ShriDattaMadhira_Assignment_6

October 24, 2021

[53]:
```
from sklearn.datasets import load_boston

boston = load_boston()
print(boston.feature_names)

X, y = load_boston(return_X_y=True)
```

```
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
```

[4]:
```
X[:5]
```

[4]:
```
array([[6.3200e-03, 1.8000e+01, 2.3100e+00, 0.0000e+00, 5.3800e-01,
        6.5750e+00, 6.5200e+01, 4.0900e+00, 1.0000e+00, 2.9600e+02,
        1.5300e+01, 3.9690e+02, 4.9800e+00],
       [2.7310e-02, 0.0000e+00, 7.0700e+00, 0.0000e+00, 4.6900e-01,
        6.4210e+00, 7.8900e+01, 4.9671e+00, 2.0000e+00, 2.4200e+02,
        1.7800e+01, 3.9690e+02, 9.1400e+00],
       [2.7290e-02, 0.0000e+00, 7.0700e+00, 0.0000e+00, 4.6900e-01,
        7.1850e+00, 6.1100e+01, 4.9671e+00, 2.0000e+00, 2.4200e+02,
        1.7800e+01, 3.9283e+02, 4.0300e+00],
       [3.2370e-02, 0.0000e+00, 2.1800e+00, 0.0000e+00, 4.5800e-01,
        6.9980e+00, 4.5800e+01, 6.0622e+00, 3.0000e+00, 2.2200e+02,
        1.8700e+01, 3.9463e+02, 2.9400e+00],
       [6.9050e-02, 0.0000e+00, 2.1800e+00, 0.0000e+00, 4.5800e-01,
        7.1470e+00, 5.4200e+01, 6.0622e+00, 3.0000e+00, 2.2200e+02,
        1.8700e+01, 3.9690e+02, 5.3300e+00]])
```

[5]:
```
y[:5]
```

[5]:
```
array([24. , 21.6, 34.7, 33.4, 36.2])
```

[71]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

```
linearReg = LinearRegression()
```

0.1 STEP1 - Splitting data into training set (80%) and testing set (20%).

[43]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.80,␣
↪test_size=0.20)
```

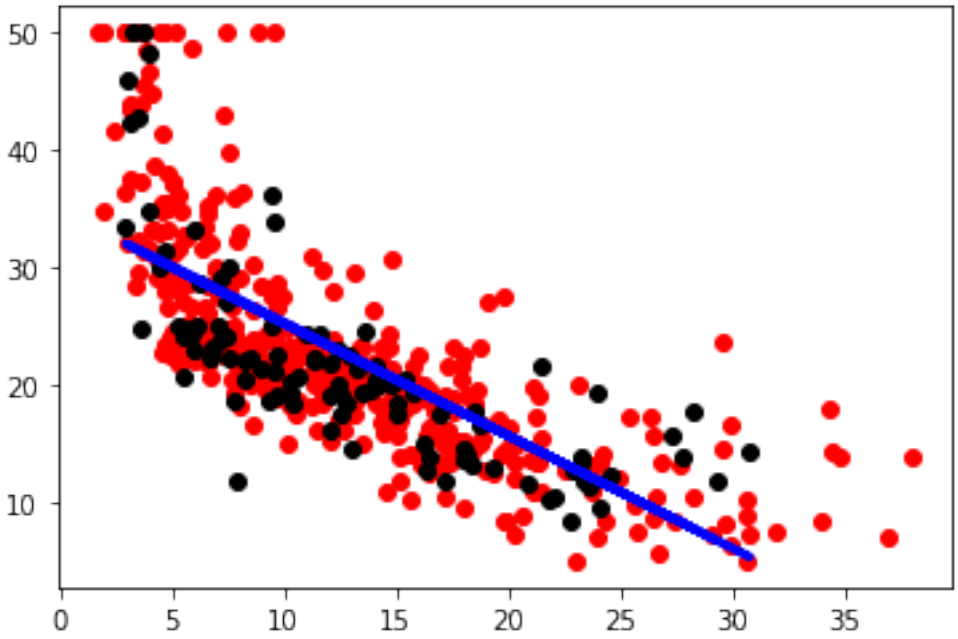
0.2 STEP2(a) - Using all 13 features to fit the linear regression model for target feature 14.

[44]:
```
regr = lReg.fit(X_train, y_train)
y_pred = lReg.predict(X_test)
```

0.3 STEP2(b) - Report the coefficients, mean squared error and variance score for the model on the test set.

[130]:
```
print("---------------------- ALL FEATURES ----------------------")
# Explained variance score :  1 is perfect prediction
r_sq = lReg.score(X_test, y_test)
print('Variance score:', r_sq)

# The coefficients:
print('Coefficients:', lReg.coef_)

# The mean squared error
print("Mean Squared Error:", mean_squared_error(y_pred, y_test))

plt.scatter(X_train, y_train, color='red')
plt.scatter(X_test, y_test,  color='black')
plt.plot(X_test, y_pred, color='blue', linewidth=3)
plt.show()
```

