In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
```

In [2]:
```
import pandas as pd
import numpy as np

data_url = "http://lib.stat.cmu.edu/datasets/boston"
raw_df = pd.read_csv(data_url, sep="\s+", skiprows=22, header=None)
data = np.hstack([raw_df.values[::2, :], raw_df.values[1::2, :2]])
target = raw_df.values[1::2, 2]
```

In [3]:
```
X = data
print(X.shape)
y = target
print(y.shape)
```

```
(506, 13)
(506,)
```

In [4]:
```
#Train test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

In [5]:
```
#initializing the regressor model
regressor = linear_model.LinearRegression()
```

In [6]:
```
#fitting the train data into the model
regressor.fit(X_train, y_train)
```

Out[6]:
```
LinearRegression()
```

In [7]:
```
#predicting the test values
y_pred = regressor.predict(X_test)
```

In [8]:
```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(X_test ,y_test))
```

```
Coefficients: [-1.42593189e-01  4.12283216e-02  5.66084481e-02  2.30234404e+00
 -1.82698129e+01  3.57430939e+00  1.40746493e-02 -1.31113142e+00
  3.31531677e-01 -1.29975512e-02 -8.93177289e-01  9.10943388e-03
 -6.07655616e-01]
Mean squared error: 27.15
Variance score: 0.73
```

In [ ]:

```python

```

In [ ]:

```python
#Feature 1
```

In [9]:

```python
#taking only column 0 for feature 1
X[:,0].shape
```

Out[9]:

```
(506,)
```

In [10]:

```python
X1 = X[:,0]
X1 = X1.reshape(-1,1)
```

In [11]:

```python
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [12]:

```python
regressor.fit(x_train, y_train)
```

Out[12]:

```
LinearRegression()
```

In [13]:

```python
y_pred = regressor.predict(x_test)
```

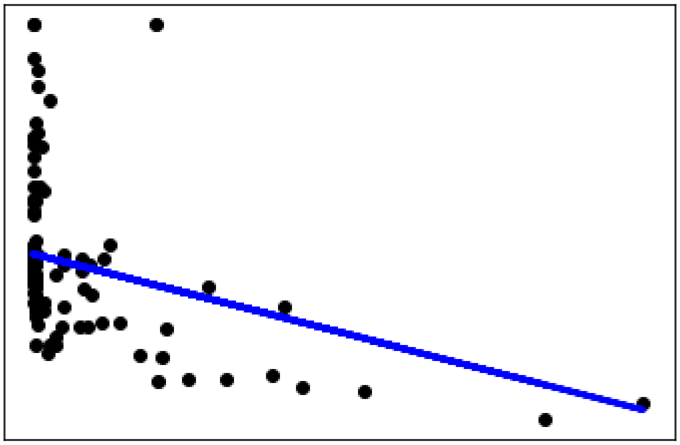
In [14]:

```python
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.3866244]
Mean squared error: 72.08
Variance score: 0.18
```

In [15]:

```python
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 2
```

In [16]:

```
X1 = X[:,1]
X1 = X1.reshape(-1,1)
```

In [17]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [18]:

```
regressor.fit(x_train, y_train)
```

Out[18]:

```
LinearRegression()
```

In [19]:

```
y_pred = regressor.predict(x_test)
```

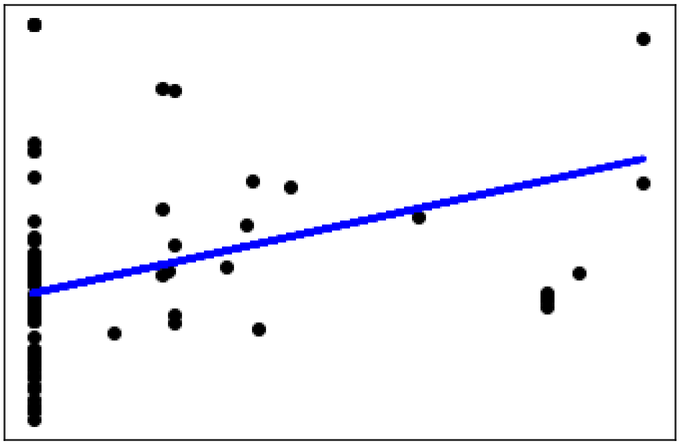
In [20]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [0.15289428]
Mean squared error: 85.21
Variance score: 0.04
```

