

Practical Question 6-KMeans

Q6. Use Simple Kmeans, DBScan, Hierarchical clustering algorithms for clustering. Compare the performance of clusters by changing the parameters involved in the algorithms.

```
from google.colab import drive
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.cluster import hierarchical
from sklearn.cluster import DBSCAN
from sklearn import preprocessing
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
```

```
iris=pd.read_csv('/content/gdrive/MyDrive/q6_iris.csv')
print(iris)
#print(iris.keys())
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
..
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

[150 rows x 5 columns]

```
sep_l=iris.values[:,0]
print(sep_l)
#print(len(sep_l))
sep_w=iris.values[:,1]
print(sep_w)
#print(len(sep_w))
pet_l=iris.values[:,2]
print(pet_l)
```

```

#petal_w=iris.values[:,2]
#print(len(pet_w))
pet_w=iris.values[:,3]
print(pet_w)
#print(len(pet_w))

```

```

[5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4 5.1
 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5 4.9 5.0
 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0 6.4 6.9 5.5
 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.7 5.6 5.8 6.2 5.6 5.9 6.1
 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4 6.0 6.7 6.3 5.6 5.5
 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.8 7.1 6.3 6.5 7.6 4.9 7.3
 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7 6.0 6.9 5.6 7.7 6.3 6.7 7.2
 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8
 6.7 6.7 6.3 6.5 6.2 5.9]
[3.5 3.0 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 3.7 3.4 3.0 3.0 4.0 4.4 3.9 3.5
 3.8 3.8 3.4 3.7 3.6 3.3 3.4 3.0 3.4 3.5 3.4 3.2 3.1 3.4 4.1 4.2 3.1 3.2
 3.5 3.1 3.0 3.4 3.5 2.3 3.2 3.5 3.8 3.0 3.8 3.2 3.7 3.3 3.2 3.2 3.1 2.3
 2.8 2.8 3.3 2.4 2.9 2.7 2.0 3.0 2.2 2.9 2.9 3.1 3.0 2.7 2.2 2.5 3.2 2.8
 2.5 2.8 2.9 3.0 2.8 3.0 2.9 2.6 2.4 2.4 2.7 2.7 3.0 3.4 3.1 2.3 3.0 2.5
 2.6 3.0 2.6 2.3 2.7 3.0 2.9 2.9 2.5 2.8 3.3 2.7 3.0 2.9 3.0 3.0 2.5 2.9
 2.5 3.6 3.2 2.7 3.0 2.5 2.8 3.2 3.0 3.8 2.6 2.2 3.2 2.8 2.8 2.7 3.3 3.2
 2.8 3.0 2.8 3.0 2.8 3.8 2.8 2.8 2.6 3.0 3.4 3.1 3.0 3.1 3.1 3.1 2.7 3.2
 3.3 3.0 2.5 3.0 3.4 3.0]
[1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 1.5 1.6 1.4 1.1 1.2 1.5 1.3 1.4
 1.7 1.5 1.7 1.5 1.0 1.7 1.9 1.6 1.6 1.5 1.4 1.6 1.6 1.5 1.5 1.4 1.5 1.2
 1.3 1.5 1.3 1.5 1.3 1.3 1.3 1.6 1.9 1.4 1.6 1.4 1.5 1.4 4.7 4.5 4.9 4.0