```
---------------------- ALL FEATURES ----------------------
Variance score: 0.5188064744787702
Coefficients: [-0.95841767]
Mean Squared Error: 35.38735898619001
```



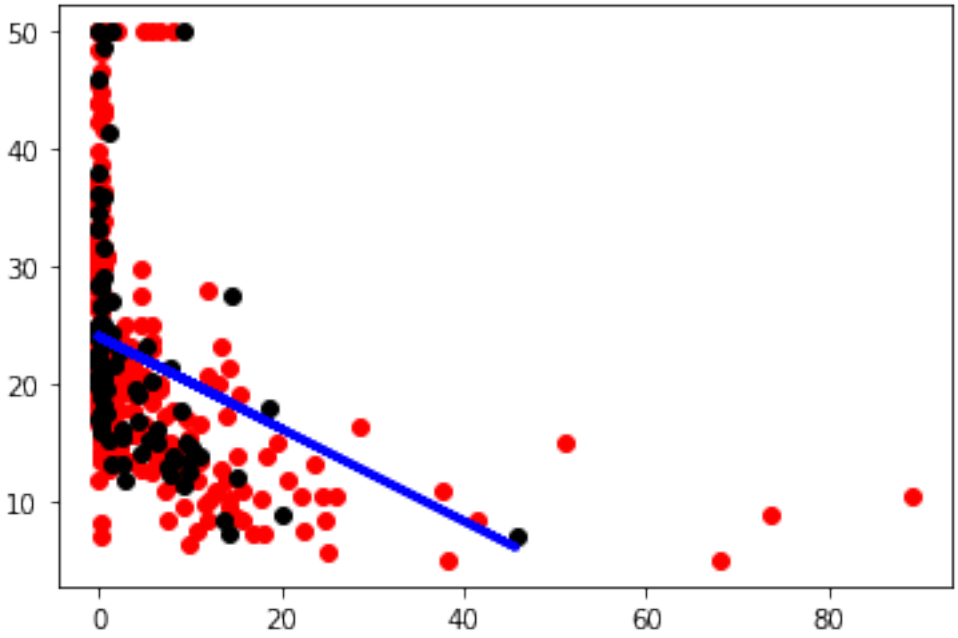
0.4 STEP3(a) - Use each feature alone - to fit a linear regression model on the training set.

0.5 STEP3(b) - Reporting metrics and plotting graphs.

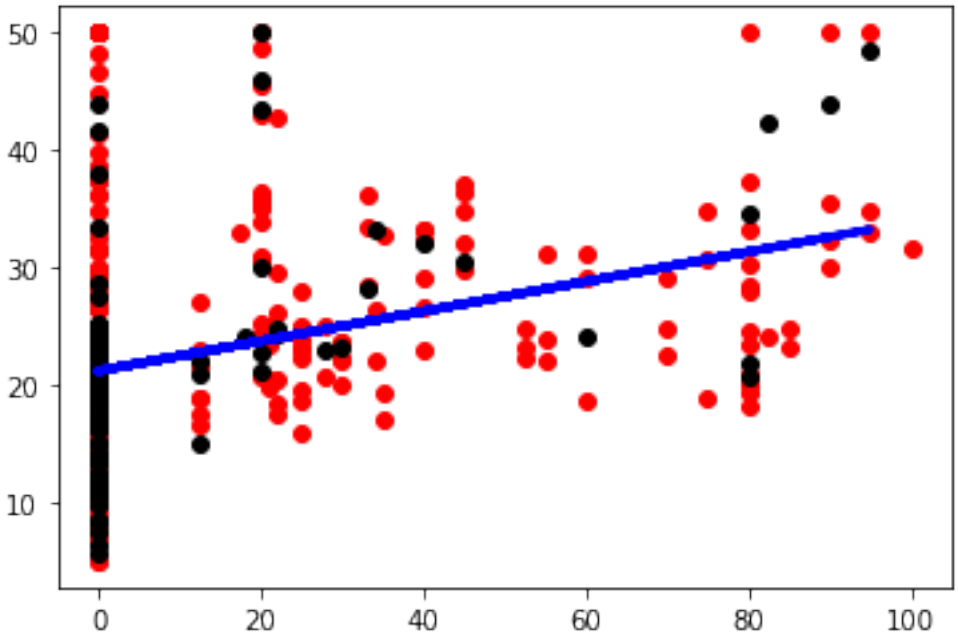
[129]:
```
features = ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',␣
 ↪'TAX', 'PTRATIO', 'B', 'LSTAT']
for (i, feature) in enumerate(features):
    tempX = X[:,i].reshape(-1, 1)

    X_train, X_test, y_train, y_test = train_test_split(tempX, y, train_size=0.
 ↪80, test_size=0.20)
    regr = lReg.fit(X_train, y_train)
    y_pred = lReg.predict(X_test)

    print("----------------------", feature, "----------------------")
    print("Variance Score for feature: ", lReg.score(X_test, y_test))
    print("Mean Squared Error for feature: ", mean_squared_error(y_pred,␣
 ↪y_test))
    print("Coefficients for feature: ", lReg.coef_)

    # Plot outputs
    plt.scatter(X_train, y_train, color='red')
    plt.scatter(X_test, y_test,  color='black')
    plt.plot(X_test, y_pred, color='blue', linewidth=3)
```