In [21]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 3
```

In [22]:

```
X1 = X[:,2]
X1 = X1.reshape(-1,1)
```

In [23]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [24]:

```
regressor.fit(x_train, y_train)
```

Out[24]:

```
LinearRegression()
```

In [25]:

```
y_pred = regressor.predict(x_test)
```

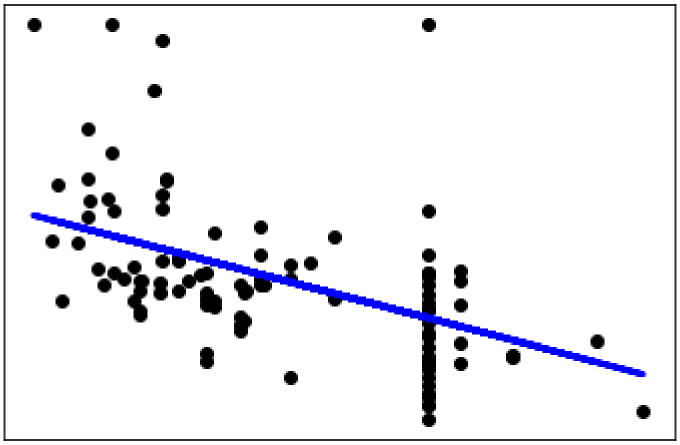
In [26]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.62927206]
Mean squared error: 52.80
Variance score: 0.29
```

In [27]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 4
```

In [28]:

```
X1 = X[:,3]
X1 = X1.reshape(-1,1)
```

In [29]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [30]:

```
regressor.fit(x_train, y_train)
```

Out[30]:

```
LinearRegression()
```

In [31]:

```
y_pred = regressor.predict(x_test)
```

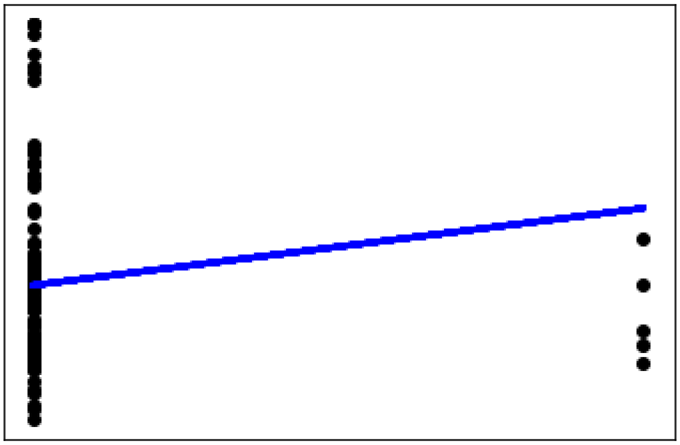
In [32]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [8.33181818]
Mean squared error: 111.70
Variance score: -0.08
```

In [33]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 5
```

In [34]:

```
X1 = X[:,4]
X1 = X1.reshape(-1,1)
```

In [35]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [36]:

```
regressor.fit(x_train, y_train)
```

Out[36]:

```
LinearRegression()
```

In [37]:

```
y_pred = regressor.predict(x_test)
```

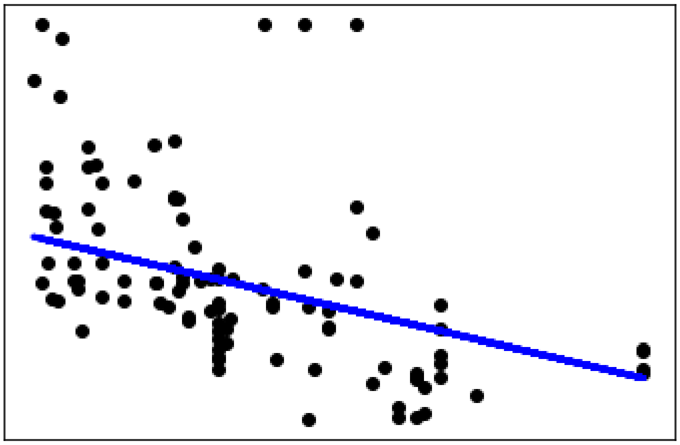
In [38]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-31.28773919]
Mean squared error: 70.78
Variance score: 0.25
```

In [39]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:
```
#Feature 6
```

In [40]:
```
X1 = X[:,5]
X1 = X1.reshape(-1,1)
```

In [41]:
```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [42]:
```
regressor.fit(x_train, y_train)
```

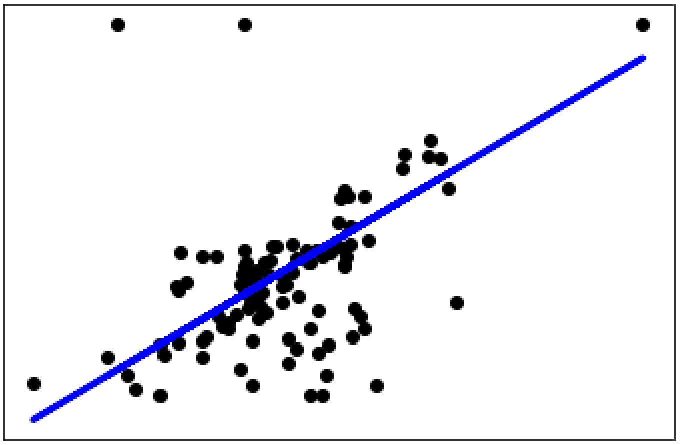
Out[42]:
```
LinearRegression()
```

In [43]:
```
y_pred = regressor.predict(x_test)
```

In [44]:
```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [9.53429989]
Mean squared error: 59.88
Variance score: 0.13
```

In [45]:
```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 7
```

In [46]:

```
X1 = X[:,6]
X1 = X1.reshape(-1,1)
```

In [47]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [48]:

```
regressor.fit(x_train, y_train)
```

Out[48]:

```
LinearRegression()
```

In [49]:

```
y_pred = regressor.predict(x_test)
```

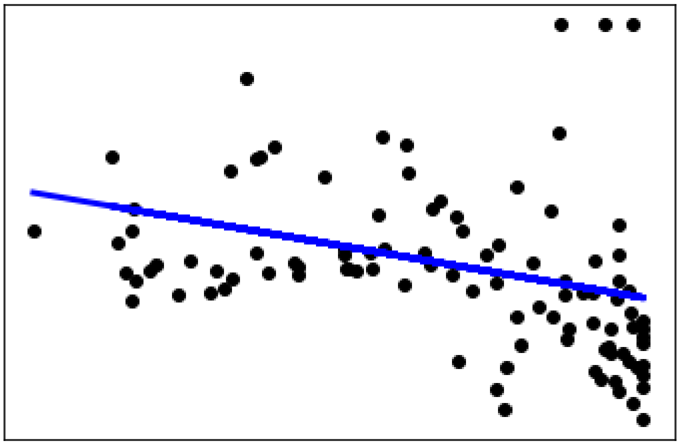
In [50]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.12319795]
Mean squared error: 73.26
Variance score: 0.13
```

In [51]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:

```
#Feature 8
```

In [52]:

```
X1 = X[:,7]
X1 = X1.reshape(-1,1)
```

In [53]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [54]:

```
regressor.fit(x_train, y_train)
```