 4.6 4.5 4.7 3.3 4.6 3.9 3.5 4.2 4.0 4.7 3.6 4.4 4.5 4.1 4.5 3.9 4.8 4.0
 4.9 4.7 4.3 4.4 4.8 5.0 4.5 3.5 3.8 3.7 3.9 5.1 4.5 4.5 4.7 4.4 4.1 4.0
 4.4 4.6 4.0 3.3 4.2 4.2 4.2 4.3 3.0 4.1 6.0 5.1 5.9 5.6 5.8 6.6 4.5 6.3
 5.8 6.1 5.1 5.3 5.5 5.0 5.1 5.3 5.5 6.7 6.9 5.0 5.7 4.9 6.7 4.9 5.7 6.0
 4.8 4.9 5.6 5.8 6.1 6.4 5.6 5.1 5.6 6.1 5.6 5.5 4.8 5.4 5.6 5.1 5.1 5.9
 5.7 5.2 5.0 5.2 5.4 5.1]
[0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 0.2 0.2 0.1 0.1 0.2 0.4 0.4 0.3
 0.3 0.3 0.2 0.4 0.2 0.5 0.2 0.2 0.4 0.2 0.2 0.2 0.2 0.4 0.1 0.2 0.1 0.2
 0.2 0.1 0.2 0.2 0.3 0.3 0.2 0.6 0.4 0.3 0.2 0.2 0.2 0.2 1.4 1.5 1.5 1.3
 1.5 1.3 1.6 1.0 1.3 1.4 1.0 1.5 1.0 1.4 1.3 1.4 1.5 1.0 1.5 1.1 1.8 1.3
 1.5 1.2 1.3 1.4 1.4 1.7 1.5 1.0 1.1 1.0 1.2 1.6 1.5 1.6 1.5 1.3 1.3 1.3
 1.2 1.4 1.2 1.0 1.3 1.2 1.3 1.3 1.1 1.3 2.5 1.9 2.1 1.8 2.2 2.1 1.7 1.8
 1.8 2.5 2.0 1.9 2.1 2.0 2.4 2.3 1.8 2.2 2.3 1.5 2.3 2.0 2.0 1.8 2.1 1.8
 1.8 1.8 2.1 1.6 1.9 2.0 2.2 1.5 1.4 2.3 2.4 1.8 1.8 2.1 2.4 2.3 1.9 2.3
 2.5 2.3 1.9 2.0 2.3 1.8]

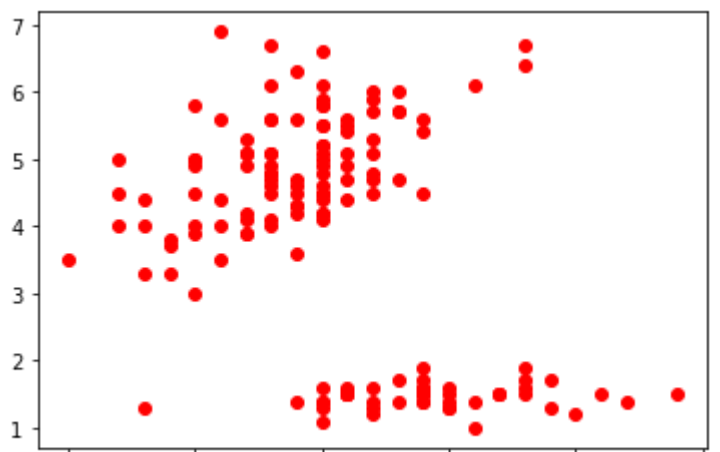
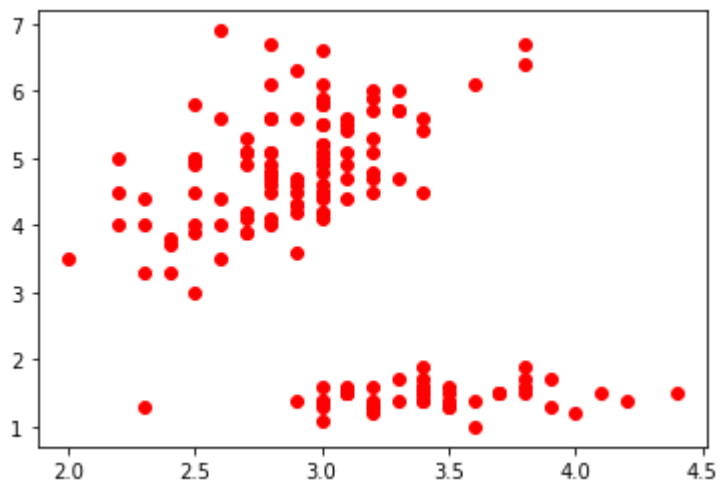
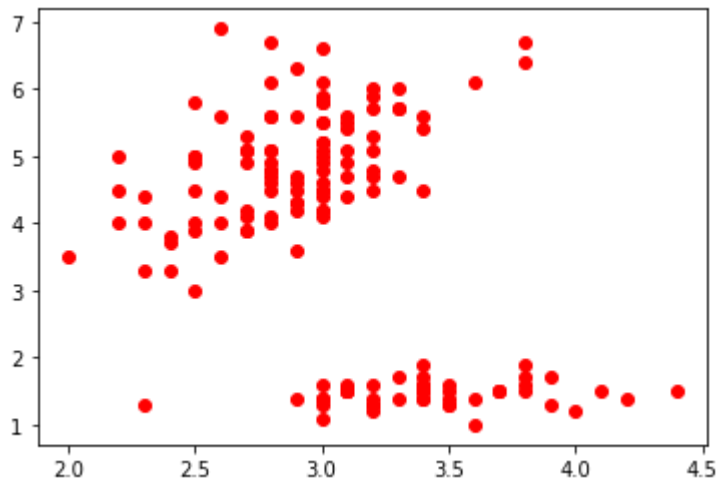
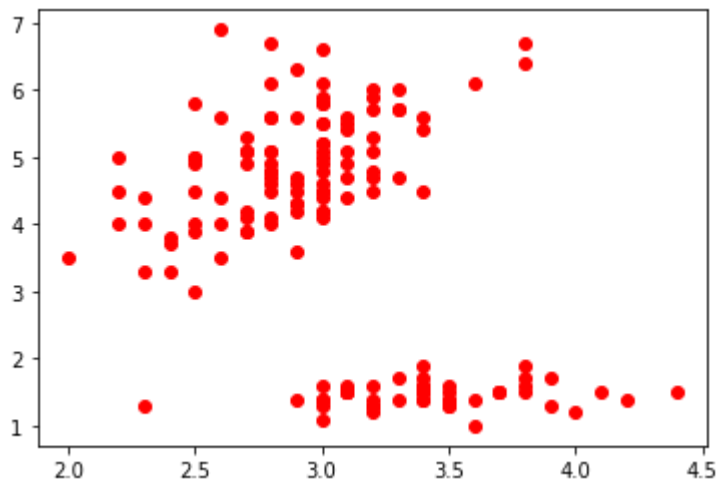
