

```
In [99]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn import datasets
```

```
In [100]: dataset = datasets.load_boston()
dir(dataset)
```

```
Out[100]: ['DESCR', 'data', 'feature_names', 'filename', 'target']
```

```
In [101]: feature = pd.DataFrame(dataset.data, columns=dataset.feature_names)
feature.head(3)
```

```
Out[101]:
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03

```
In [102]: label = pd.DataFrame(dataset.target)
label.head(3)
```

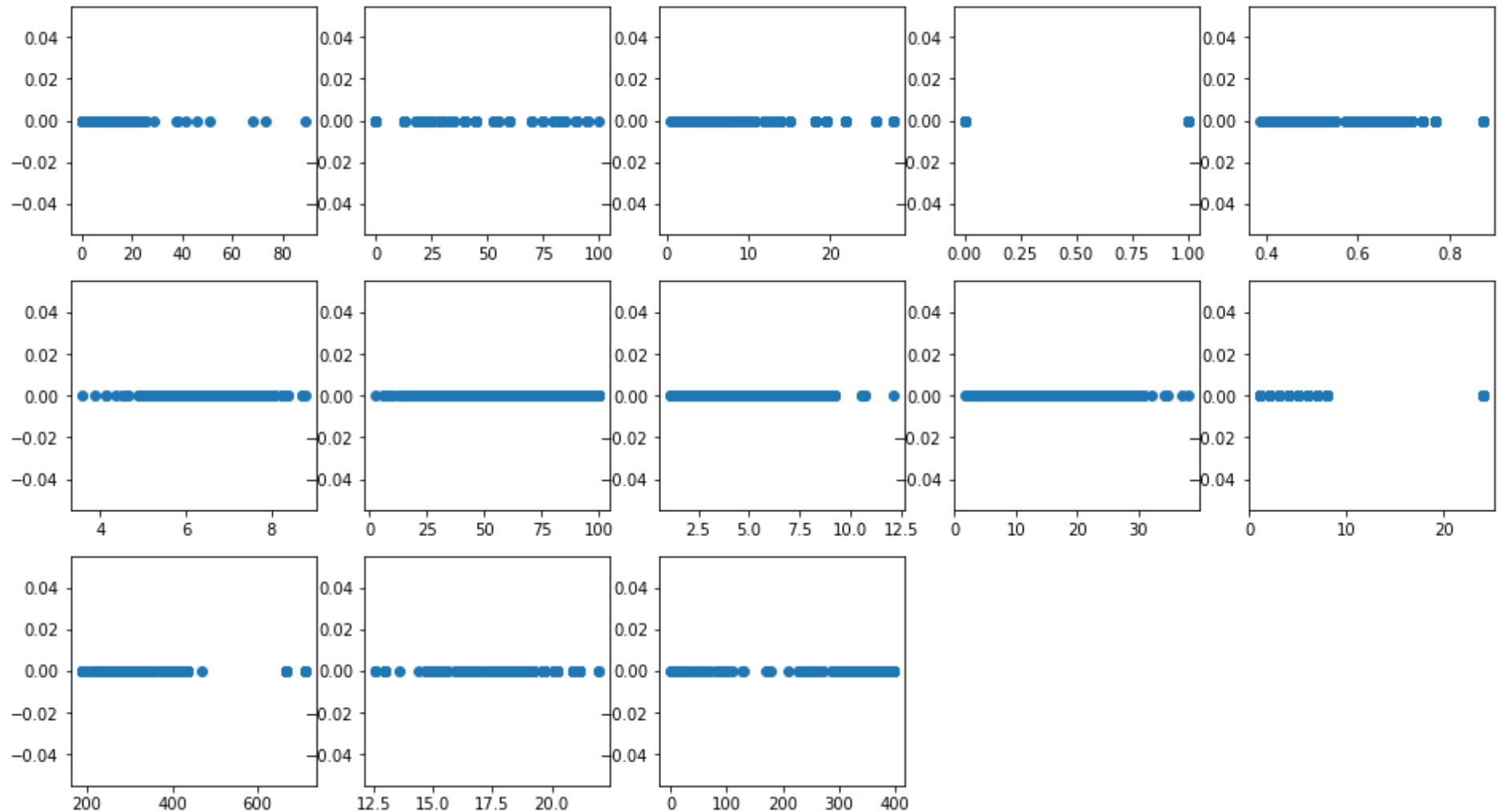
```
Out[102]:
```

	0
0	24.0
1	21.6
2	34.7

Univariate Analysis

