EDA & Classification report on Boston Housing Data

As my ID is 20231040, mod(40,2)=0; so Boston Housing Data has been used EDA, interpret and classification.

1. Data Description

All the work has been done using python programming language on Google Colab platform. After downloading the dataset from http://lib.stat.cmu.edu/datasets/boston the dataset has imported by using pandas.

1.1 Summary of the dataset

Python Script:

```
import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/ASDS/boston.csv')
```

Table 1: Variable summary information of Boston housing price dataset

Features	Description	Value Level	Level of measurements	Measures of central tendency
CRIM	per capita crime rate by town		RATIO	Mean
ZN	proportion of residential land zoned for lots over 25,000 square feet.		RATIO	Mean
INDUS	proportion of non-retail business acres per town.		RATIO	Mean
CHAS	Charles River dummy variable	if tract bounds river; 0 otherwise 1	NOMINAL	Mode
NOX	nitric oxides concentration (parts per 10 million)		RATIO	Mean
RM	average number of rooms per dwelling		RATIO	Mean
AGE	proportion of owner-occupied units built prior to 1940		RATIO	Mean
DIS	weighted distances to five Boston employment centers		RATIO	Mean
RAD	index of accessibility to radial highways		ORDINAL	Mean
TAX	full-value property-tax rate per $10,000		RATIO	Mean
PTRATIO	pupil-teacher ratio by town		RATIO	Mean
B	1000(Bk - 0.63) ^2 where Bk is the proportion of blacks by town		RATIO	Mean
LSTAT	% lower status of the population		RATIO	Mean
MEDV	Median value of owner-occupied homes in $1000		RATIO	Mean

The following **5 rows as mentioned** have been appended by using the code provided below.

Python Script:

```
datarowsSeries =
[pd.Series([0.069,10,2.3,0,0.53,6.5,65.2,4.01,1,290,15,395,4.9,24],index=df.columns),
pd.Series([40.69,50,2.7,0,0.9,6.9,65.6,4.5,1,330,55,435,5.3,24.7],index=df.columns),
pd.Series([40.68,51,2.8,0,1,7,65.5,4.4,1,331,53,430,4.6,24.6],index=df.columns),
pd.Series([40.67,52,2.9,0,0.8,6.9,65.7,4.6,1,332,54,432,4.7,24.5],index=df.columns),
pd.Series([40.66,53,2.8,0,1.1,6.9,65.8,4.5,1,333,56,431,4.8,24.6],index=df.columns)]
df_boston=df.append(datarowsSeries,ignore_index=True)
```

```
df_boston.tail()
```

Table 2: Last 5 rows of the dataset after appending.

Row	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
506	0.069	10	2.3	0	0.53	6.5	65.2	4.01	1	290	15	395	4.9	24
507	40.69	50	2.7	0	0.9	6.9	65.6	4.5	1	330	55	435	5.3	24.7
508	40.68	51	2.8	0	1	7	65.5	4.4	1	331	53	430	4.6	24.6
509	40.67	52	2.9	0	0.8	6.9	65.7	4.6	1	332	54	432	4.7	24.5
510	40.66	53	2.8	0	1.1	6.9	65.8	4.5	1	333	56	431	4.8	24.6

So the final shape of the dataset is (511,14) which can be found by **`df_boston.shape`**

1.2 Summary Measures:

The statistical description of the whole dataset (append 5 rows included) is given below.

Python Script: By using the commands, the following table has been made.

```
df_boston.describe()
df_boston.mode()
```

Table 3: Summary measures of different variables of the dataset.

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX
count	506	506	506	506	506	506	506	506	506	506
mean	3.61	11.36	11.14	-	0.55	6.28	68.57	3.80	-	408.24
std	8.60155	23.32	6.86	-	0.12	0.70	28.15	2.11	-	168.54
min	0.00632	0.00	0.46	0.00	0.39	3.56	2.90	1.13	-	187.00
25%	0.08205	0.00	5.19	-	0.45	5.89	45.03	2.10	-	279.00
50%	0.25651	0.00	9.69	-	0.54	6.21	77.50	3.21	-	330.00
75%	3.67708	12.50	18.10	-	0.62	6.62	94.08	5.19	-	666.00
max	88.9762	100.00	27.74	1.00	0.87	8.78	100.00	12.13	24.00	711.00
mode	-	-	-	0.00	-	-	-	-	24.00	-

	B	LSTAT	MEDV
count	506	506	506
mean	356.67	12.65	22.53
std	91.29	7.14	9.20
min	0.32	1.73	5.00
25%	375.38	6.95	17.03
50%	391.44	11.36	21.20
75%	396.23	16.96	25.00
max	396.90	37.97	50.00
mode	-	-	-

As the column CHAS and RAD is not interval or ratio, the measurement of central tendency of from the Table 3 is not applicable.

2. Univariate Analysis

For ratio or interval variables, boxplot or histogram and for ordinal or nominal variables pie chart or frequency distributed bar chart have been visualized here.

2.1 Graphical Representation

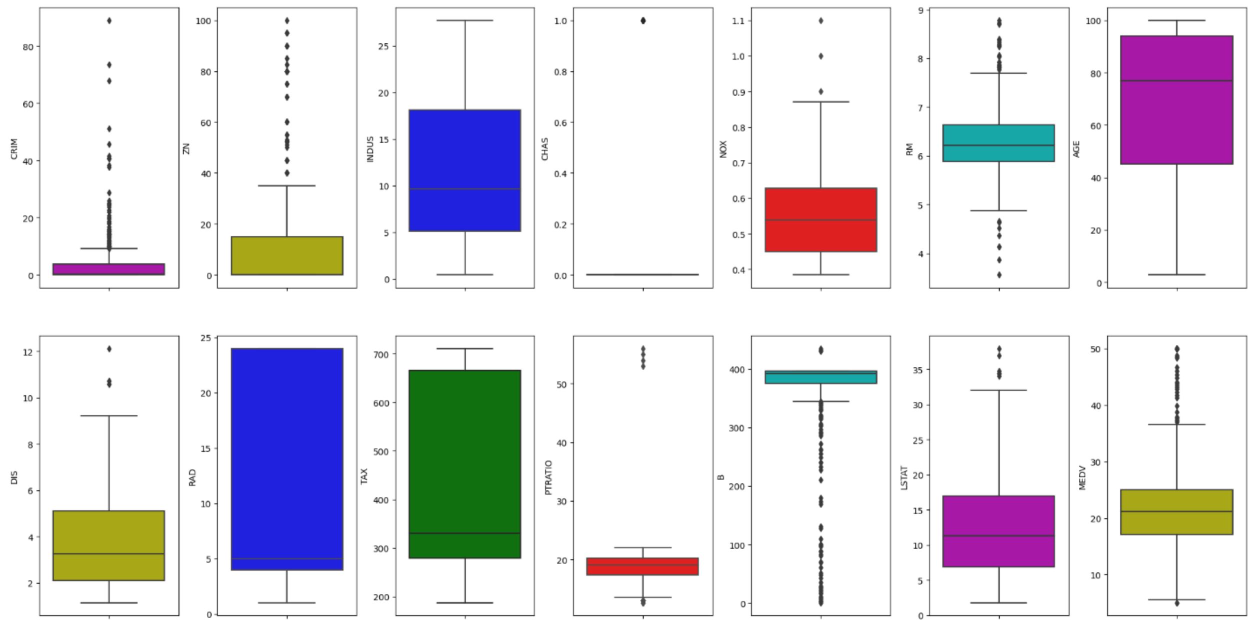


Fig 1: Boxplot of all the features of the Boston housing price dataset

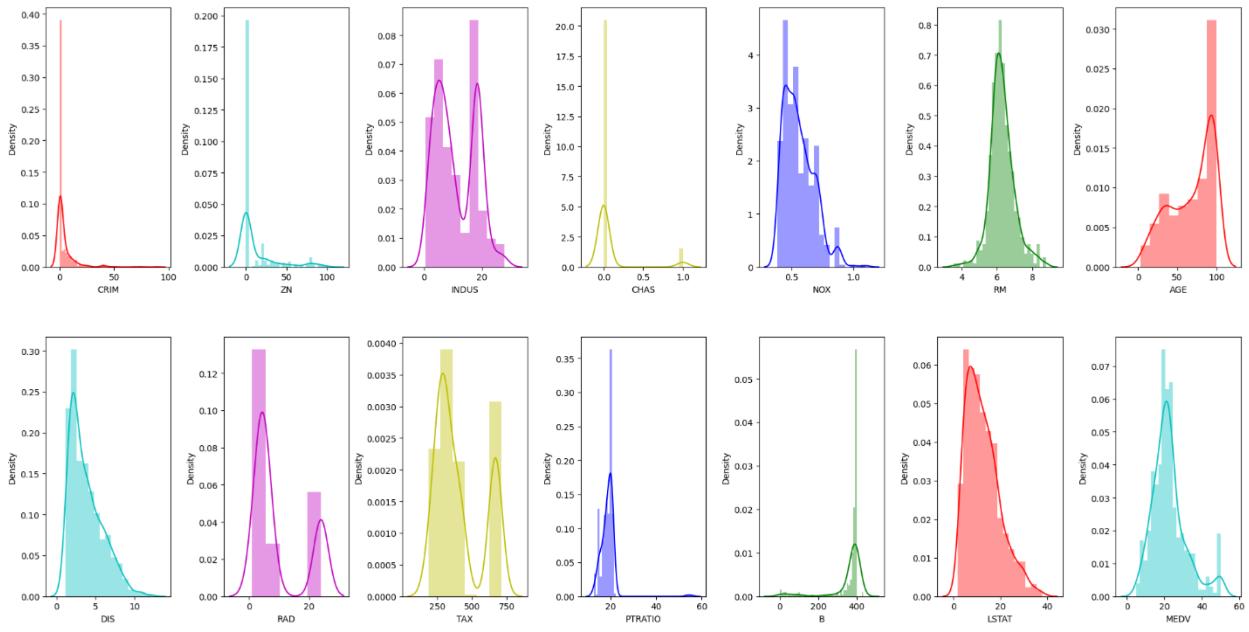


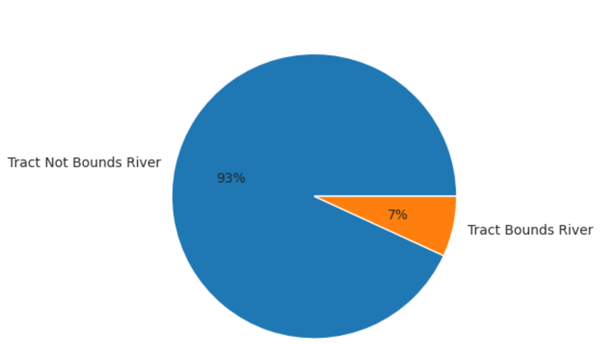
Fig 2: Histogram of all the features of the Boston housing price dataset

Observations from Fig. 1 & 2:

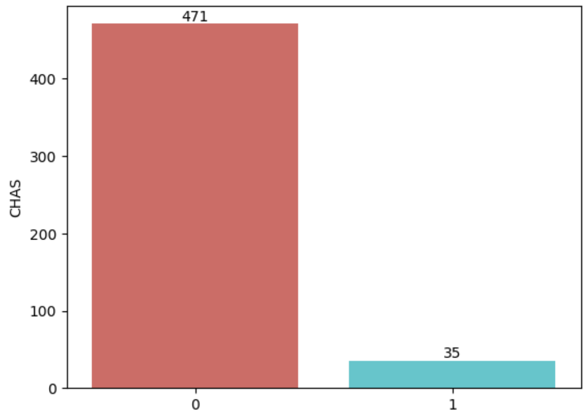
- The columns CRIM, ZN are positively as well as B are negatively heavily skewed, and TAX is moderately skewed. This is due to the presence of the Outliers present in our dataset.
- We can see that the values in the column CHAS are almost 0. Also, RAD is highly positively skewed but the level of measurements CHAS and RAD is nominal and ordinal respectively.
- The column RM and MEDV are symmetrically distributed with few outliers.