```

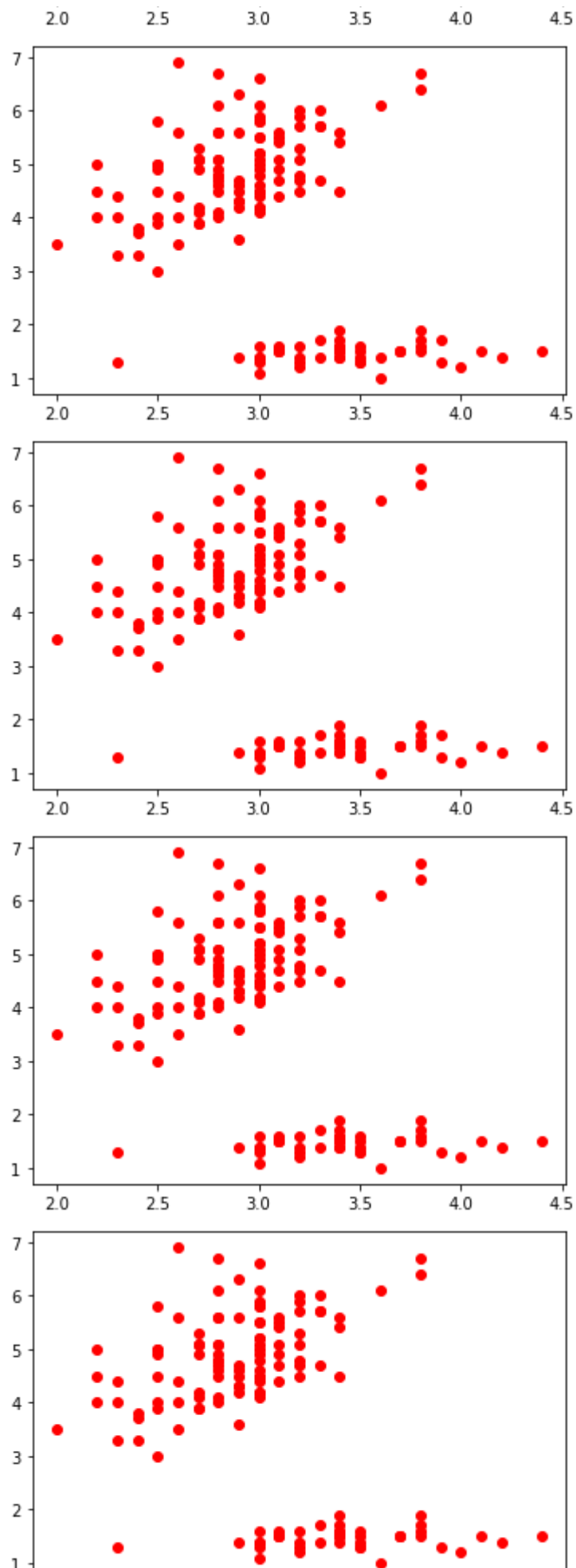
Taking Sepal Width and Petal Length as our two features for clustering

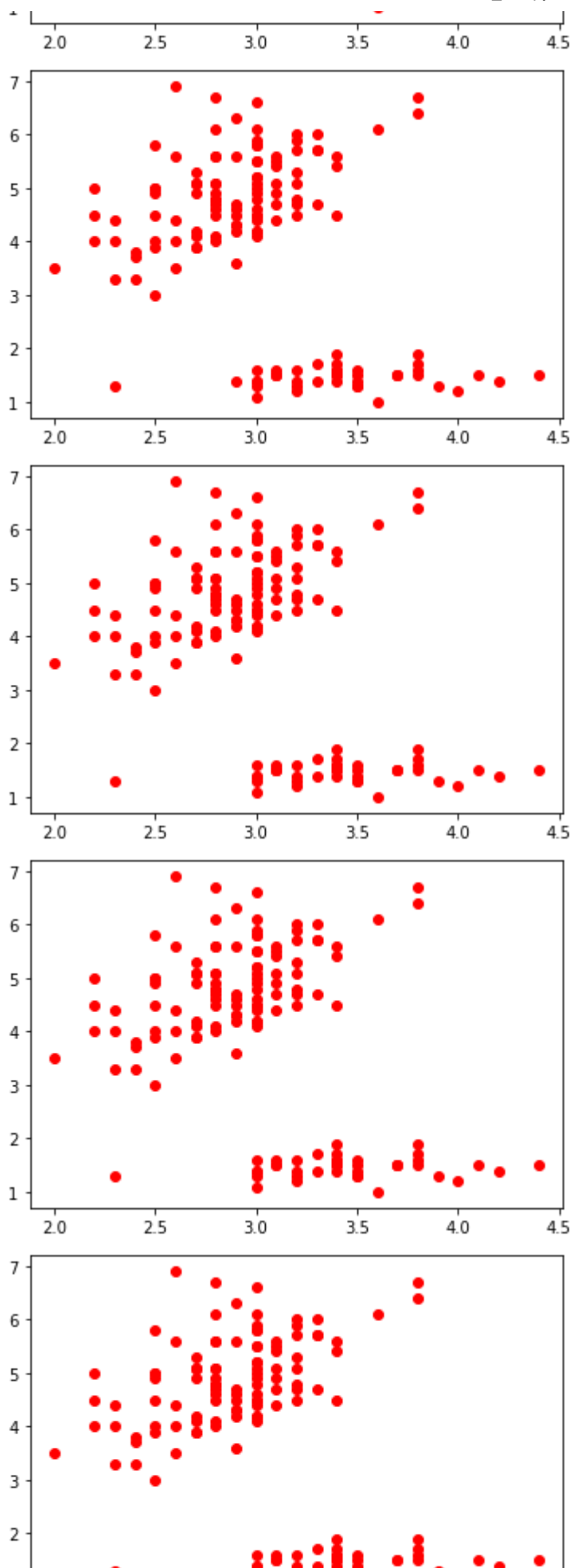
```

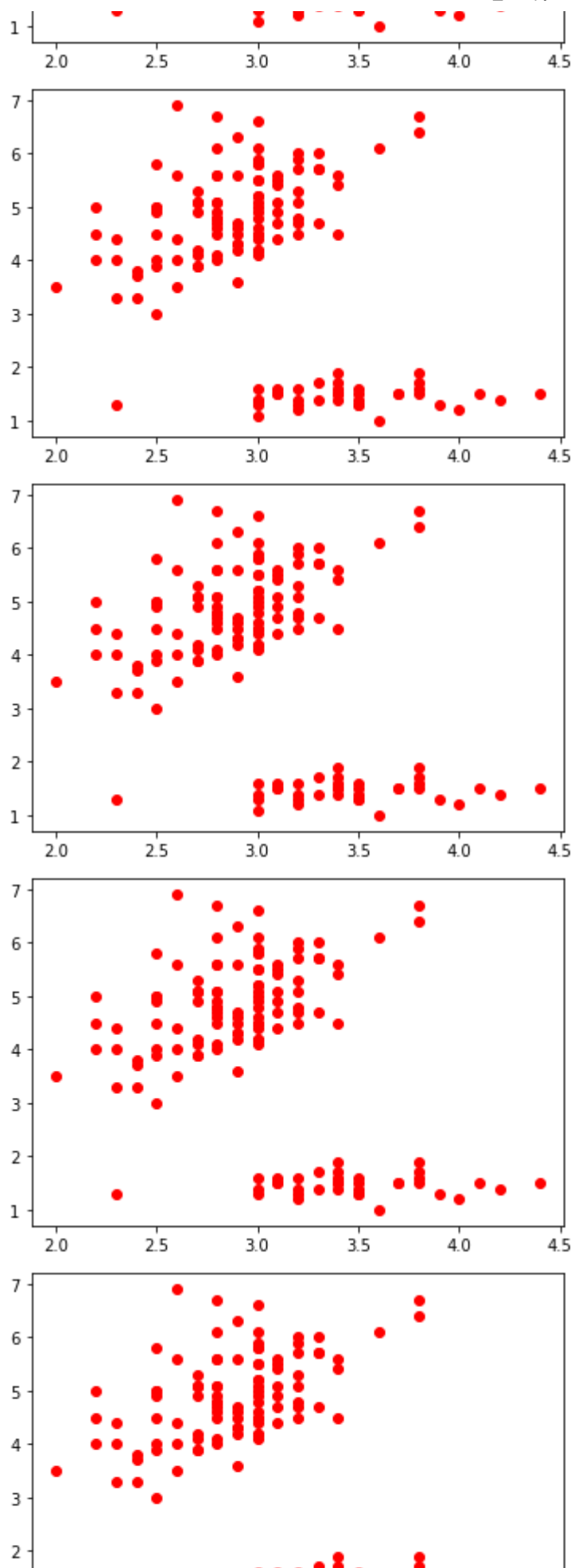
for i in range(150):
    if i<=49:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'ro')
    if i>49 and i<=99:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'bo')
    if i>99:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'go')
plt.show()

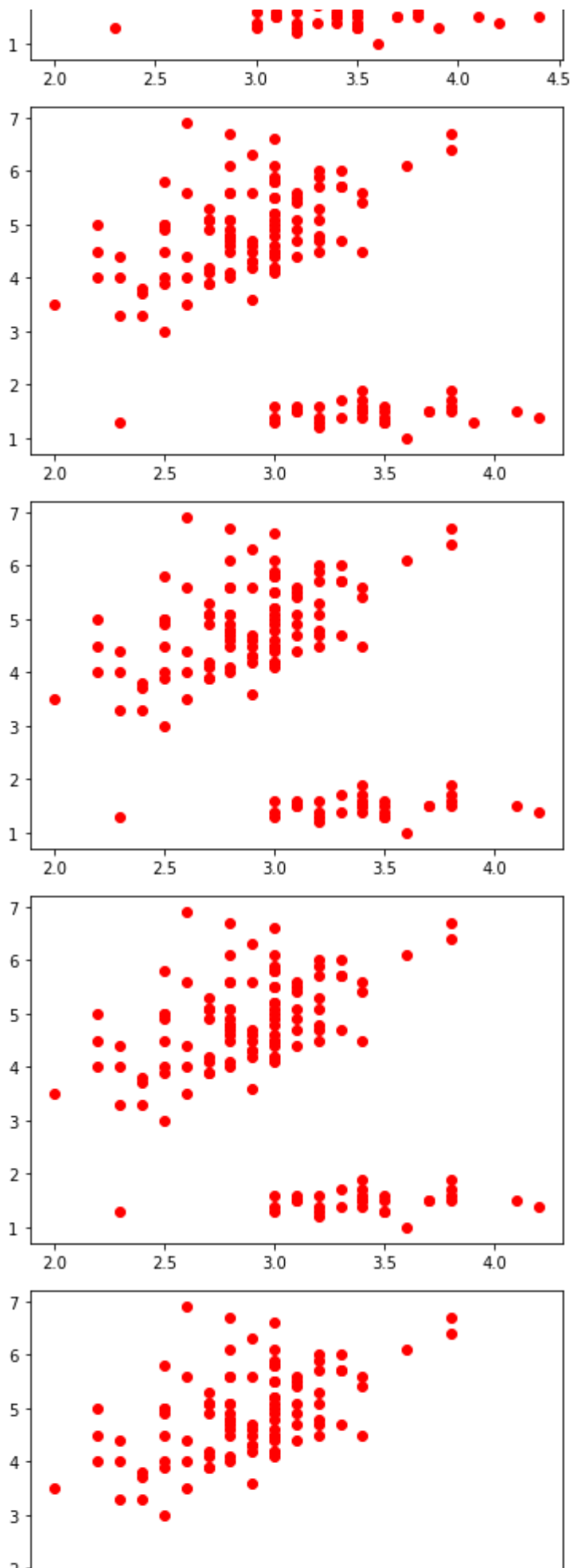
```

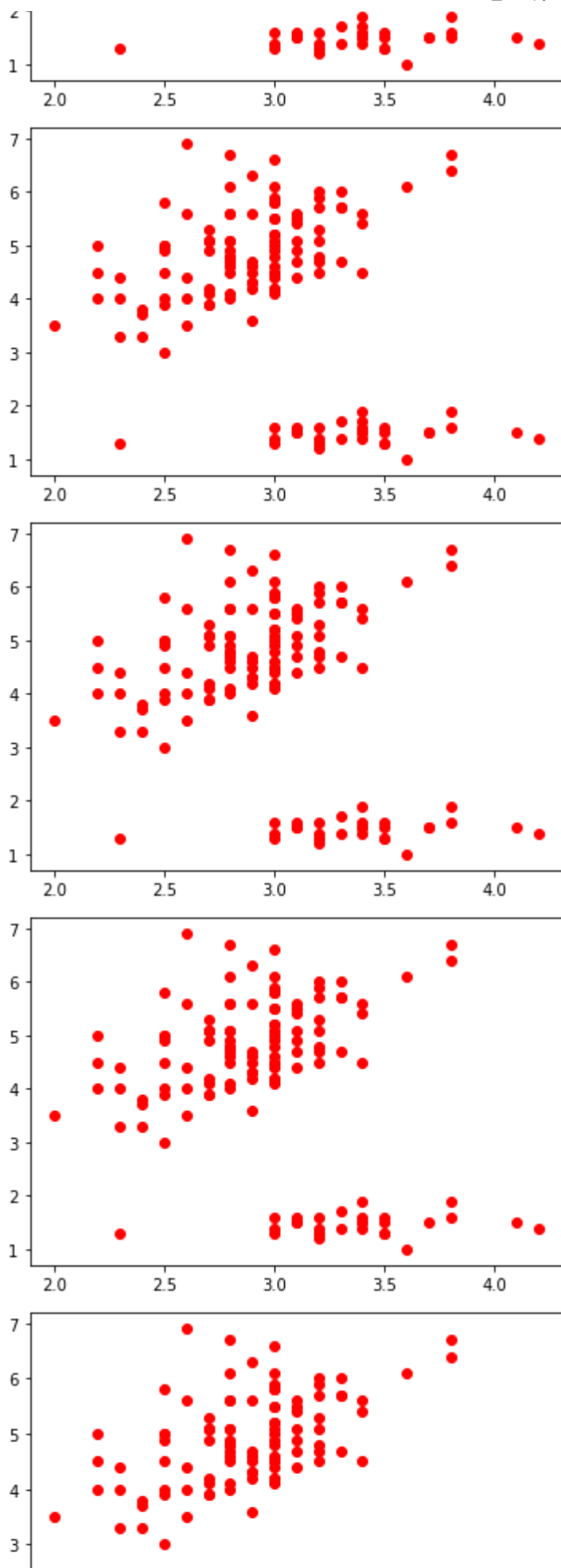



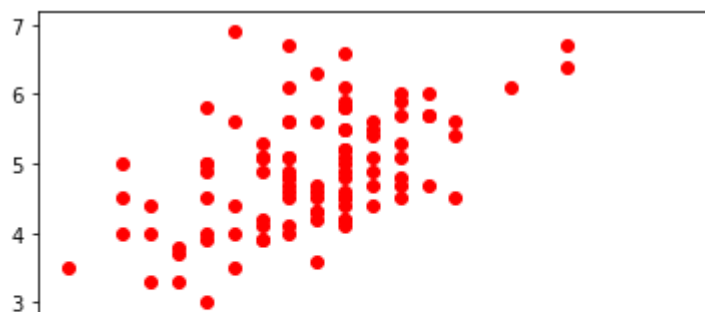
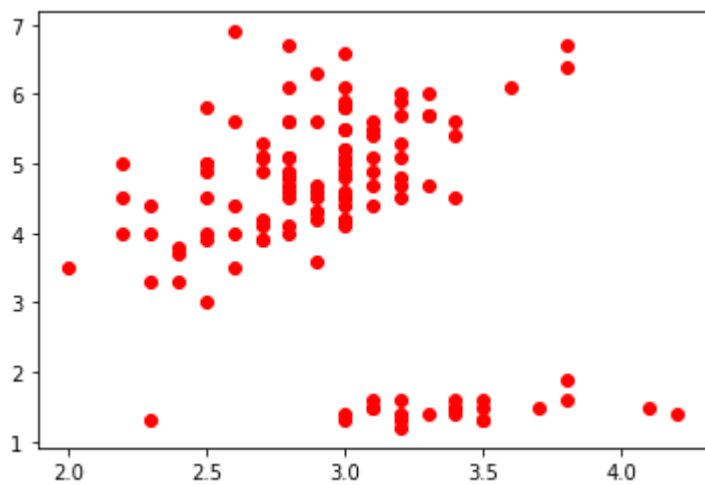
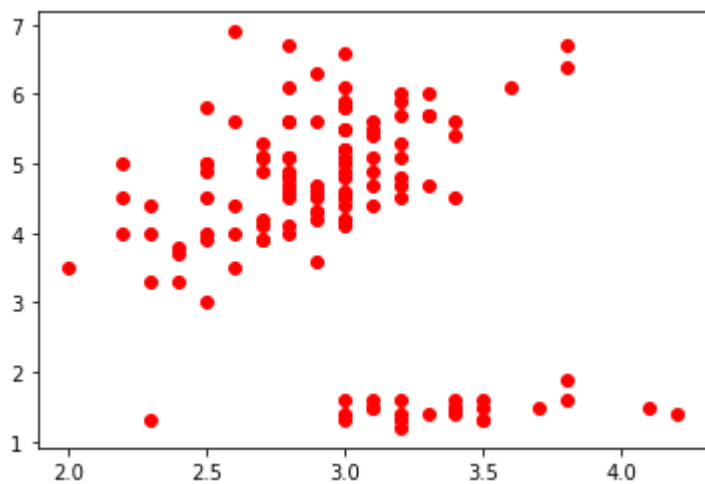
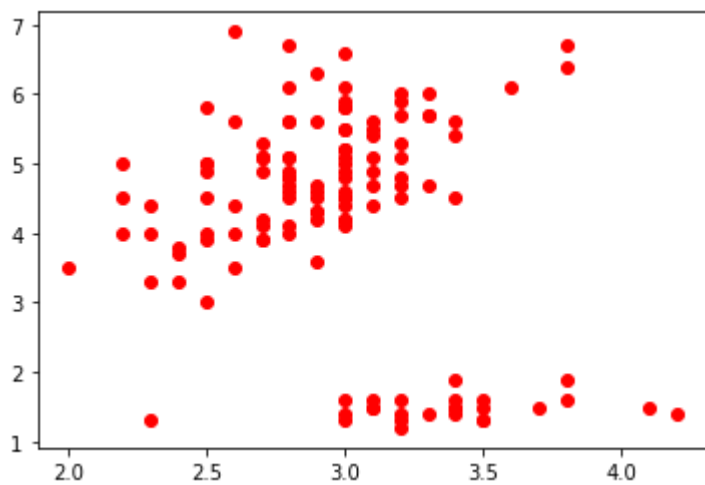
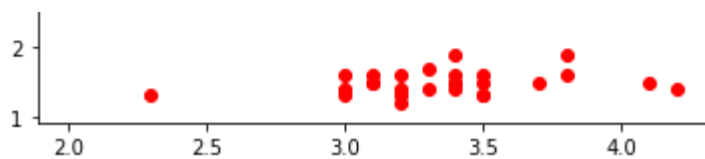


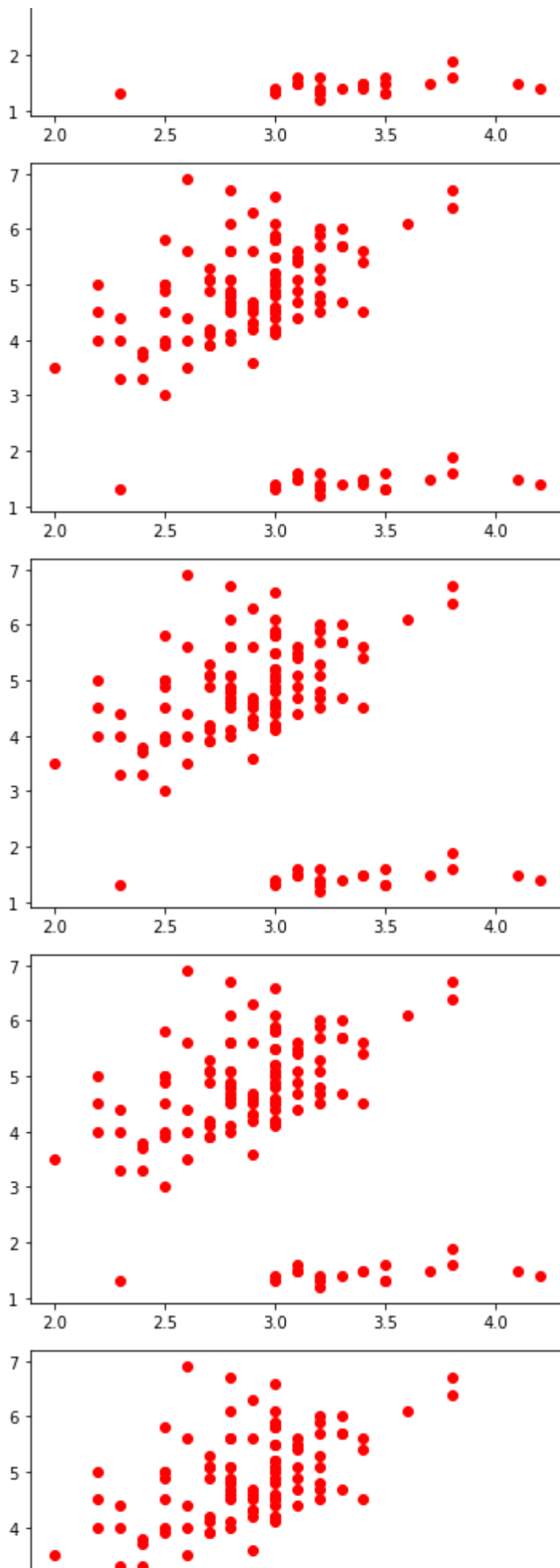


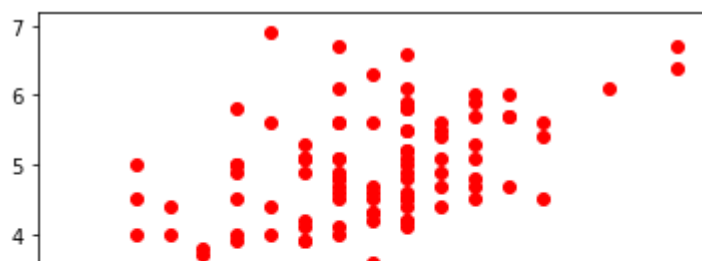
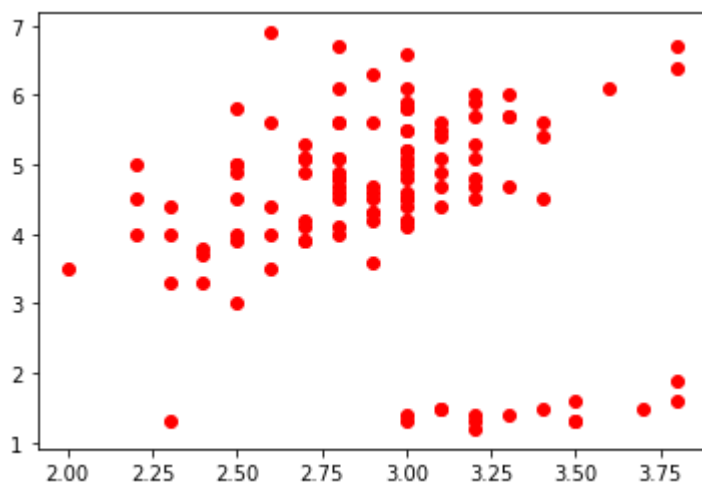
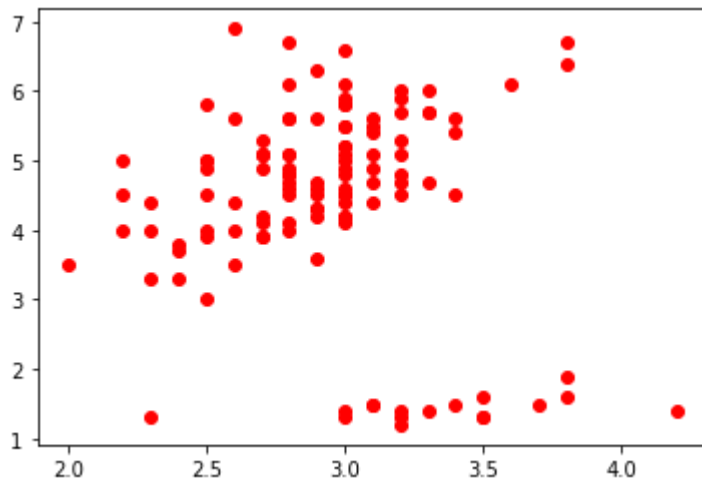
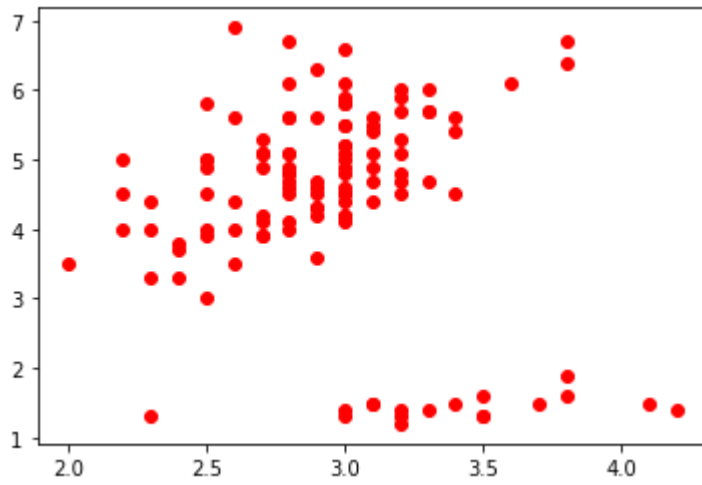
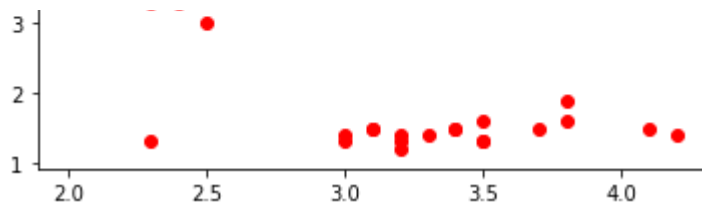


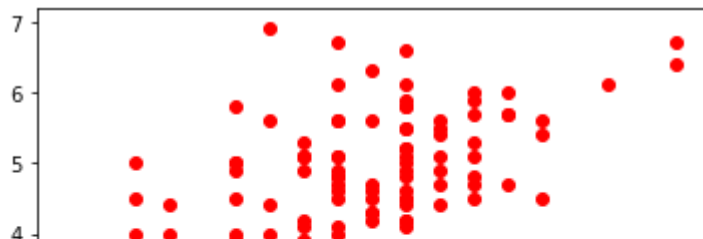
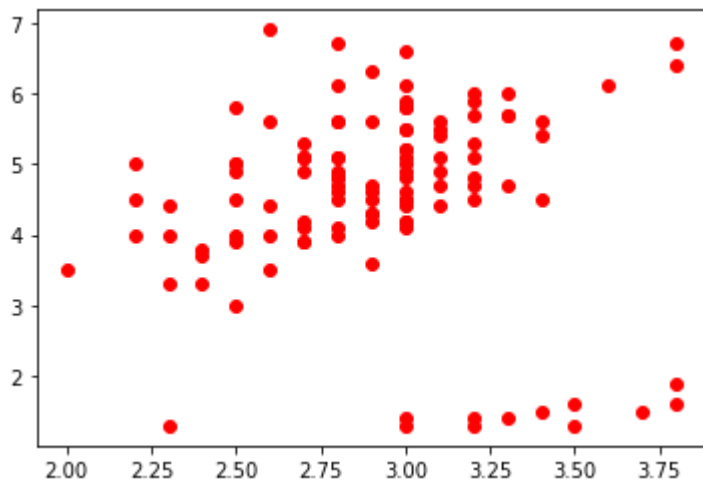
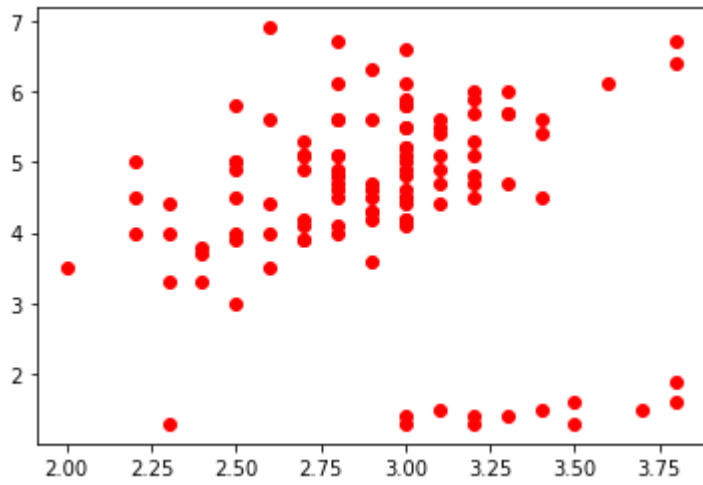
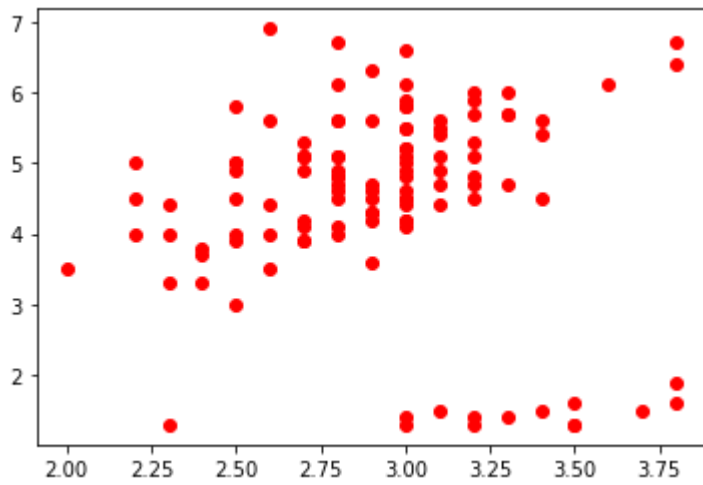