Out[54]:

```
LinearRegression()
```

In [55]:

```
y_pred = regressor.predict(x_test)
```

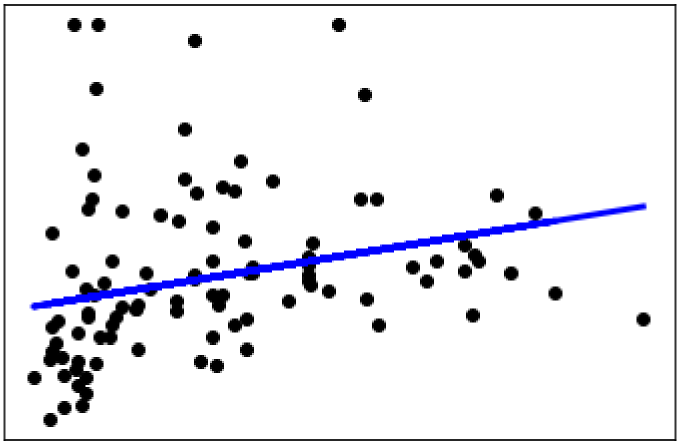
In [56]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [1.1509228]
Mean squared error: 77.33
Variance score: 0.03
```

In [57]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 9
```

In [58]:

```
X1 = X[:,8]
X1 = X1.reshape(-1,1)
```

In [59]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [60]:

```
regressor.fit(x_train, y_train)
```

Out[60]:

```
LinearRegression()
```

In [61]:

```
y_pred = regressor.predict(x_test)
```

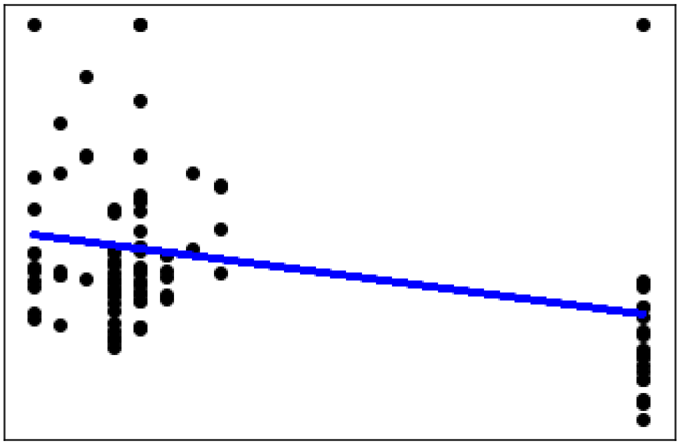
In [62]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.3928012]
Mean squared error: 70.59
Variance score: 0.16
```

In [63]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:
```
#Feature 10
```

In [64]:
```
X1 = X[:,9]
X1 = X1.reshape(-1,1)
```

In [65]:
```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [66]:
```
regressor.fit(x_train, y_train)
```

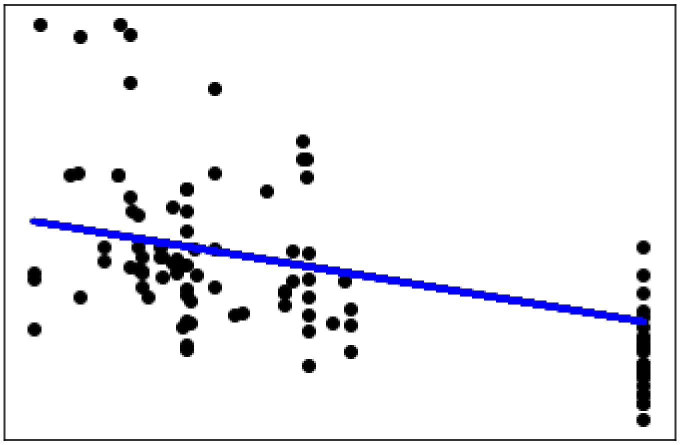
Out[66]:
```
LinearRegression()
```

In [67]:
```
y_pred = regressor.predict(x_test)
```

In [68]:
```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.02357828]
Mean squared error: 59.65
Variance score: 0.31
```

In [69]:
```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:
```
#Feature 11
```

In [70]:
```
X1 = X[:,10]
X1 = X1.reshape(-1,1)
```

In [71]:
```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [72]:
```
regressor.fit(x_train, y_train)
```