```
    plt.show()

    print("\n")
```

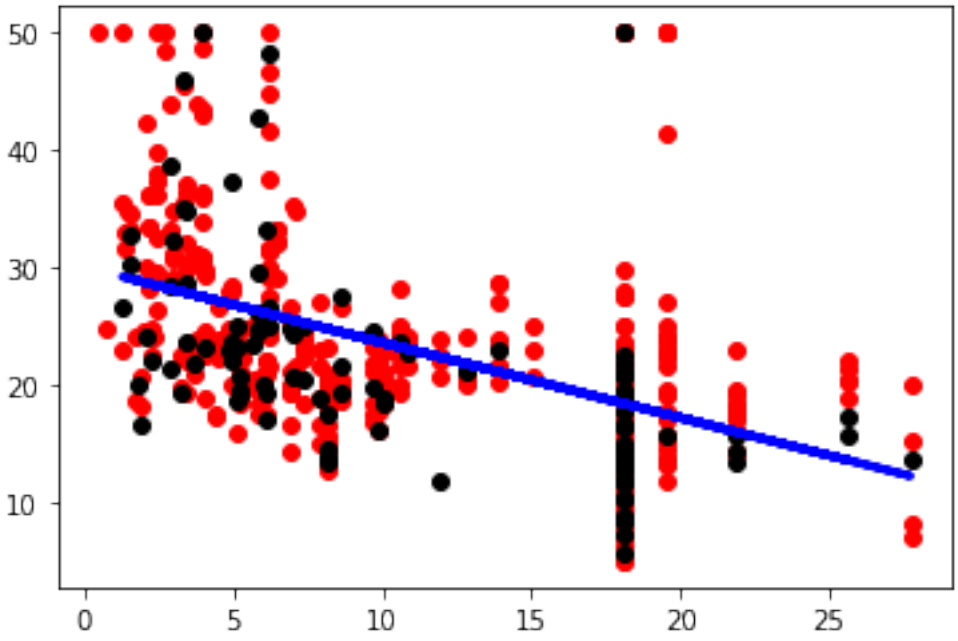
```
---------------------- CRIM ----------------------
Variance Score for feature:  0.15306403547850067
Mean Squared Error for feature:  67.2427365143848
Coefficients for feature:  [-0.39323747]
```



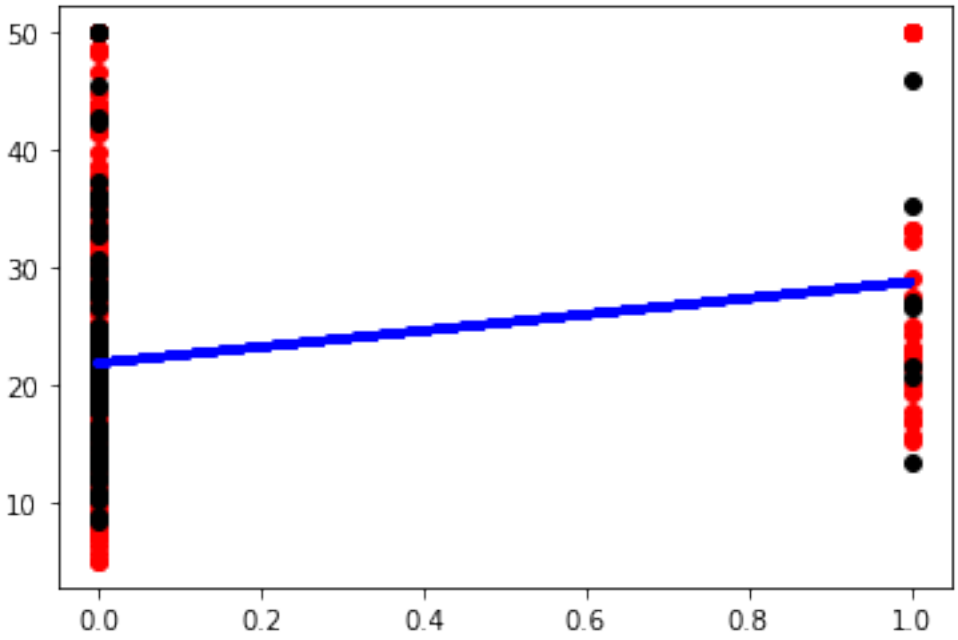

```
---------------------- ZN ----------------------
Variance Score for feature:  0.21465376040947437
Mean Squared Error for feature:  65.34703389395708
Coefficients for feature:  [0.12659577]
```



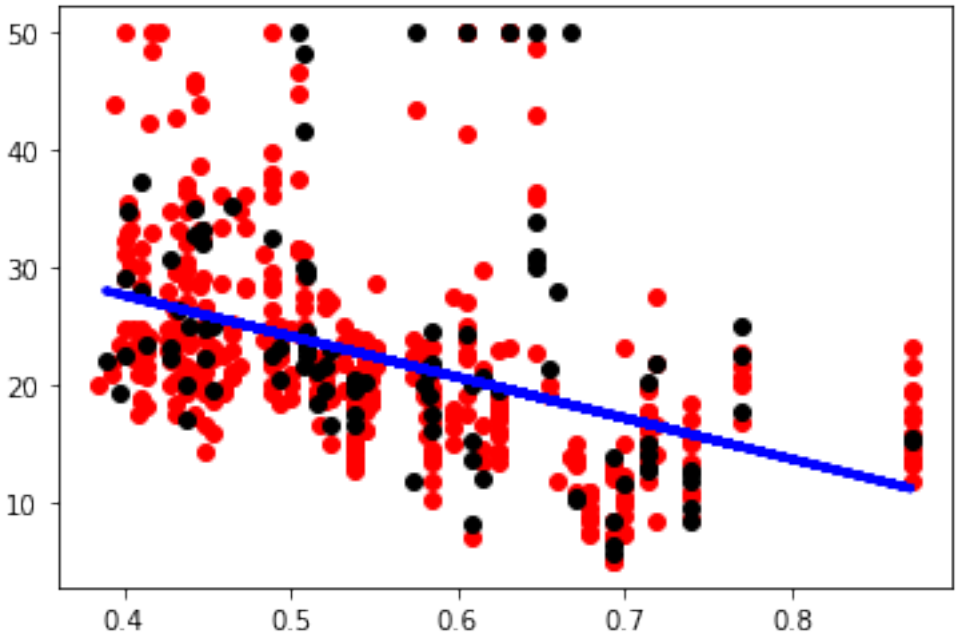

