In [132]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  1967 non-null   float64
 1   blue           2000 non-null   object 
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   object 
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            1757 non-null   float64
 14  sc_h           2000 non-null   int64  
 15  sc_w           2000 non-null   int64  
 16  talk_time      2000 non-null   int64  
 17  three_g        2000 non-null   int64  
 18  touch_screen   2000 non-null   int64  
 19  wifi           2000 non-null   int64  
 20  price_range    2000 non-null   int64  
dtypes: float64(4), int64(15), object(2)
memory usage: 328.2+ KB
```

In [133]:
```
df.to_csv('Mobile_Price_Classifiation_train_missing.csv')
```

--- outliers ---

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
#from sklearn.datasets import load_boston
```

In [3]:
```
df = pd.read_csv('./dataset/boston_train.csv')
```

In [50]:
```
df.head()
```

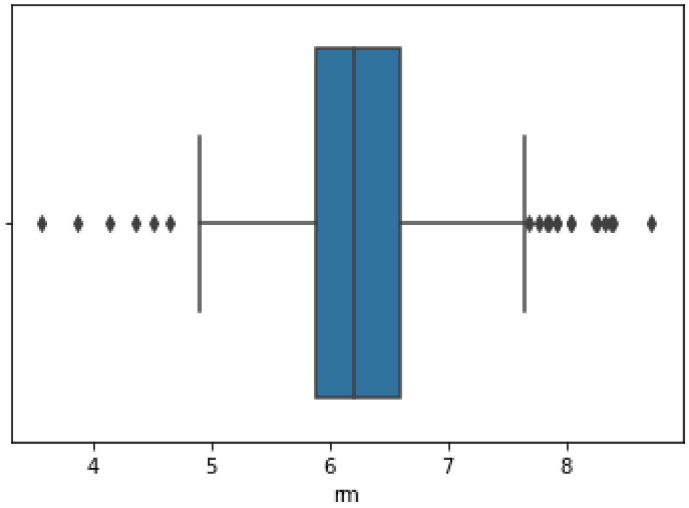
Out[50]:

	ID	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
3	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
4	7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9

In [4]:
```
sns.boxplot(df.rm)
```

```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following
variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[4]: <AxesSubplot:xlabel='rm'>



In [51]:
```
dfS = df[['lstat','rm','crim']]
```

In [52]:
```
dfS.columns = ['LSTAT','RM','CRIM']
```

In [53]:
```
dfS
```

Out[53]:

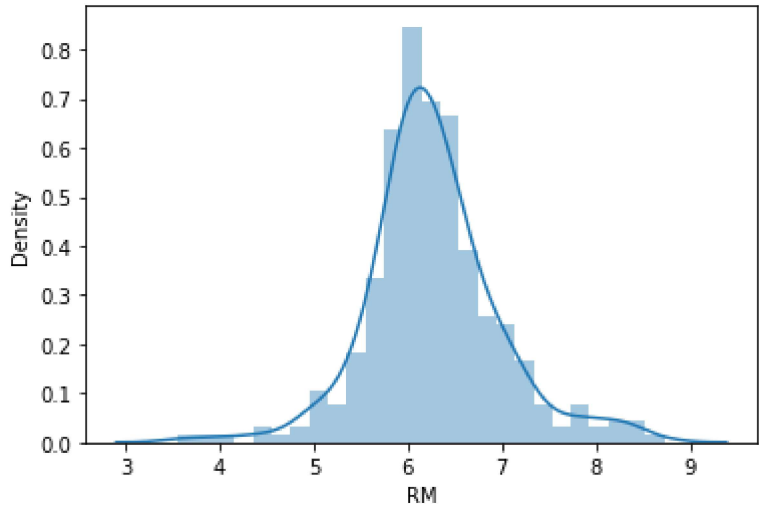
	LSTAT	RM	CRIM
0	4.98	6.575	0.00632
1	9.14	6.421	0.02731
2	2.94	6.998	0.03237
3	5.33	7.147	0.06905
4	12.43	6.012	0.08829
...	...	...	...
328	15.10	5.569	0.17783
329	9.67	6.593	0.06263
330	9.08	6.120	0.04527
331	5.64	6.976	0.06076
332	7.88	6.030	0.04741

333 rows × 3 columns

In [54]:
```
sns.distplot(dfS['RM'])
```