Python Script for Fig 1 & 2:

```
from scipy import stats
from itertools import cycle
cycol = cycle('bgrcmy')
fig, axs = plt.subplots(ncols=7, nrows=2, figsize=(20, 10))
index = 0
axs = axs.flatten()
for k,v in df_boston.items():
    sns.boxplot(y=k, data=df_boston, ax=axs[index], color=next(cycol))
    #sns.distplot(v, ax=axs[index], color=next(cycol))
    index += 1
plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=5.0)
```



(a)



(b)

Fig 3: (a) Pie chart & (b) Bar diagram of column CHAS (Charles River dummy variable)

Observations from Fig 3:

- The column CHAS has two categories where count of tract does not bound river (encoded as 0) is 471 which is 93 % of total count. Tract bounds river (encoded as 1) is 7% of the total count.

Python Script for Fig 3:

```
#pie-chart
keys=['Tract Not Bounds River', 'Tract Bounds River']
plt.pie(df_boston.CHAS.value_counts(), labels=keys,autopct='%.0f%%')
plt.show()
#bar-chart
ax=sns.barplot( x=[0,1],y=df["CHAS"].value_counts(),palette = 'hls')
ax.bar_label(ax.containers[0])
plt.show()
```

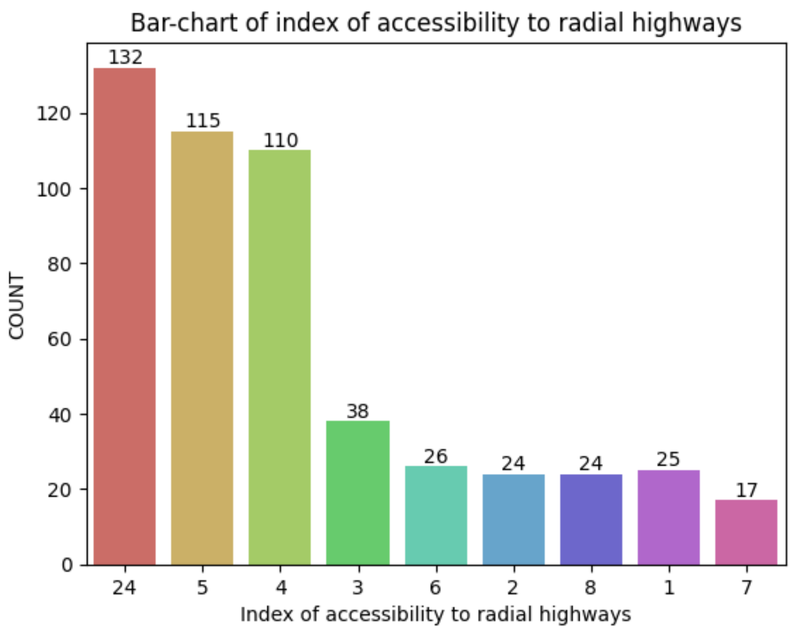


Fig 4: Bar diagram of accessibility of radial highways

Observations from Fig. 4:

- In column RAD, Index 24 has the highest count which is 132 where index 7 has lowest with 17 counts.

Python Script for Fig 4:

```
ax=sns.countplot(data=df_boston, x='RAD', order=df.RAD.value_counts().index, palette = 'hls')
ax.bar_label(ax.containers[0])
plt.xlabel('Index of accessibility to radial highways')
plt.ylabel('COUNT')
plt.title('Bar-chart of index of accessibility to radial highways')
plt.show()
```

2.2 Outlier detection

The percentage of outliers of each variable is calculated here. The data points which fall below **Q1 – 1.5 IQR or above Q3 + 1.5 IQR** are outliers.

Python Script:

```
for k, v in df_boston.items():
    q1 = v.quantile(0.25)
    q3 = v.quantile(0.75)
    irq = q3 - q1
    v_col = v[(v <= q1 - 1.5 * irq) | (v >= q3 + 1.5 * irq)]
    perc = np.shape(v_col)[0] * 100.0 / np.shape(df_boston)[0]
    print("Column %s outliers = %.2f%%" % (k, perc))
```

```
Output:
Column CRIM outliers = 13.50%
Column ZN outliers = 12.13%
Column INDUS outliers = 0.00%
Column CHAS outliers = 100.00%
Column NOX outliers = 0.59%
Column RM outliers = 5.87%
Column AGE outliers = 0.00%
Column DIS outliers = 0.98%
Column RAD outliers = 0.00%
Column TAX outliers = 0.00%
Column PTRATIO outliers = 3.72%
Column B outliers = 15.85%
Column LSTAT outliers = 1.17%
Column MEDV outliers = 7.83%
```

3. Bivariate Analysis

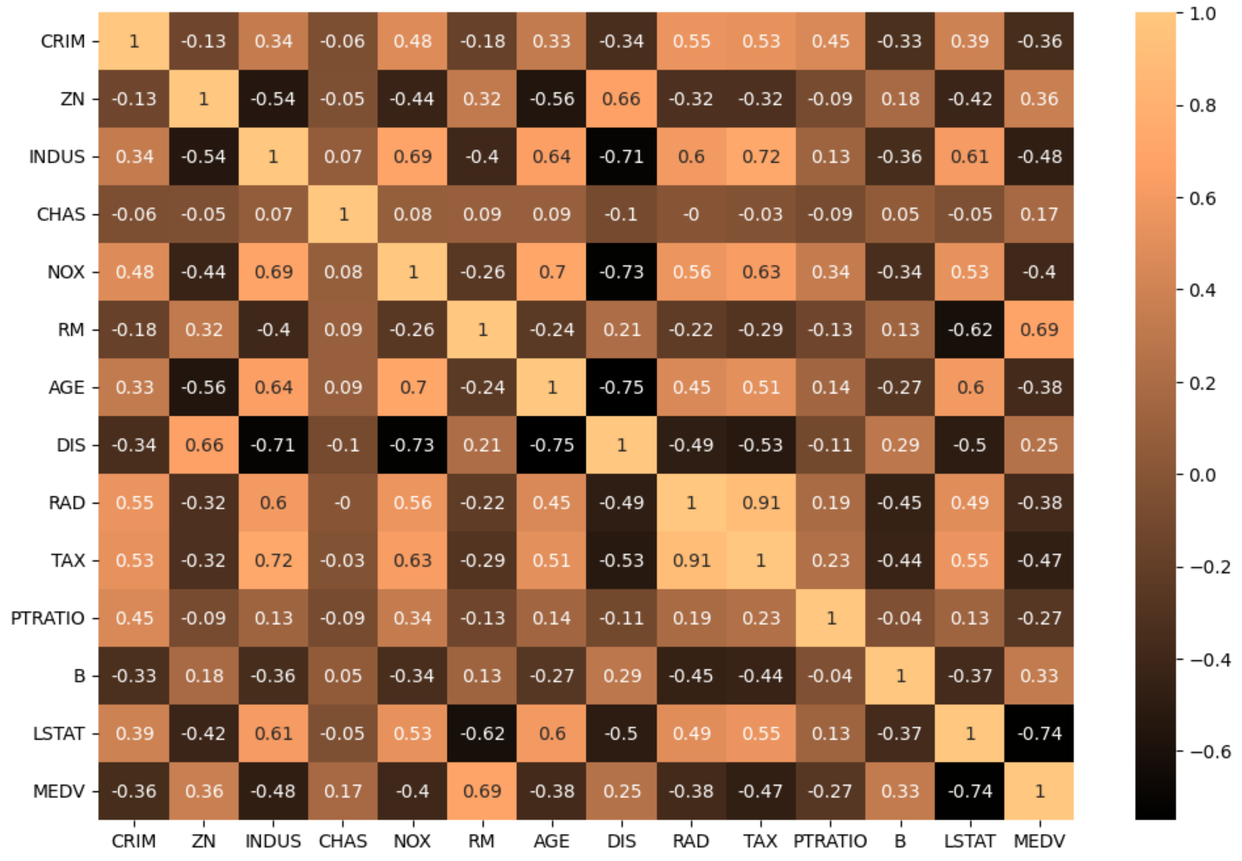
Based on correlation matrix, some variables which are co-related to each other moderately or strongly have been visualized by scatter plot.

3.1 Correlation Matrix:

A correlation matrix is just a table with the correlation coefficients for different variables. The matrix shows how all the possible pairs of values in a table are related to each other. It is a powerful tool for summarizing a large data set and finding and showing patterns in the data. The Pearson correlation method is usually used as a primary check for the relationship between two variables.

Python Script for Fig 5:

```
plt.figure(figsize=(12,8))
correlation_matrix = df_boston.corr().round(2)
# annot = True to print the values inside the square
sns.heatmap(data=correlation_matrix, annot=True, cmap='copper')
plt.yticks(rotation=0)
plt.show()
```



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	1	-0.13	0.34	-0.06	0.48	-0.18	0.33	-0.34	0.55	0.53	0.45	-0.33	0.39	-0.36
ZN	-0.13	1	-0.54	-0.05	-0.44	0.32	-0.56	0.66	-0.32	-0.32	-0.09	0.18	-0.42	0.36
INDUS	0.34	-0.54	1	0.07	0.69	-0.4	0.64	-0.71	0.6	0.72	0.13	-0.36	0.61	-0.48
CHAS	-0.06	-0.05	0.07	1	0.08	0.09	0.09	-0.1	-0	-0.03	-0.09	0.05	-0.05	0.17
NOX	0.48	-0.44	0.69	0.08	1	-0.26	0.7	-0.73	0.56	0.63	0.34	-0.34	0.53	-0.4
RM	-0.18	0.32	-0.4	0.09	-0.26	1	-0.24	0.21	-0.22	-0.29	-0.13	0.13	-0.62	0.69
AGE	0.33	-0.56	0.64	0.09	0.7	-0.24	1	-0.75	0.45	0.51	0.14	-0.27	0.6	-0.38
DIS	-0.34	0.66	-0.71	-0.1	-0.73	0.21	-0.75	1	-0.49	-0.53	-0.11	0.29	-0.5	0.25
RAD	0.55	-0.32	0.6	-0	0.56	-0.22	0.45	-0.49	1	0.91	0.19	-0.45	0.49	-0.38
TAX	0.53	-0.32	0.72	-0.03	0.63	-0.29	0.51	-0.53	0.91	1	0.23	-0.44	0.55	-0.47
PTRATIO	0.45	-0.09	0.13	-0.09	0.34	-0.13	0.14	-0.11	0.19	0.23	1	-0.04	0.13	-0.27
B	-0.33	0.18	-0.36	0.05	-0.34	0.13	-0.27	0.29	-0.45	-0.44	-0.04	1	-0.37	0.33
LSTAT	0.39	-0.42	0.61	-0.05	0.53	-0.62	0.6	-0.5	0.49	0.55	0.13	-0.37	1	-0.74
MEDV	-0.36	0.36	-0.48	0.17	-0.4	0.69	-0.38	0.25	-0.38	-0.47	-0.27	0.33	-0.74	1

Fig 5: Correlation matrix of different variables of Boston housing price dataset

The correlation coefficient ranges from -1 to 1. If the value is close to 1, it means that there is a strong positive correlation between the two variables. When it is close to -1, the variables have a strong negative correlation.