```
---------------------- INDUS ----------------------
Variance Score for feature:  0.25899055606250176
Mean Squared Error for feature:  56.54903459855737
Coefficients for feature:  [-0.64089221]
```



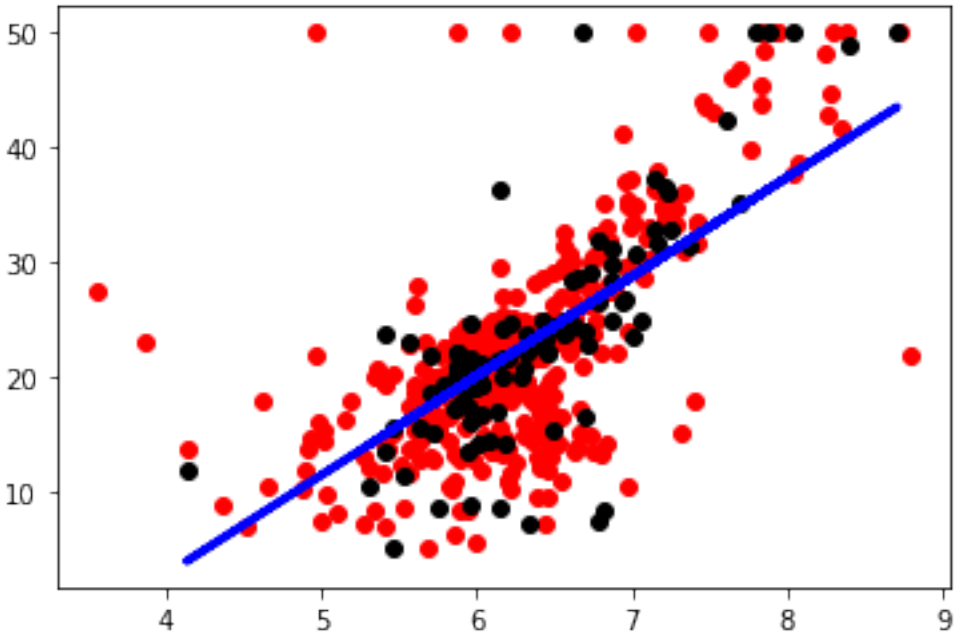
```
---------------------- CHAS ----------------------
Variance Score for feature:  -0.007926576430969723
Mean Squared Error for feature:  79.43937467001003
Coefficients for feature:  [6.90604103]
```



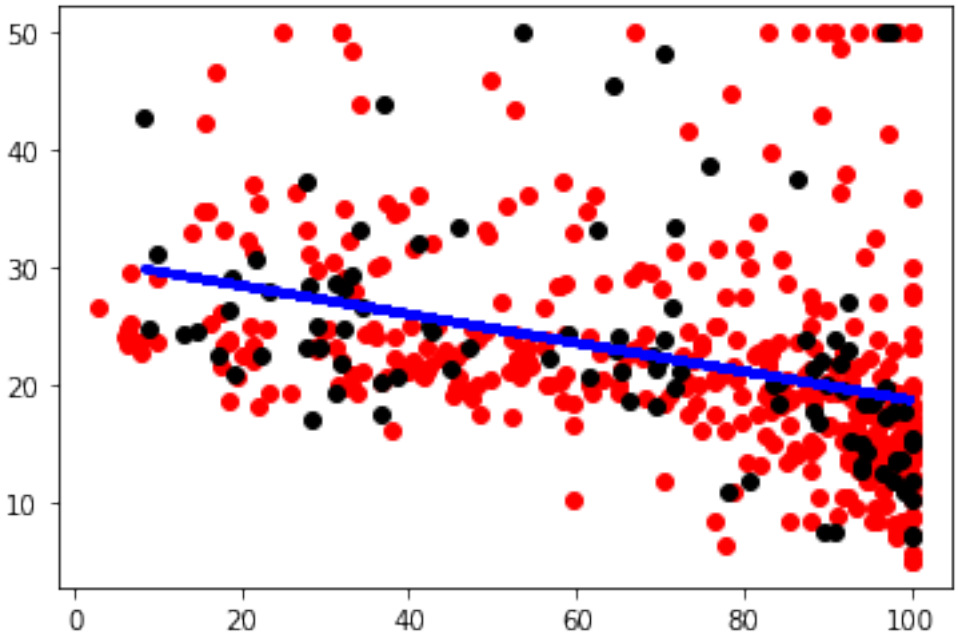

