Load the dataset

In [1]:

```
from sklearn.datasets import load_boston
boston = load_boston()
```

Dataset characteristics

In [2]:

```
print(boston.DESCR)
```

```
.. _boston_dataset:

Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical pred
ictive. Median Value (attribute 14) is usually the tar
get.

    :Attribute Information (in order):
        - CRIM     per capita crime rate by town
        - ZN       proportion of residential land zone
d for lots over 25,000 sq.ft.
        - INDUS    proportion of non-retail business a
cres per town
        - CHAS     Charles River dummy variable (= 1 i
f tract bounds river; 0 otherwise)
        - NOX      nitric oxides concentration (parts
per 10 million)
        - RM       average number of rooms per dwellin
g
        - AGE      proportion of owner-occupied units
built prior to 1940
```

```
        - DIS      weighted distances to five Boston e
mployment centres
        - RAD      index of accessibility to radial hi
ghways
        - TAX      full-value property-tax rate per $1
0,000
        - PTRATIO  pupil-teacher ratio by town
        - B        1000(Bk - 0.63)^2 where Bk is the p
roportion of blacks by town
        - LSTAT    % lower status of the population
        - MEDV     Median value of owner-occupied home
s in $1000's

    :Missing Attribute Values: None

    :Creator: Harrison, D. and Rubinfeld, D.L.

This is a copy of UCI ML housing dataset.
https://archive.ics.uci.edu/ml/machine-learning-databa
ses/housing/ (https://archive.ics.uci.edu/ml/machine-l
earning-databases/housing/)


This dataset was taken from the StatLib library which
is maintained at Carnegie Mellon University.

The Boston house-price data of Harrison, D. and Rubinf
eld, D.L. 'Hedonic
prices and the demand for clean air', J. Environ. Econ
omics & Management,
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch,
'Regression diagnostics
...', Wiley, 1980.   N.B. Various transformations are
used in the table on
pages 244-261 of the latter.

The Boston house-price data has been used in many mach
ine learning papers that address regression
problems.

.. topic:: References

   - Belsley, Kuh & Welsch, 'Regression diagnostics: I
dentifying Influential Data and Sources of Collinearit
y', Wiley, 1980. 244-261.
```

```
   - Quinlan,R. (1993). Combining Instance-Based and M
odel-Based Learning. In Proceedings on the Tenth Inter
national Conference of Machine Learning, 236-243, Univ
ersity of Massachusetts, Amherst. Morgan Kaufmann.
```

In [3]:

```
print(f'  Data shape = {boston.data.shape}')
print(f'Target shape = {boston.target.shape}')
```

```
  Data shape = (506, 13)
Target shape = (506,)
```

In [4]:

```
boston.feature_names
```

Out[4]:

```
array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AG
E', 'DIS', 'RAD',
       'TAX', 'PTRATIO', 'B', 'LSTAT'], dtype='<U7')
```

Load data into a dataframe and explore

In [5]:

```
import pandas as pd

boston_df = pd.DataFrame(boston.data,
                         columns=boston.feature_names)
boston_df.head()
```

Out[5]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	F
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

target value MEDV is missing, we create a new column for MEDV and add into data frame

In [6]:

```
boston_df['MEDV'] = pd.Series(boston.target)
boston_df.head()
```

Out[6]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	F
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

In [7]:

```
boston_df.describe()
```

Out[7]:

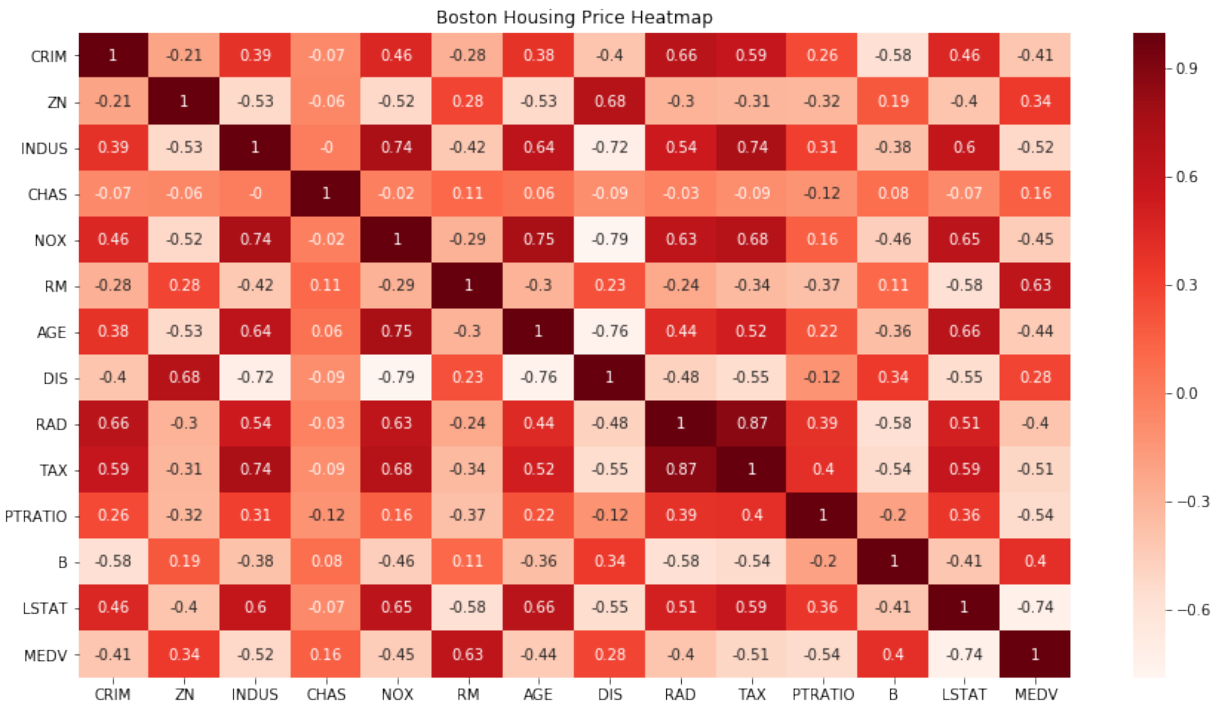
	CRIM	ZN	INDUS	CHAS	NOX	
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.00
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.28
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.70
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.56
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.88
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.20
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.62
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.78