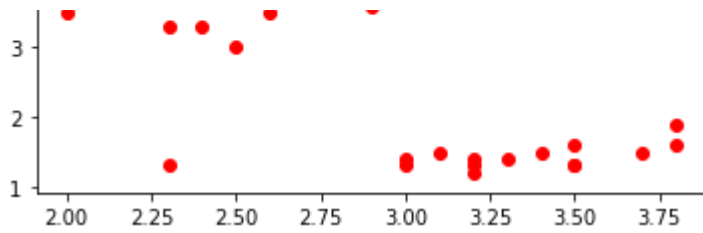


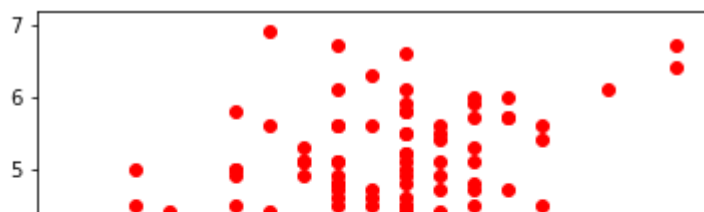
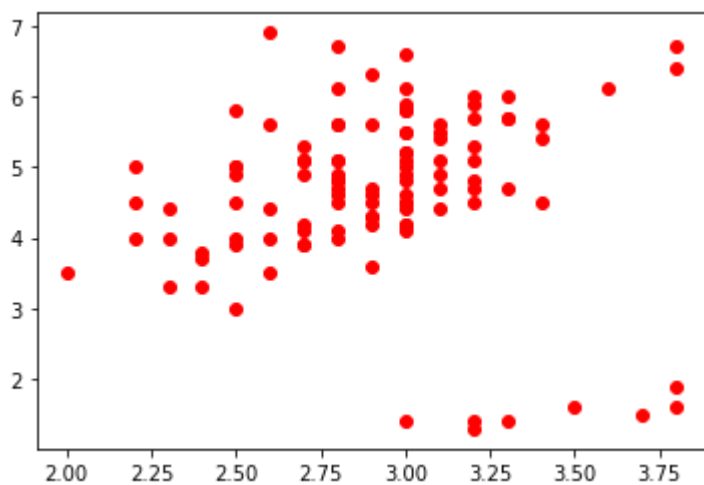
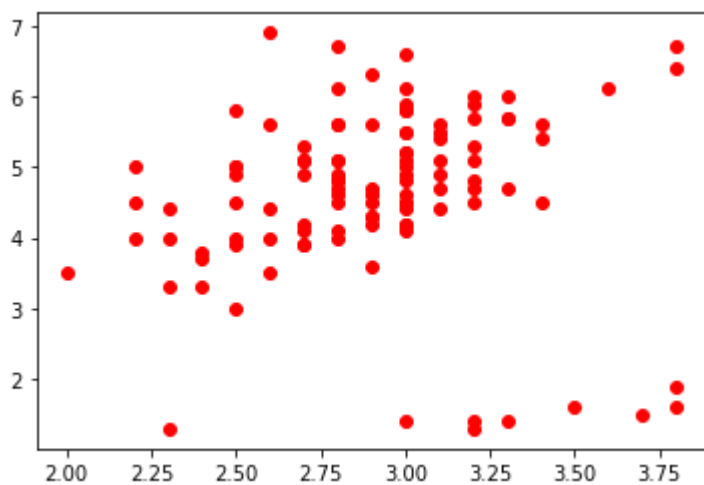
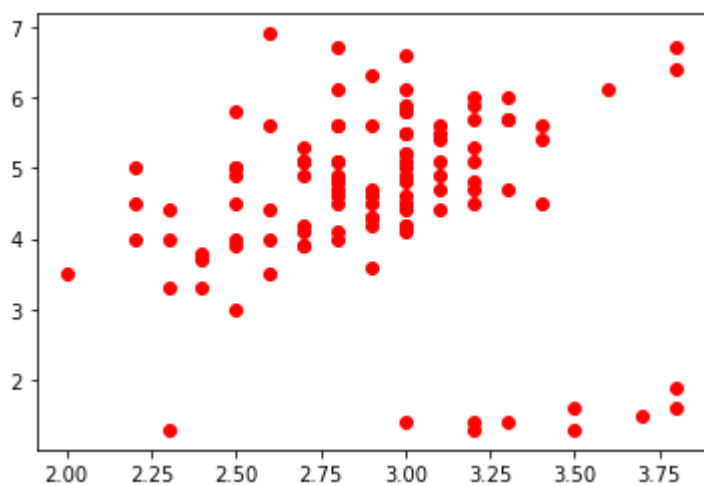
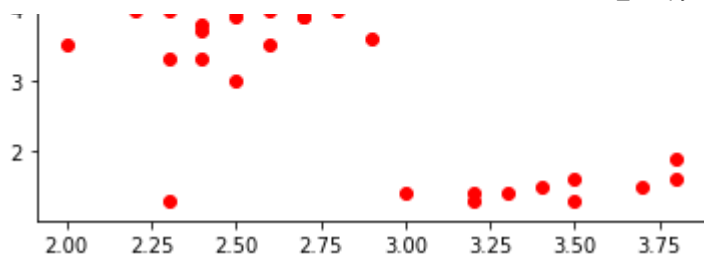


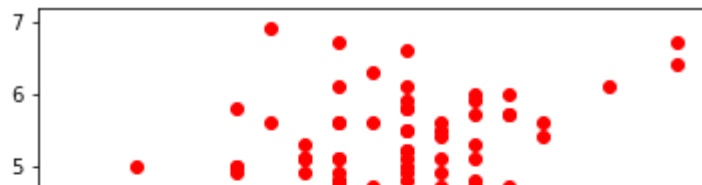
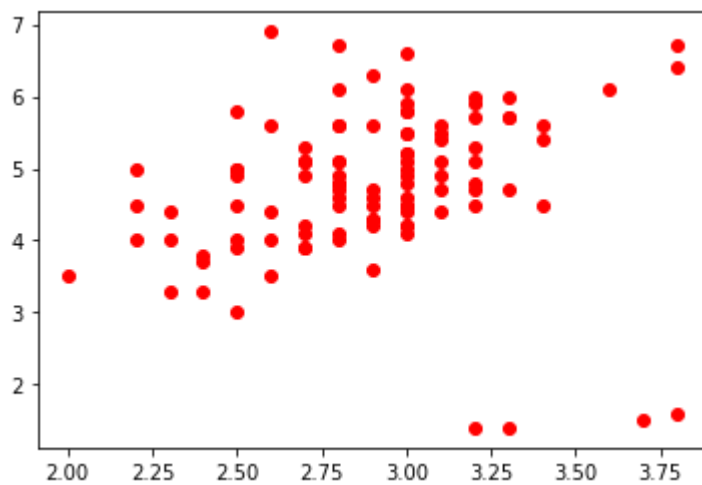
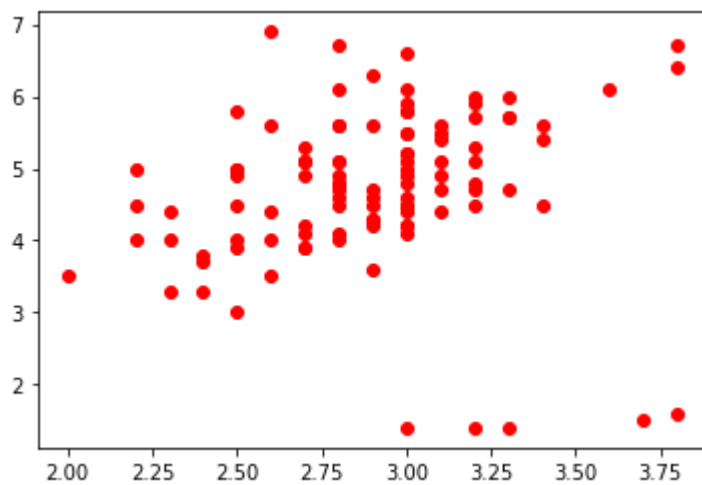
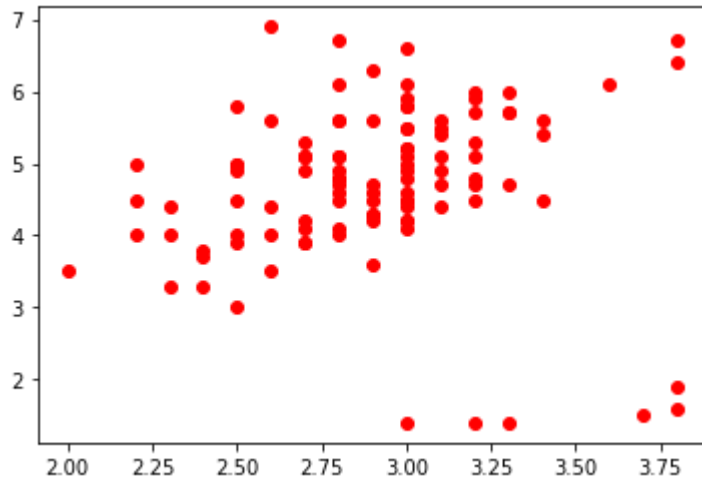
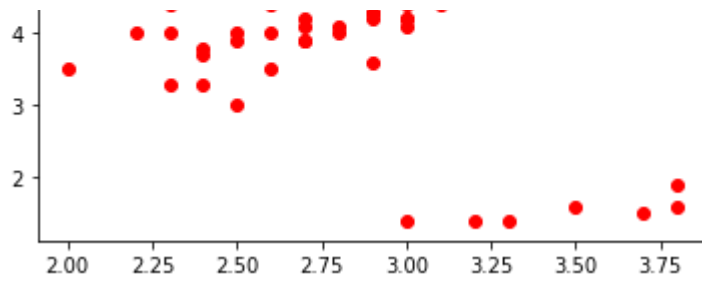


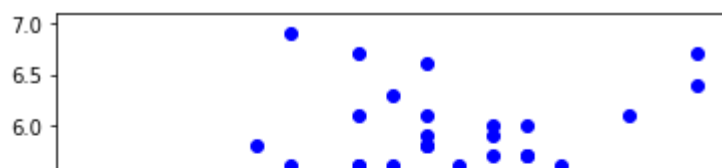
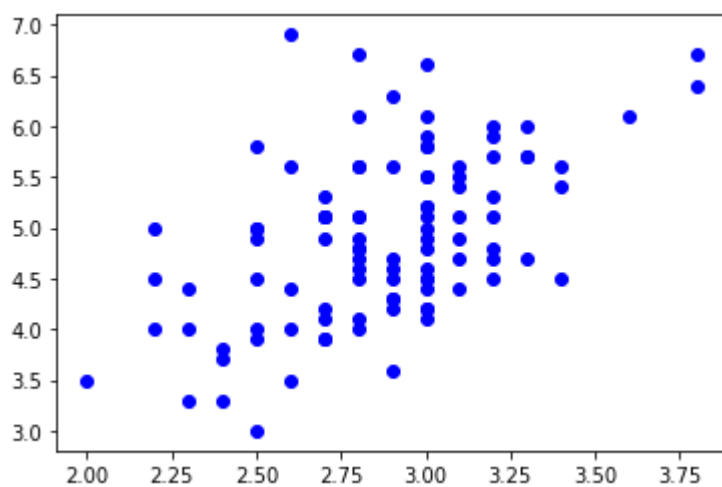
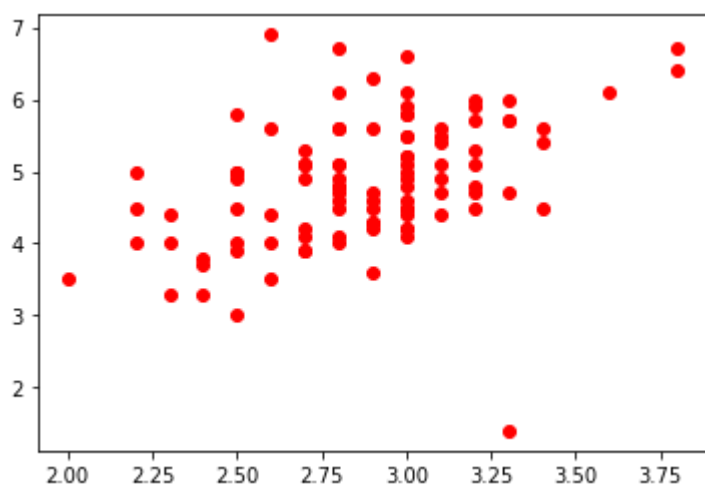
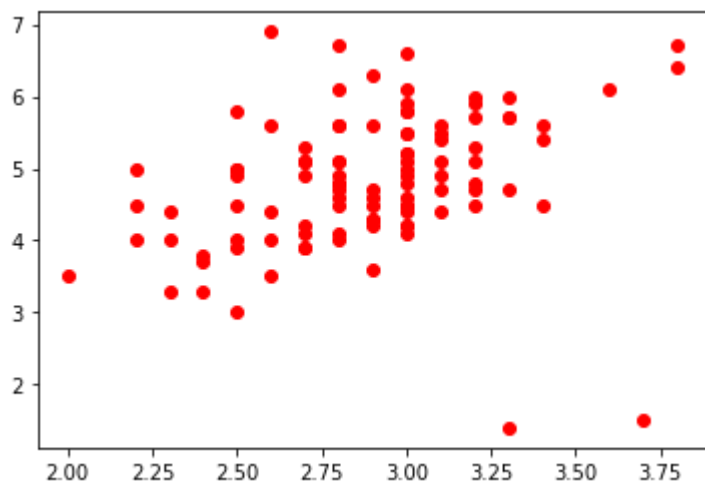
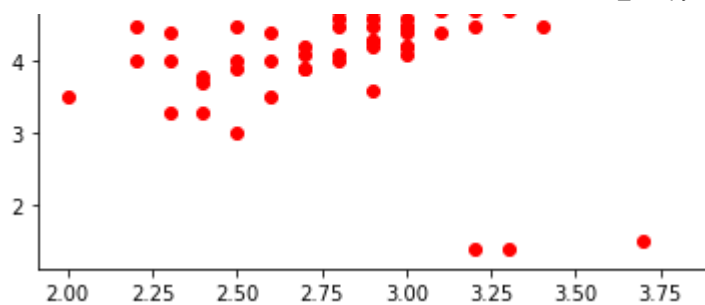


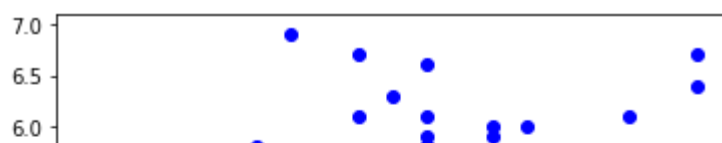
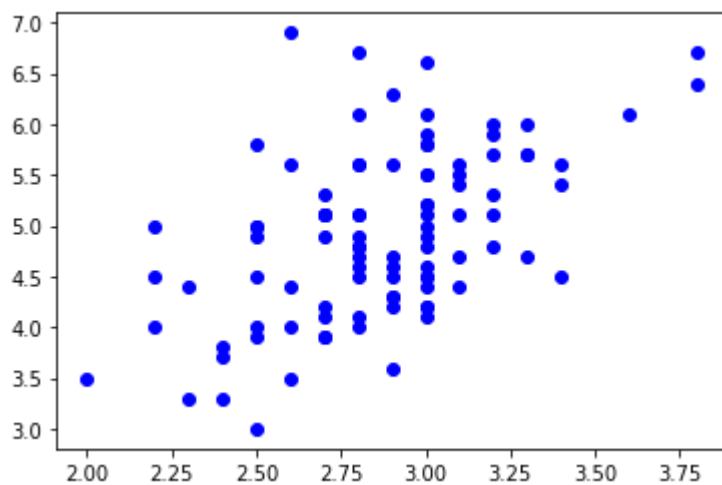
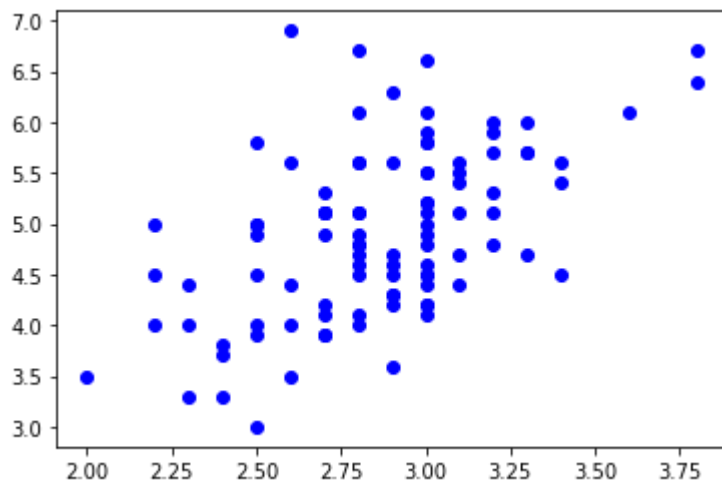
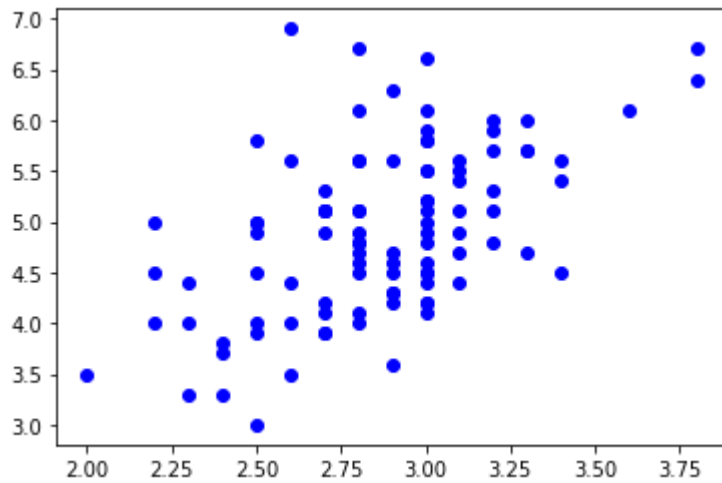
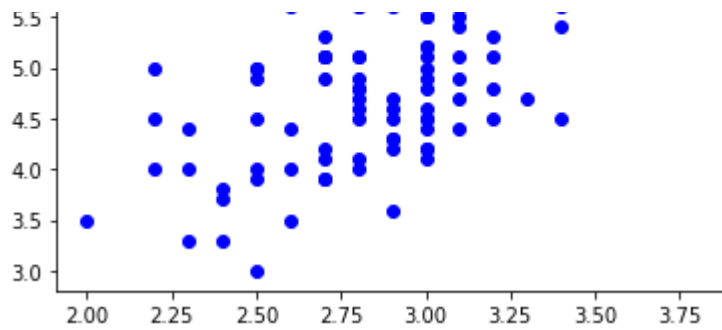


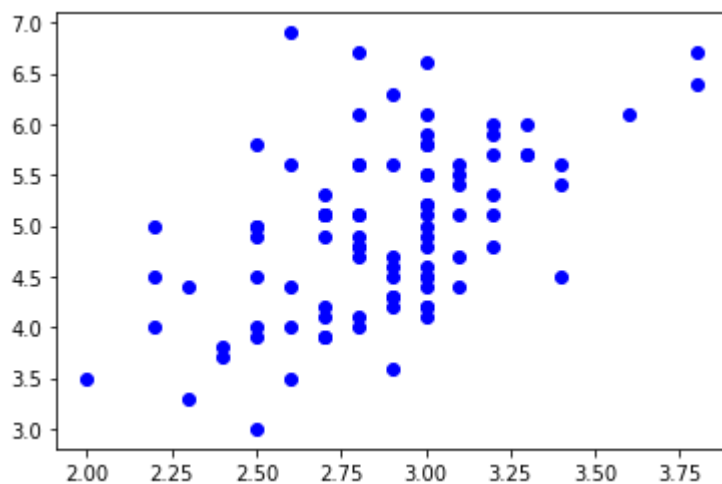
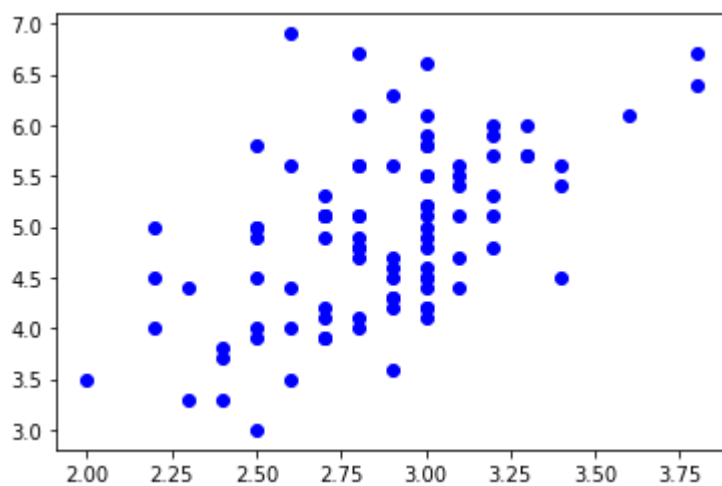
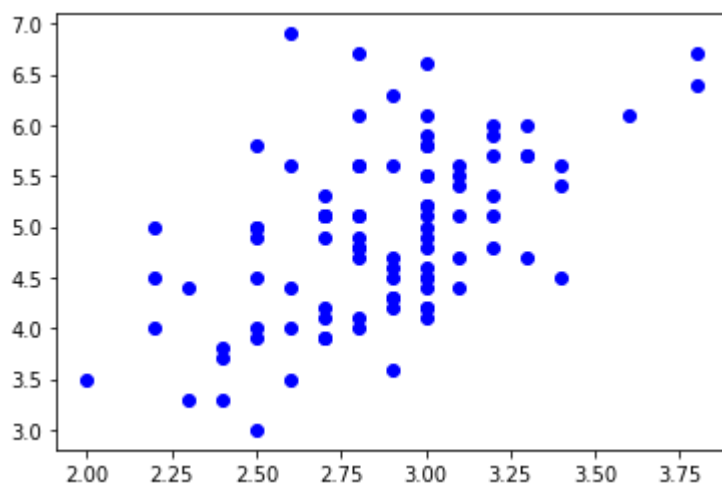
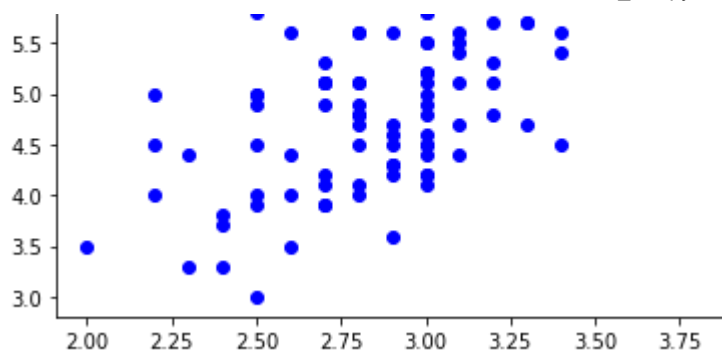


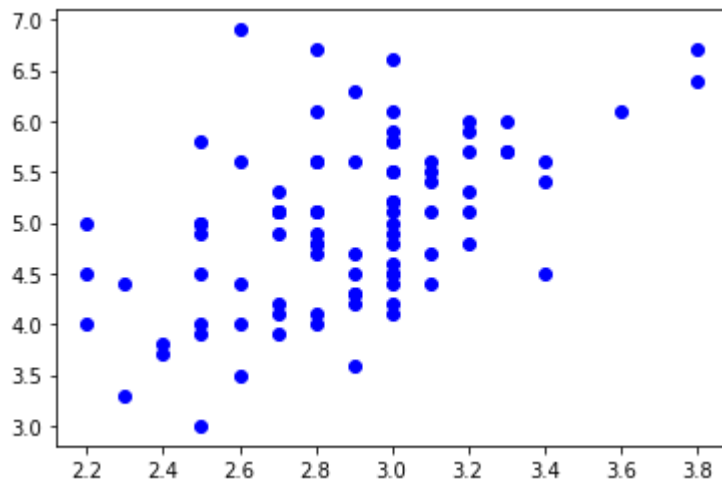
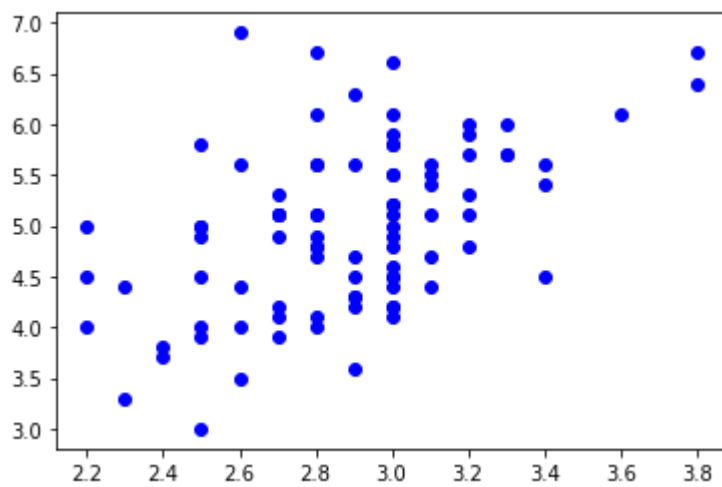
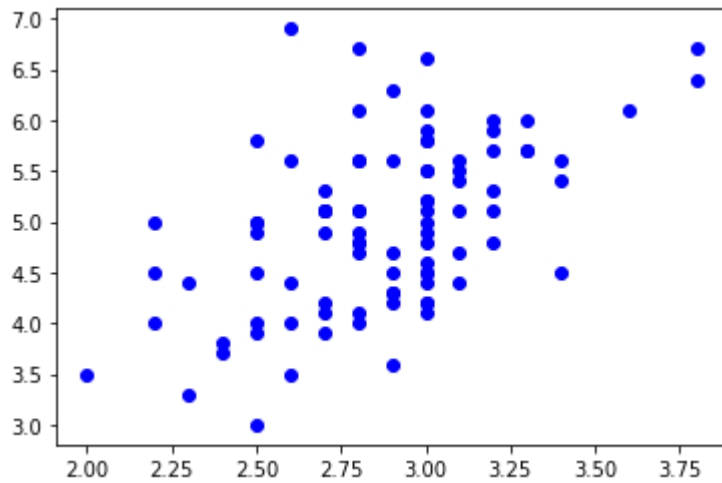
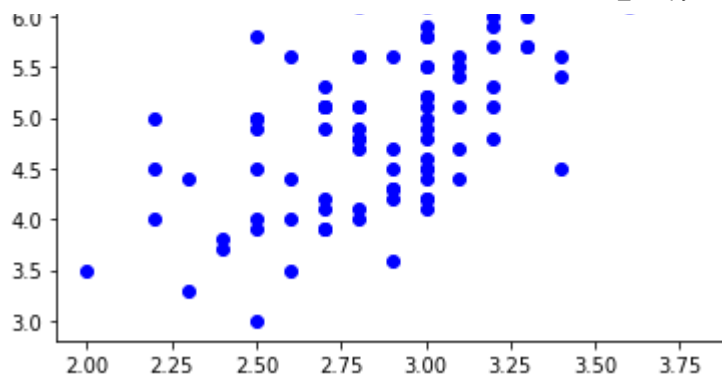