```
---------------------- NOX ----------------------
Variance Score for feature:  0.10074745442959154
Mean Squared Error for feature:  93.17561861387429
Coefficients for feature:  [-34.9265619]
```



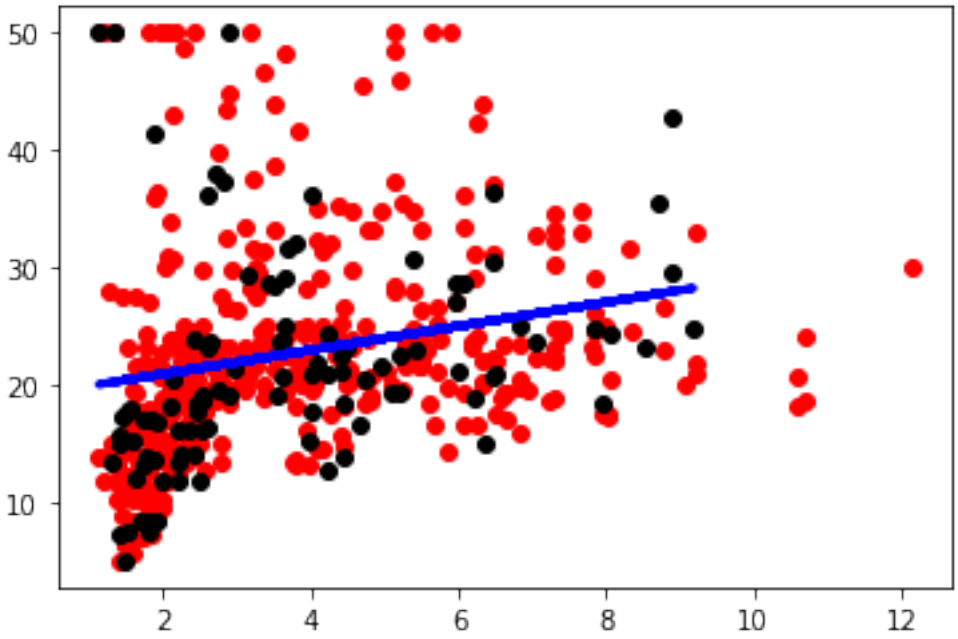

```
---------------------- RM ----------------------
Variance Score for feature:  0.5807983285514245
Mean Squared Error for feature:  39.23478476829386
Coefficients for feature:  [8.66592677]
```



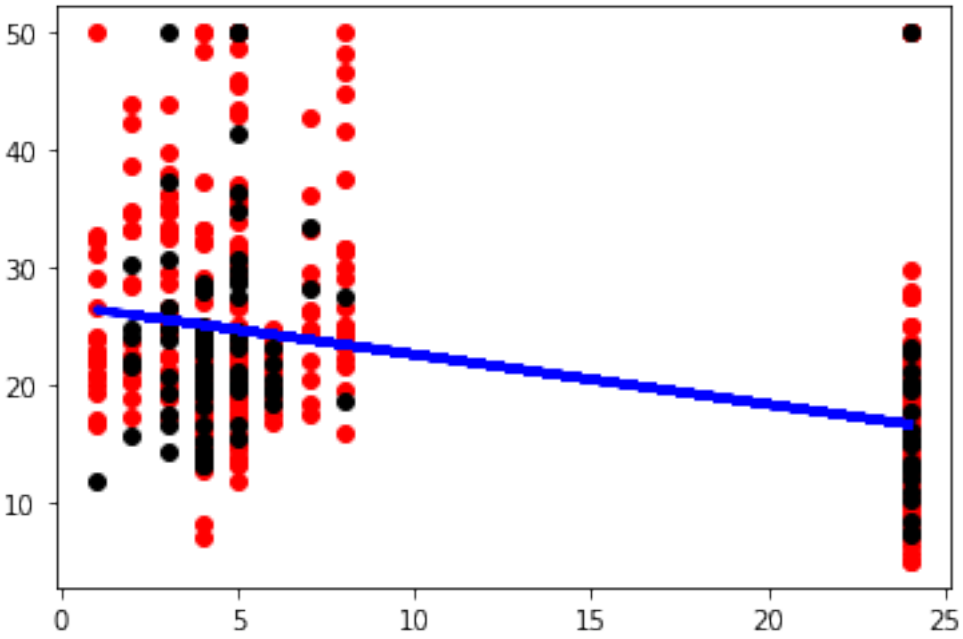
```
---------------------- AGE ----------------------
Variance Score for feature:  0.16486841944341923
Mean Squared Error for feature:  72.83704993028348
Coefficients for feature:  [-0.1215668]
```



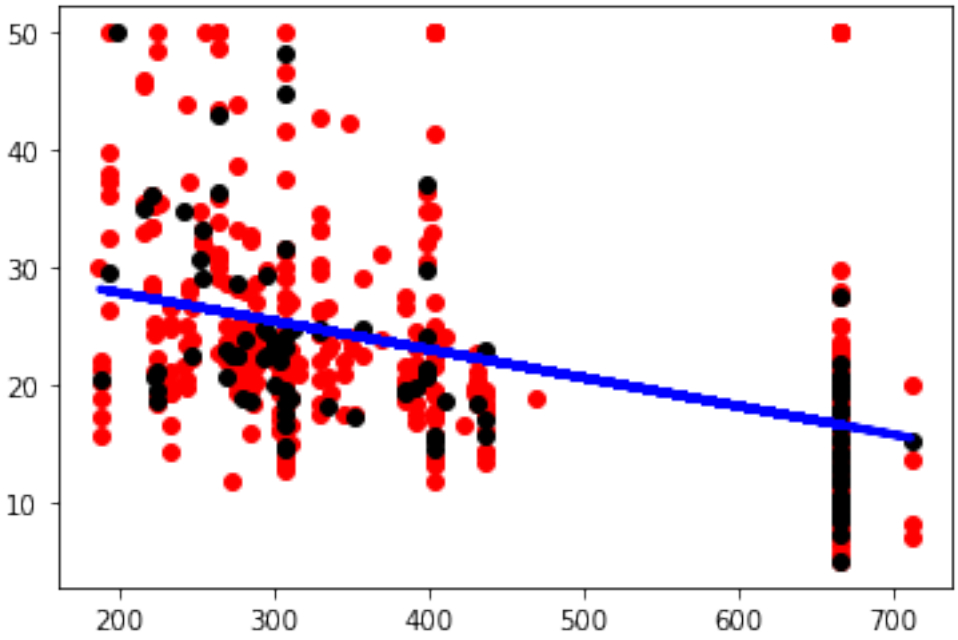