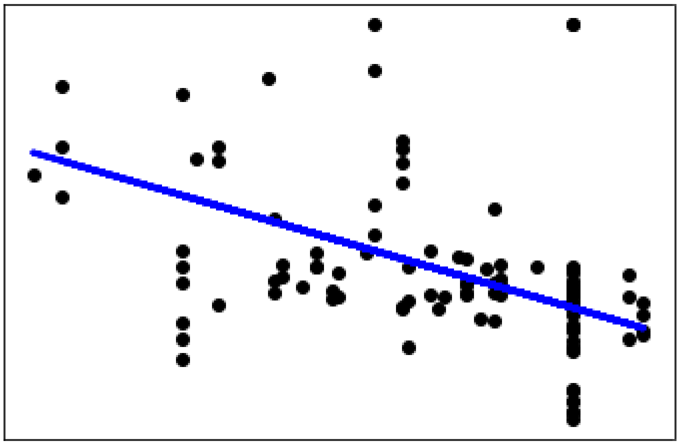
Out[72]:
```
LinearRegression()
```

In [73]:
```
y_pred = regressor.predict(x_test)
```

In [74]:
```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-2.25350668]
Mean squared error: 67.54
Variance score: 0.17
```

In [75]:
```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:

```
#Feature 12
```

In [76]:

```
X1 = X[:,11]
X1 = X1.reshape(-1,1)
```

In [77]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [78]:

```
regressor.fit(x_train, y_train)
```

Out[78]:

```
LinearRegression()
```

In [79]:

```
y_pred = regressor.predict(x_test)
```

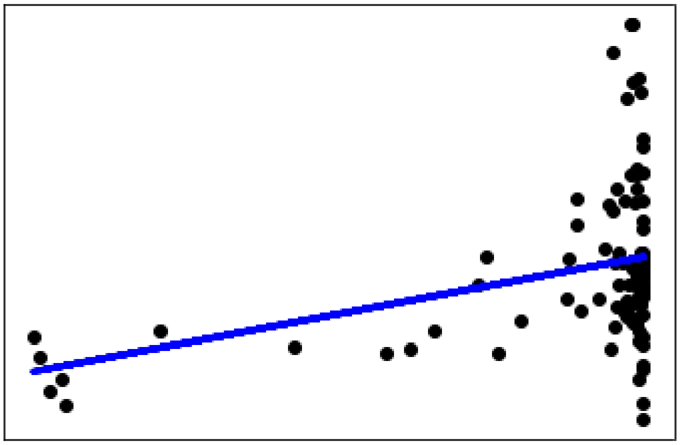
In [80]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [0.03285761]
Mean squared error: 71.50
Variance score: 0.13
```

In [81]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

```

```

In [ ]:

```
#Feature 13
```

In [82]:

```
X1 = X[:,12]
X1 = X1.reshape(-1,1)
```

In [83]:

```
x_train, x_test, y_train, y_test = train_test_split(X1, y, test_size = 0.2)
```

In [84]:

```
regressor.fit(x_train, y_train)
```

Out[84]:

```
LinearRegression()
```

In [85]:

```
y_pred = regressor.predict(x_test)
```

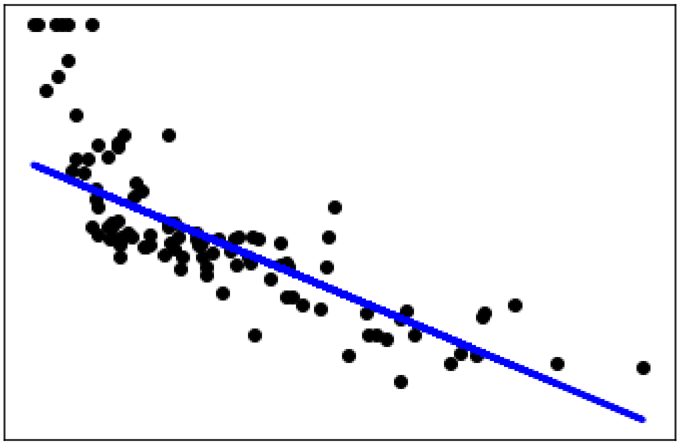
In [86]:

```
# The coefficients
print("Coefficients:", regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(y_test,y_pred))
# Explained variance score : 1 is perfect prediction
print("Variance score: %.2f" % regressor.score(x_test ,y_test))
```

```
Coefficients: [-0.90923013]
Mean squared error: 42.32
Variance score: 0.60
```