Data visualizations

We use heatmap to visualize the linear relationship between variables. In this case, we are only looking at the realtionship to MEDV column.

In [8]:

```
sample_df = boston_df.sample(frac=0.4) # sample data to 40%
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(15,8))
correlation_matrix = sample_df.corr().round(2)
# annot = True to print the values inside the square
axe=sns.heatmap(data=correlation_matrix, annot=True,cmap="Reds")
axe.set_title('Boston Housing Price Heatmap');
```



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	1	-0.21	0.39	-0.07	0.46	-0.28	0.38	-0.4	0.66	0.59	0.26	-0.58	0.46	-0.41
ZN	-0.21	1	-0.53	-0.06	-0.52	0.28	-0.53	0.68	-0.3	-0.31	-0.32	0.19	-0.4	0.34
INDUS	0.39	-0.53	1	-0	0.74	-0.42	0.64	-0.72	0.54	0.74	0.31	-0.38	0.6	-0.52
CHAS	-0.07	-0.06	-0	1	-0.02	0.11	0.06	-0.09	-0.03	-0.09	-0.12	0.08	-0.07	0.16
NOX	0.46	-0.52	0.74	-0.02	1	-0.29	0.75	-0.79	0.63	0.68	0.16	-0.46	0.65	-0.45
RM	-0.28	0.28	-0.42	0.11	-0.29	1	-0.3	0.23	-0.24	-0.34	-0.37	0.11	-0.58	0.63
AGE	0.38	-0.53	0.64	0.06	0.75	-0.3	1	-0.76	0.44	0.52	0.22	-0.36	0.66	-0.44
DIS	-0.4	0.68	-0.72	-0.09	-0.79	0.23	-0.76	1	-0.48	-0.55	-0.12	0.34	-0.55	0.28
RAD	0.66	-0.3	0.54	-0.03	0.63	-0.24	0.44	-0.48	1	0.87	0.39	-0.58	0.51	-0.4
TAX	0.59	-0.31	0.74	-0.09	0.68	-0.34	0.52	-0.55	0.87	1	0.4	-0.54	0.59	-0.51
PTRATIO	0.26	-0.32	0.31	-0.12	0.16	-0.37	0.22	-0.12	0.39	0.4	1	-0.2	0.36	-0.54
B	-0.58	0.19	-0.38	0.08	-0.46	0.11	-0.36	0.34	-0.58	-0.54	-0.2	1	-0.41	0.4
LSTAT	0.46	-0.4	0.6	-0.07	0.65	-0.58	0.66	-0.55	0.51	0.59	0.36	-0.41	1	-0.74
MEDV	-0.41	0.34	-0.52	0.16	-0.45	0.63	-0.44	0.28	-0.4	-0.51	-0.54	0.4	-0.74	1

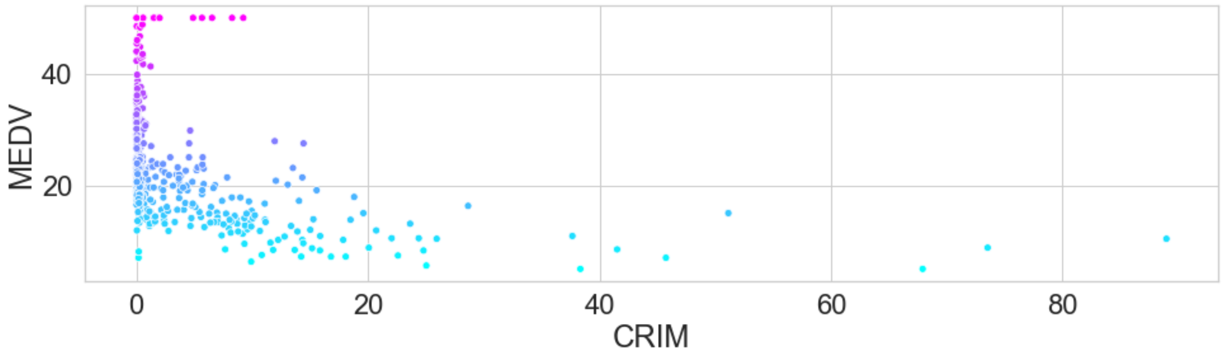
From heatmap, we can see: CHAS, DIS,B have week corrlation between MEDV with coefficient of 0.16, 0.28 and 0.4 respectively.