```
---------------------- DIS ----------------------
Variance Score for feature:  0.07940085758205617
Mean Squared Error for feature:  76.080592505996
Coefficients for feature:  [1.01775179]
```



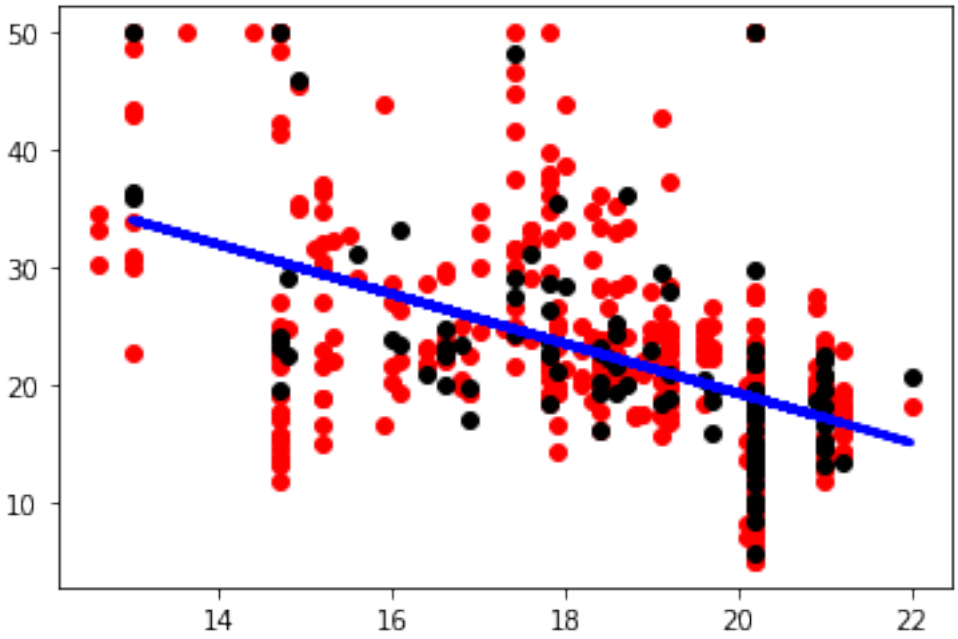

```
---------------------- RAD ----------------------
Variance Score for feature:  0.07254921398439018
Mean Squared Error for feature:  67.32947898741656
Coefficients for feature:  [-0.42367678]
```



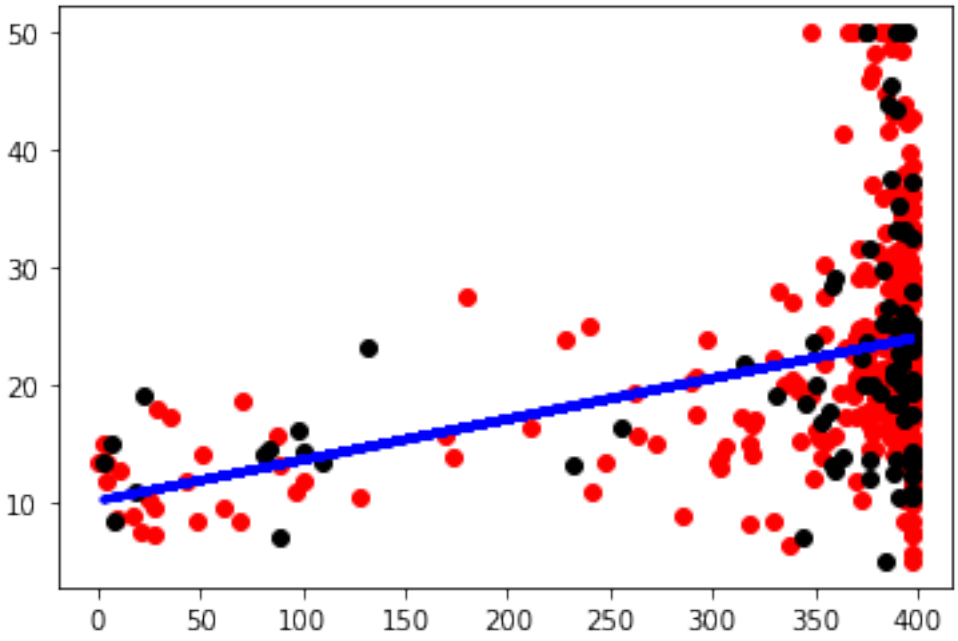
```
---------------------- TAX ----------------------
Variance Score for feature:  0.3385680801497324
Mean Squared Error for feature:  47.2268239652044
Coefficients for feature:  [-0.02419154]
```



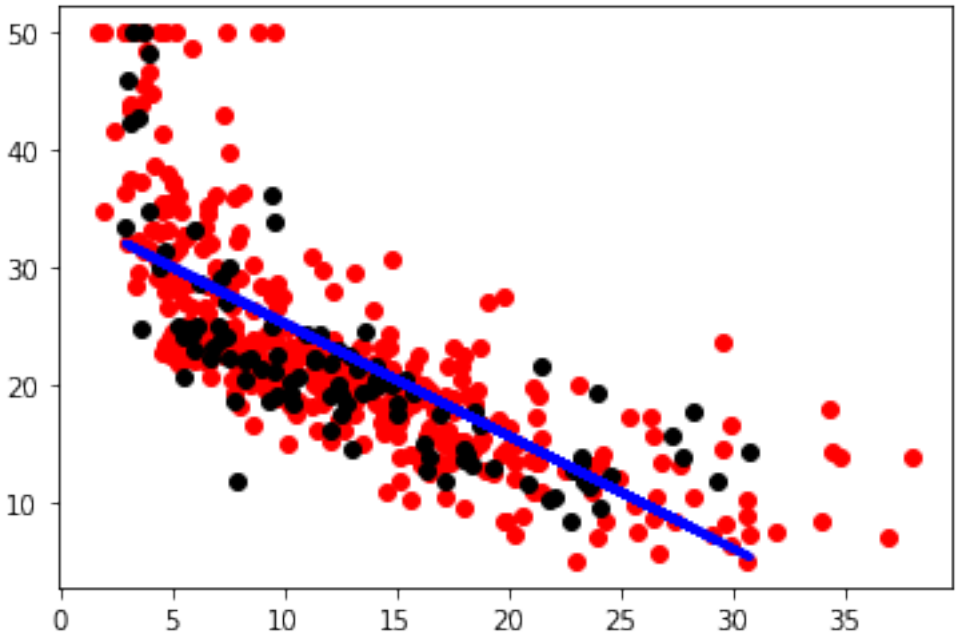