Observation from Fig 5:

- The proportion of non-retail business acres (INDUS) is positively correlated with oxide concentration (NOX). This implies towns with high non-retail business areas have higher acids emissions. Could be a causation as well.
- Also, the proportion of non-retail business acres (INDUS) is positively correlated with taxes (TAX). This implies that higher taxes are imposed on houses in town with high non-retail business areas.
- The number of rooms (RM) is positively correlated with house value (MEDV), which makes sense.
- Also, the number of rooms (RM) is negatively correlated with the low status of population (LSTAT) which makes sense. That means if the number of rooms increases, the percentage of low status of population decreases.

- Oxide concentration (NOX) is positively correlated with old owner-occupied buildings (AGE). This could transitively imply that old houses are more centered around non-retail business areas (which is positively correlated with oxide emissions).
- Distance from employment centers (DIS) is negatively correlated with proportion of non-retail business acres (INDUS), oxide concentration (NOX), and proportion of owner-occupied buildings (AGE). This says towns away from employment centers have more recent houses and less oxide concentration.
- The prices of houses (MEDV) are negatively correlated with the low status of population (LSTAT) which makes much sense.
- The index of accessibility to radial highways (RAD) is strongly positively correlated with taxes (TAX). An important point in selecting features for a model is to check for multi-co-linearity.

3.2 Graphical Representation

Now, the relationship between the pairs of features having significant correlations has been visualized here.