In [87]:

```
plt.scatter(x_test, y_test, color="black")
plt.plot(x_test, y_pred, color="blue", linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```



In [ ]:

In [ ]:

```
#Average of values of feature 1
```

In [203…

```
c1 = {}
m1 = {}
v1 = {}
```

In [204…

```
#storing the 10 values of the iterations in a dictionary of lists
regressor = linear_model.LinearRegression()
for i in range(0,10):
  x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
  regressor.fit(x_train, y_train)
  y_pred = regressor.predict(x_test)
  if 0 not in c1:
    c1[0] = []
    m1[0] = []
    v1[0] = []
  c1[0].append(regressor.coef_)
  m1[0].append(mean_squared_error(y_test,y_pred))
  v1[0].append(regressor.score(x_test ,y_test))
  for j in range(0,13):
    regressor.fit(x_train[:,j].reshape(-1,1), y_train)
    y_pred = regressor.predict(x_test[:,j].reshape(-1,1))
    if (j+1) not in c1:
      #print(i,j,c1.keys)
      c1[j+1] = []
      m1[j+1] = []
      v1[j+1] = []
    c1[j+1].append(regressor.coef_)
    m1[j+1].append(mean_squared_error(y_test,y_pred))
    v1[j+1].append(regressor.score(x_test[:,j].reshape(-1,1) ,y_test))
```

In [205…

```
#finding the average values of the 10 iteration values of coeffients. 0th elemen
#the 13 features are represented.
coef_avg = {}
for k in c1.keys():
  if k not in coef_avg:
```

```
    coef_avg[k] = []
  t = np.array(c1[k])
  coef_avg[k] = t.mean(axis=0)
```

In [197…

```
print(coef_avg)
```

```
{0: array([-1.08576949e-01,  4.69650080e-02,  3.36717969e-03,  2.61640185e+00,
       -1.69075154e+01,  3.64192363e+00,  2.19195019e-03, -1.51047968e+00,
        3.17410608e-01, -1.26597496e-02, -9.56959209e-01,  9.59915241e-03,
       -5.40074665e-01]), 1: array([-0.40660438]), 2: array([0.13565969]), 3: ar
ray([-0.65041286]), 4: array([6.59670025]), 5: array([-33.83124193]), 6: array
([9.04885899]), 7: array([-0.11942163]), 8: array([1.06422678]), 9: array([-0.40
907926]), 10: array([-0.02587526]), 11: array([-2.17523208]), 12: array([0.03417
115]), 13: array([-0.94712004])}
```

In [206…

```
#finding the average values of the 10 iteration values of mean squared error. 0t
#the 13 features are represented.
mean_avg = {}
for q in m1.keys():
  if q not in mean_avg:
    mean_avg[q] = []
  t = np.array(m1[q])
  mean_avg[q] = t.mean(axis=0)
```

In [213…

```
print(mean_avg)
```

```
{0: 26.23755383606642, 1: 74.49042238777979, 2: 78.82820745915694, 3: 68.1267626
3104126, 4: 83.17471034647133, 5: 72.3217439741164, 6: 47.830675345899614, 7: 7
5.83184797523081, 8: 82.96056661720618, 9: 75.72844655872007, 10: 69.95779601507
938, 11: 64.93101549138404, 12: 76.37464047198947, 13: 42.05034469135267}
```

In [211…

```
#finding the average values of the 10 iteration values of variance. 0th element
#the 13 features are represented.
var_avg = {}
for o in v1.keys():
  if o not in var_avg:
    var_avg[o] = []
  t = np.array(v1[o])
  var_avg[o] = t.mean(axis=0)
```

In [212…