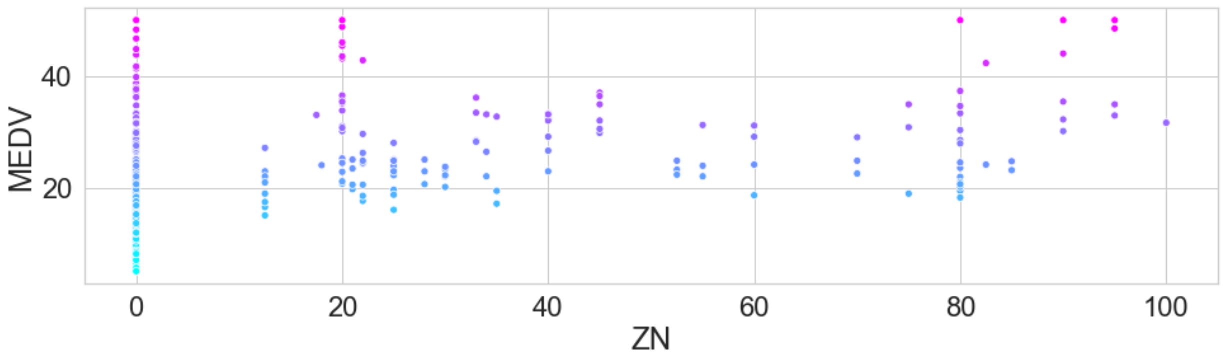
In [9]:

```
sns.set(font_scale=2)
sns.set_style('whitegrid')
```

In [18]:

```python
# We used the full data set to plot these graphs because there is n
for feature in boston.feature_names:
    plt.figure(figsize=(16, 4))
    sns.scatterplot(data=boston_df, x=feature, y='MEDV',
                    hue='MEDV',
                    palette='cool', legend=False)
```





Split the data for training and testing

In [16]:

```
from sklearn.model_selection import train_test_split
import numpy as np
boston_data_new = np.delete(boston.data,3,1)
# We found that CHAS contains only variables 0 and 1, which may hur
X_train, X_test, y_train, y_test = \
    train_test_split(boston_data_new, boston.target, random_state=5

print(f'Shape of training set = {X_train.shape}')
print(f'Shape of testing  set = {X_test.shape}')
```

```
Shape of training set = (379, 12)
Shape of testing  set = (127, 12)
```

Train the model

In [13]:

```
from sklearn.linear_model import LinearRegression

linear_regression = LinearRegression()
linear_regression.fit(X=X_train, y=y_train)

for i, name in enumerate(np.delete(boston.feature_names,3,0)):
    print(f'{name:>11}: {linear_regression.coef_[i]:24.20f}')

print()
print(f'y-intercept: {linear_regression.intercept_:23.20f}')
```

```
       CRIM:  -0.15913839634194795747
         ZN:   0.04890147232976368302
      INDUS:   0.01445084249427397612
        NOX: -13.69906809612140463628
         RM:   3.95110016068859470906
        AGE:  -0.00539332527178060499
        DIS:  -1.45933358556167180886
        RAD:   0.36650893681391799594
        TAX:  -0.01395246623283373857
    PTRATIO:  -0.95282684633969105814
          B:   0.01359438499293804704
      LSTAT:  -0.53257535640300879276

y-intercept: 32.62557602342805296303
```

Test the model

In [14]:

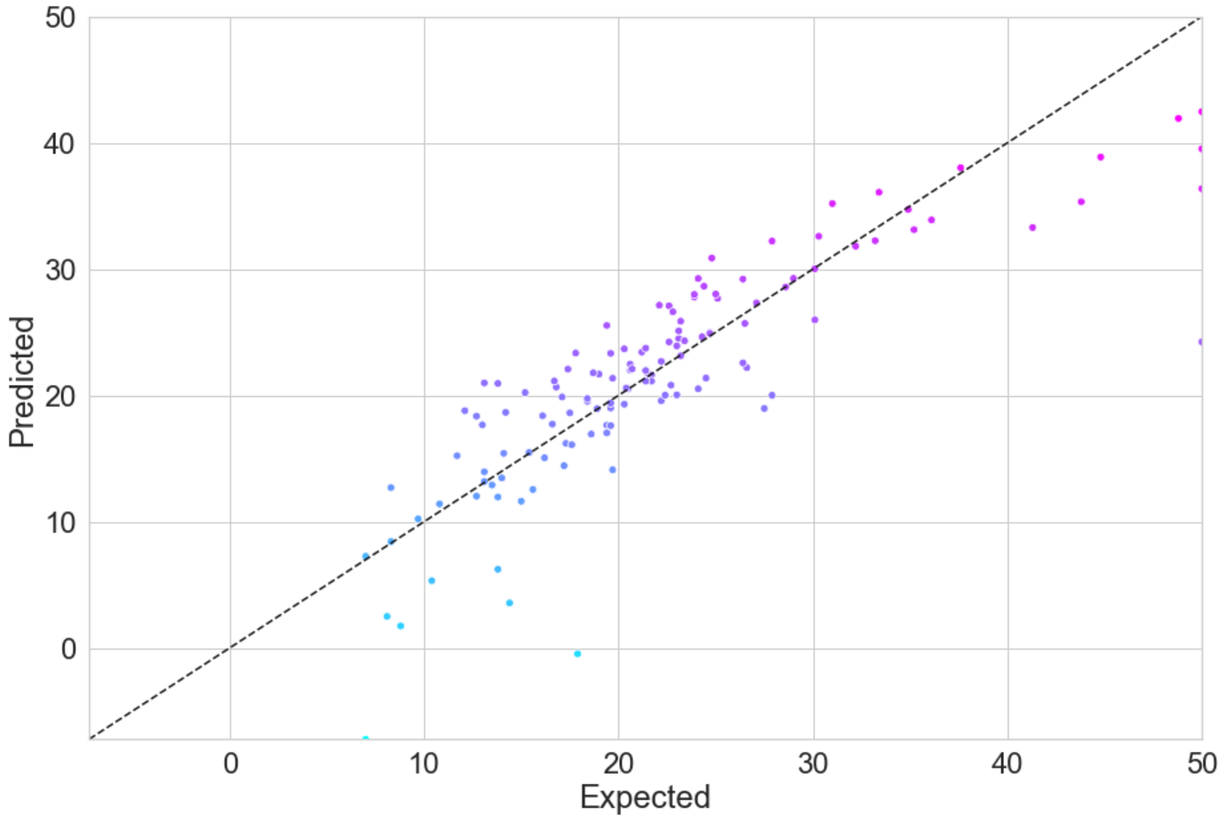
```
import math
from sklearn import metrics

predicted = linear_regression.predict(X_test)
expected  = y_test

r2 = metrics.r2_score(expected, predicted)
r  = math.sqrt(r2)

print(f'coefficient of determination = {r2:.1f}')
print(f'     correlation coefficient = {r:.1f}')
```

```
coefficient of determination = 0.7
     correlation coefficient = 0.8
```