<u>Python Scirpt:</u>

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='NOX', y='INDUS', data=df_boston)
plt.show()
```

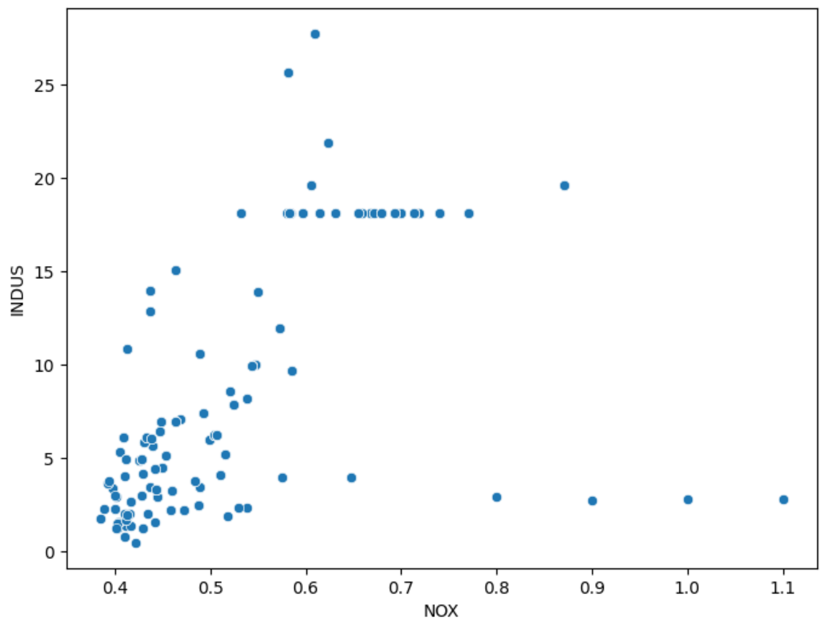


Fig 6: Scatter diagram of nitric oxides concentration and non-retail business acres per town

Observations for Fig 6:

➢ It appears that there is no pattern in this correlation between the two features.

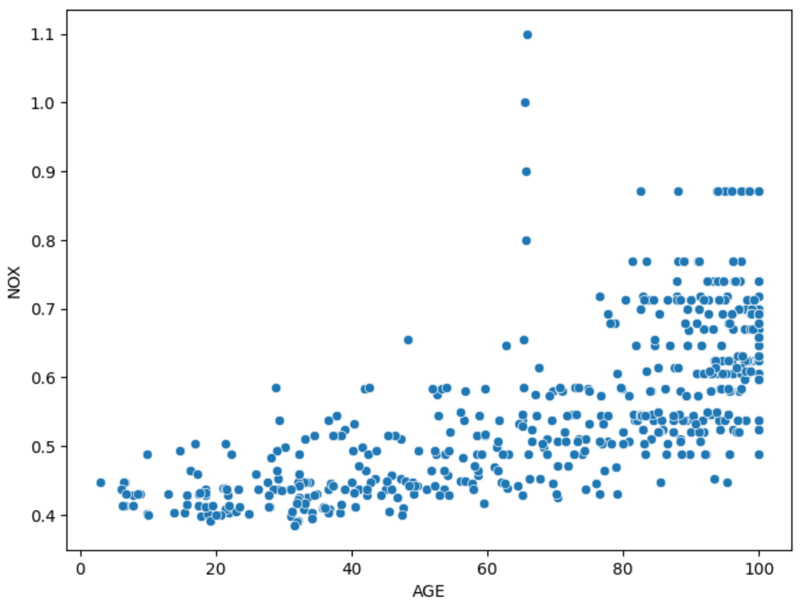


Fig 7: Scatter diagram of nitric oxides concentration and owner-occupied units built prior to 1940.

Observations for Fig 7:

- The more proportion of owner-occupied units built prior to 1940 exist, the more oxides concentration. This implies that old owner-occupied houses are in a geographical location closer to the oxide source than more recently built houses.

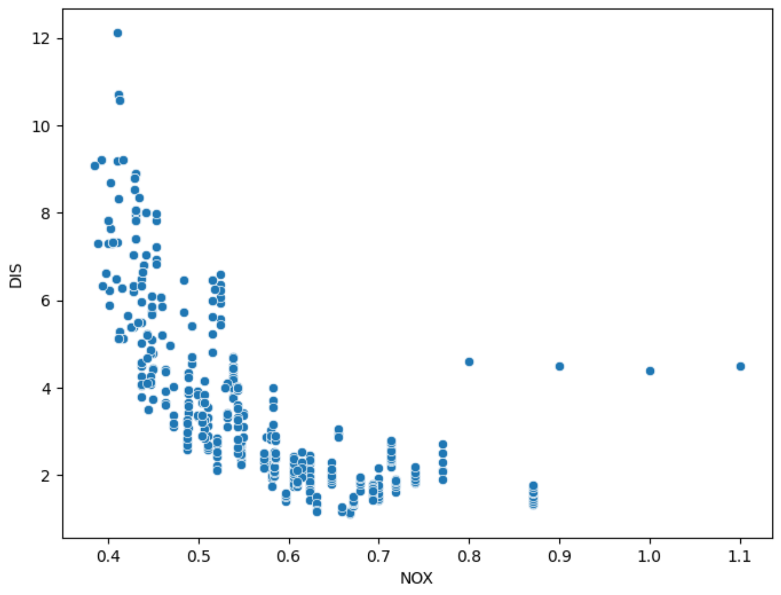


Fig 8: Scatter diagram of distances to Boston employment centers and nitric oxides concentration

Observations for Fig 8:

- ➢ The distance the house is from employment centers; the less oxide concentration is there. This implies that the employment center's location is where the oxide source lies.

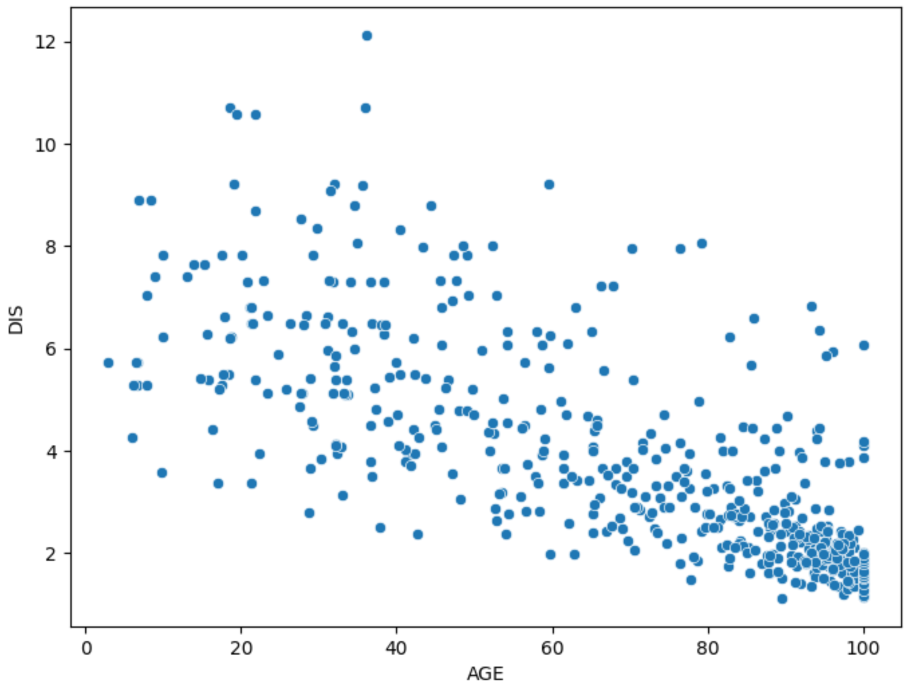


Fig 9: Scatter diagram of distances to Boston employment centers and owner-occupied units built prior to 1940.

Observations for Fig 9:

- The distance of the houses to the Boston employment centers appears to decrease moderately as the proportion of the old houses increase in the town. It is possible that the Boston employment centers are in the established towns where proportion of owner-occupied units built prior to 1940 is comparatively high.
- This, along with the previous observations, supports the assumption that old owner-occupied houses are closer to employment centers and employment centers are emitting oxides.

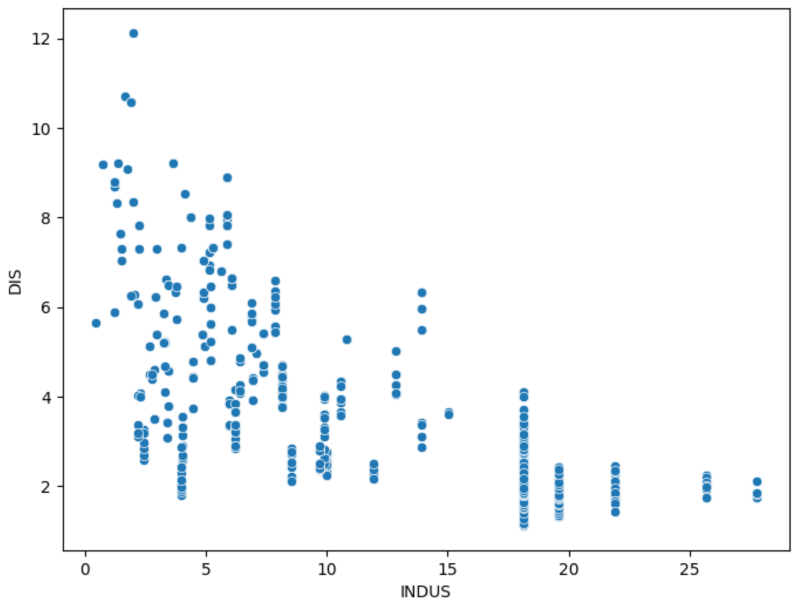


Fig 10: Scatter diagram of distances to Boston employment centers and proportion of non-retail business acres per town.

Observations for Fig 10:

- The distance of the houses to the Boston employment centers appears to decrease moderately as the proportion of the old houses increase in the town.

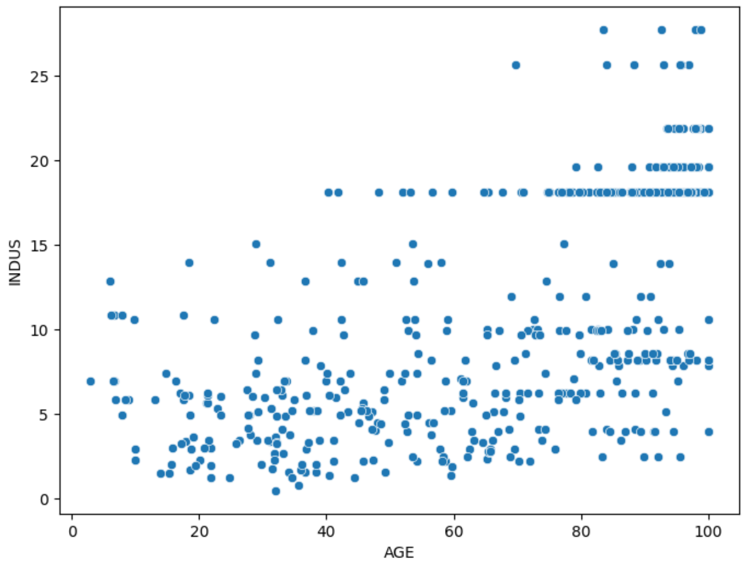


Fig 11: Scatter diagram of non-retail business acres per town and owner-occupied units built prior to 1940.

Observations for Fig 11:

- No trend between the two variables is visible in the above plot.

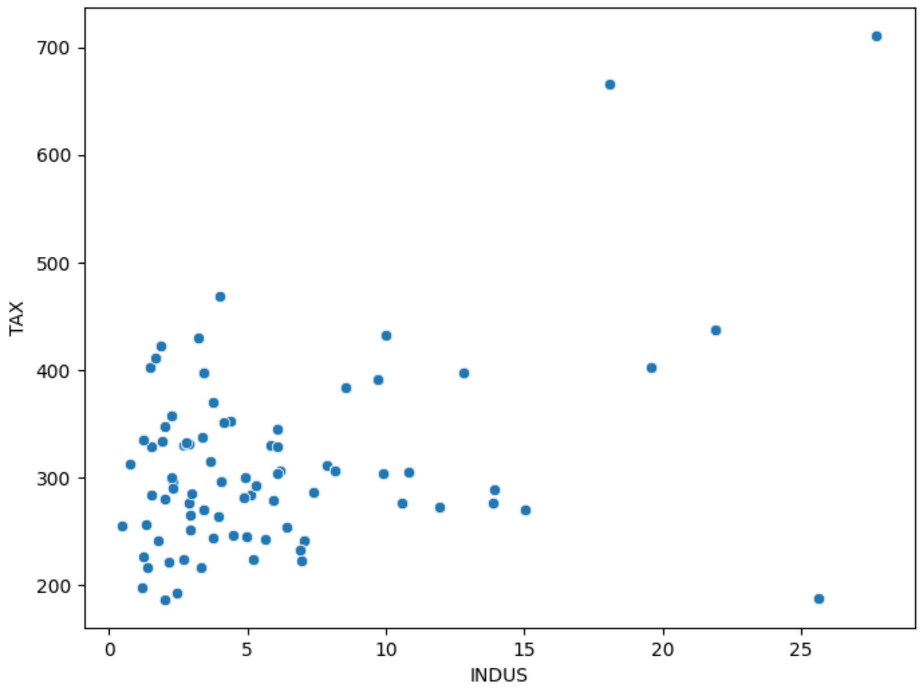


Fig 12: Scatter diagram of property-tax rate per $10,000 and non-retail business acres per town.

Observations for Fig 12:

- The tax rate appears to increase with an increase in the proportion of non-retail business acres per town. This might be due to the reason that the variables TAX and INDUS are related to a third variable.

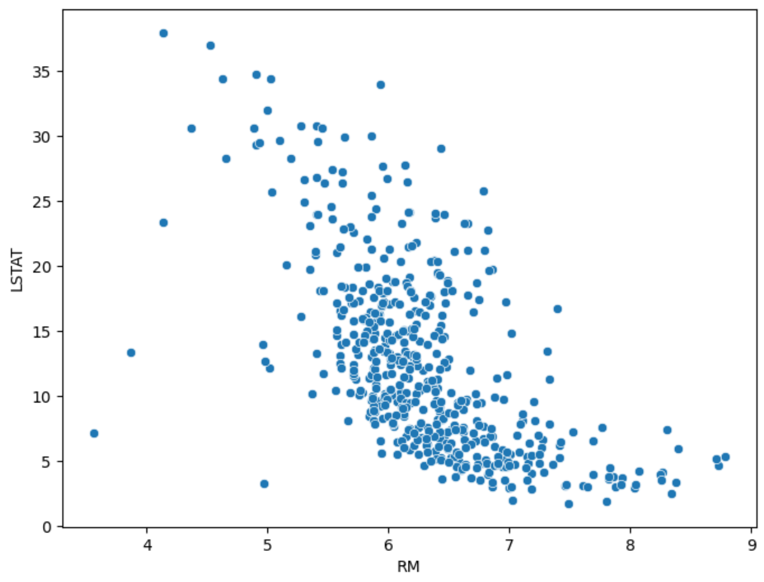


Fig 13: Scatter diagram of lower status of the population and average number of rooms per dwelling

Observation for Fig 13:

- The percentage of lower status of the population appears to decrease moderately as the average number of rooms per dwelling.

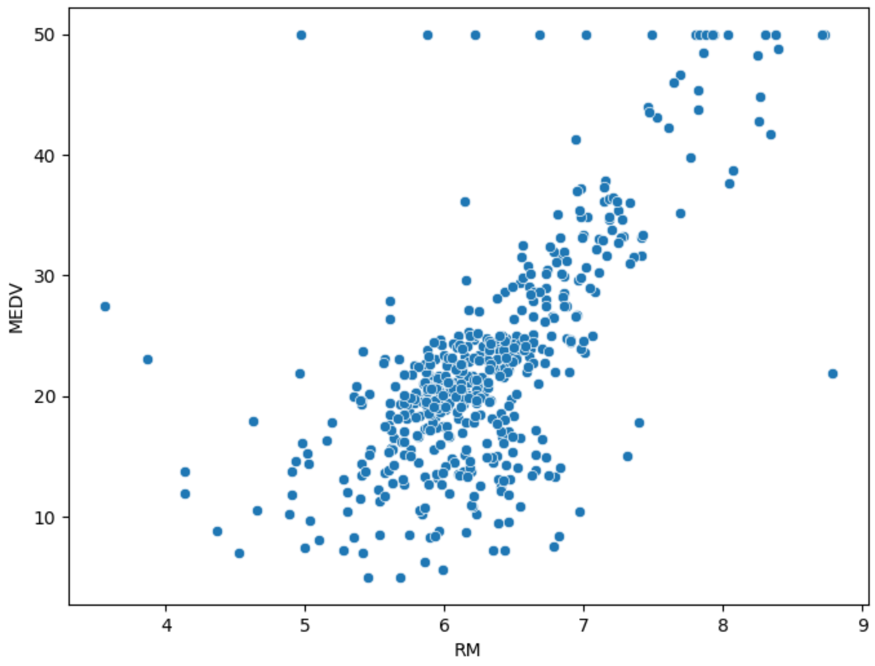


Fig 14: Scatter diagram of owner-occupied homes and average number of rooms per dwelling

Observations for Fig 14:

- The price of the house seems to increase as the value of RM increases. This is expected as the price is generally higher for more rooms.
- There are a few outliers in a horizontal line as the MEDV value seems to be capped at 50.

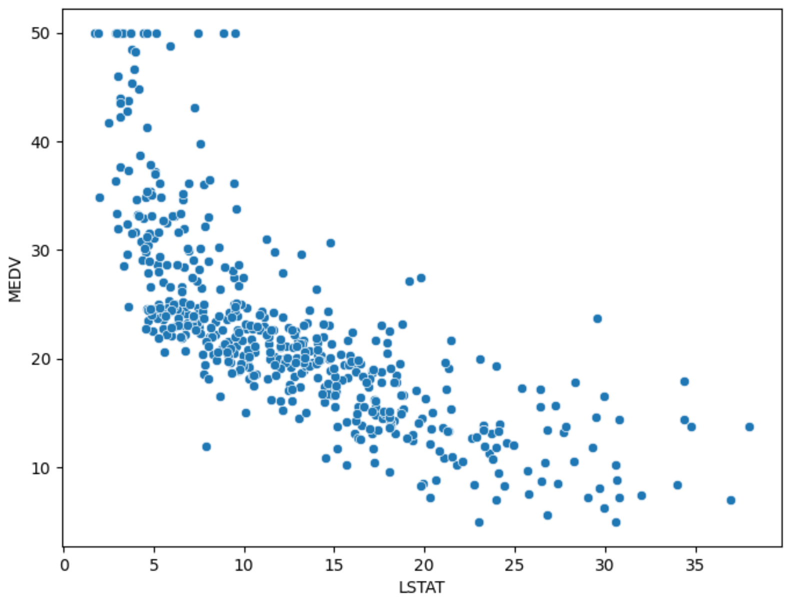


Fig 15: Scatter diagram of owner-occupied homes and lower status of the population

Observations for Fig 15:

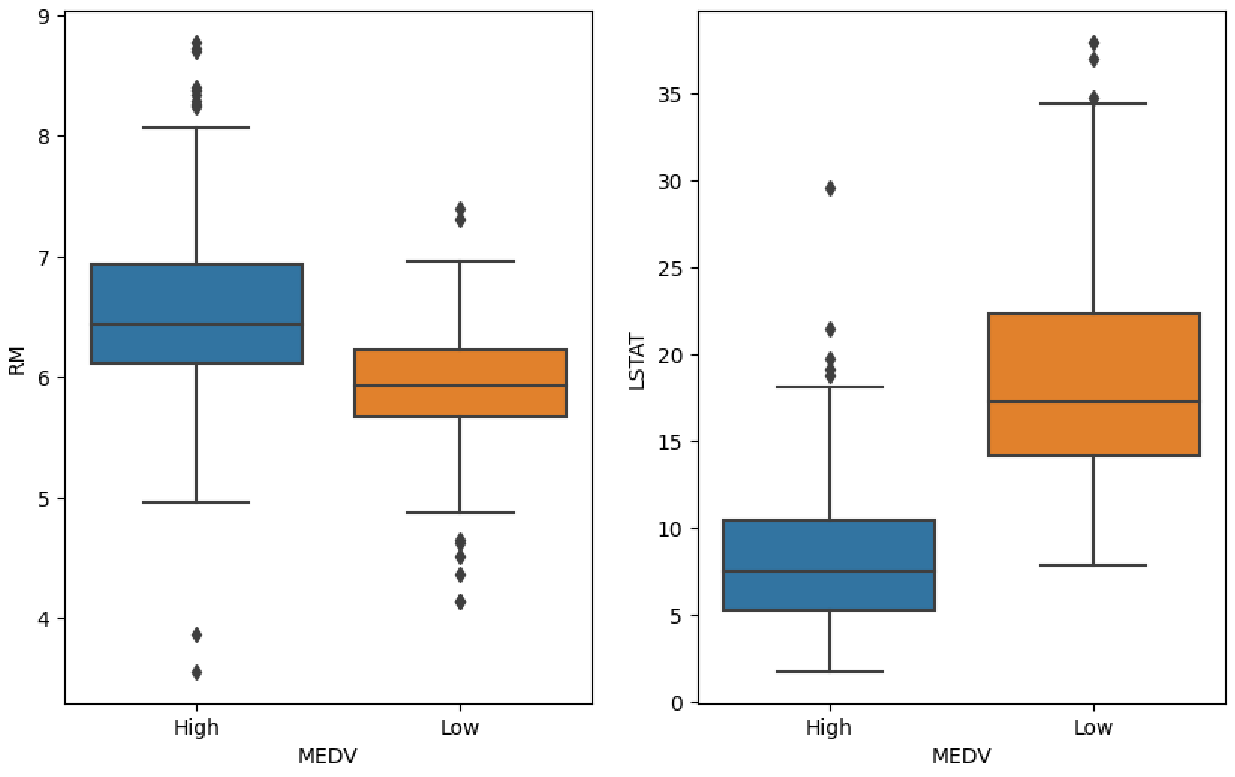
- The prices tend to decrease with an increase in LSTAT. This is also possible as the house price is lower in areas where lower status people live.

- There are few outliers, and the data seems to be capped at 50.

We have seen that the variables LSTAT and RM have a linear relationship with the dependent variable MEDV. Also, there are significant relationships among a few independent variables.

Python Script for Fig 16:

```
fig, axs = plt.subplots(ncols=2,nrows=1, figsize=(10, 6))
sns.boxplot(x='MEDV', y='RM', data=df_boston, ax=axs[0])
sns.boxplot(x='MEDV', y='LSTAT', data=df_boston, ax=axs[1])
```



(a) (b)

Fig 16: Boxplot of (a) average number of rooms per dwelling (b) lower status of the population at price status of owner-occupied homes

Observation for Fig 16:

- Boxplot of average number of rooms per dwelling at price status of owner-occupied homes represent that MEDV is slightly positive skewed for high pricing of homes and symmetric for low price. A few outliers contain for both price status.
- Boxplot of lower status of the population at price status of owner-occupied homes depicts that MEDV is symmetric for high pricing of homes and positive skewed for low pricing of homes.

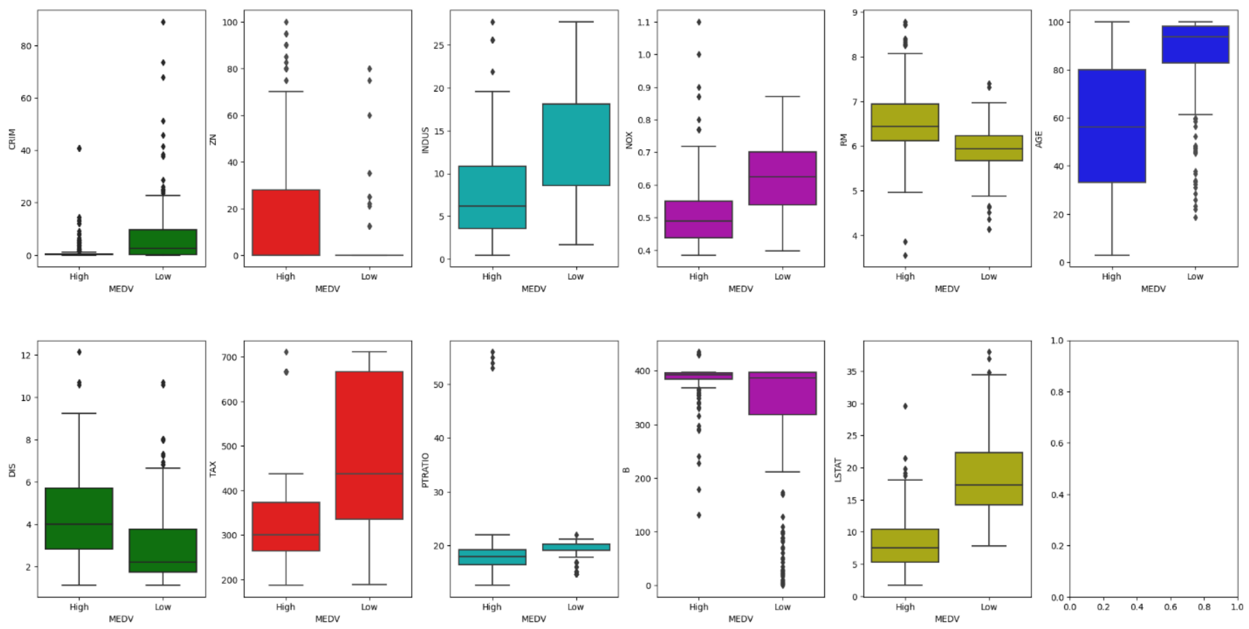


Fig 17: Boxplot of all ratio/interval variables at price status of owner-occupied homes.

3.3 Covariance Matrix

Covariance is a measure of how changes in one variable are associated with changes in a second variable. Specifically, it's a measure of the degree to which two variables are linearly associated. A covariance matrix is a square matrix that shows the covariance between different variables in a dataset. Table 4 is drawn by using `df_boston.cov().round(2)`

Table 4: Covariance matrix of different features of Boston housing price dataset

index	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD
CRIM	83.98	-28.23	21.42	-0.14	0.53	-1.13	83.75	-6.61	43.99
ZN	-28.23	551.16	-87.15	-0.27	-1.26	5.26	-371.14	32.53	-65.37
INDUS	21.42	-87.15	47.29	0.11	0.58	-1.91	123.54	-10.18	35.9
CHAS	-0.14	-0.27	0.11	0.06	0	0.02	0.61	-0.05	-0.01
NOX	0.53	-1.26	0.58	0	0.01	-0.02	2.35	-0.18	0.59
RM	-1.13	5.26	-1.91	0.02	-0.02	0.49	-4.72	0.3	-1.32
AGE	83.75	-371.14	123.54	0.61	2.35	-4.72	784.68	-43.91	110.93
DIS	-6.61	32.53	-10.18	-0.05	-0.18	0.3	-43.91	4.39	-9.03
RAD	43.99	-65.37	35.9	-0.01	0.59	-1.32	110.93	-9.03	75.78
TAX	815.29	-1247.9	832.19	-1.45	12.69	-34.68	2381.65	-188.27	1329.72
PTRATIO	15.77	-8.31	3.36	-0.09	0.16	-0.36	14.99	-0.85	6.34
B	-277.98	393.45	-226.92	1.07	-3.75	8.52	-698.01	55.92	-355.45
LSTAT	25.52	-70.52	29.93	-0.09	0.46	-3.09	120.12	-7.45	30.73
MEDV	-29.83	77.2	-30.38	0.4	-0.44	4.46	-96.69	4.8	-30.42

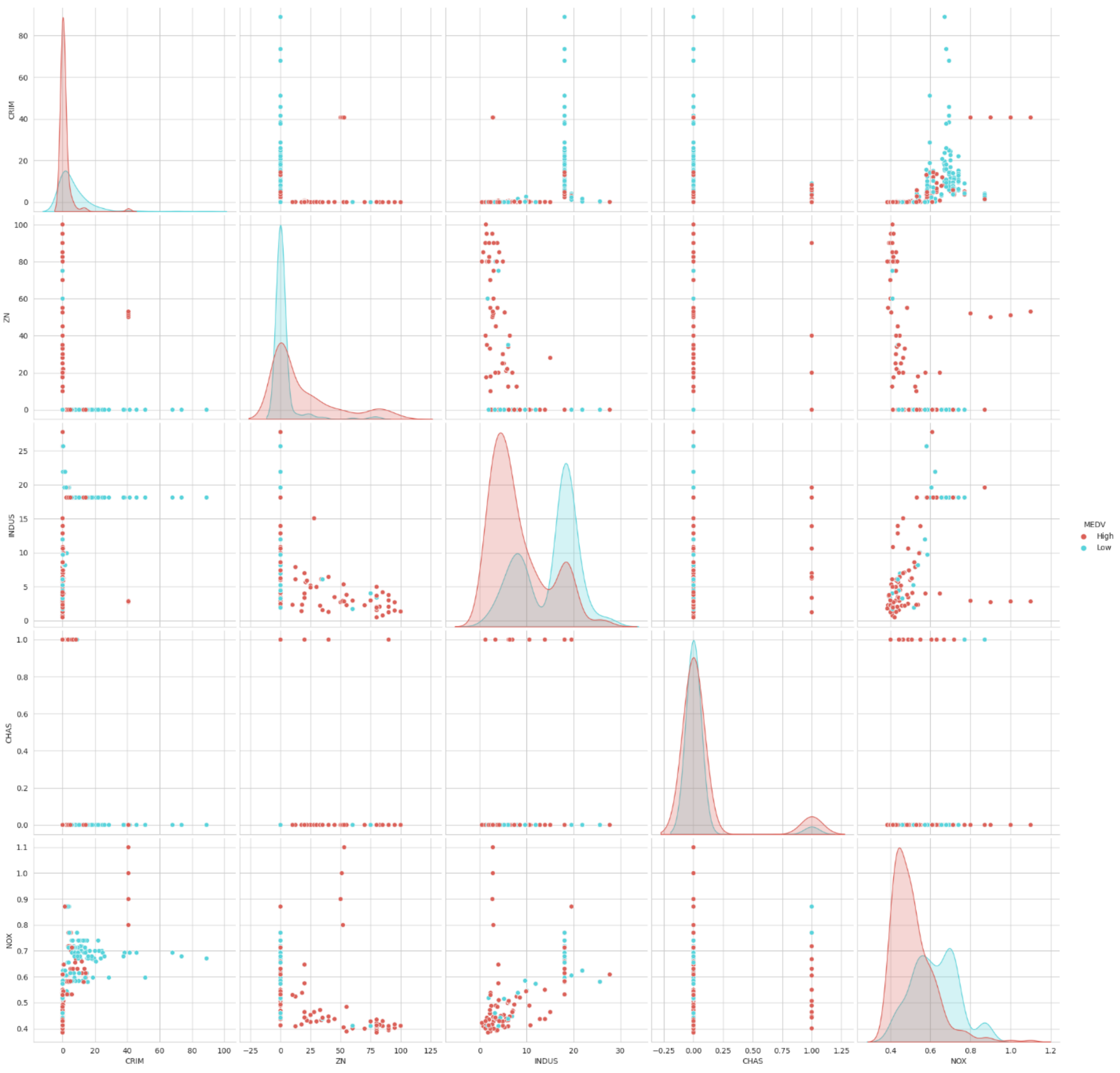
index	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	815.29	15.77	-277.98	25.52	-29.83
ZN	-1247.9	-8.31	393.45	-70.52	77.2
INDUS	832.19	3.36	-226.92	29.93	-30.38
CHAS	-1.45	-0.09	1.07	-0.09	0.4
NOX	12.69	0.16	-3.75	0.46	-0.44
RM	-34.68	-0.36	8.52	-3.09	4.46
AGE	2381.65	14.99	-698.01	120.12	-96.69
DIS	-188.27	-0.85	55.92	-7.45	4.8
RAD	1329.72	6.34	-355.45	30.73	-30.42
TAX	28199.19	145.85	-6784.94	654.72	-720.7
PTRATIO	145.85	14.79	-13.86	3.59	-9.44
B	-6784.94	-13.86	8300.01	-241.47	278.56
LSTAT	654.72	3.59	-241.47	51.08	-48.12
MEDV	-720.7	-9.44	278.56	-48.12	83.79

3.4 Pair plots

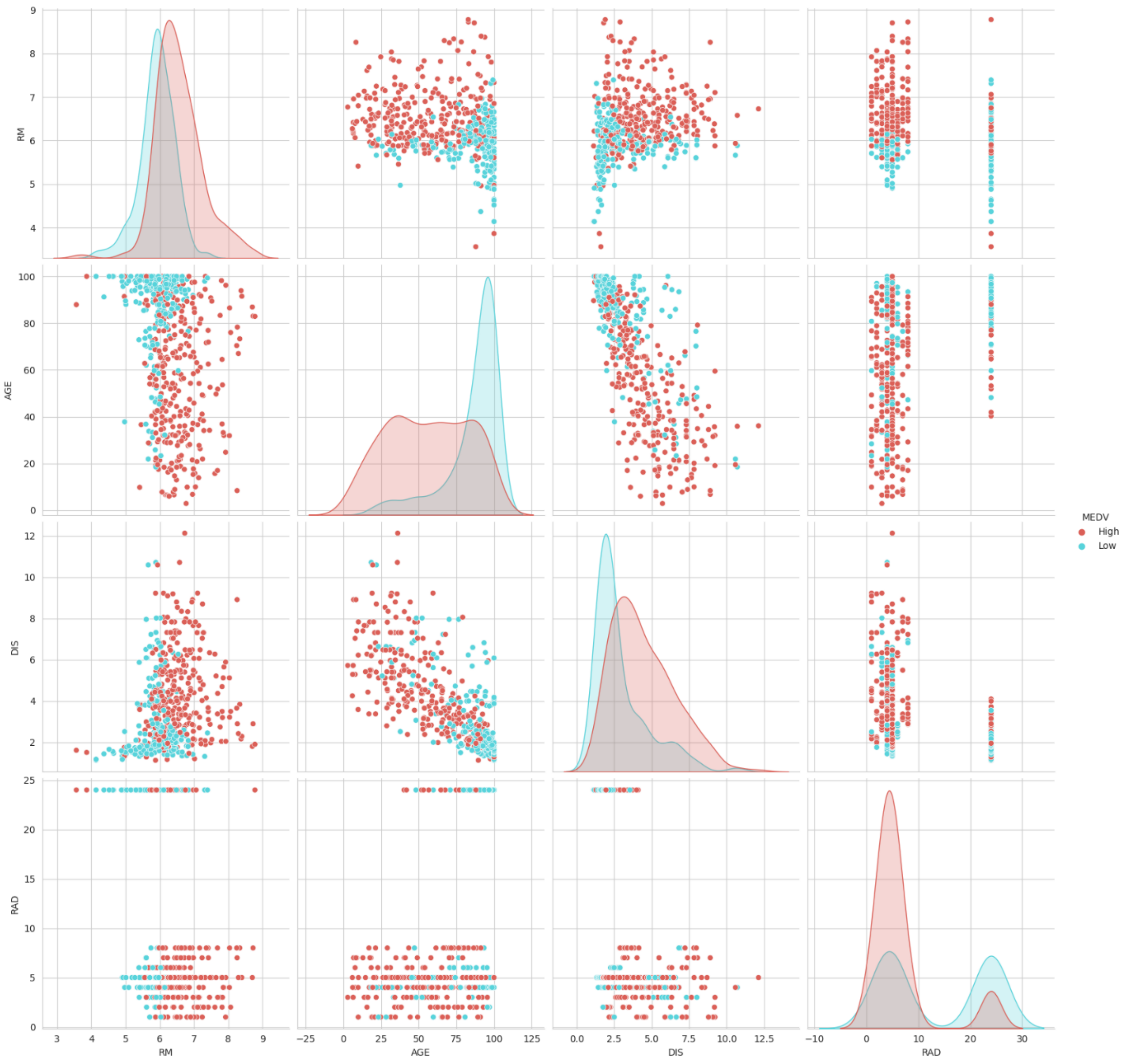
Pair plots are used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or making linear separation in our dataset.

Python Script:

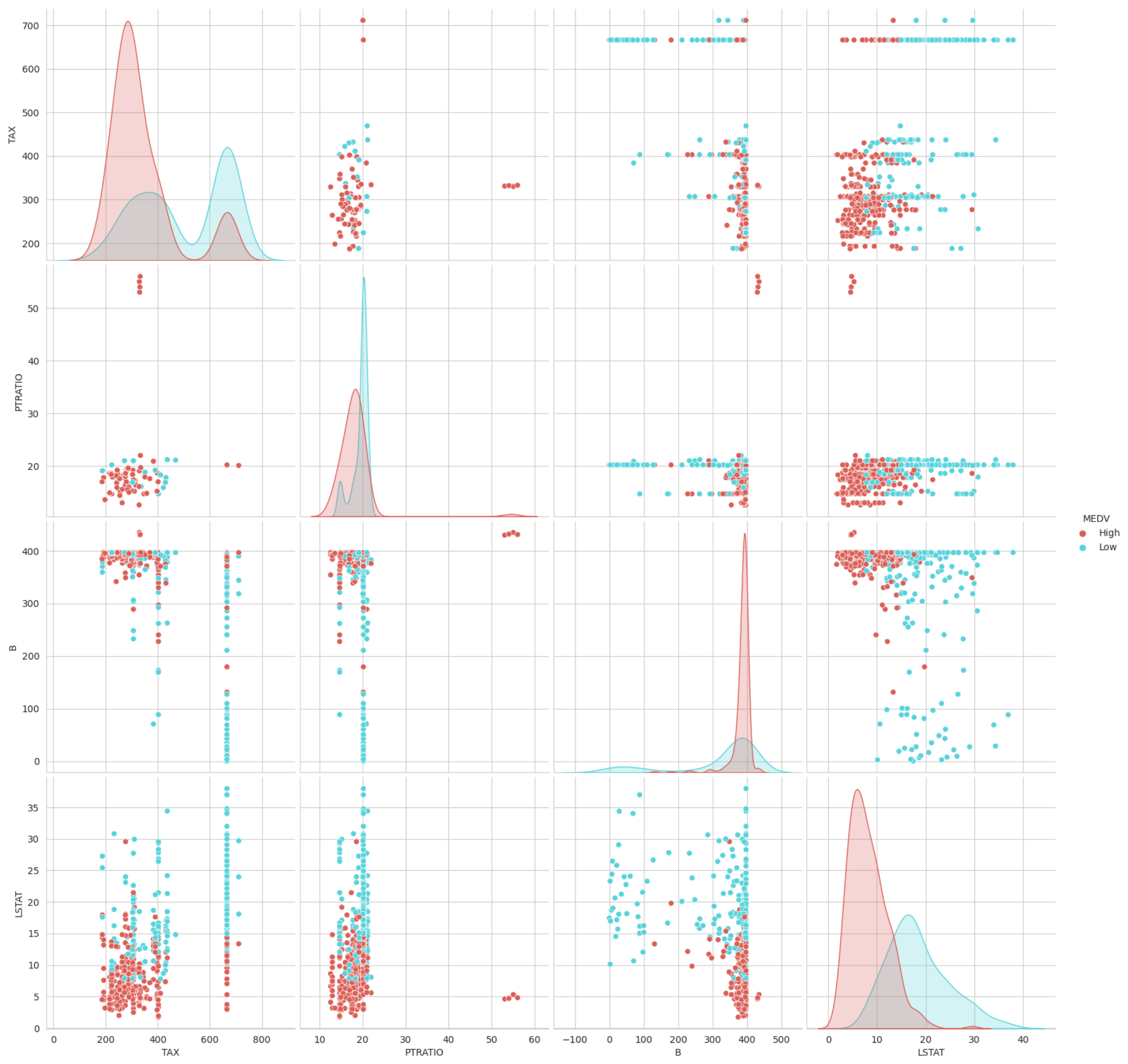
```
sns.set_style('whitegrid');
sns.pairplot(df_boston, vars=[ 'CRIM', 'ZN','INDUS','CHAS','NOX'],hue = 'MEDV', height
= 4, palette = 'hls')
sns.pairplot(df_boston,  vars=['RM','AGE','DIS','RAD'],hue  =  'MEDV',  height  =  4,
palette = 'hls')
sns.pairplot(df_boston, vars=['TAX','PTRATIO' ,'B'   ,'LSTAT'],hue = 'MEDV', height =
4, palette = 'hls')
plt.show()
```



(a)



(b)



(c)

Fig 18: Pair plots of all variables at price status of Boston housing price dataset

Observations for Fig 18:

- Fig explores histograms for each variable and scatter diagrams between each pair of variables at price status 'high' with red dots and at price status 'low' with blue dots.
- Histogram of each variable for price status (high or low) overlapping each other.
- Price is high where crime rate is in range 20 to 80 or more and low where per capita crime rate is less than 20.
- Housing prices are high where the proportion of residential land zoned is less than or equal to 35 and low price for greater than 35.

- Proportion of non-retail business acres per town higher than 22 is conducted higher housing price. (Bi-modal shape).
- Where nitric oxides concentration is greater than 0.7, the price of houses is higher here.
- Average number of rooms per dwelling less than 4 are more likely to high price.
- The proportion of owner-occupied units built prior to 1940 less than 40 are more likely to high price as well as more than 40 conducted lower housing price.
- Housing prices are more likely to high where distances to Boston employment centers less than 4.
- Tax rate greater than 400 is driven to high prices (Bi-modal shape).
- Housing prices are more likely to be low where pupil-teacher ratio by town is greater than 35.
- The proportion of blacks by town is conducted high price where proportion value greater than 100.
- Housing prices are high where the percentage of lower status of the population is greater than 15.

4. Classification of housing prices

In order to classify the different house prices (high > 20, low <= 20) in Boston Housing data, four different classifier have been used and analyzed which model is best.

Before passing the input vector through the model, the dataset has been split into train and test data by 70%-30% ratio with random state 42. [*In Douglas Adams's popular 1979 science-fiction novel The Hitchhiker's Guide to the Galaxy, towards the end of the book, the supercomputer Deep Thought reveals that the answer to the great question of "life, the universe and everything" is 42.*]

<u>Python Script:</u>