```
In [103]: plt.figure(figsize=(16,9))
```

```
plt.subplot(3,5,1)
plt.plot(feature['CRIM'],np.zeros_like(feature['CRIM']),'o')
plt.subplot(3,5,2)
plt.plot(feature['ZN'],np.zeros_like(feature['ZN']),'o')
plt.subplot(3,5,3)
plt.plot(feature['INDUS'],np.zeros_like(feature['INDUS']),'o')
plt.subplot(3,5,4)
plt.plot(feature['CHAS'],np.zeros_like(feature['CHAS']),'o')
plt.subplot(3,5,5)
plt.plot(feature['NOX'],np.zeros_like(feature['NOX']),'o')
plt.subplot(3,5,6)
plt.plot(feature['RM'],np.zeros_like(feature['RM']),'o')
plt.subplot(3,5,7)
plt.plot(feature['AGE'],np.zeros_like(feature['AGE']),'o')
plt.subplot(3,5,8)
plt.plot(feature['DIS'],np.zeros_like(feature['DIS']),'o')
plt.subplot(3,5,9)
plt.plot(feature['LSTAT'],np.zeros_like(feature['LSTAT']),'o')
plt.subplot(3,5,10)
plt.plot(feature['RAD'],np.zeros_like(feature['RAD']),'o')
plt.subplot(3,5,11)
plt.plot(feature['TAX'],np.zeros_like(feature['TAX']),'o')
plt.subplot(3,5,12)
plt.plot(feature['PTRATIO'],np.zeros_like(feature['PTRATIO']),'o')
plt.subplot(3,5,13)
plt.plot(feature['B'],np.zeros_like(feature['B']),'o')
plt.show()
```



Multivariate Analysis

```
In [104... data = pd.concat([feature, label], axis=1)
data.head(2)
```

```
Out[104...   CRIM  ZN  INDUS  CHAS  NOX   RM  AGE  DIS  RAD  TAX  PTRATIO   B  LSTAT   0
0  0.00632  18.0    2.31    0.0  0.538  6.575  65.2  4.0900  1.0  296.0    15.3  396.9  4.98  24.0
```


[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

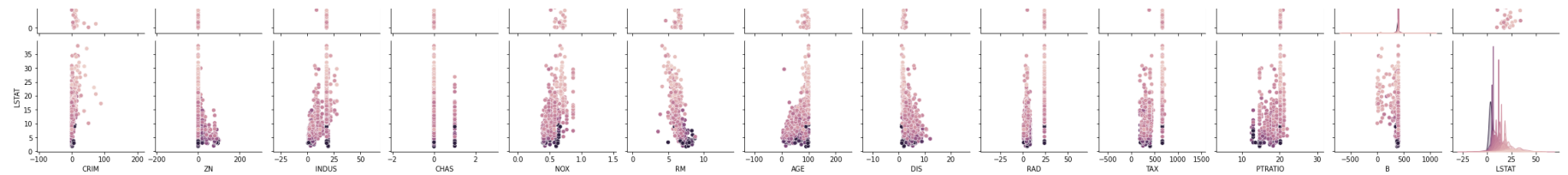
[illegible]

[illegible]

[illegible]

[illegible]

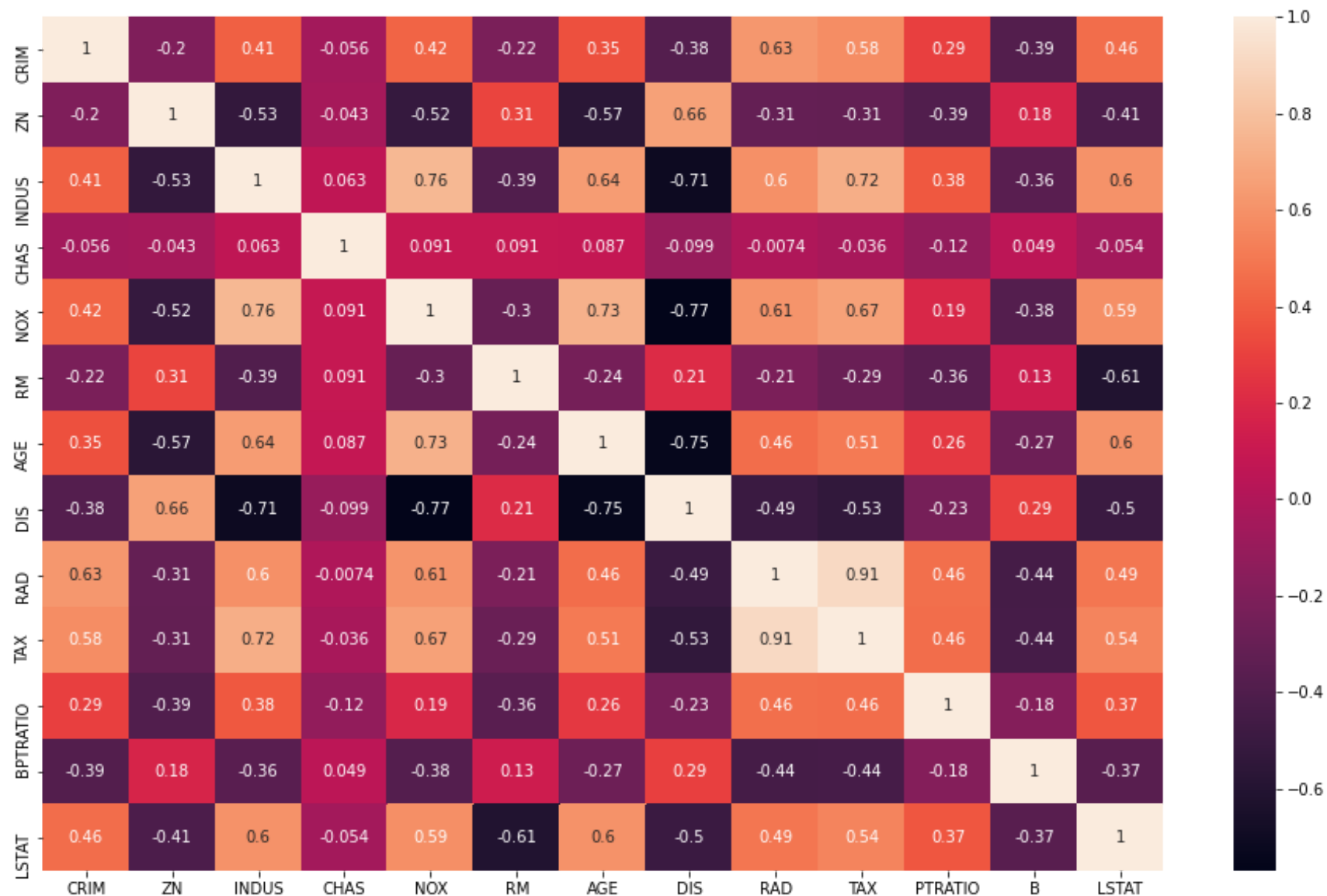




