

# ex3-multivariate-linear-regression

August 5, 2024

```
[1]: import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

## 1 Load the Boston Housing DataSet

```
[2]: boston = pd.read_csv("../datasets/boston_house_prices.csv")

boston.head()
```

```
[2]:
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	\
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	

	B	LSTAT	MEDV
0	396.90	4.98	24.0
1	396.90	9.14	21.6
2	392.83	4.03	34.7
3	394.63	2.94	33.4
4	396.90	5.33	36.2

## 2 Check if our data has null values and count them up for each column

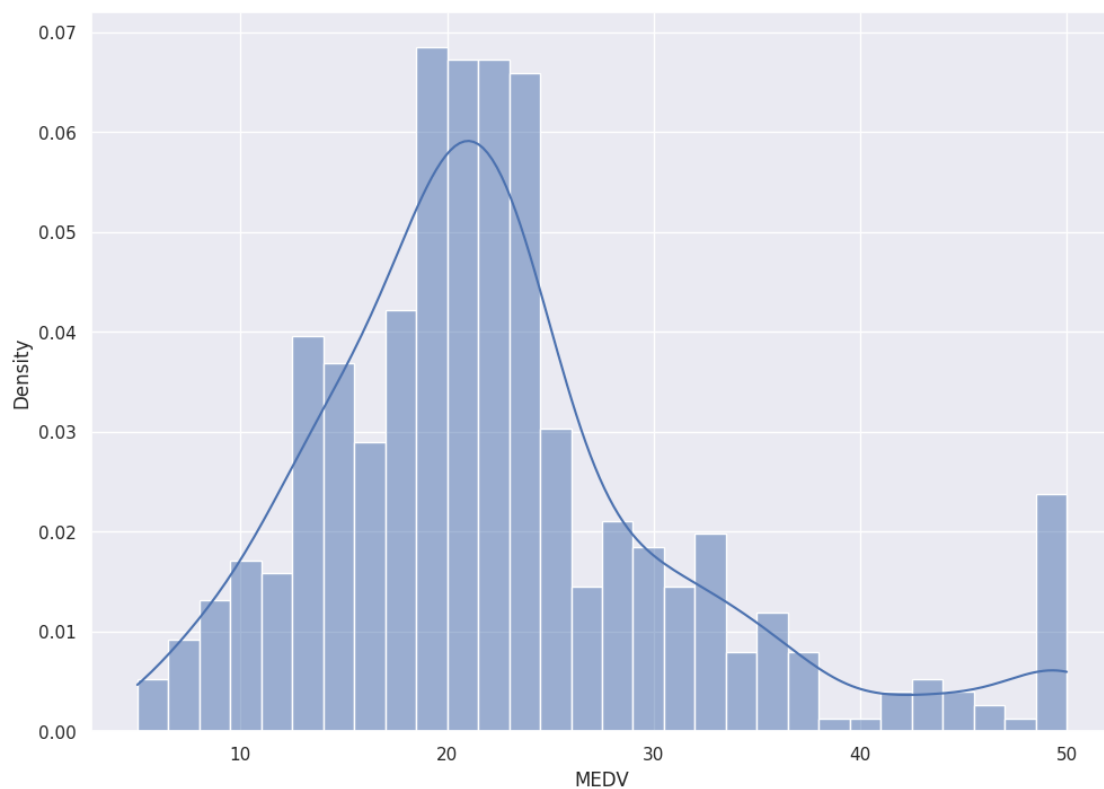
```
[3]: boston.isnull().sum()
```

```
[3]: CRIM      0
      ZN       0
      INDUS   0
      CHAS    0
      NOX     0
      RM      0
```

```
AGE          0
DIS          0
RAD          0
TAX          0
PTRATIO     0
B           0
LSTAT       0
MEDV       0
dtype: int64
```

### 3 Data Visualization

```
[4]: # set the size of the figure
sns.set(rc={"figure.figsize": (11.7, 8.27)})
# plot a histogram showing the distribution of the target values
sns.histplot(boston["MEDV"], bins=30, kde=True, stat="density")
plt.show()
```



## 4 Correlation matrix

```
[5]: # compute the pair wise correlation for all columns
correlation_matrix = boston.corr().round(2)
```

```
[6]: # use the heatmap function from seaborn to plot the correlation matrix
# annot = True to print the values inside the square
sns.heatmap(data=correlation_matrix, annot=True)
```

```
[6]: <Axes: >
```



## 5 Observations

From the above correlation plot we can see that MEDV is strongly correlated to LSTAT, RM

RAD and TAX are strongly correlated, so we don't include this in our features together to avoid multi-collinearity

```
[7]: plt.figure(figsize=(20, 5))

features = ["LSTAT", "RM"]
target = boston["MEDV"]
```

```
for i, col in enumerate(features):
    plt.subplot(1, len(features), i + 1)
    x = boston[col]
    y = target
    plt.scatter(x, y, marker="o")
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel("MEDV")
```