```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

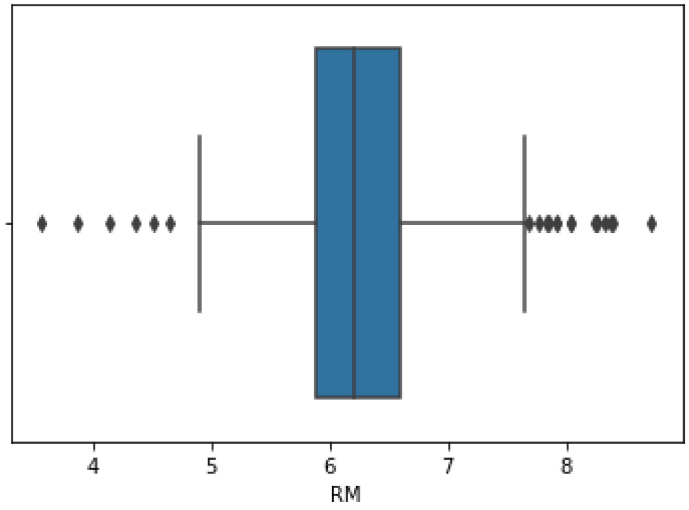
Out[54]:
```
<AxesSubplot:xlabel='RM', ylabel='Density'>
```



In [55]:
```
sns.boxplot(dfS['RM'])
```

```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following
variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[55]:
```
<AxesSubplot:xlabel='RM'>
```



--- removing the outliers ---

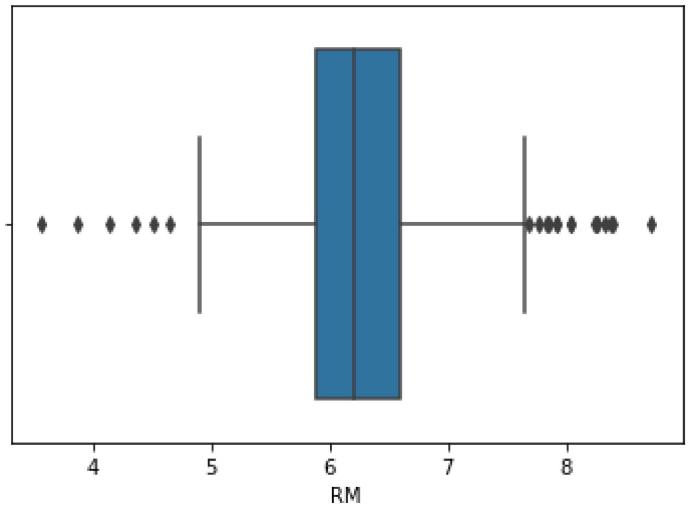
--- outliers boundires function ---

In [56]:
```
def find_boundaries(df, variable, distance):
    Q1 =  df[variable].quantile(0.25)
    Q3 =  df[variable].quantile(0.75)
    IQR = Q3 - Q1
    lower_boundary = Q1 - (IQR * distance)
    upper_boundary = Q3 + (IQR * distance)
    return upper_boundary, lower_boundary
```

In [57]:
```
sns.boxplot(dfS.RM)
```

```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following
variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[57]: <AxesSubplot:xlabel='RM'>



In [58]:
```
RM_upper_limit, RM_lower_limit = find_boundaries(dfS,'RM',1.5)
```

In [59]:
```
RM_upper_limit, RM_lower_limit
```

Out[59]: (7.661499999999998, 4.817500000000001)

Let's create a Boolean vector to flag the outliers in RM:

In [60]:
```
outliers_RM = np.where(dfS['RM'] > RM_upper_limit, True, np.where(dfS['RM'] < RM_lower_limit, True,False))
```

In [61]:
```
dfS.shape
```

Out[61]: (333, 3)

In [62]:
```
dfS['RM'][outliers_RM].count()   # count the outlier
```

Out[62]: 21

--- Finally, let's remove the outliers from the dataset: ---

In [63]:
```
dfS_trimmed = dfS.loc[~(outliers_RM)]
```

In [64]:
```
dfS_trimmed.shape
```

Out[64]: (312, 3)

In [65]:
```
dfS_trimmed.head()
```

Out[65]:

	LSTAT	RM	CRIM
0	4.98	6.575	0.00632
1	9.14	6.421	0.02731
2	2.94	6.998	0.03237
3	5.33	7.147	0.06905
4	12.43	6.012	0.08829

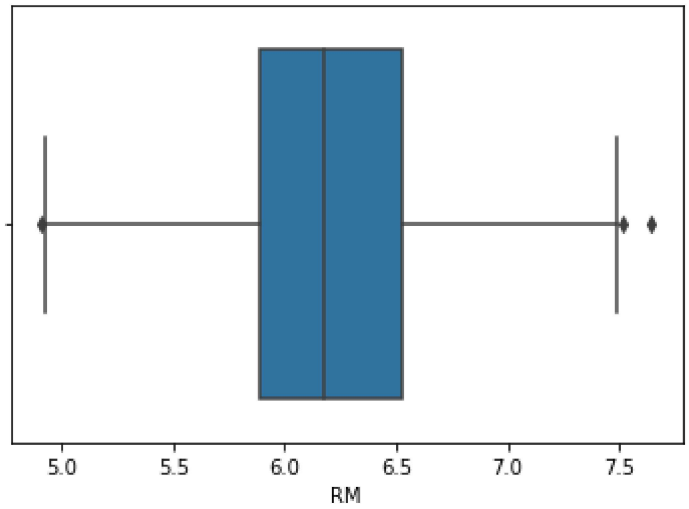
In [19]:
```
dfS.RM.min()
```

Out[19]: 3.561

In [16]:
```
sns.boxplot(dfS_trimmed.RM)
```

```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following
variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

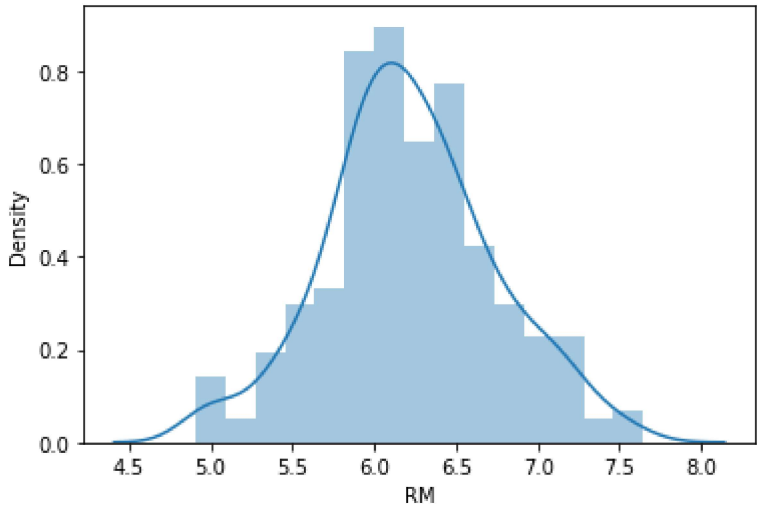
Out[16]: <AxesSubplot:xlabel='RM'>



In [38]:
```
sns.distplot(dfS_trimmed['RM'])
```

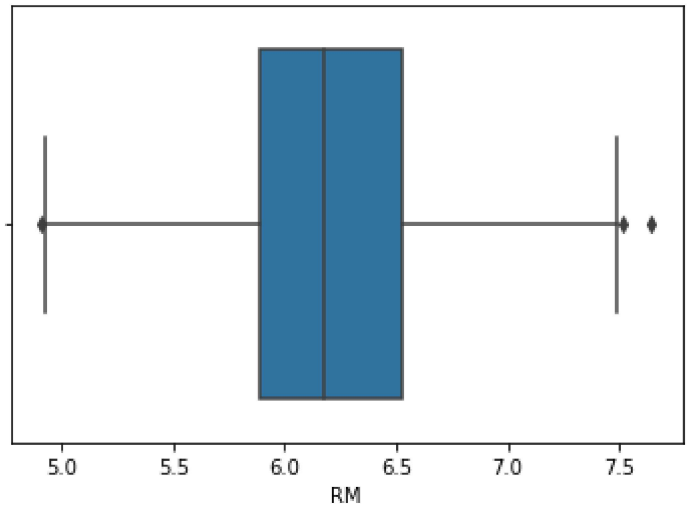
```
C:\Users\Khalid Khan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[38]:
```
<AxesSubplot:xlabel='RM', ylabel='Density'>
```