```
X=df_boston.iloc[:,0:13]
y=df_boston.MEDV
x_train, x_test, y_train, y_test=train_test_split(X,y,test_size=0.3,random_state=42)
```

Breakdown of the target variable 'MEDV' before and after splitting is given below:

<u>Python Script:</u>

```
df_boston.MEDV.value_counts()
unique, counts = np.unique(y_test, return_counts=True)
print(np.asarray((unique, counts)).T)
```

Table 4: Breakdown of price data before and after splitting

MEDV	Total	Low (1)	High (2)
Total	511	215	296
Train Data	357	142	215
Test Data	154	73	81

4.1 Confusion Matrix

A confusion matrix is a summary of prediction results on a classification problem. The confusion matrix shows the ways in which your classification model is confused when it makes predictions.

Table 5: Confusion matrix for classification of housing price

Predicted \ Actual	Low	High
Low	True Positive (TP)	False Negative (FN)
High	False Positive (FP)	True Negative (TN)

From the confusion matrix, different types of metrics can be calculated. Here, four different metrics are described here.

Accuracy: Accuracy determines the fraction of correctly classified instances to all the instances in the test set.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

Precision: Precision refers to the percentage of low-price predictions that are actually low price.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall: Defines also as Sensitivity, True Positive Rate (TPR), or Detection Rate (DR), it represents the fraction of correctly identified high prices samples over the total number of low price of houses present in the test set.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Specificity: Specificity also known as the True Negative Rate (TNR) represents the portion of high price samples that is correctly labeled by the model.

$$Specificity = \frac{TN}{TN + FP} \tag{4}$$

Negative Precision: Negative Precision represents a percentage of high-price predictions that are actually high price.

$$Negative\ Precision = \frac{TN}{TN + FN} \tag{5}$$

Python Script:

```
rf=RandomForestClassifier(n_estimators=42)
dt=DecisionTreeClassifier(random_state=42)
lr=LogisticRegression(random_state=42)
svm=SVC(kernel='rbf',probability=True)

#define a function
def classifier(clf):
  model=clf
  model.fit(x_train,y_train)
  y_pred=model.predict(x_test)
  y_pred_prob=model.predict_proba(x_test)
  cm=confusion_matrix(y_test,y_pred)

  accuracy=accuracy_score(y_test,y_pred)
  print('Accuracy : ', accuracy )
  recall = recall_score(y_test,y_pred)
  print('Recall : ', recall )
  precision = precision_score(y_test, y_pred)
  print('Precision : ', precision )
  f1 = f1_score(y_test, y_pred)
  print('F1_Score : ', f1 )

  sensitivity = cm[0,0]/(cm[0,1]+cm[0,0])
  print('Sensitivity/Recall/TPR/Detection Rate : ', sensitivity )
  specificity = cm[1,1]/(cm[1,1]+cm[1,0])
  print('Specificity : ', specificity )
  FPR= cm[1,0]/(cm[1,0]+cm[1,1])
  print('FPR : ', FPR )
  TNR= cm[1,1]/(cm[0,1]+cm[1,1])
  print('Negative Precision / TNR: ', TNR)
  error= 1-accuracy
  print('Error : ', error )
  roc_auc=roc_auc_score(y_test, y_pred)
  print('ROC_AUC Score : ', roc_auc )

  label=['Low','High']
  disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=label)
  plt.figure(figsize=(1000, 800), dpi=700)
  disp.plot(cmap='hot_r',colorbar=True)
  plt.grid(False)
  plt.show()

  return y_pred_prob, roc_auc

classifier(rf)
classifier(dt)
classifier(svm)
classifier(lr)
```

4.2 Random Forest

All the metrics which are used for performance evaluation of the model have been calculated from the confusion matrix of Fig 19.

```
Output:
Accuracy:  0.8831168831168831
Recall:  0.8493150684931506
Precision:  0.8985507246376812
F1_Score:  0.8732394366197183
Sensitivity/Recall/TPR/Detection Rate:  0.8493150684931506
Specificity:  0.9135802469135802
FPR:  0.08641975308641975
Negative Precision / TNR:  0.8705882352941177
Error:  0.11688311688311692
ROC_AUC Score:  0.8814476577033656
```

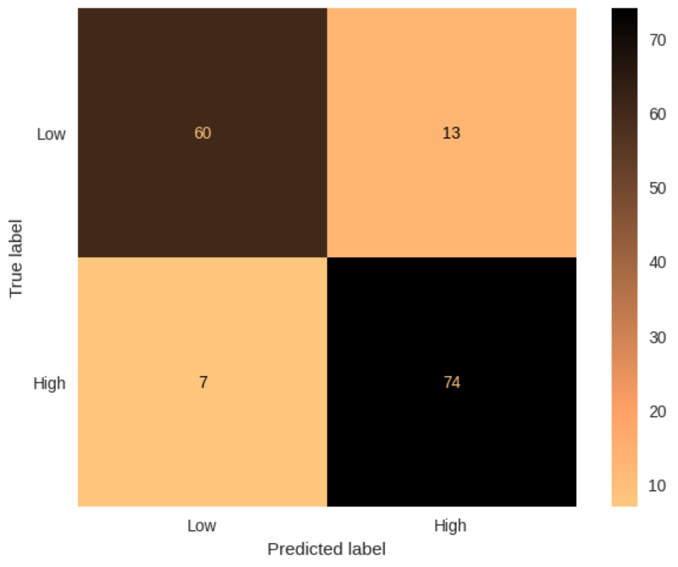


Fig 19: Confusion matrix for classification of housing price using random forest.

4.3 Decision Tree

All the metrics which are used for performance evaluation of the model have been calculated from the confusion matrix of Fig 20.

```
Output:
Accuracy:  0.7792207792207793
Recall:  0.7808219178082192
Precision:  0.76
F1_Score:  0.7702702702702703
Sensitivity/Recall/TPR/Detection Rate:  0.7808219178082192
Specificity:  0.7777777777777778
```

```
FPR:  0.2222222222222222
Negative Precision / TNR:  0.7974683544303798
Error:  0.22077922077922074
ROC_AUC Score:  0.7792998477929984
```

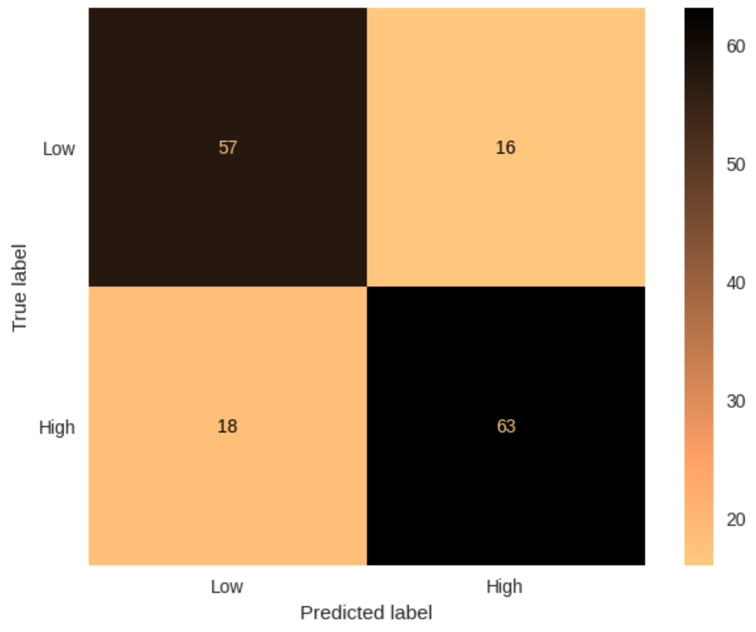


Fig 20: Confusion matrix for classification of housing price using decision tree.

4.4 Support Vector Machine

All the metrics which are used for performance evaluation of the model have been calculated from the confusion matrix of Fig 21.

```
Output:
Accuracy:  0.7207792207792207
Recall:  0.5205479452054794
Precision:  0.8260869565217391
F1_Score:  0.6386554621848739
Sensitivity/Recall/TPR/Detection Rate:  0.5205479452054794
Specificity:  0.9012345679012346
FPR:  0.09876543209876543
Negative Precision / TNR:  0.6759259259259259
Error:  0.27922077922077926
ROC_AUC Score:  0.710891256553357
```

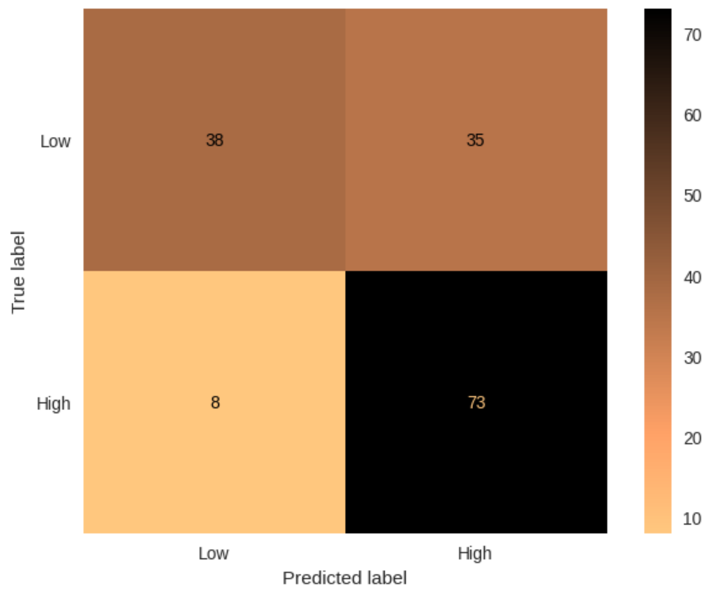


Fig 21: Confusion matrix for classification of housing price using support vector machine.

4.5 Logistic Regression

All the metrics which are used for performance evaluation of the model have been calculated from the confusion matrix of Fig 22.

Output:

```
Accuracy:  0.8246753246753247
Recall:  0.726027397260274
Precision:  0.8833333333333333
F1_Score:  0.7969924812030076
Sensitivity/Recall/TPR/Detection Rate:  0.726027397260274
Specificity:  0.9135802469135802
FPR:  0.08641975308641975
Negative Precision / TNR:  0.7872340425531915
Error:  0.17532467532467533
ROC_AUC Score:  0.8198038220869271
```

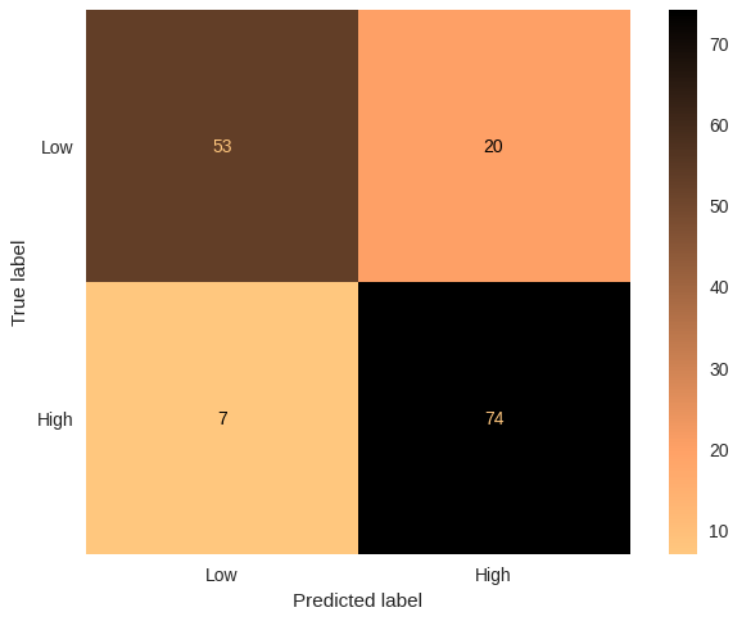


Fig 22: Confusion matrix for classification of housing price using logistic regression.

Table 6: Comparative table of different classifiers for classifying housing prices.

Metrics	RF (%)	DT (%)	SVM (%)	LR (%)
Accuracy	88.31	77.92	72.08	82.47
Sensitivity	84.93	78.08	52.05	72.60
Specificity	91.36	77.78	90.12	91.36
Predictive value positive	89.86	76.00	82.61	88.33
Predictive value negative	87.06	79.75	67.59	78.72

From Table 6 it can be observed that the Random Forest classifier provides better results with higher detection rate than other three classifiers. So, it can be declared that Random Forest model is perform better than Decision Tree, Support Vector Machine and Logistic Regression model for Boston housing price dataset.

4.6 ROC Curve

In ML, the **Receiver Operator Characteristics** (ROC) curve is a graphical way of representing the tradeoff between true positive rate (TPR) and false positive rate (FPR) of the model. The optimal ROC curve has a higher true positive rate together with a lower false positive rate. While the area under the ROC curve (AUC) tells the usefulness of the model. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

Python Script for Fig 23:

```
fpr1, tpr1, thresholds1 = roc_curve(y_test, y_pred_prob_rf[:,1], pos_label=2)
fpr2, tpr2, thresholds2 = roc_curve(y_test, y_pred_prob_dt[:,1], pos_label=2)
fpr3, tpr3, thresholds3 = roc_curve(y_test, y_pred_prob_svm[:,1], pos_label=2)
fpr4, tpr4, thresholds4 = roc_curve(y_test, y_pred_prob_lr[:,1], pos_label=2)

# roc curve for tpr = fpr
random_probs = [0 for i in range(len(y_test))]
p_fpr, p_tpr, _ = roc_curve(y_test, random_probs, pos_label=2)

plt.style.use('seaborn')

# plot roc curves
plt.plot(fpr1, tpr1, linestyle='--',color='red',lw=2,label='RF(AUC = %0.4f)' %
roc_auc_rf)
plt.plot(fpr2, tpr2, linestyle='--',color='green',lw=2, label='DT(AUC = %0.4f)' %
roc_auc_dt)
plt.plot(fpr3, tpr3, linestyle='--',color='blue',lw=2, label='SVM(AUC = %0.4f)' %
roc_auc_svm)
plt.plot(fpr4, tpr4, linestyle='--',color='purple',lw=2, label='LR(AUC = %0.4f)' %
roc_auc_lr)
plt.plot(p_fpr, p_tpr, linestyle='--', color='black',lw=2)

# title
plt.title('ROC curve')
# x label
plt.xlabel('False Positive Rate')
# y label
plt.ylabel('True Positive rate')

plt.legend(loc='best')
plt.show()
```

`

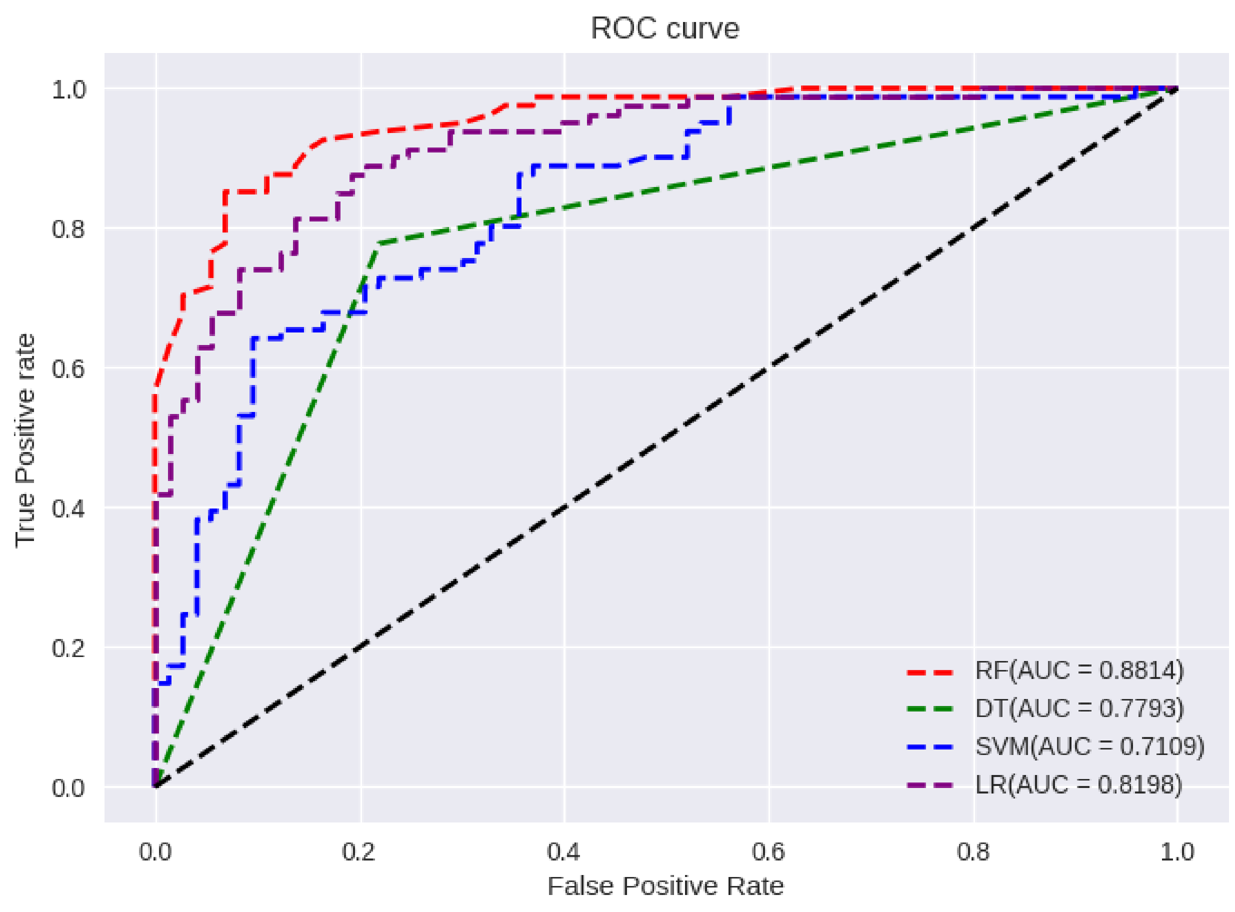


Fig 23: Comparison of ROC curve for the four different classification models.

In Fig 23, it can be seen that the RF classifier covers the greater area in the ROC curve with higher TPR than the other three classifier. The AUC score is highest for RF then LR. (0.8814>0.8198). So, it can be said that RF is the best classification model for identifying different housing prices (High/Low) with 88.14% ROC_AUC score.

4.7 Visualization of Tree

As the random forest tree is too large to visualize, a small tree extracted from the random forest model and has been visualized the reduced version in Fig 24.

Python Script:

```
tree_small = rf.estimators_[5]
# Save the tree as a png image
export_graphviz(tree_small, out_file = 'small_tree.dot', feature_names = feature_list,
rounded = True, precision = 1)
(graph, ) = pydot.graph_from_dot_file('small_tree.dot')
graph.write_png('small_tree.png');
```

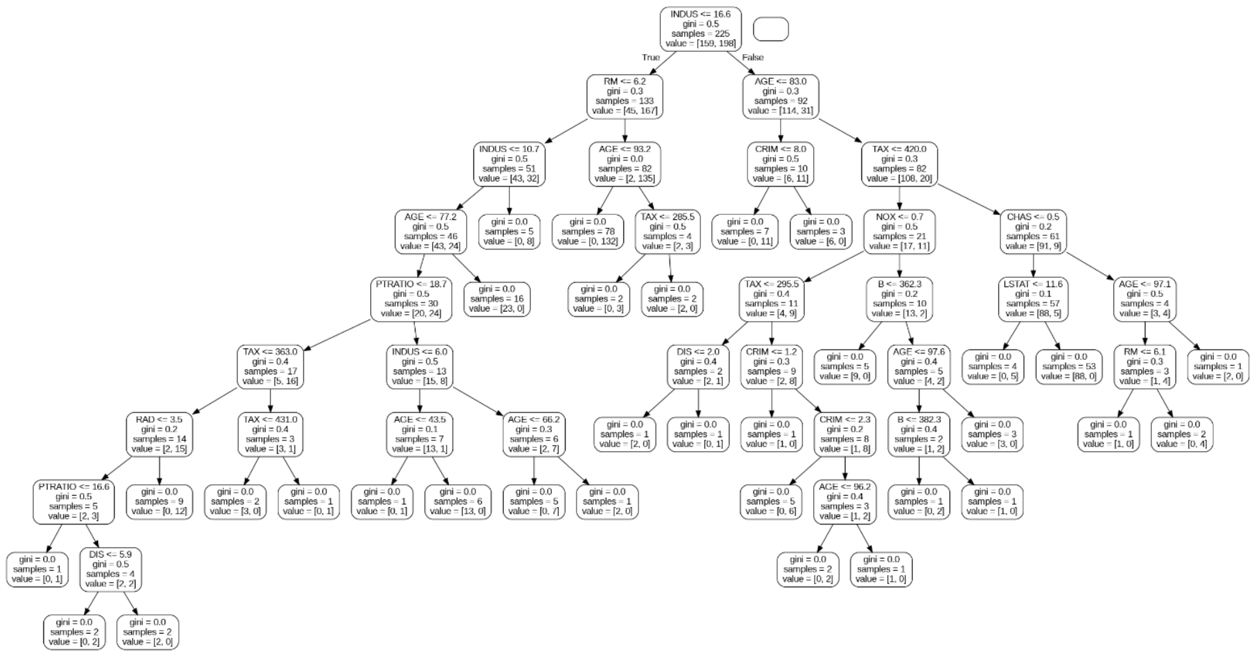


Fig 24: Reduced size tree for random forest model

#OLS Regression Results:

OLS (ordinary least squares) is a common technique used in analyzing linear regression. In brief, it compares the difference between individual points in your data set and the predicted best fit line to measure the amount of error produced.

<u>Python Script:</u>

```
import statsmodels.formula.api as smf
# Fit model
rf =
smf.ols('MEDV~CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT',data=df_bost
on)
result = rf.fit()
# Make predictions
predictions = result.predict(x_test)
print(result.summary())
```

Output:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   MEDV   R-squared:                       0.719
Model:                            OLS   Adj. R-squared:                  0.712
Method:                 Least Squares   F-statistic:                     97.84
Date:                Sat, 05 Aug 2023   Prob (F-statistic):         9.00e-128
Time:                        09:21:49   Log-Likelihood:                -1531.7
No. Observations:                 511   AIC:                             3091.
Df Residuals:                     497   BIC:                             3151.
Df Model:                          13
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     16.1997      4.131      3.921      0.000       8.083      24.317
CRIM          -0.0585      0.033     -1.772      0.077      -0.123       0.006
ZN             0.0711      0.014      5.215      0.000       0.044       0.098
INDUS         -0.0546      0.062     -0.879      0.380      -0.177       0.067
CHAS           3.0505      0.890      3.426      0.001       1.301       4.800
NOX           -4.8461      3.302     -1.468      0.143     -11.334       1.642
RM             4.2137      0.428      9.847      0.000       3.373       5.055
AGE           -0.0082      0.014     -0.605      0.546      -0.035       0.018
DIS           -1.4761      0.206     -7.152      0.000      -1.882      -1.071
RAD            0.1909      0.066      2.902      0.004       0.062       0.320
TAX           -0.0133      0.004     -3.409      0.001      -0.021      -0.006
PTRATIO       -0.2359      0.069     -3.439      0.001      -0.371      -0.101
B              0.0099      0.003      3.543      0.000       0.004       0.015
LSTAT         -0.5561      0.052    -10.659      0.000      -0.659      -0.454
==============================================================================
Omnibus:                      171.606   Durbin-Watson:                   1.015
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              730.987
Skew:                           1.457   Prob(JB):                    1.85e-159
Kurtosis:                       8.084   Cond. No.                     1.13e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.13e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```