```
---------------------- PTRATIO ----------------------
Variance Score for feature:  0.3401728775192031
Mean Squared Error for feature:  47.665908791191846
Coefficients for feature:  [-2.10591135]
```




```
---------------------- B ----------------------
Variance Score for feature:  0.1051480517155381
Mean Squared Error for feature:  89.37473084289245
Coefficients for feature:  [0.03478472]
```



```
---------------------- LSTAT ----------------------
Variance Score for feature:  0.5188064744787702
Mean Squared Error for feature:  35.38735898619001
Coefficients for feature:  [-0.95841767]
```



1 STEPS - 4(a) and 4(b) - Repeating steps 1, 2(a), 3(a) for 10 iterations

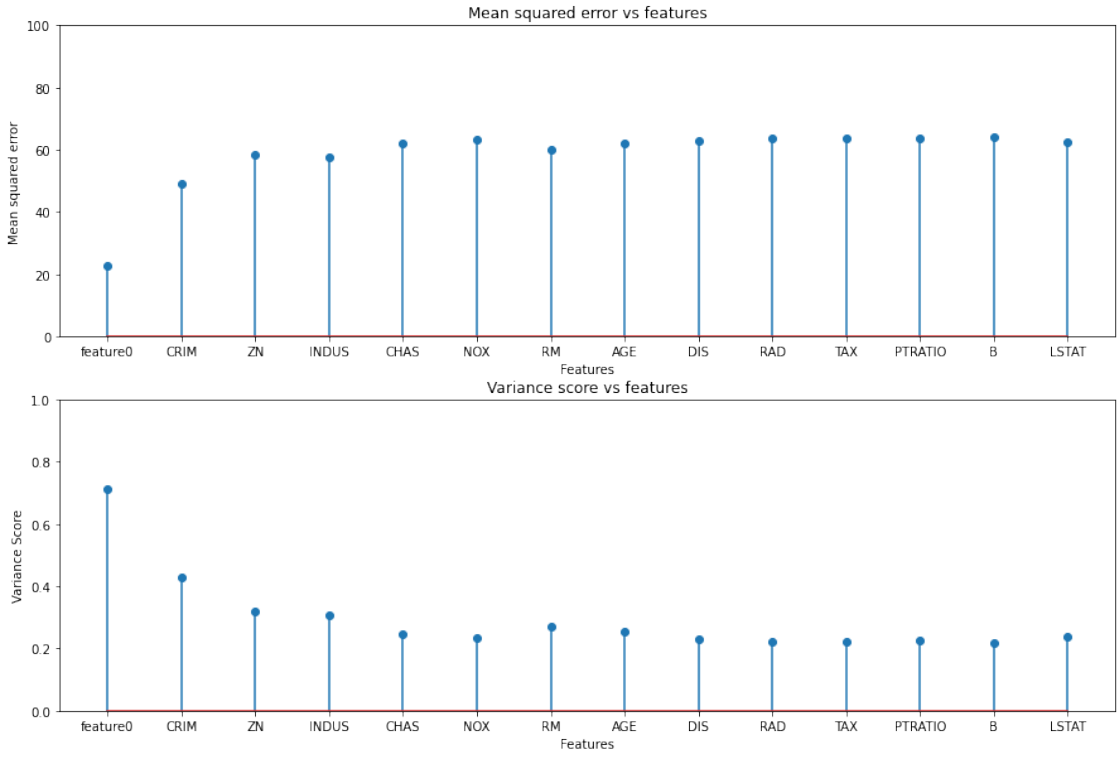
[121]:
```
mse_per_feature, varianceScore_per_feature = [], []
# all features.
var_score, mse = [], []
for i in range(10):
    X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.80,␣
 ↪test_size=0.20)
    regr = lReg.fit(X_train, y_train)
    y_pred = lReg.predict(X_test)

    var_score.append(lReg.score(X_test, y_test))
```

```
    mse.append(mean_squared_error(y_pred, y_test))

print("Variance Score after 10 iterations with all features included: ", np.
 ↪mean(var_score))
print("Mean Squared Error after 10 iterations with all features included: ", np.
 ↪mean(mse))
print("\n")

mse_per_feature.append(np.mean(mse))
varianceScore_per_feature.append(np.mean(var_score))