In [37]:
```
sns.boxplot(dfS_trimmed['RM'])
```

Out[37]: <AxesSubplot:xlabel='RM'>



--- Making NaN the outlier ---

In [66]:
```
RM_UB, RM_LB = find_boundaries(dfS,'RM',1.5)
```

In [67]:
```
dfS['RM_alt_trim'] = dfS.RM[(dfS.RM < RM_UB) & (dfS.RM > RM_LB)]  # This for the trim the outliers
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_10716\2531889305.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS['RM_alt_trim'] = dfS.RM[(dfS.RM < RM_UB) & (dfS.RM > RM_LB)]  # This for the trim the outliers
```

In [68]:
```
dfS.isnull().sum()
```

Out[68]:
```
LSTAT          0
RM             0
CRIM           0
RM_alt_trim    21
dtype: int64
```

In [30]:
```
dfS.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 333 entries, 0 to 332
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   LSTAT        333 non-null    float64
 1   RM           333 non-null    float64
 2   CRIM         333 non-null    float64
 3   RM_alt_trim  312 non-null    float64
dtypes: float64(4)
memory usage: 10.5 KB
```

In [36]:
```
dfS.dropna(inplace=True)
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_16232\557178398.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS.dropna(inplace=True)
```

In [37]:
```
dfS.shape
```

Out[37]:
```
(312, 4)
```

--- 2. Making NaN outliers ---

In [39]:
```
dfS_new_NaN = dfS.RM[(dfS.RM < RM_upper_limit) & (dfS.RM > RM_lower_limit)]
```

In [42]:
```
dfS['RM_new'] = dfS_new_NaN
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_20980\3764809249.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS['RM_new'] = dfS_new_NaN
```

In [43]:
```
dfS.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 333 entries, 0 to 332
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   LSTAT   333 non-null    float64
 1   RM      333 non-null    float64
 2   CRIM    333 non-null    float64
 3   RM_new  312 non-null    float64
dtypes: float64(4)
memory usage: 10.5 KB
```

In [63]:
```
dfS.RM.min()
```

Out[63]:
```
3.561
```

In [64]:
```
dfS.RM.max()
```

Out[64]:
```
8.725
```

--- 3. Binning technique for outlier ---

In [72]:
```
dfS['RM_bin_3'] = pd.qcut(dfS['RM'], 4 , labels=[1,2,3,4] ) # fix bining
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_10716\234028650.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS['RM_bin_3'] = pd.qcut(dfS['RM'], 4 , labels=[1,2,3,4] ) # fix bining
```

In [73]:
```
dfS.RM_bin_3.value_counts()
```

Out[73]:
```
1    84
2    83
3    83
4    83
Name: RM_bin_3, dtype: int64
```

In [72]:
```
dfS['RM_bin_3'] = pd.cut(dfS['RM'], [2,3.8,5.2,6.9,8.8], labels=[1,2,3,4] , include_lowest=True) # Vriable binin
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_16232\2845768594.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS['RM_bin_3'] = pd.cut(dfS['RM'], [2,3.8,5.2,6.9,8.8], labels=[1,2,3,4] , include_lowest=True) # Vriable b
ining
```

In [73]:
```
dfS.RM_bin_3.value_counts()
```

Out[73]:
```
3    268
4     48
2     16
1      1
Name: RM_bin_3, dtype: int64
```

In [68]:
```
dfS.RM_bin_2.value_counts()
```

Out[68]:
```
3    268
4     48
2     16
1      1
Name: RM_bin_2, dtype: int64
```

In [57]:
```
dfS.RM_bin.value_counts()
```

Out[57]:
```
3    162
2    146
4     19
1      6
Name: RM_bin, dtype: int64
```

In [74]:
```
dfS['RM_bin_2'] = pd.qcut(dfS['RM'], 5, labels=[0,1,2,3,4])
```

```
C:\Users\Khalid Khan\AppData\Local\Temp\ipykernel_16232\4283262163.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  dfS['RM_bin_2'] = pd.qcut(dfS['RM'], 5, labels=[0,1,2,3,4])
```

In [75]:
```
dfS.RM_bin_2.value_counts()
```

Out[75]:
```
0    67
2    67
4    67
1    66
3    66
Name: RM_bin_2, dtype: int64
```

---Practice on selection of values ---