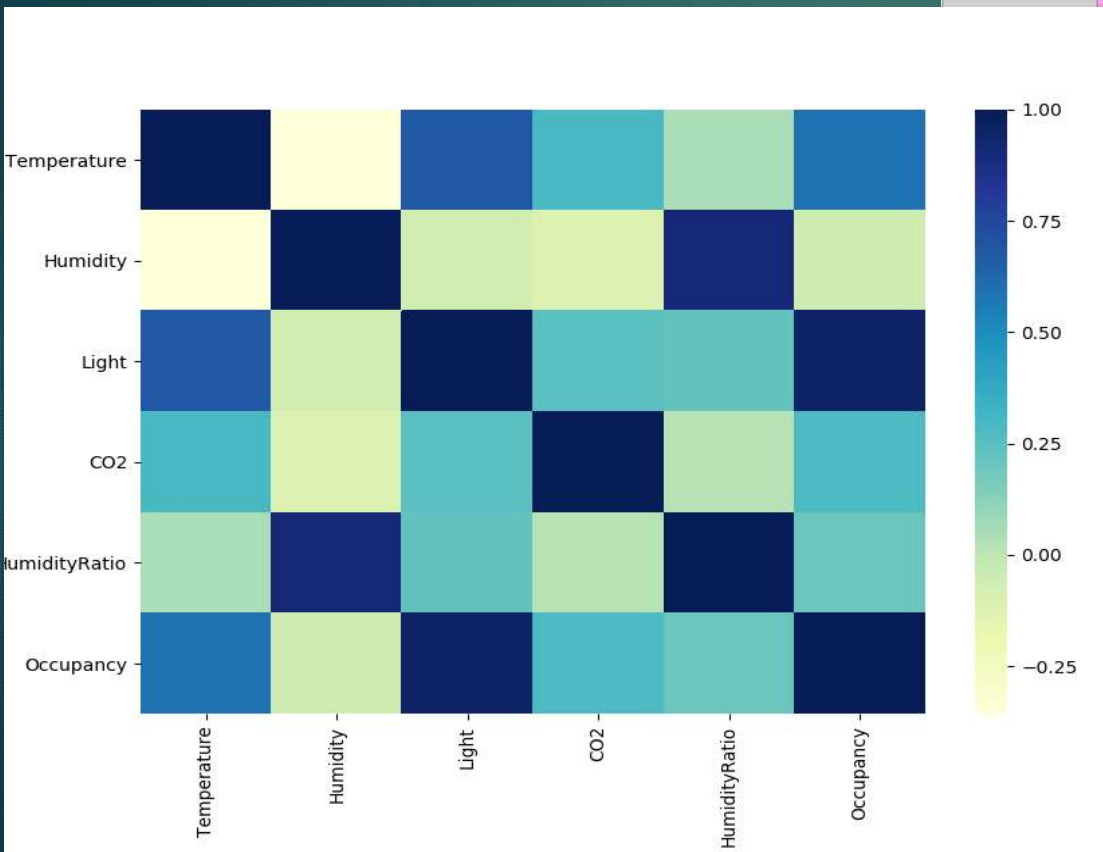

Light


Temperature


CO2


Occupancy

Index	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
Temperature	1	-0.42178	0.672874	0.238632	0.00990336	0.519537
Humidity	-0.42178	1	-0.11962	-0.10454	0.901006	-0.0584362
Light	0.672874	-0.11962	1	0.225416	0.191345	0.92952
CO2	0.238632	-0.10454	0.225416	1	0.0218437	0.279379
	.00990336	0.901006	0.191345	0.0218437	1	0.193572
	.519537	-0.0584362	0.92952	0.279379	0.193572	1



HeatMap

Splitting

```
In [49]: trainX, testX, trainy, testy = train_test_split(X, y, test_size=0.3, shuffle=False,
random_state=1)
```

Acc Score

```
In [55]: print('My Accuracy Score is',score)
My Accuracy Score is 0.9953933380581148
```

```
feature=0, name=Temperature, score=0.631
feature=1, name=Humidity, score=0.631
feature=2, name=Light, score=0.995
feature=3, name=CO2, score=0.515
feature=4, name=HumidityRatio, score=0.631
```

Conclusion

- The main goal of the project is to create a model that predicts whether a room is occupied through a change in room temperature, humidity, amount of light and CO_2
- The accuracy results show a high percentage in predicting room occupancy which means, our model is good

Thank You