# Individual features.
features = ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',␣
 ↪'TAX', 'PTRATIO', 'B', 'LSTAT']
for (i, feature) in enumerate(features):
    tempX = X[:,i].reshape(-1, 1)
    for j in range(10):
        X_train, X_test, y_train, y_test = train_test_split(tempX, y,␣
 ↪train_size=0.80, test_size=0.20)
        regr = lReg.fit(X_train, y_train)
        y_pred = lReg.predict(X_test)

        var_score.append(lReg.score(X_test, y_test))
        mse.append(mean_squared_error(y_pred, y_test))

    print("Variance Score for feature ", feature, " after 10 iterations is: ",␣
 ↪np.mean(var_score))
    print("Mean Squared Error for feature ", feature, " after 10 iterations is:␣
 ↪", np.mean(mse))
    print("\n")

    mse_per_feature.append(np.mean(mse))
    varianceScore_per_feature.append(np.mean(var_score))

features = ['feature0', *features]

fig, ax = plt.subplots(2, 1, figsize=(15, 10))

ax[0].stem(features, mse_per_feature)
ax[0].set_ylim(0, 100)
ax[0].set_ylabel('Mean squared error')
ax[0].set_xlabel('Features')
ax[0].set_title('Mean squared error vs features')

ax[1].stem(features, varianceScore_per_feature)
ax[1].set_ylim(0, 1)
```

```
ax[1].set_ylabel('Variance Score')
ax[1].set_xlabel('Features')
ax[1].set_title('Variance score vs features')

plt.show()
```

```
Variance Score after 10 iterations with all features included:
0.7125824665066522
Mean Squared Error after 10 iterations with all features included:
22.84532808578974
Variance Score for feature  CRIM  after 10 iterations is:  0.4263698394965714
Mean Squared Error for feature  CRIM  after 10 iterations is:  49.23346178399178


Variance Score for feature  ZN  after 10 iterations is:  0.31738807904702465
Mean Squared Error for feature  ZN  after 10 iterations is:  58.4187208704778


Variance Score for feature  INDUS  after 10 iterations is:  0.3051998364409819
Mean Squared Error for feature  INDUS  after 10 iterations is:
57.77056030012852


Variance Score for feature  CHAS  after 10 iterations is:  0.2461175712166381
Mean Squared Error for feature  CHAS  after 10 iterations is:  62.05835444663242


Variance Score for feature  NOX  after 10 iterations is:  0.23371063832234812
Mean Squared Error for feature  NOX  after 10 iterations is:  63.20908488557493


Variance Score for feature  RM  after 10 iterations is:  0.269796600517371
Mean Squared Error for feature  RM  after 10 iterations is:  60.013834975488486


Variance Score for feature  AGE  after 10 iterations is:  0.2522916358065167
Mean Squared Error for feature  AGE  after 10 iterations is:  62.11528276238569


Variance Score for feature  DIS  after 10 iterations is:  0.23148607201883106
Mean Squared Error for feature  DIS  after 10 iterations is:  62.76296444123454


Variance Score for feature  RAD  after 10 iterations is:  0.2233143298027116
Mean Squared Error for feature  RAD  after 10 iterations is:  63.67427358098711


Variance Score for feature  TAX  after 10 iterations is:  0.22279798091143432
```

```
Mean Squared Error for feature  TAX  after 10 iterations is:  63.59987978544117


Variance Score for feature  PTRATIO  after 10 iterations is:
0.22491008700006537
Mean Squared Error for feature  PTRATIO  after 10 iterations is:
63.5820896949342


Variance Score for feature  B  after 10 iterations is:  0.2172703344026751
Mean Squared Error for feature  B  after 10 iterations is:  64.10617786861725


Variance Score for feature  LSTAT  after 10 iterations is:  0.23980827409125344
Mean Squared Error for feature  LSTAT  after 10 iterations is:
62.560968911529145
```



1.0.1 Q1) Based upon the linear models you generated, which feature appears to be most predictive for the target feature? Note that you can answer this question based upon the output provided for the linear models.

Based upon the linear models generated, features LSTAT and RM are the features that appear to be most predictive for the target. The variance score of all these features is decent and they are two features with by far the least squared error scores. Apart from that TAX and PTRATIO also seem to be good, but not good enough to be considered as serious contendors to be included in the bracket of LSTAT and RM.

1.0.2 Q2) Suppose you need to select two features for a linear regression model to predict the target feature. Which two features would you select? Why?

If I need to select two features for a linear regression model to predict the target feature, I would choose RM and LSTAT because of their high variance score and less mean squared error.

1.0.3 Q3) Examine all the plots and numbers you have, do you have any comments on them? Do you find any surprising trends? Do you have any idea about what might be causing this surprising trend in the data? This is a descriptive question meant to encourage you to interpret your results and express yourself.

Looking at the plots, there is a difference in variance scores for individual feature single runs and individual feature 10 iterations. During the single run, the variance score is the highest for LSTAT and RM, while the variance score is highest for CRIM after 10 iterations. This is the same for mean squared error as well which is interesting.

The other thing is, after 10 iterations the variance score of all features increased from 58%(all feature single run) to 72%. The reason for such a low variance score is because, in my opinion, of the extremely skewed data some of the features contain. If we look at the data for features like B, RAD, ZN, and CHAS; the plots suggest that the data is so skewed.

Ths trend I have noticed is that, over a period of runs, the variance score is improving but not enough. As mentioned above this is because of the extremely skewed data.