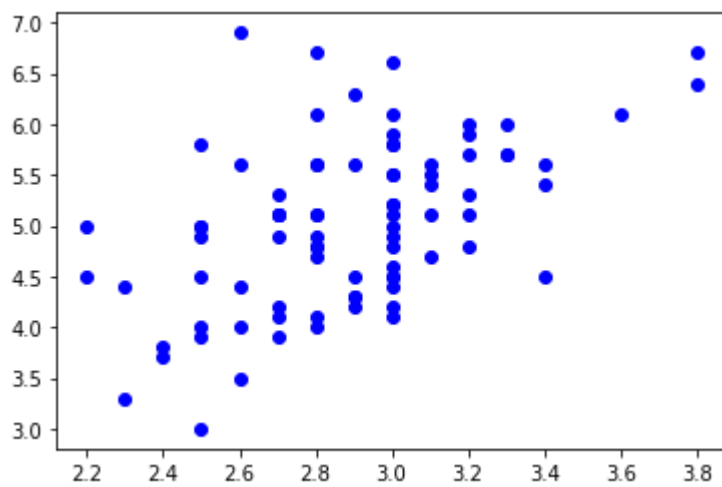
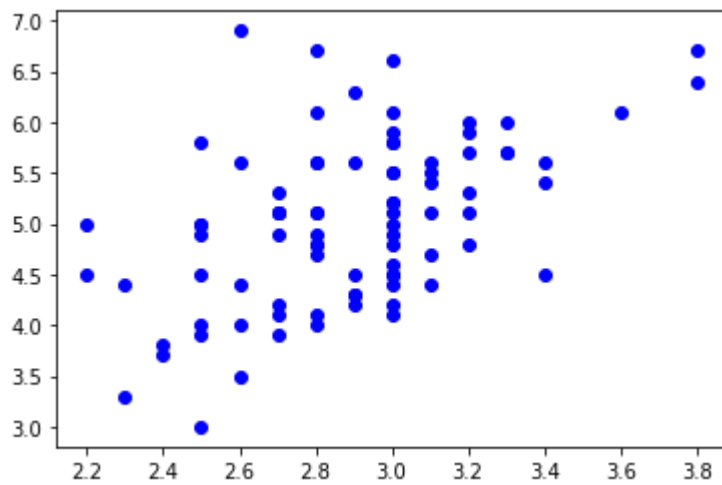
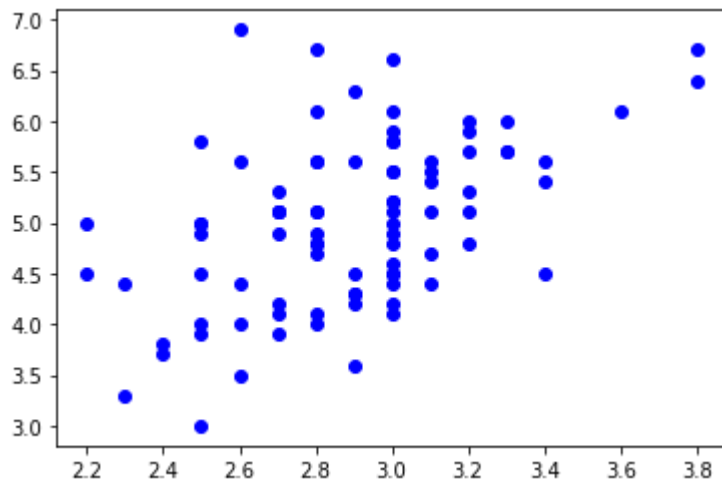
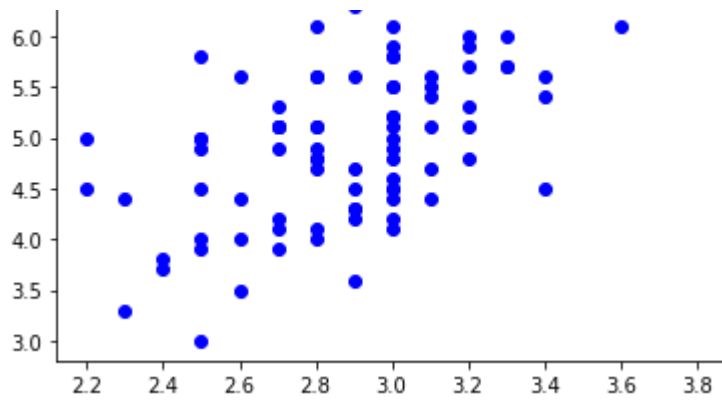


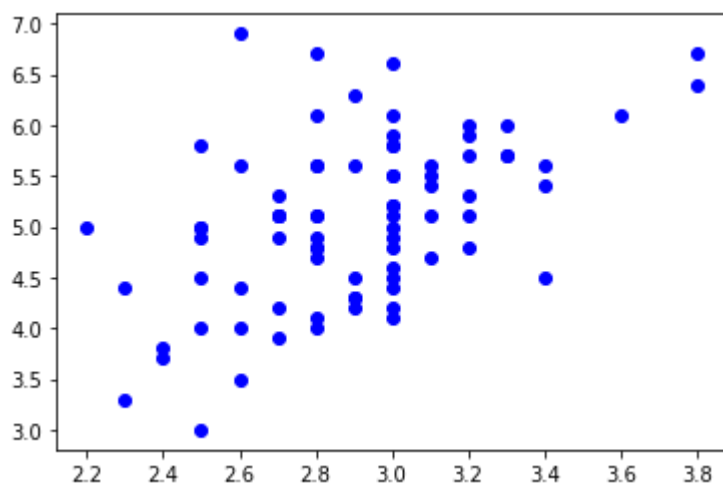
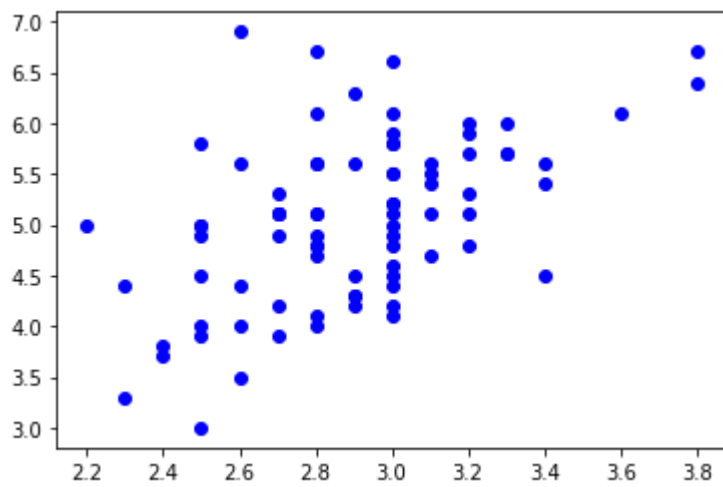
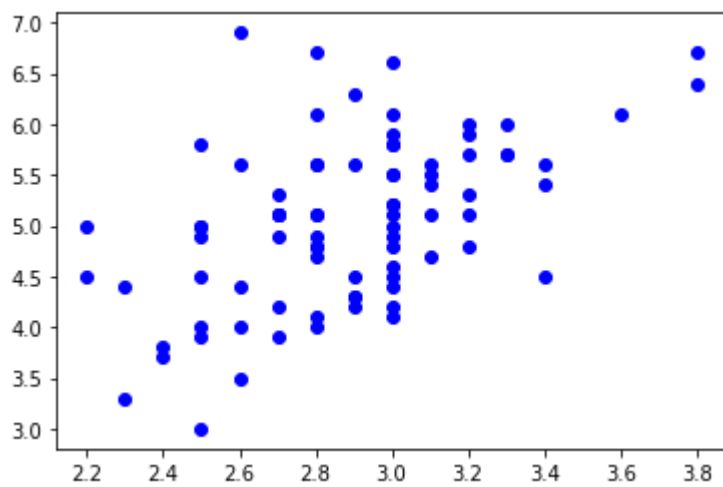
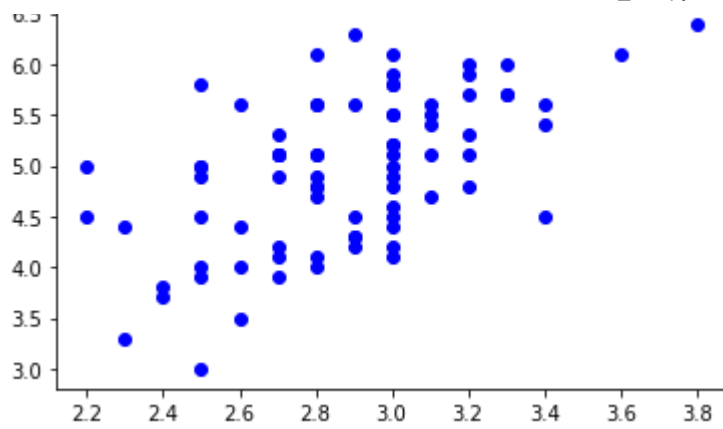


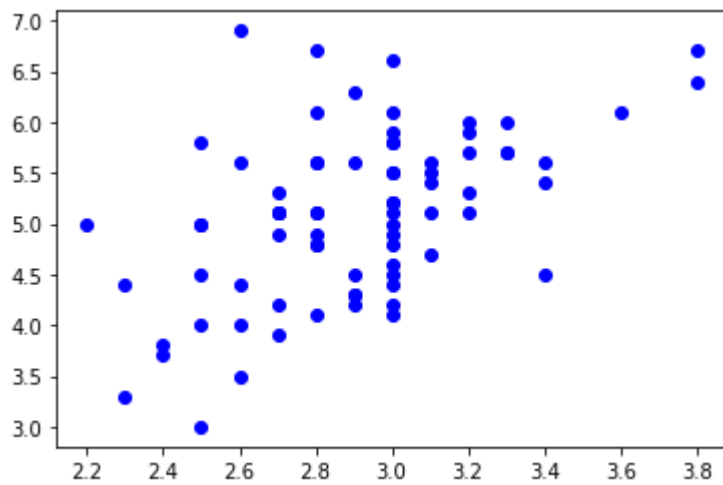
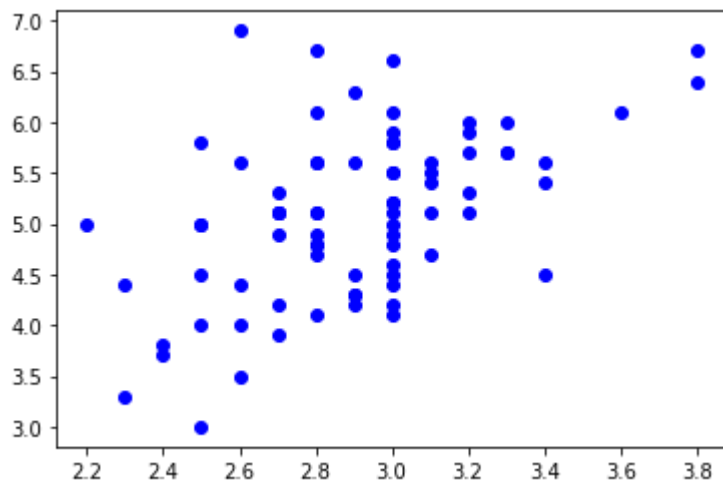
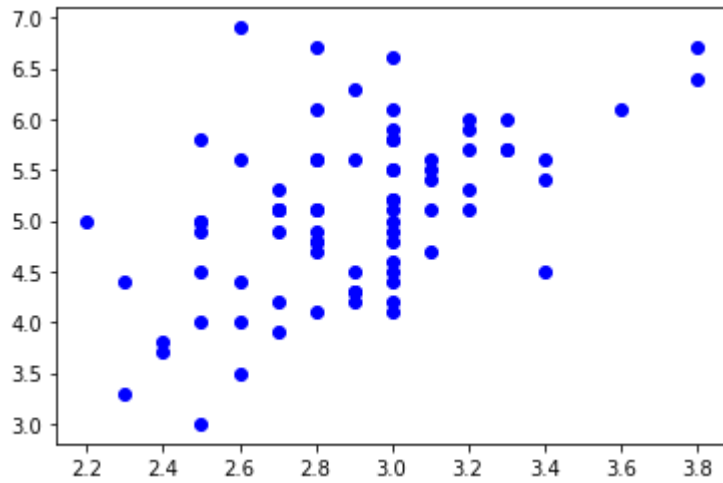
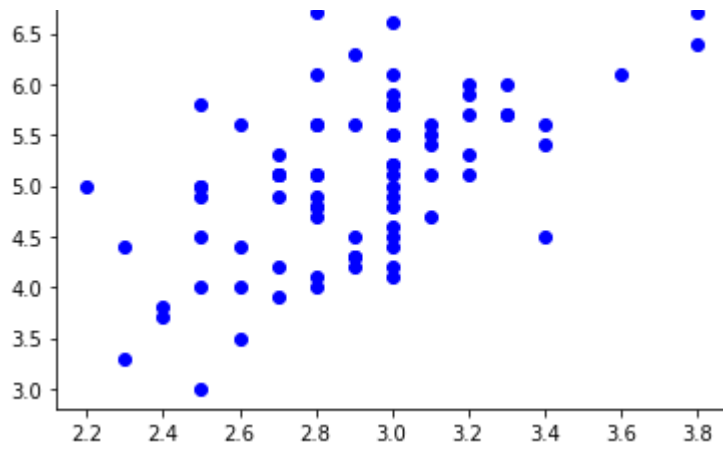


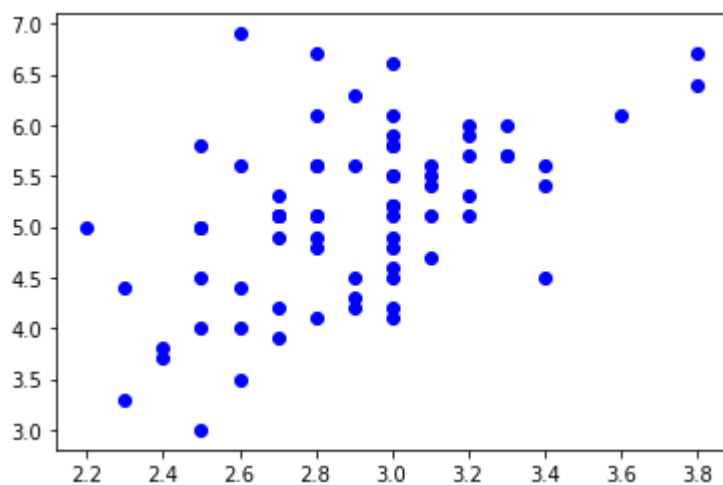
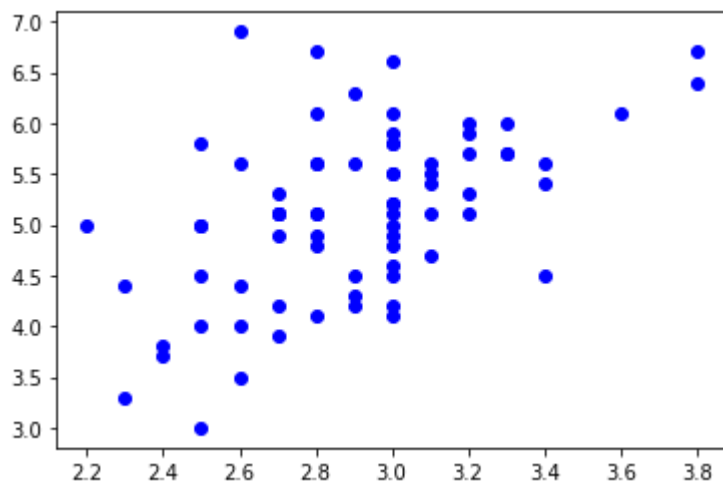
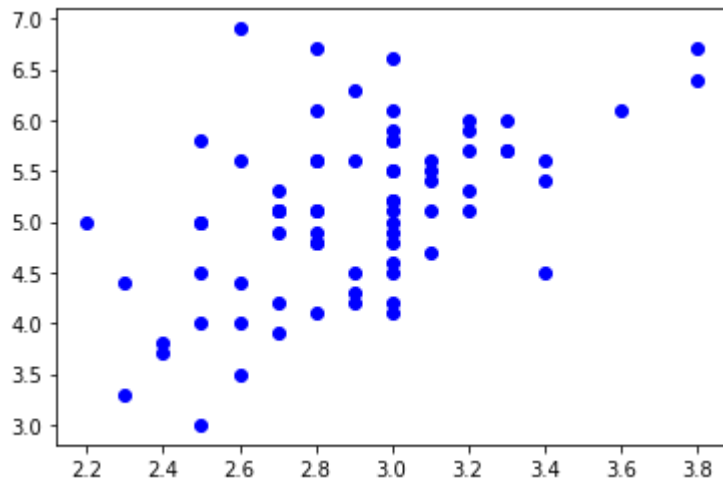
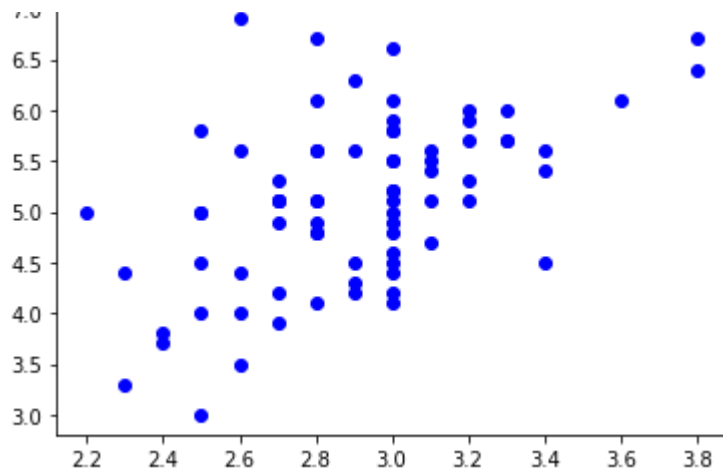


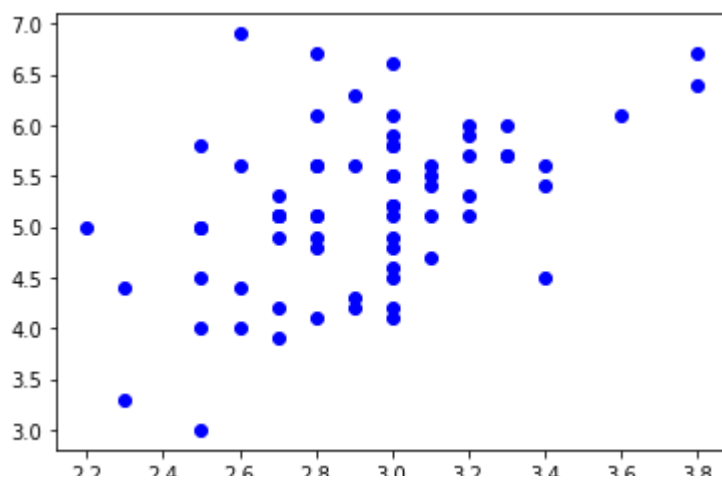
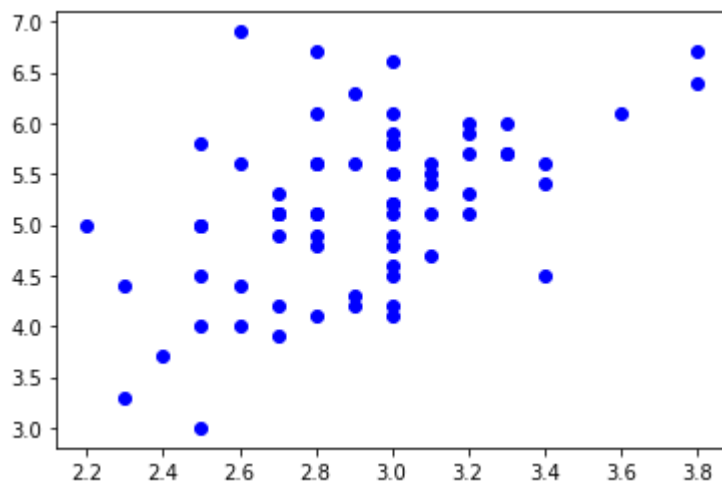
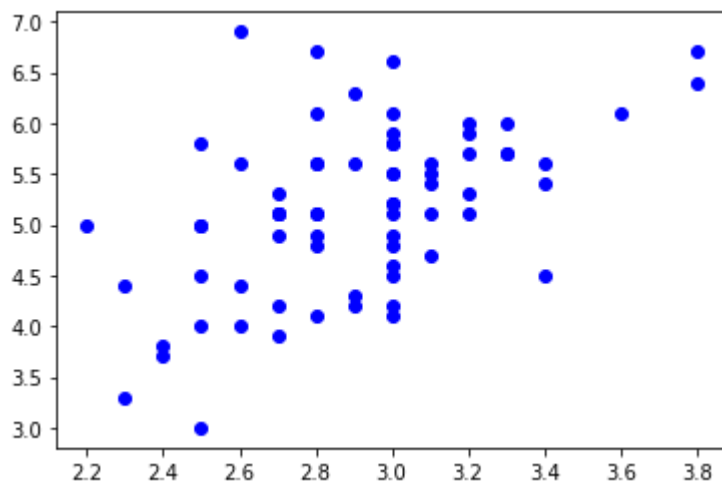
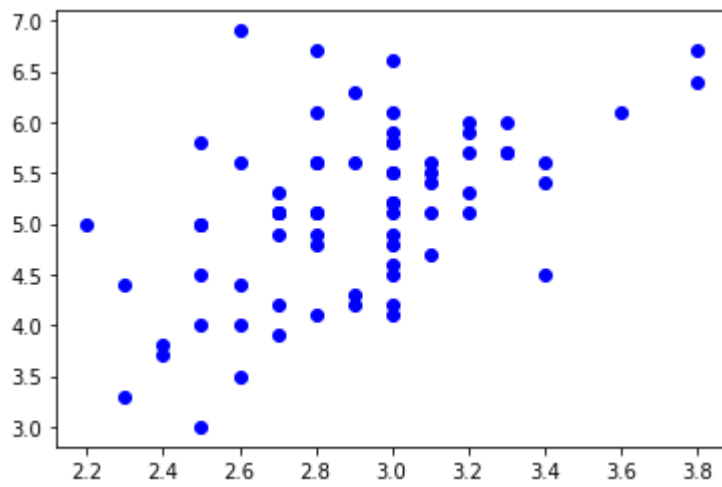


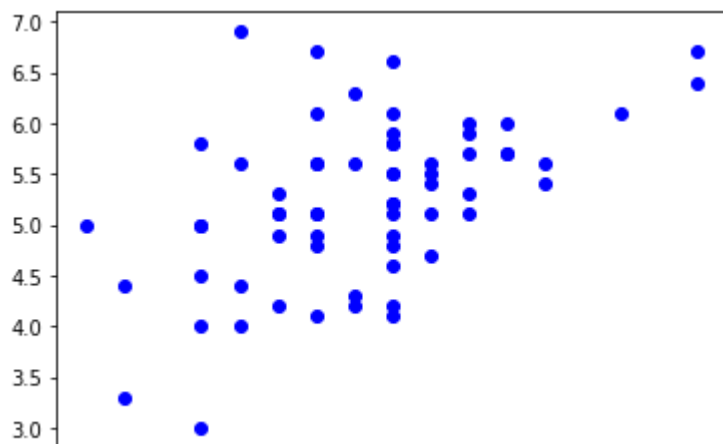
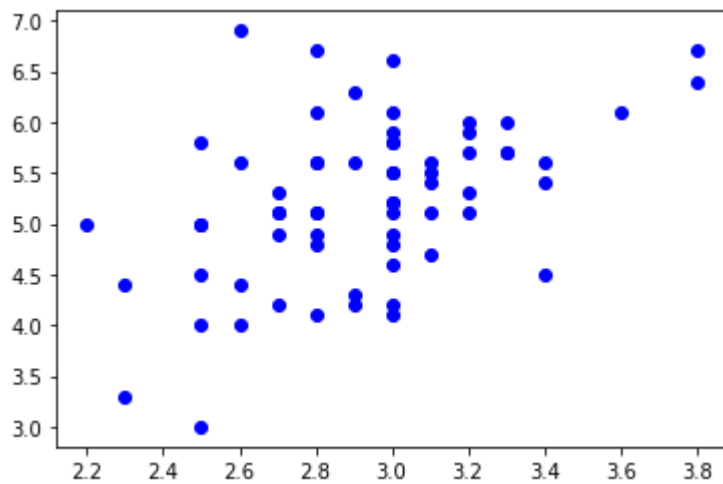
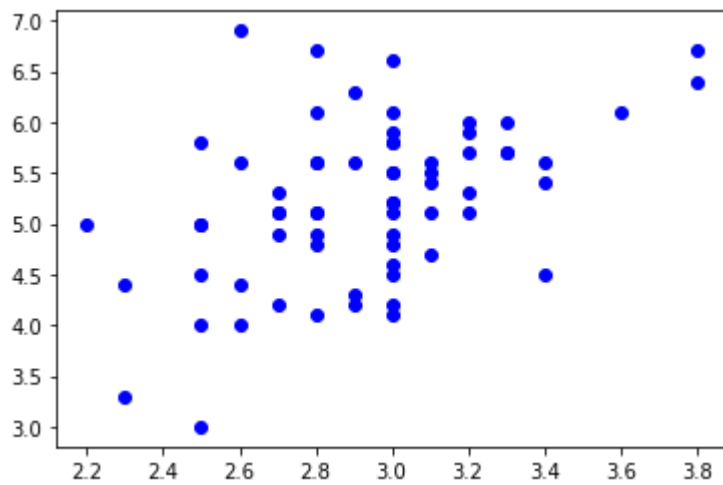
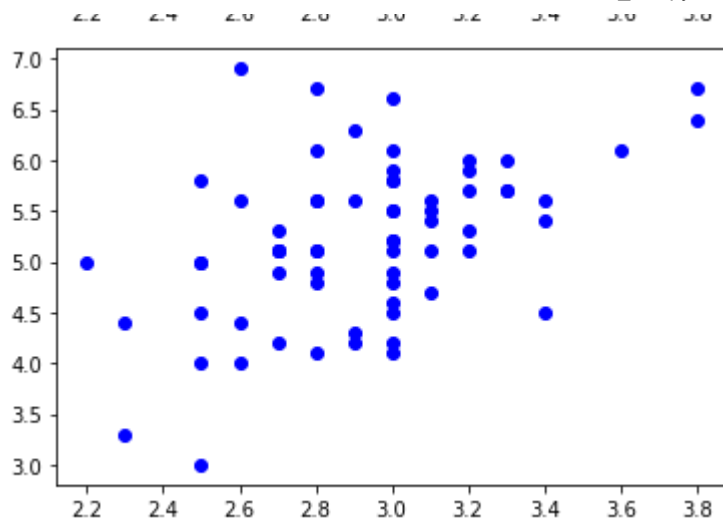


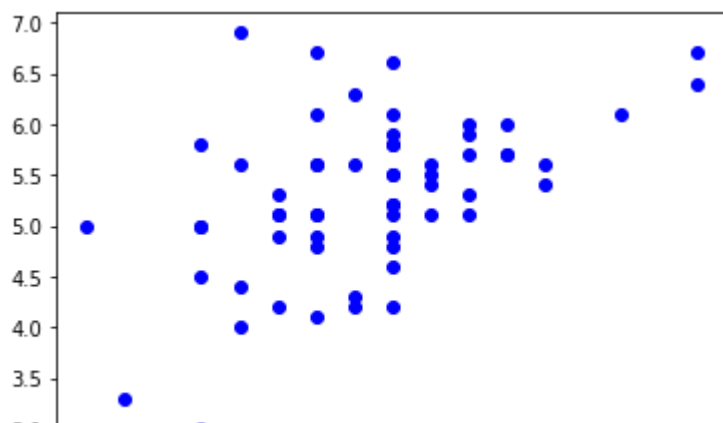
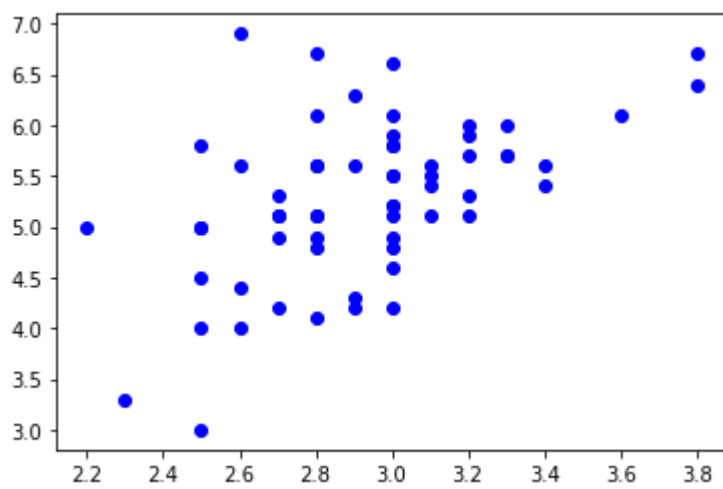
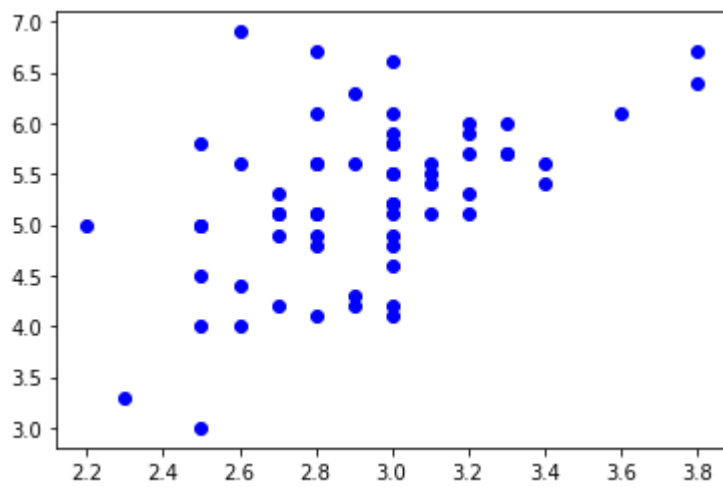
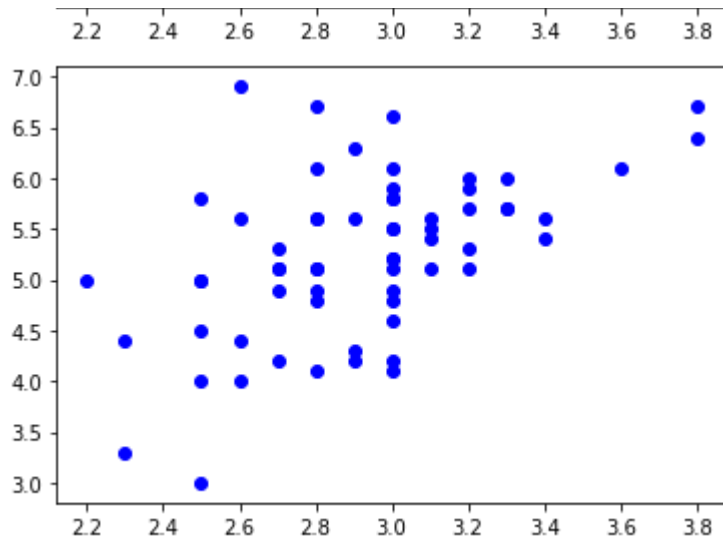


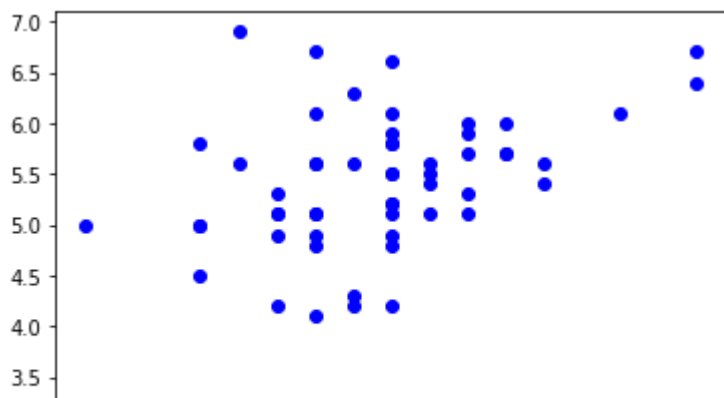