```
In [108... corr = feature.corr()
corr.head(2)
```

```
Out[108...      CRIM      ZN      INDUS      CHAS      NOX      RM      AGE      DIS      RAD      TAX      PTRATIO      B      LSTAT
CRIM  1.000000 -0.200469  0.406583 -0.055892  0.420972 -0.219247  0.352734 -0.379670  0.625505  0.582764  0.289946 -0.385064  0.455621
ZN    -0.200469  1.000000 -0.533828 -0.042697 -0.516604  0.311991 -0.569537  0.664408 -0.311948 -0.314563 -0.391679  0.175520 -0.412995
```

```
In [109... plt.figure(figsize=(16,10))
sns.heatmap(corr,annot=True)
plt.show()
```



```
In [110... x = data.drop('Target',axis=1)
y = (data['Target']).astype('int')
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25)
```

Applying random forest classifier


```
In [111... model = RandomForestRegressor()  
model.fit(x_train,y_train)
```

```
Out[111... RandomForestRegressor()
```

```
In [112... print(f"Train_model_score: {model.score(x_train,y_train)}")  
print(f"Test_model_score: {model.score(x_test,y_test)}")
```

```
Train_model_score: 0.9806544234146064  
Test_model_score: 0.8993272343814955
```

Hyper Tunning By Using GridSearchCV

```
In [113... grid_params = {"n_estimators" : [10,40,65,100],  
                  "max_depth" : range(2,20,1),  
                  "min_samples_leaf" : range(1,10,1),  
                  "min_samples_split" : range(2,10,1),  
                  "max_features" : ['auto','log2']  
                  }
```

```
In [114... from sklearn.model_selection import GridSearchCV  
grid_search = GridSearchCV(estimator=model,param_grid=grid_params,cv=5,n_jobs=-1,verbose=3)
```

```
In [115... grid_search.fit(x_train,y_train)
```

Fitting 5 folds for each of 10368 candidates, totalling 51840 fits

```
Out[115... GridSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=-1,  
              param_grid={'max_depth': range(2, 20),  
                           'max_features': ['auto', 'log2'],  
                           'min_samples_leaf': range(1, 10),  
                           'min_samples_split': range(2, 10),  
                           'n_estimators': [10, 40, 65, 100]}},  
              verbose=3)
```

```
In [116... grid_search.best_params_
```

```
Out[116... {'max_depth': 10,  
          'max_features': 'auto',  
          'min_samples_leaf': 1,  
          'min_samples_split': 2,  
          'n_estimators': 10}
```

```
In [119... model2 = RandomForestRegressor(n_estimators = 10,max_depth = 10,min_samples_split = 2,min_samples_leaf = 1,max_features = 'auto')
model2
```

```
Out[119... RandomForestRegressor(max_depth=10, n_estimators=10)
```

```
In [120... model2.fit(x_train,y_train)
```

```
Out[120... RandomForestRegressor(max_depth=10, n_estimators=10)
```

```
In [122... model2.score(x_test,y_test)
```

```
Out[122... 0.8971377179058918
```

```
In [125... y_predicted = model2.predict(x_test)
```

```
In [131... from sklearn.metrics import r2_score,mean_squared_error
print(f'R^2 : {r2_score(y_test,y_predicted)}')
print(f'MSE : {mean_squared_error(y_test,y_predicted)}')
print(f'RMSE: {np.sqrt(mean_squared_error(y_test, y_predicted))}')
```

```
R^2 : 0.8971377179058918
MSE : 7.854599319462901
RMSE: 2.802605808789902
```

```
In [ ]:
```