```
print(var_avg)
```

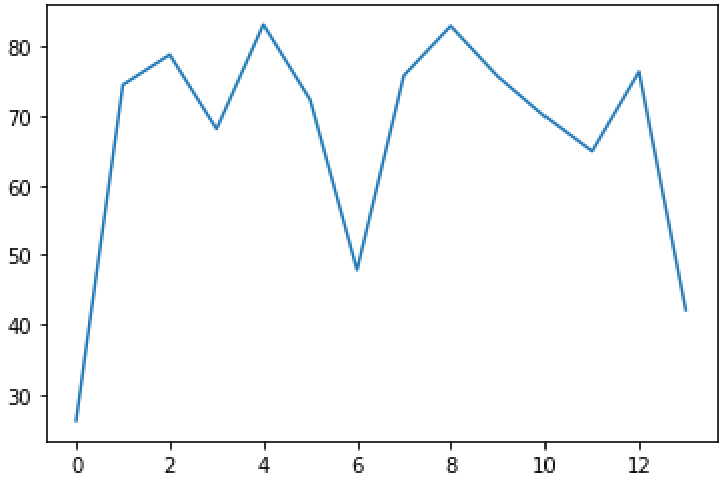
```
{0: 0.6877326542952502, 1: 0.12902090971800376, 2: 0.07256987913971263, 3: 0.199
81352207209957, 4: 0.027113064398530996, 5: 0.15507410199394694, 6: 0.4364179595
19936, 7: 0.11258727631507956, 8: 0.029788530547442182, 9: 0.1156461360014136, 1
0: 0.18077684717737066, 11: 0.2335256533494518, 12: 0.10531881924700212, 13: 0.5
044288353646085}
```

In [ ]:

In [ ]:

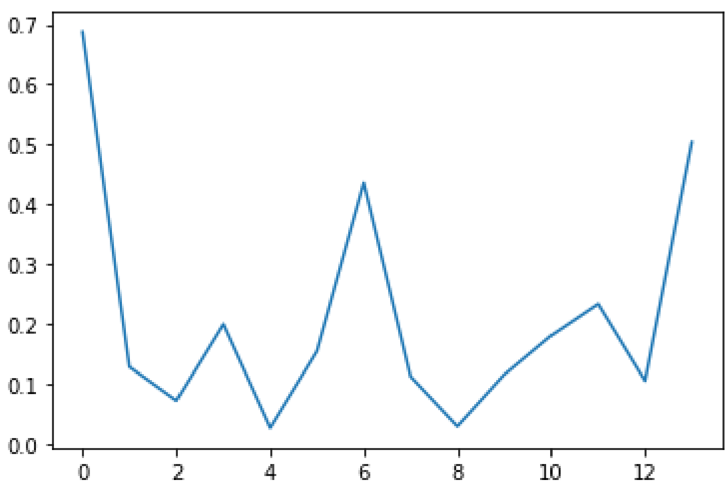
In [214…

```
#plot between mean squared error and features
import matplotlib.pylab as plt
d = mean_avg
lists = sorted(d.items()) # sorted by key, return a list of tuples
x, y = zip(*lists) # unpack a list of pairs into two tuples
plt.plot(x, y)
plt.show()
```




In [215…

```
#plot between variance and features
import matplotlib.pylab as plt
d = var_avg
lists = sorted(d.items())
x, y = zip(*lists)
plt.plot(x, y)
plt.show()
```




In [ ]:

```

```

In [ ]:

```
#1. The last feature appears to be the most predcitive one out of all the featur
#squared error
```

In [ ]:

```
#2. We would choose the 6th and the 13th feature for linear regression model to
#mean squared error and hence this shows that those features fit best.
```

In [ ]:

```
#3. If we just consider feature 4 that has low variance and high error, then we
#kind of underfitting. But if we consider all the features which have low error
#overfitting. Hence for finding the best model we need to do bias variance trend
```