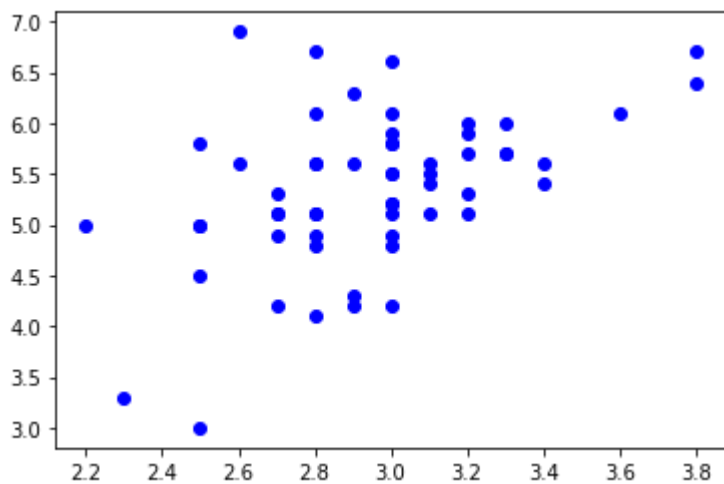
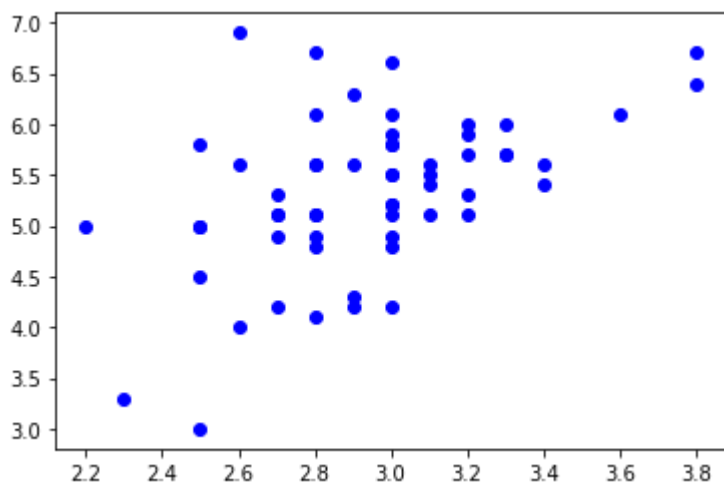
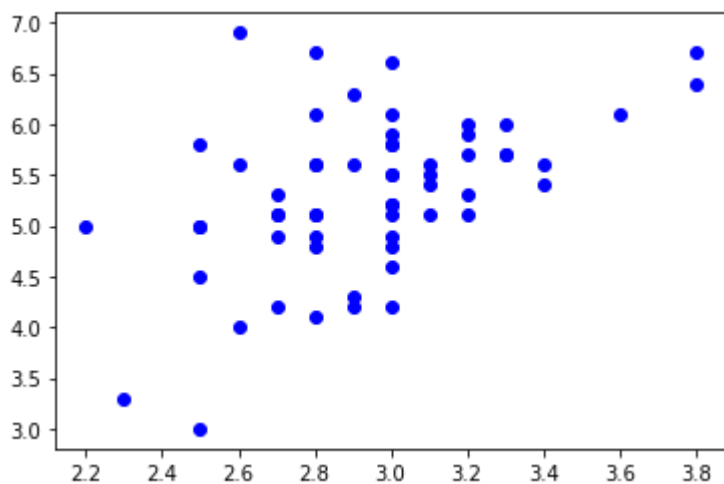
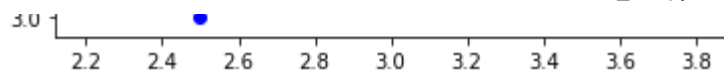


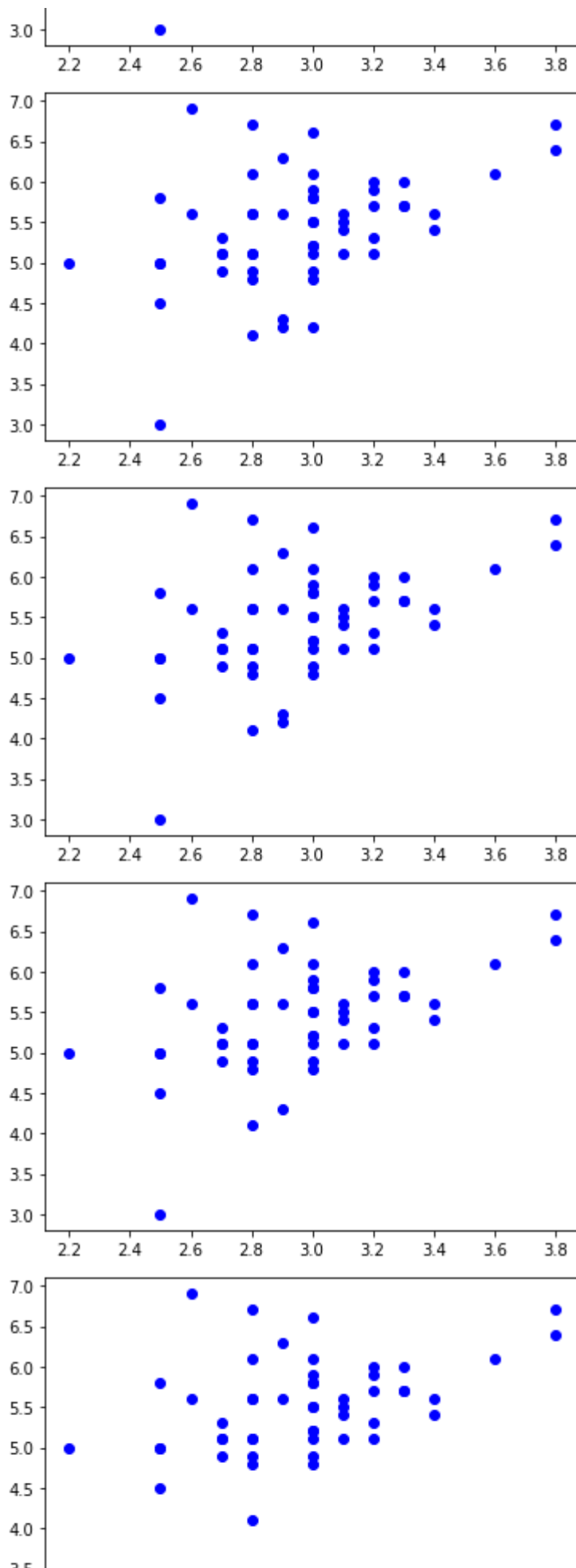


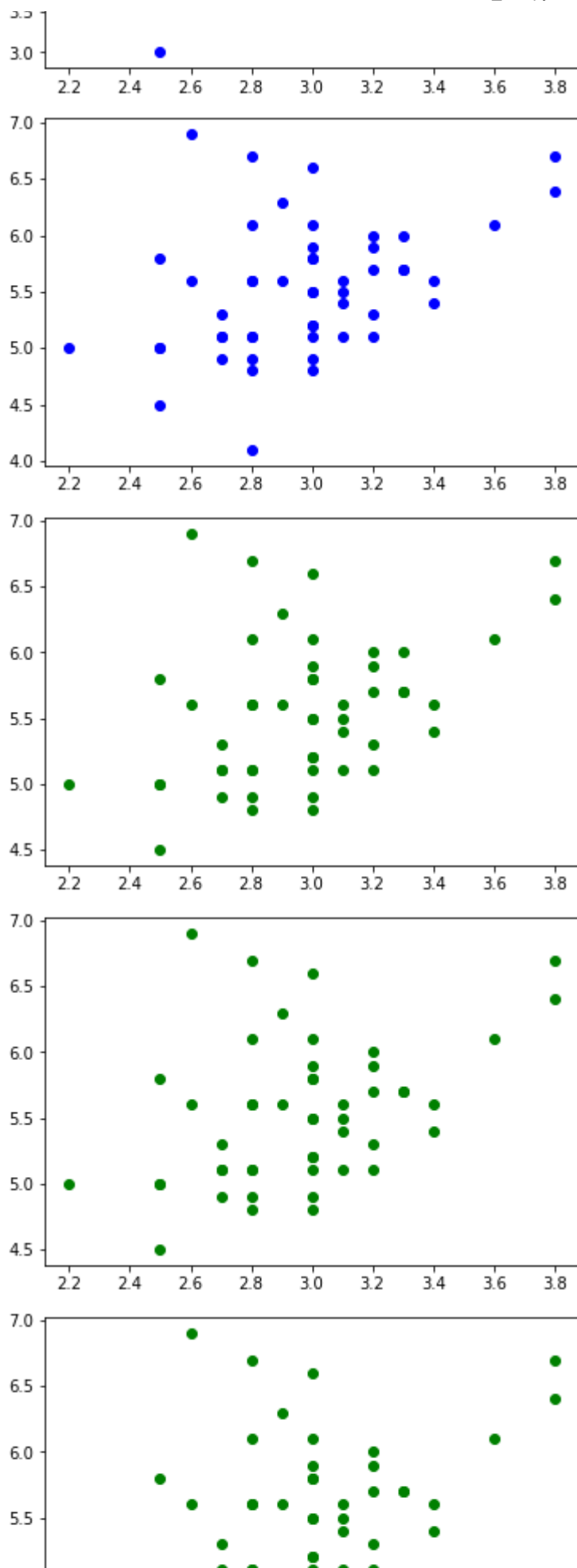


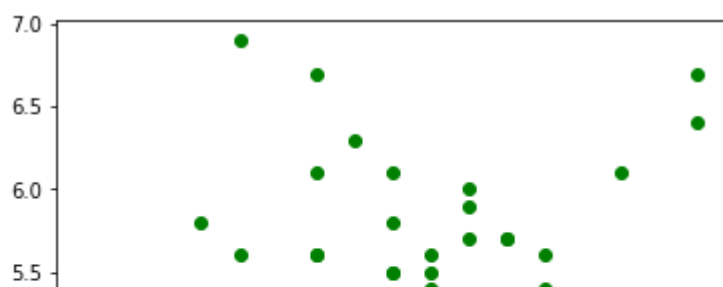
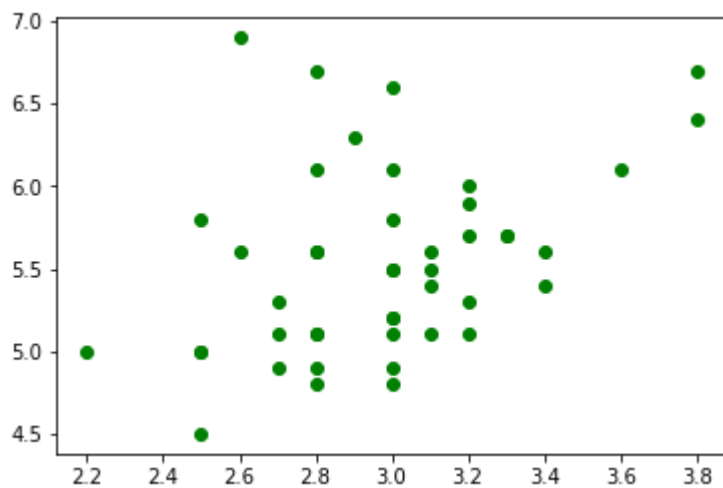
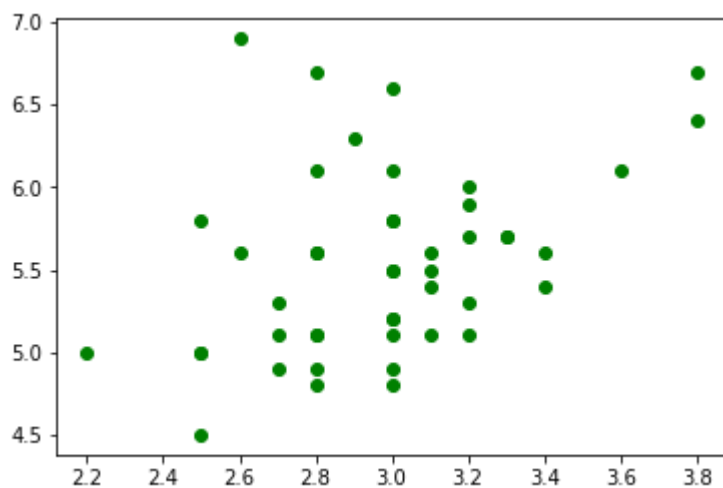
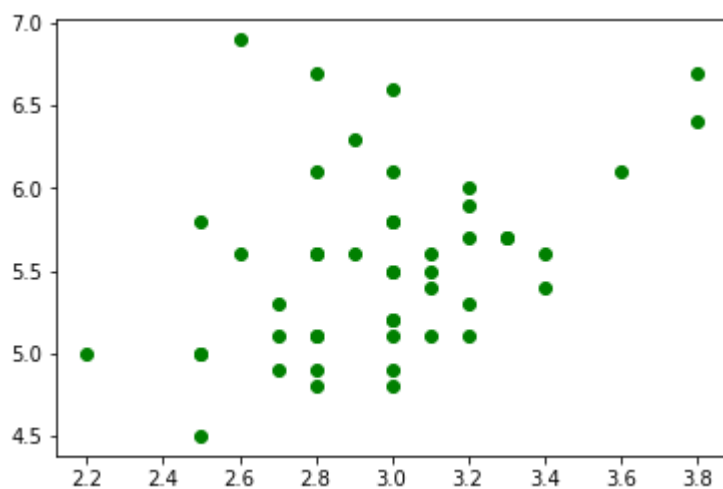
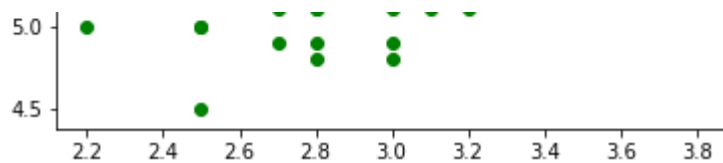


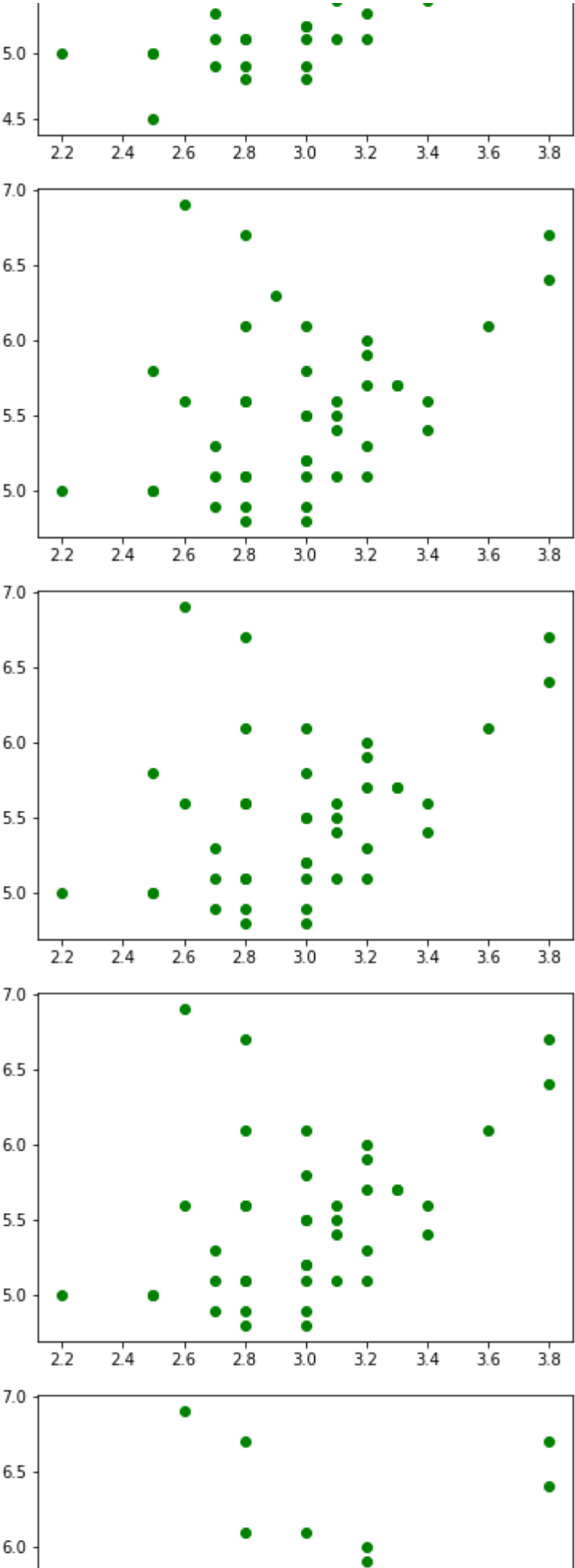


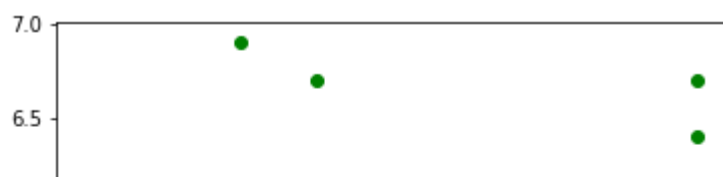
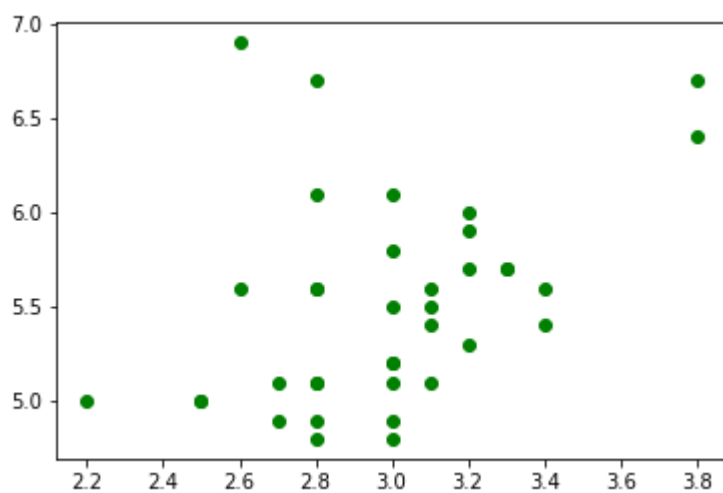
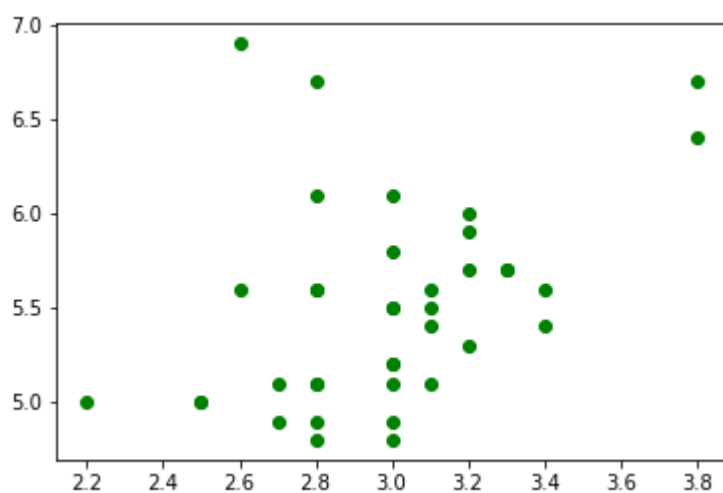
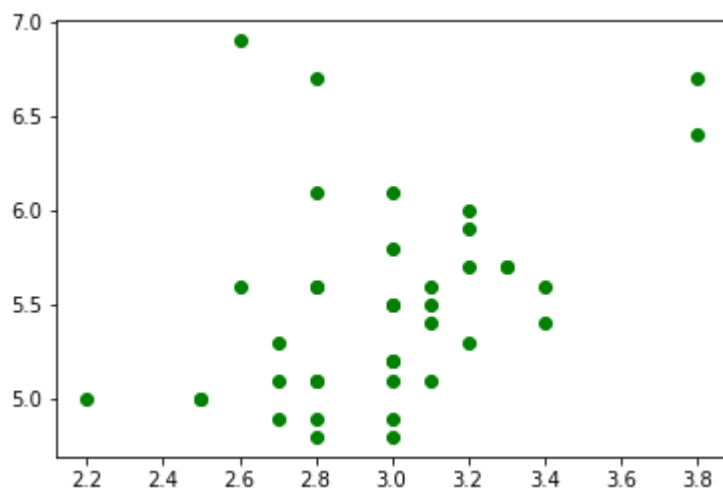
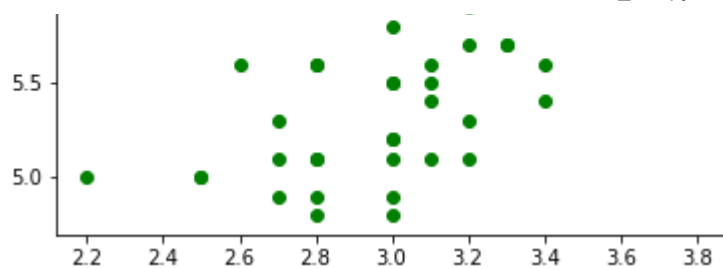


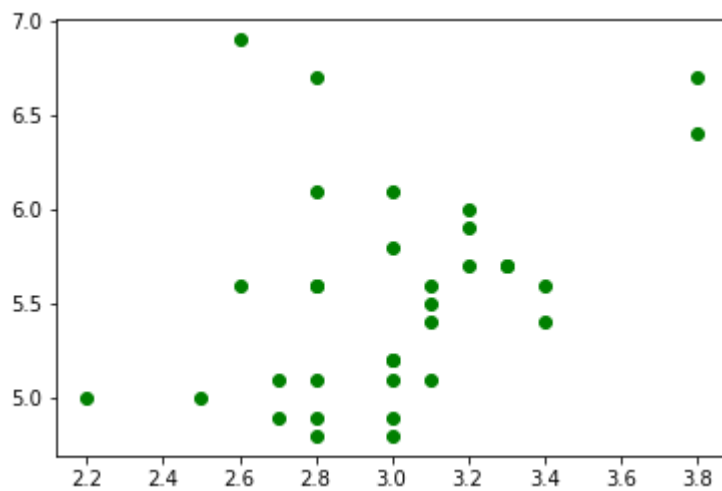
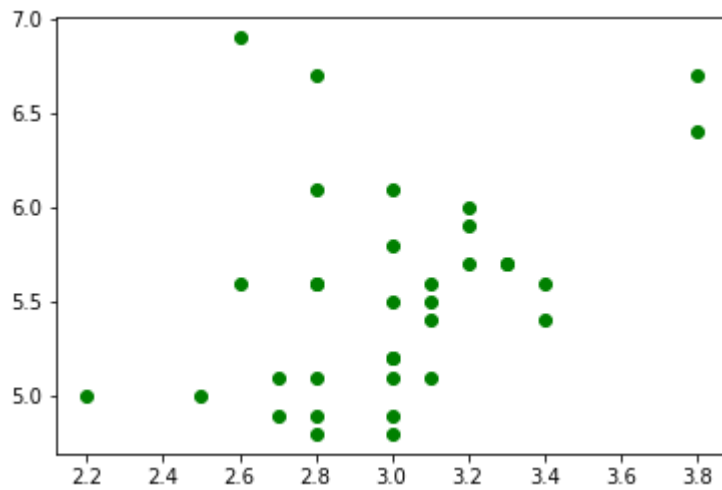
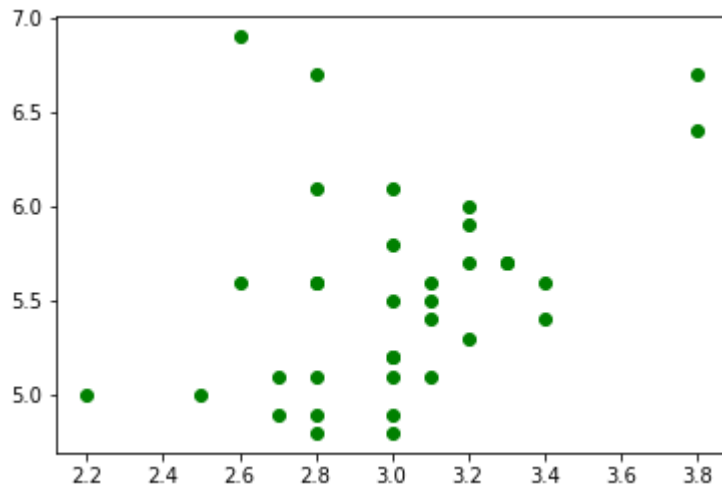
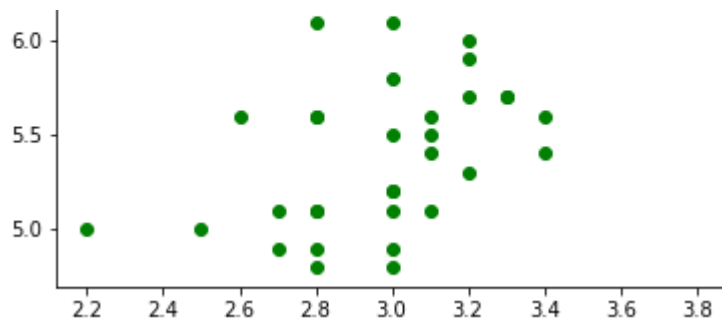


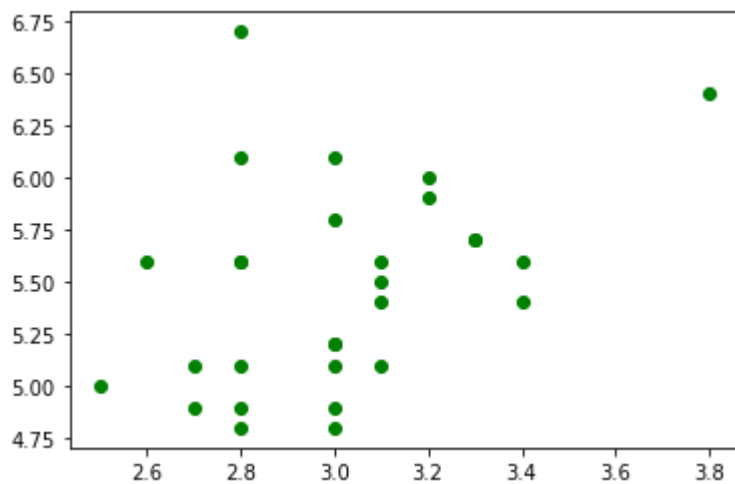
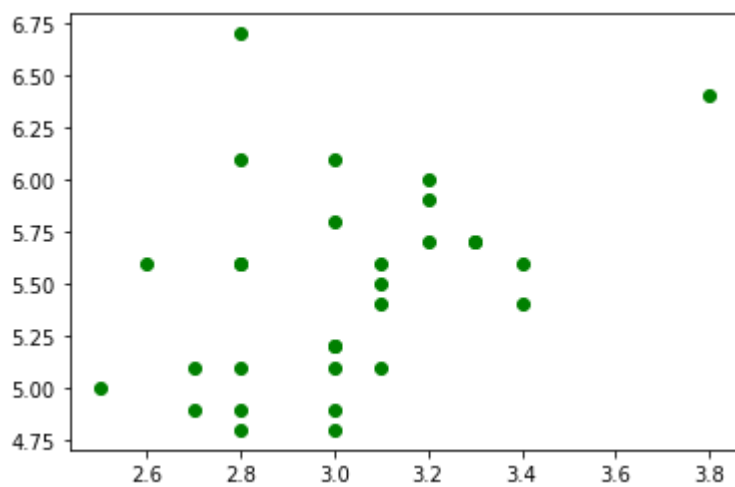
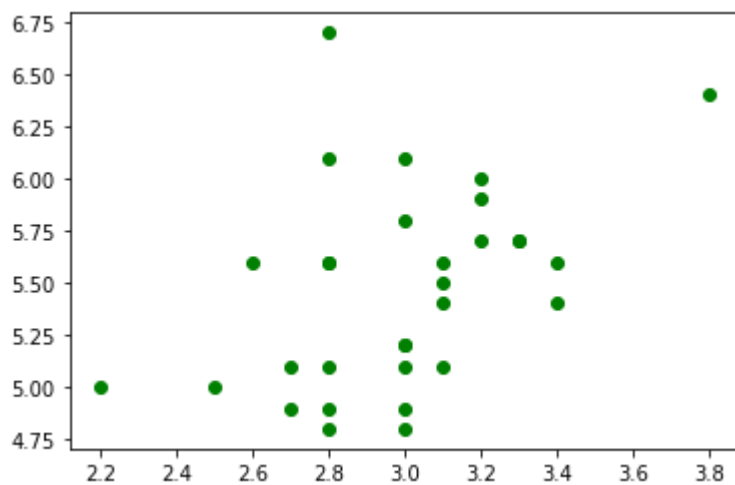
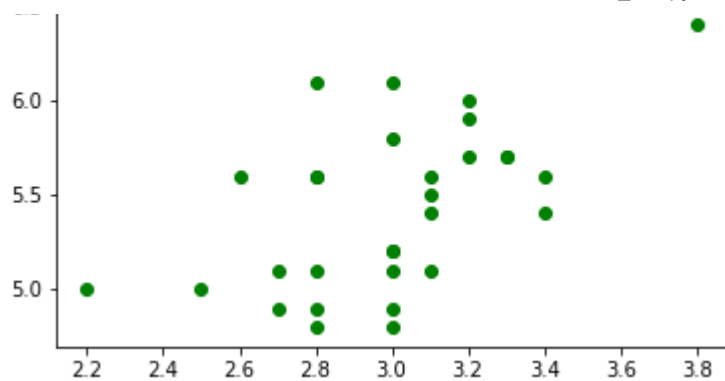


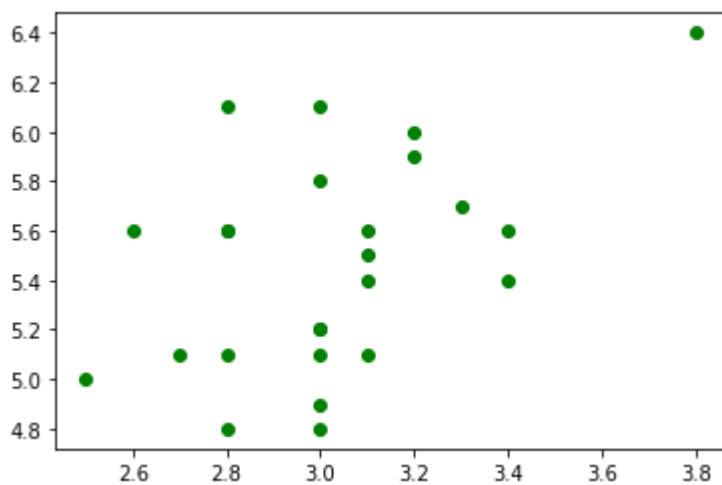
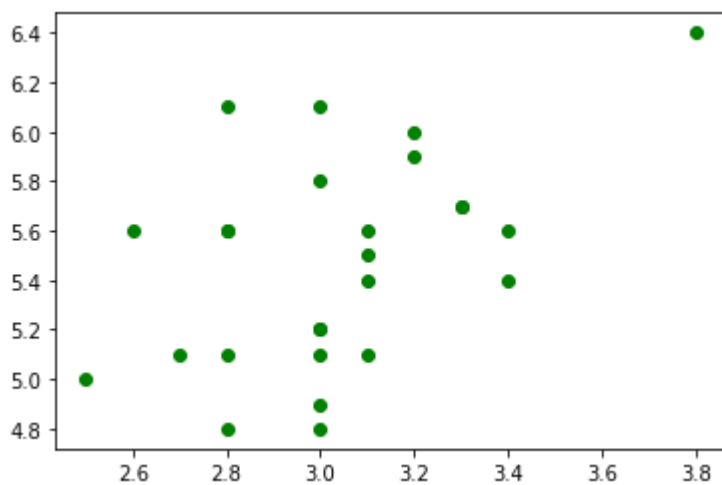
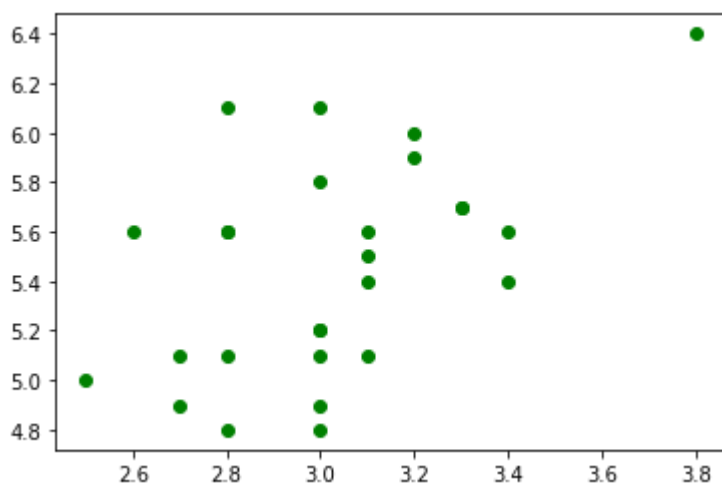