Visualize the expected vs. predicted prices

In [15]:

```
df = pd.DataFrame()

df['Expected']  = pd.Series(expected)
df['Predicted'] = pd.Series(predicted)

figure = plt.figure(figsize=(15, 10))

axes = sns.scatterplot(data=df, x='Expected', y='Predicted',
                       hue='Predicted', palette='cool',
                       legend=False)

start = min(expected.min(), predicted.min())
end   = max(expected.max(), predicted.max())

axes.set_xlim(start, end)
axes.set_ylim(start, end)

line = plt.plot([start, end], [start, end], 'k--')
```



Load the dataset

In [1]:

Dataset characteristics

In [2]:

```
.. _boston_dataset:

Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical pred
ictive. Median Value (attribute 14) is usually the tar
get.

    :Attribute Information (in order):
        - CRIM     per capita crime rate by town
        - ZN       proportion of residential land zone
d for lots over 25,000 sq.ft.
        - INDUS    proportion of non-retail business a
cres per town
```

In [3]:

```
  Data shape = (506, 13)
Target shape = (506,)
```

In [4]:

Out[4]:

```
array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AG
E', 'DIS', 'RAD',
       'TAX', 'PTRATIO', 'B', 'LSTAT'], dtype='<U7')
```

Load data into a dataframe and explore

In [5]:

Out[5]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	F
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

target value MEDV is missing, we create a new column for MEDV and add into data frame

In [6]:

Out[6]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	F
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

In [7]:

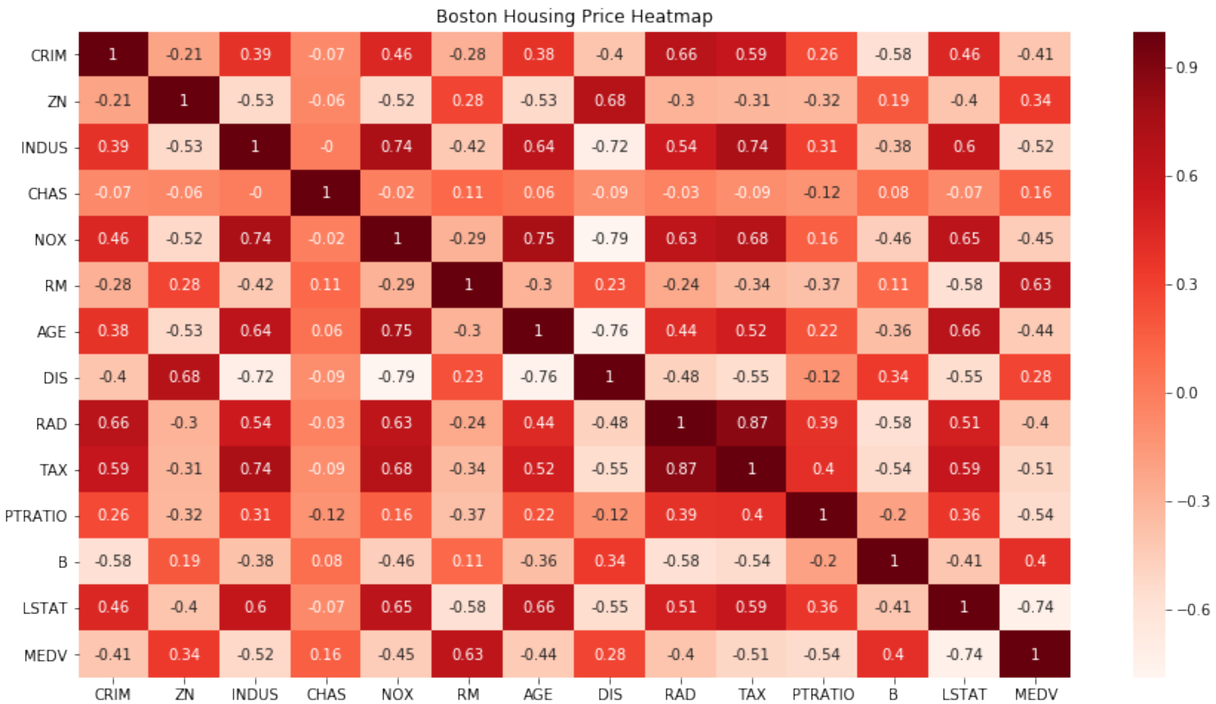
Out[7]:

	CRIM	ZN	INDUS	CHAS	NOX	
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.00
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.28
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.70
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.56
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.88
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.20
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.62
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.78

Data visualizations

We use heatmap to visualize the linear relationship between variables. In this case, we are only looking at the realtionship to MEDV column.

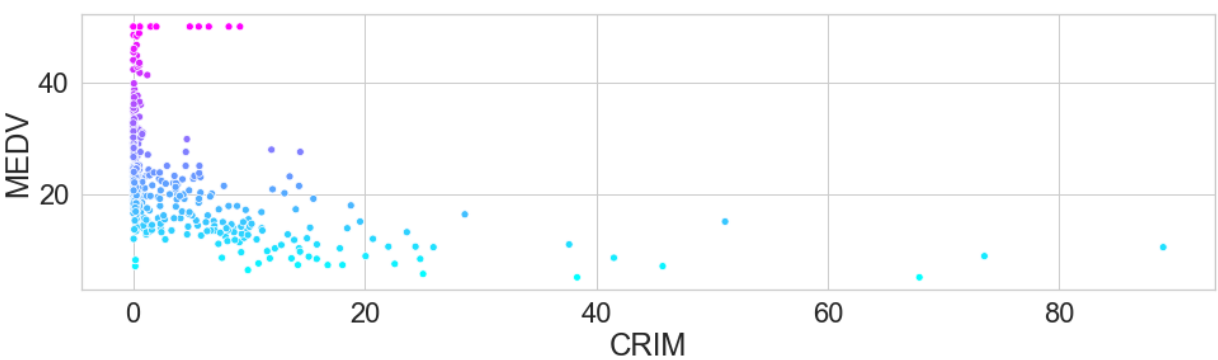
In [8]:

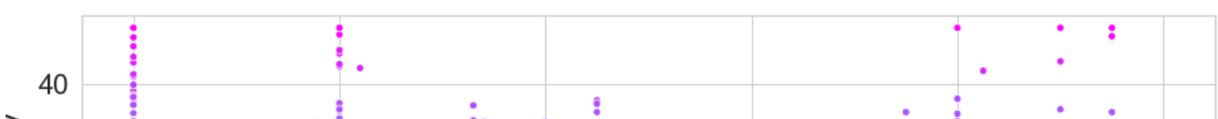


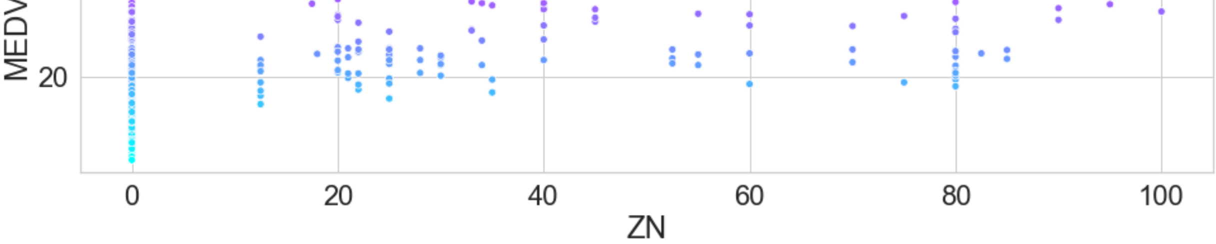
From heatmap, we can see: CHAS, DIS,B have week corrlation between MEDV with coefficient of 0.16, 0.28 and 0.4 respectively.

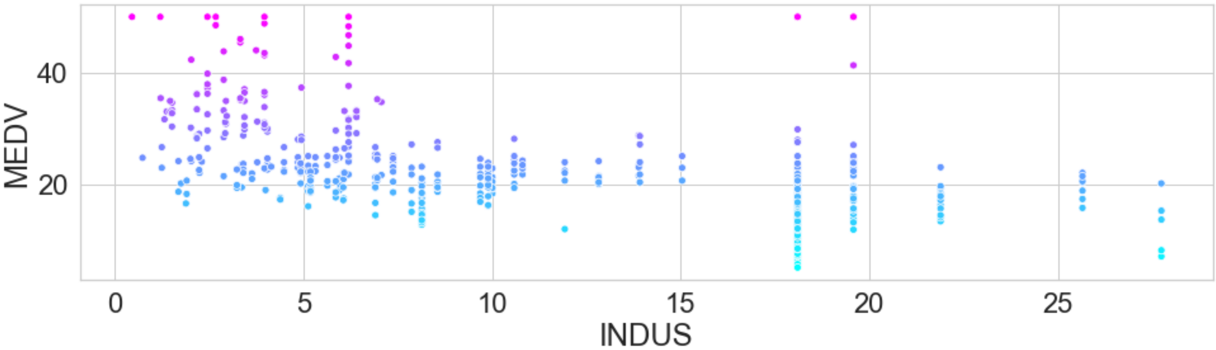
In [9]:

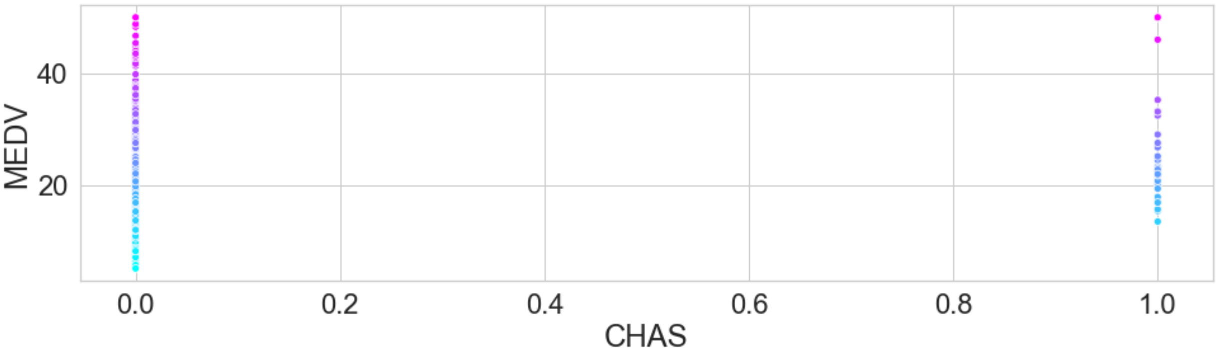
In [18]:

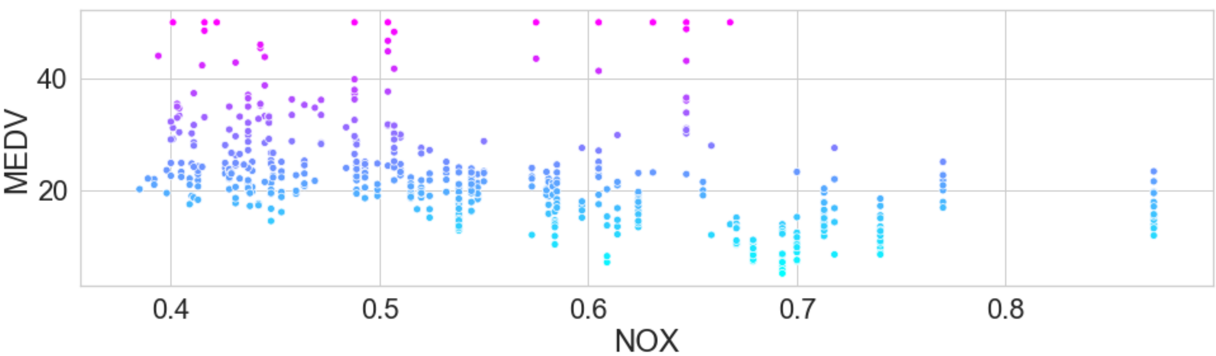


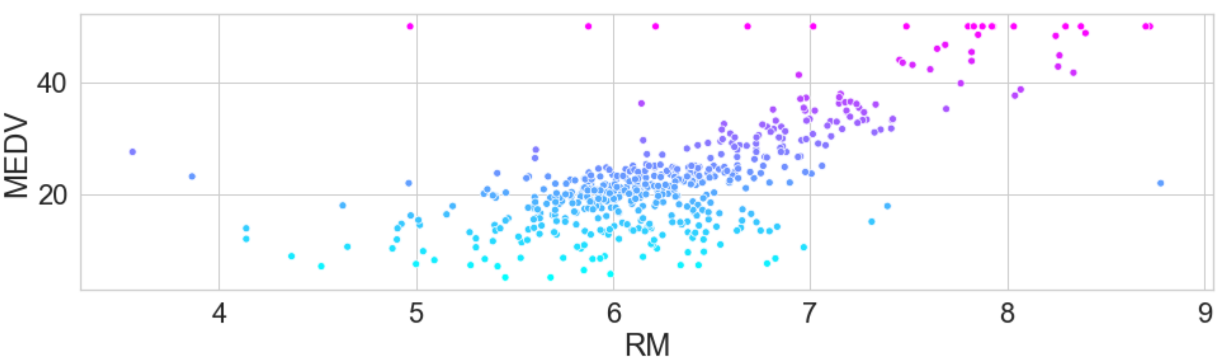


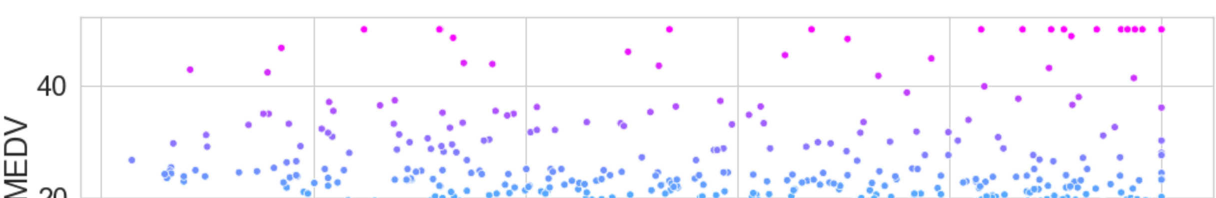

MEDV
20
0
20
40
60
80
100
ZN

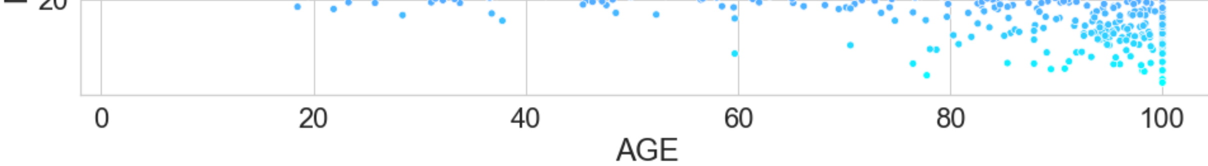

MEDV
40
20
0
5
10
15
20
25
INDUS

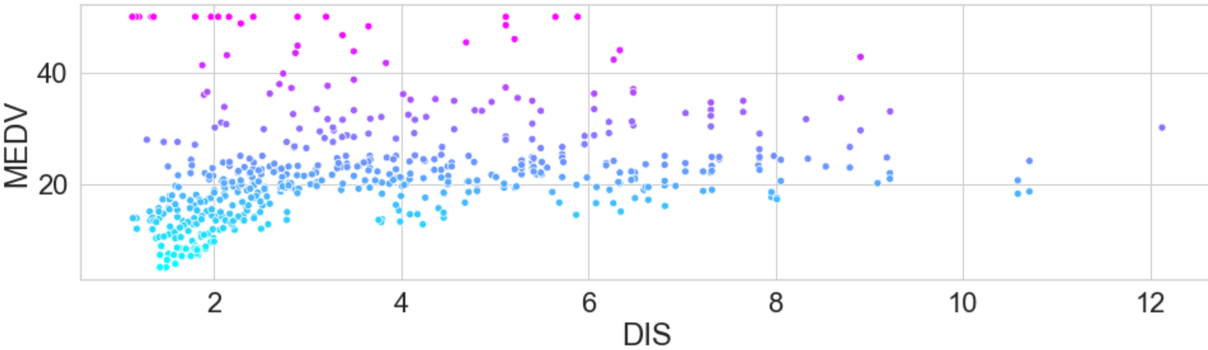

MEDV
40
20
0.0
0.2
0.4
0.6
0.8
1.0
CHAS

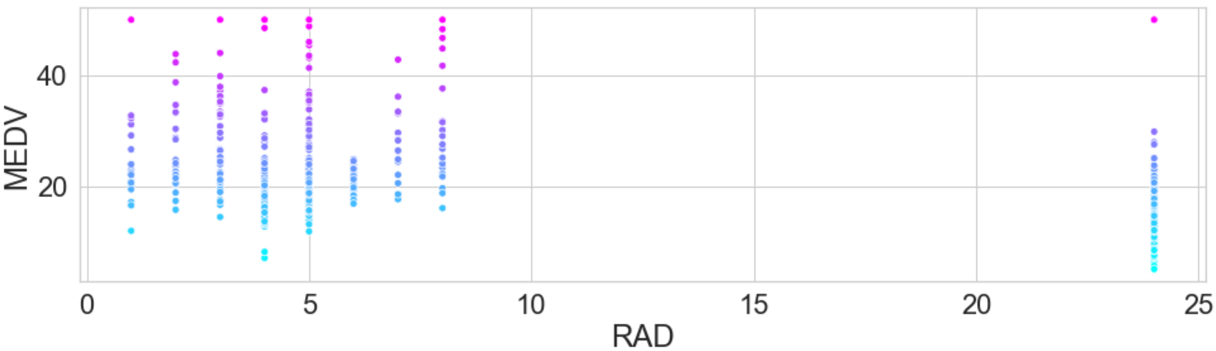

MEDV
40
20
0.4
0.5
0.6
0.7
0.8
NOX

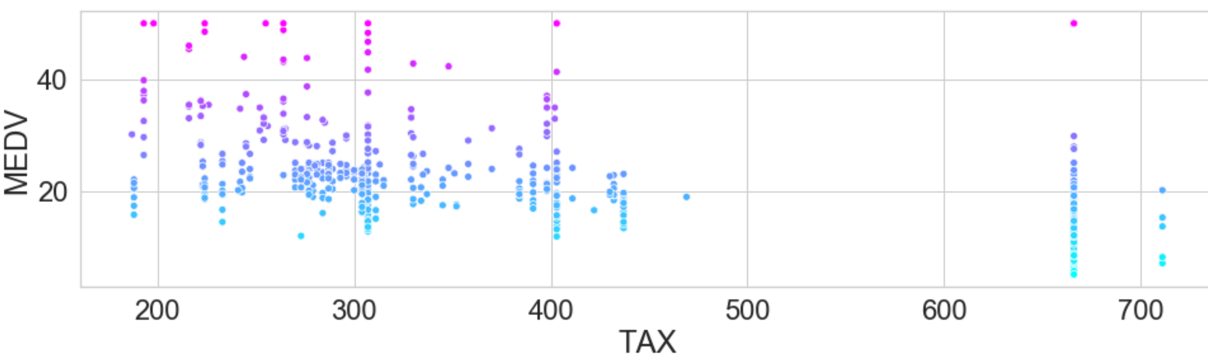

MEDV
40
20
4
5
6
7
8
9
RM

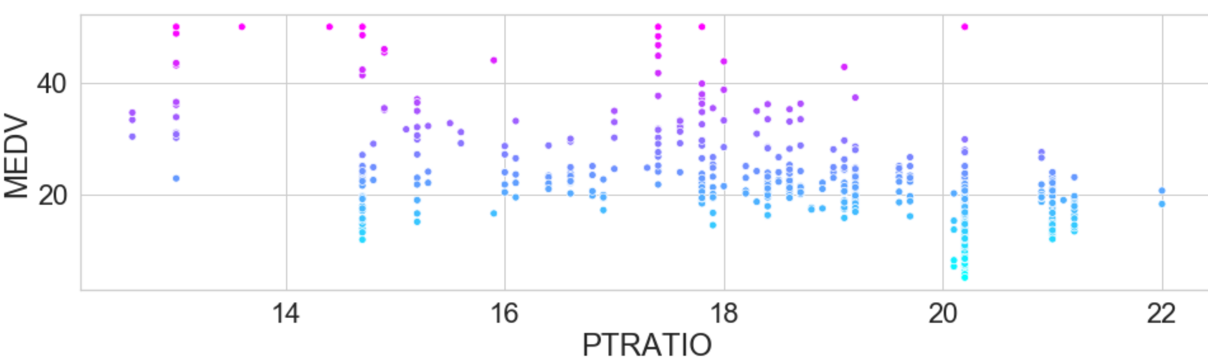

MEDV
40
20

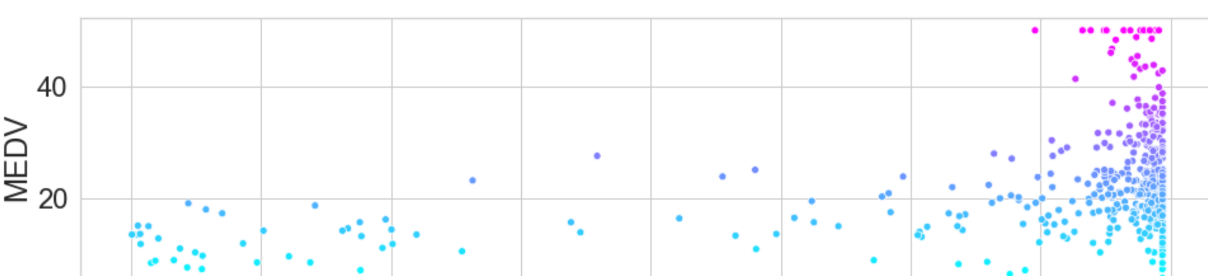

20
0
20
40
60
80
100
AGE

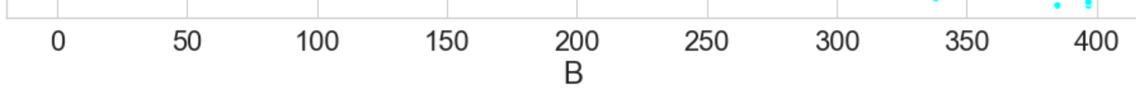

MEDV
40
20
2
4
6
8
10
12
DIS

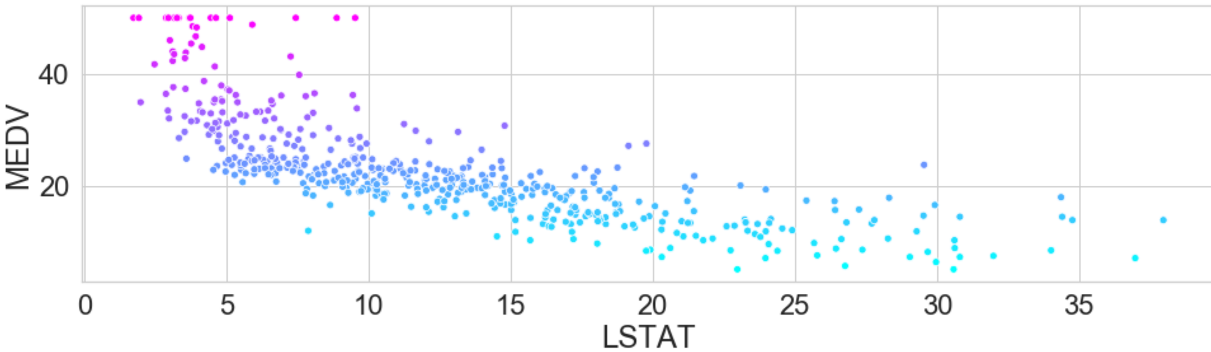

MEDV
40
20
0
5
10
15
20
25
RAD


MEDV
40
20
200
300
400
500
600
700
TAX


MEDV
40
20
14
16
18
20
22
PTRATIO


MEDV
40
20





Split the data for training and testing

In [16]:

```
Shape of training set = (379, 12)
Shape of testing  set = (127, 12)
```

Train the model

In [13]:

```
       CRIM:  -0.15913839634194795747
         ZN:   0.04890147232976368302
      INDUS:   0.01445084249427397612
        NOX: -13.69906809612140463628
         RM:   3.95110016068859470906
        AGE:  -0.00539332527178060499
        DIS:  -1.45933358556167180886
        RAD:   0.36650893681391799594
        TAX:  -0.01395246623283373857
    PTRATIO:  -0.95282684633969105814
          B:   0.01359438499293804704
      LSTAT:  -0.53257535640300879276

y-intercept: 32.62557602342805296303
```

Test the model

In [14]:

```
coefficient of determination = 0.7
     correlation coefficient = 0.8
```

Visualize the expected vs. predicted prices

In [15]:

```
```