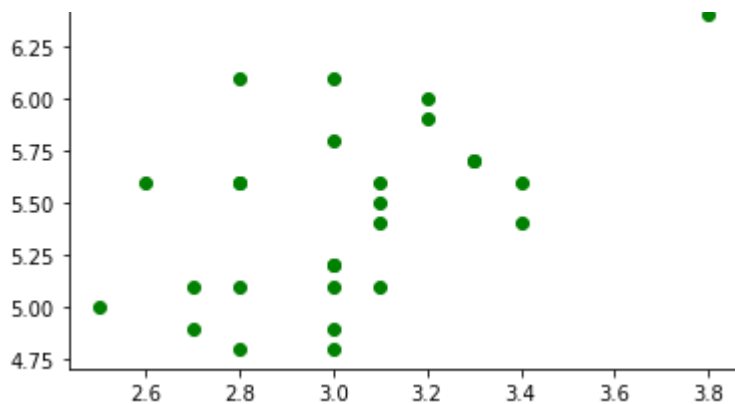


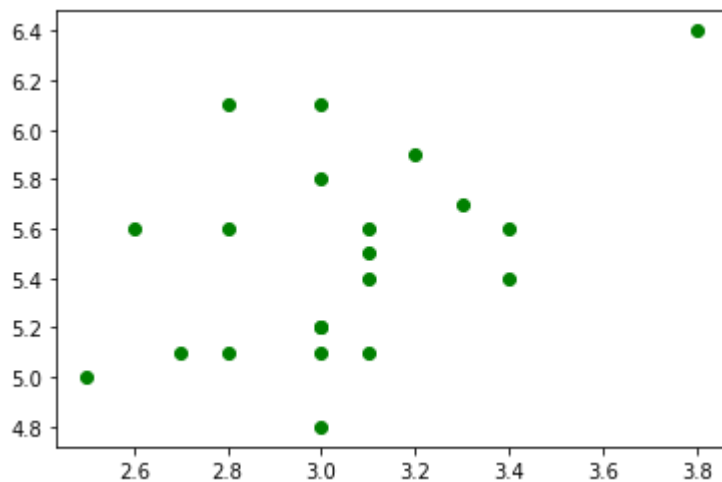
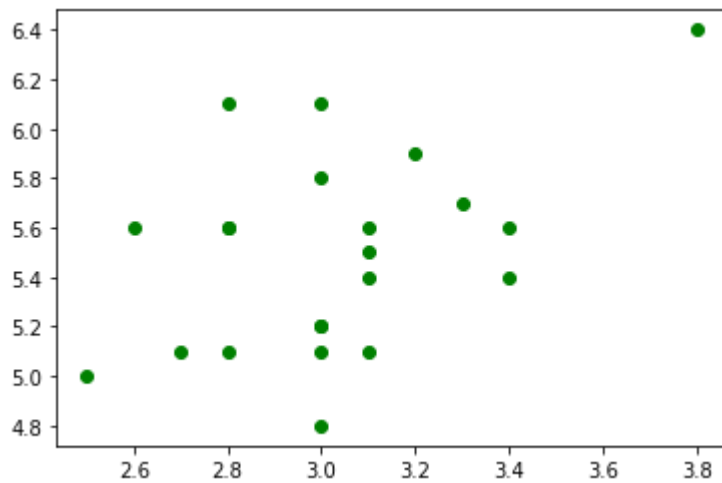
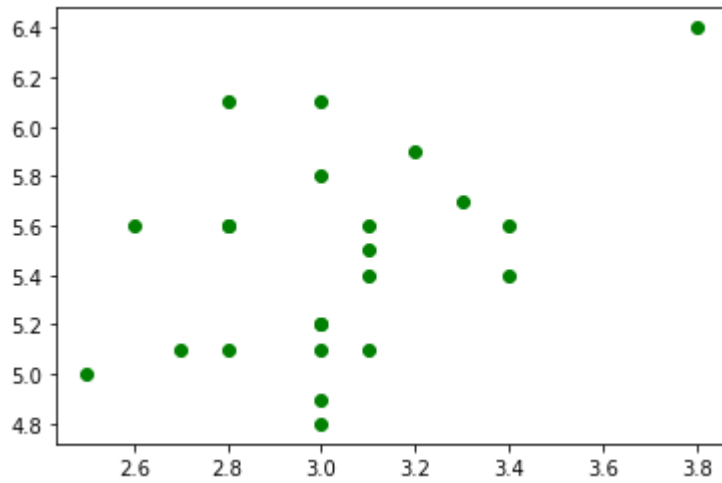
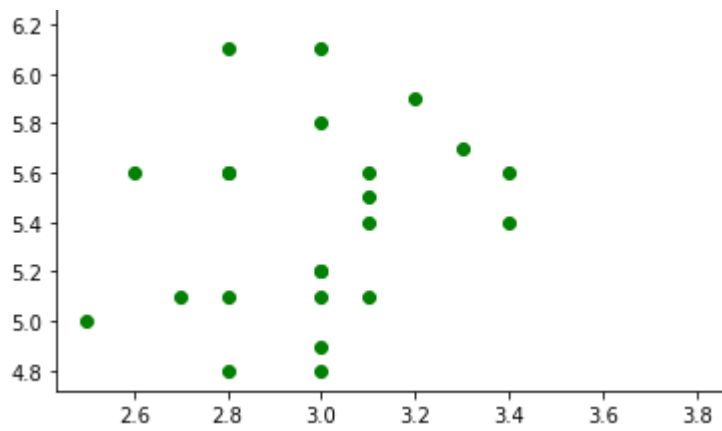


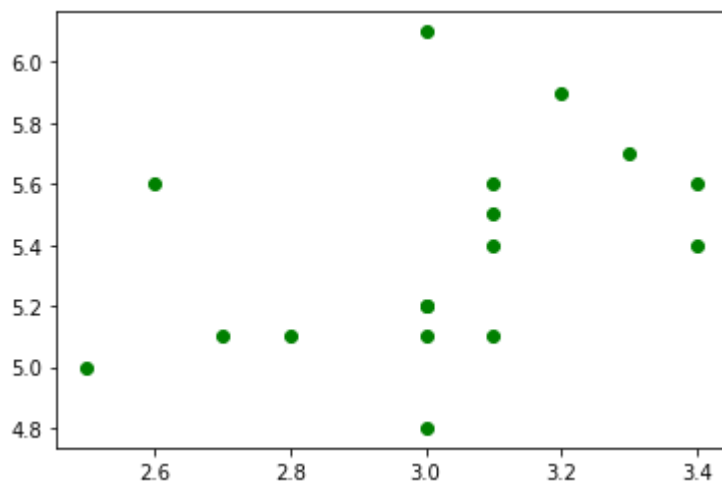
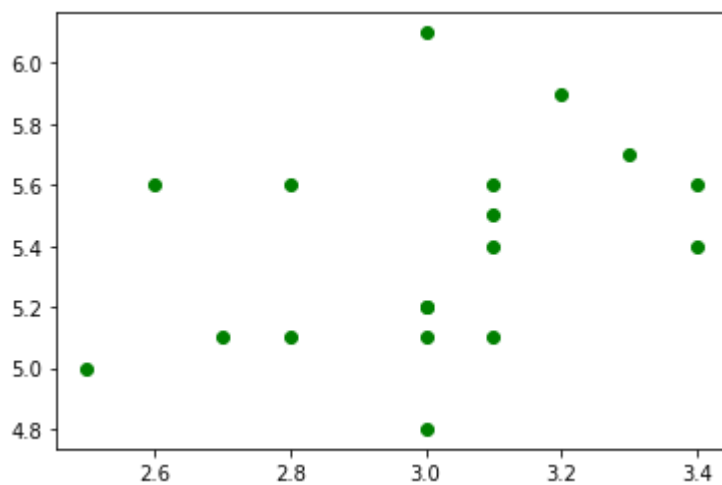
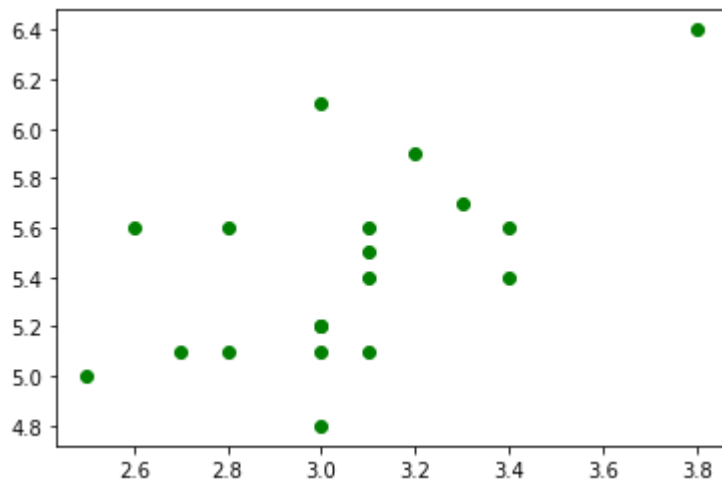
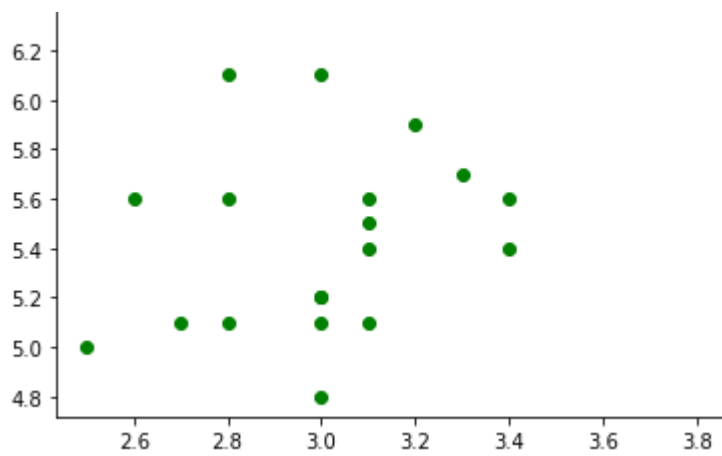


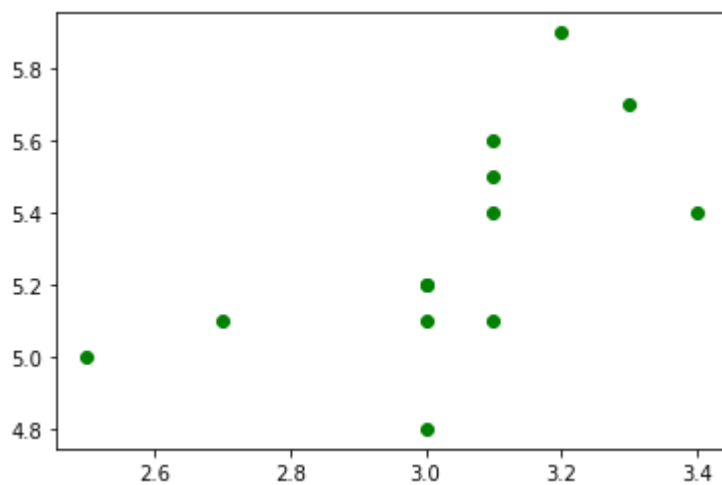
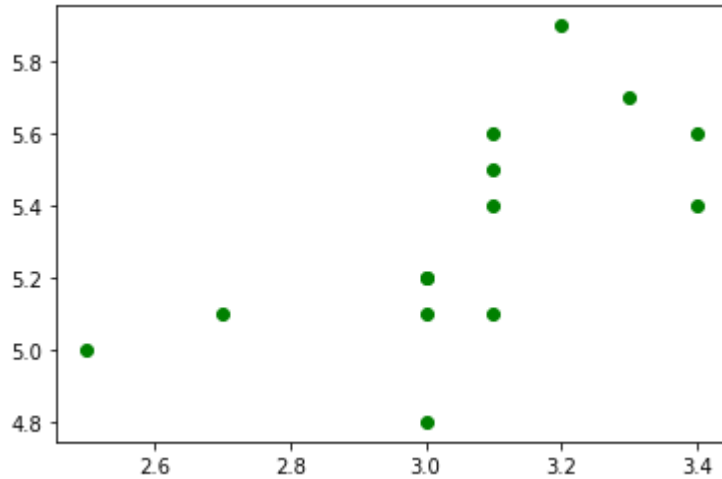
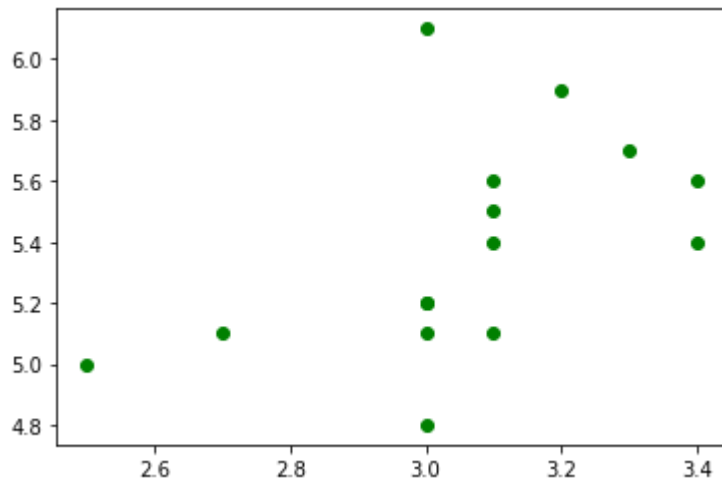
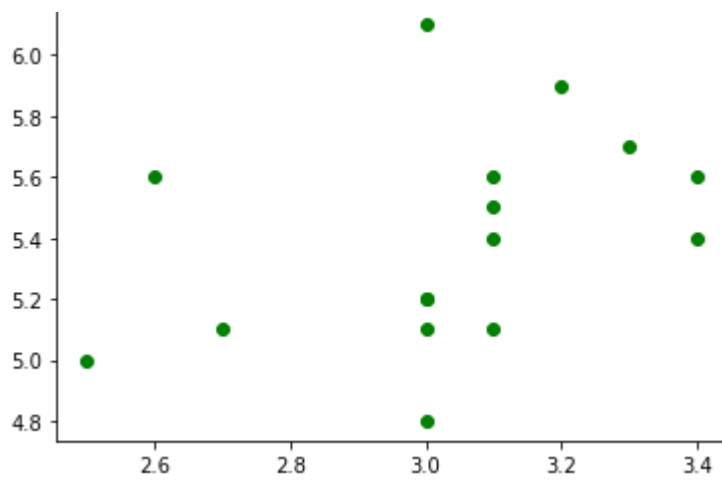


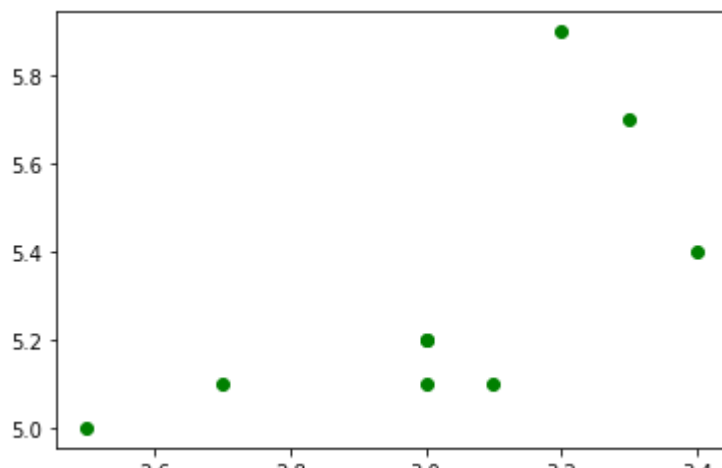
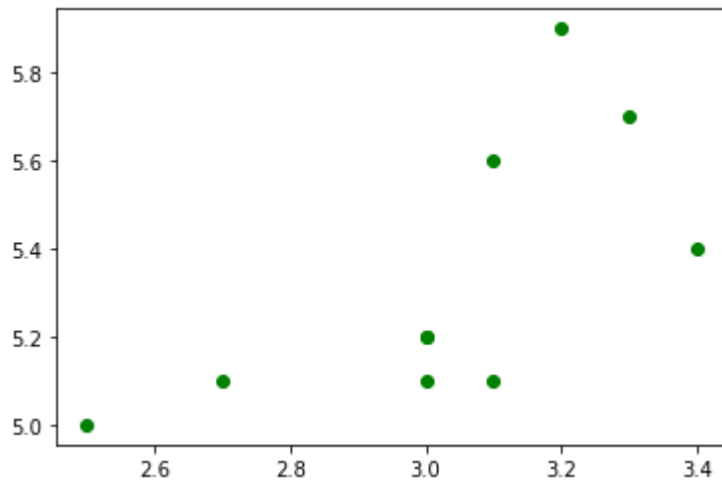
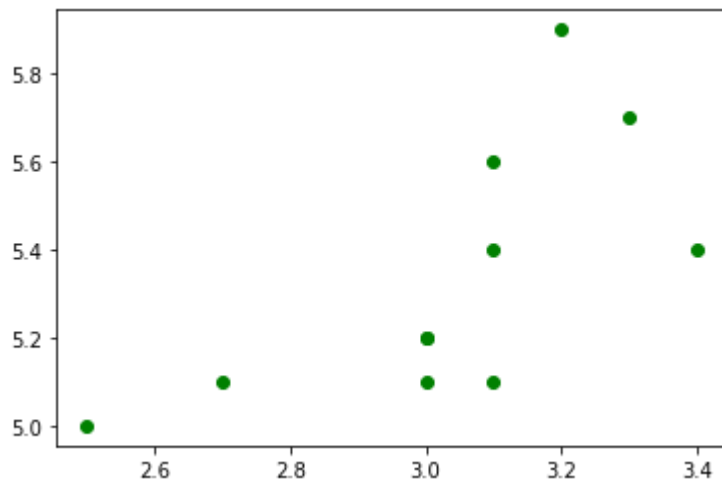
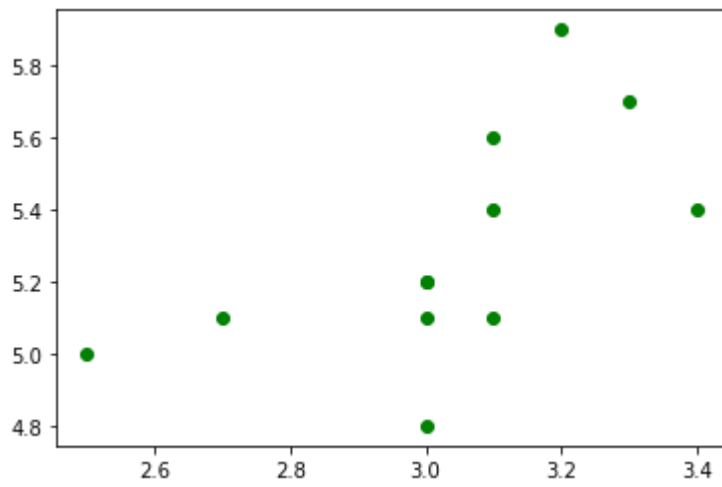


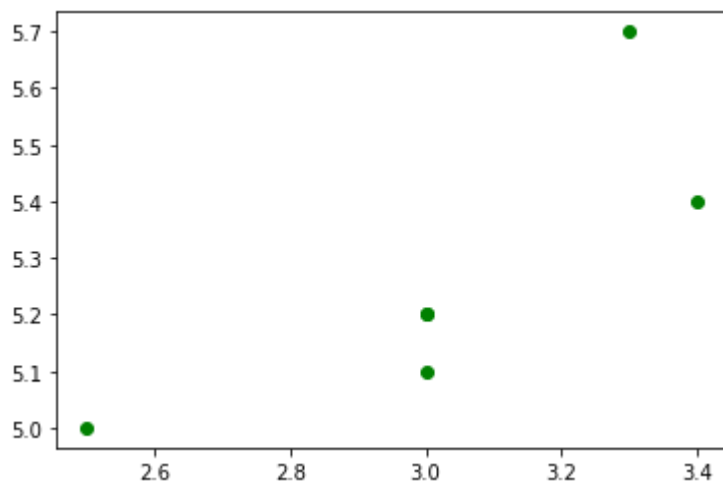
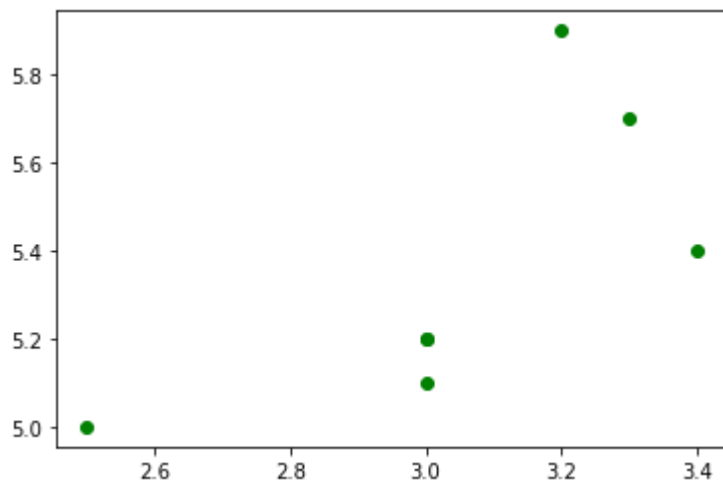
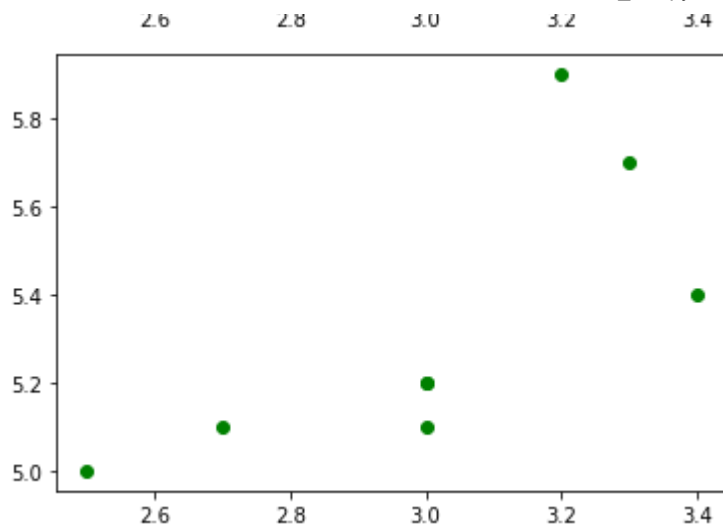


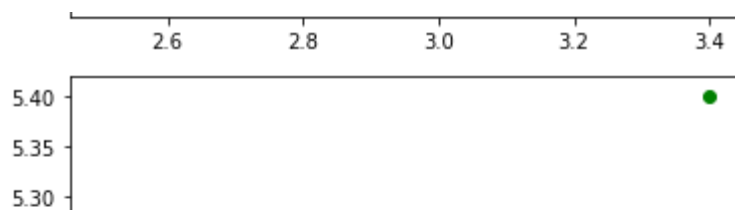






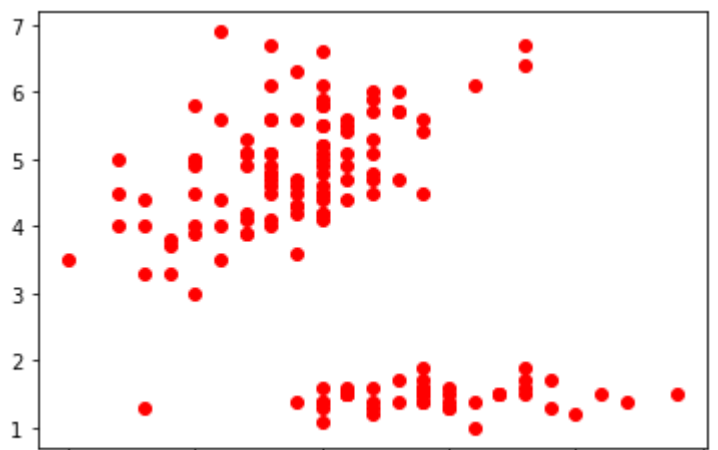
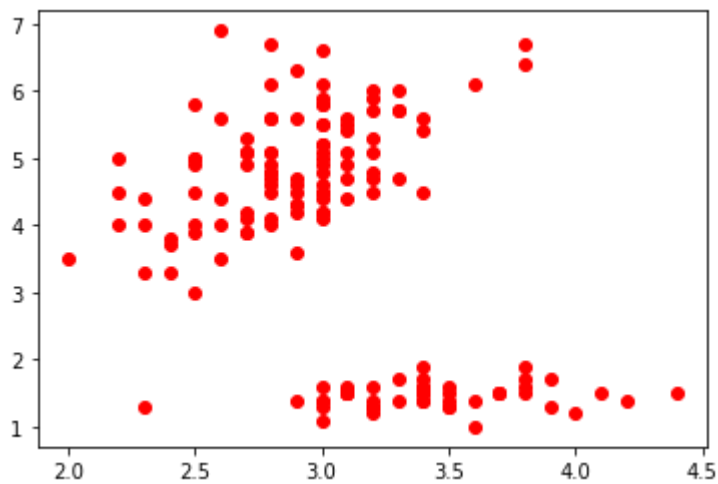
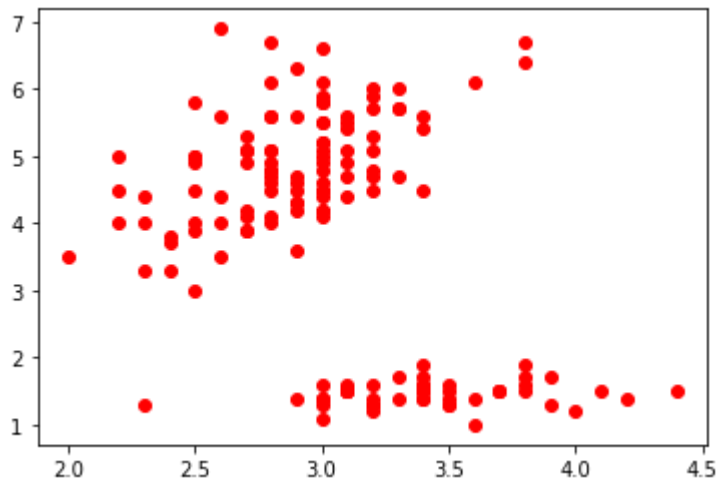
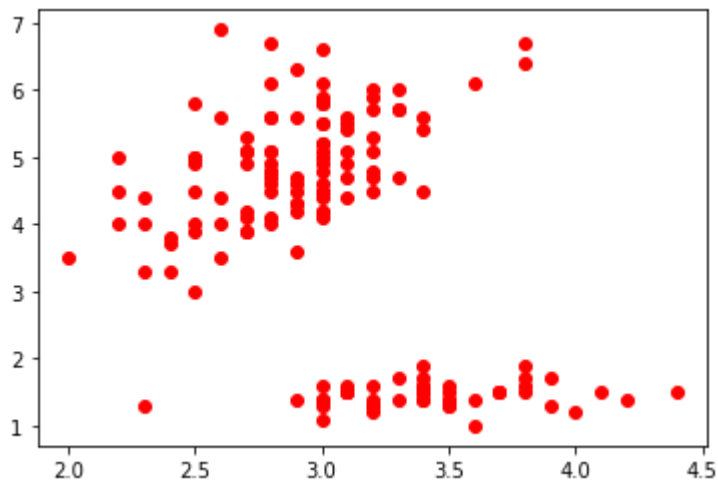


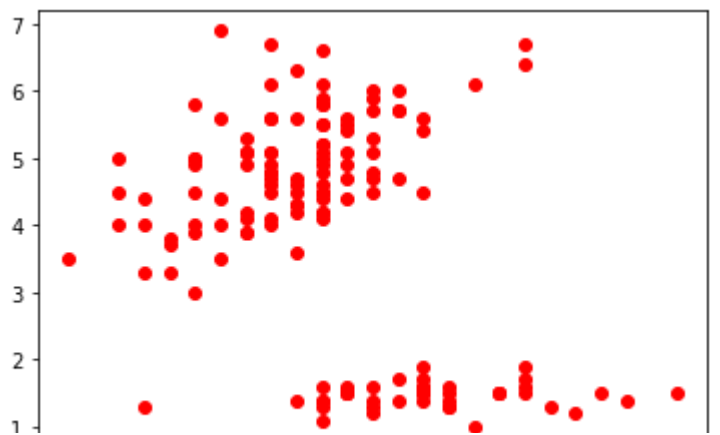
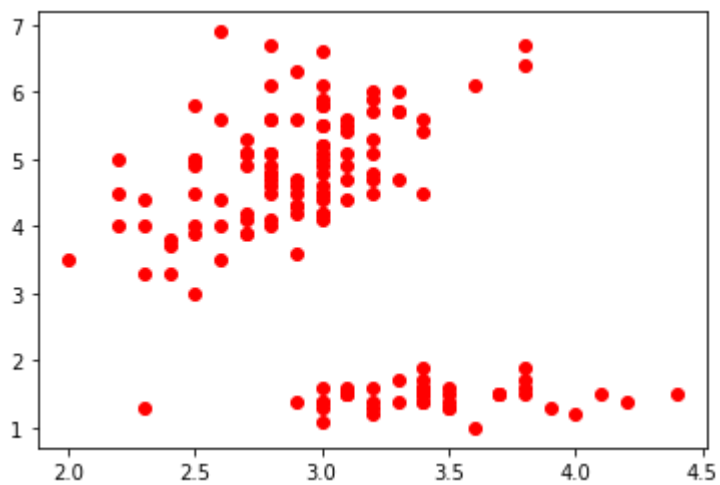
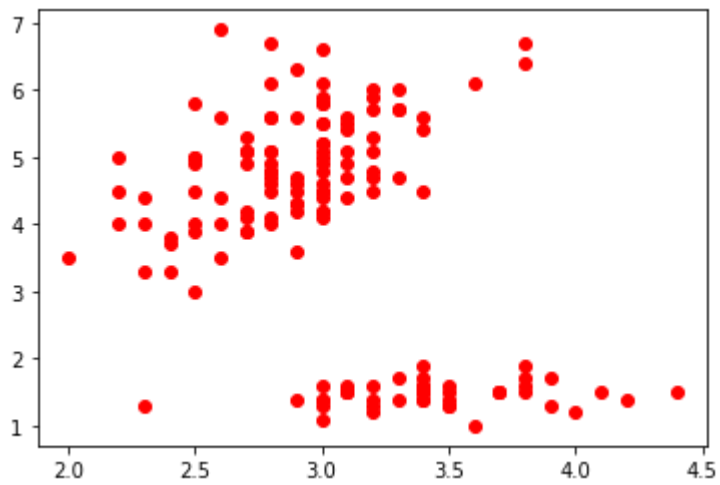
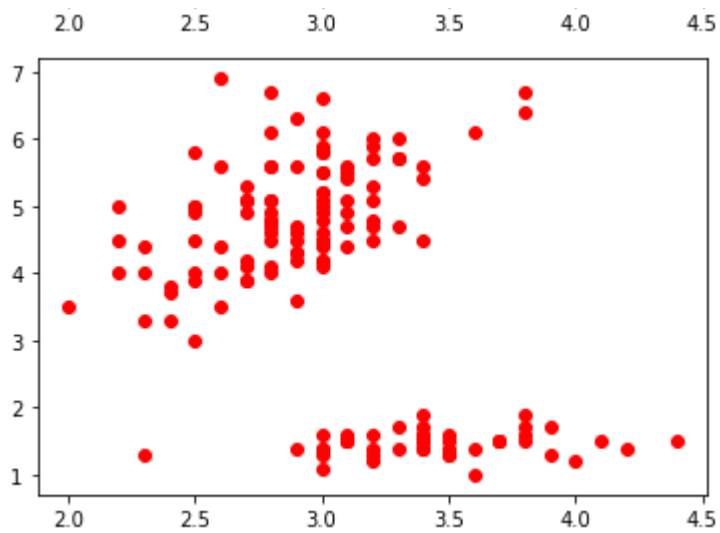


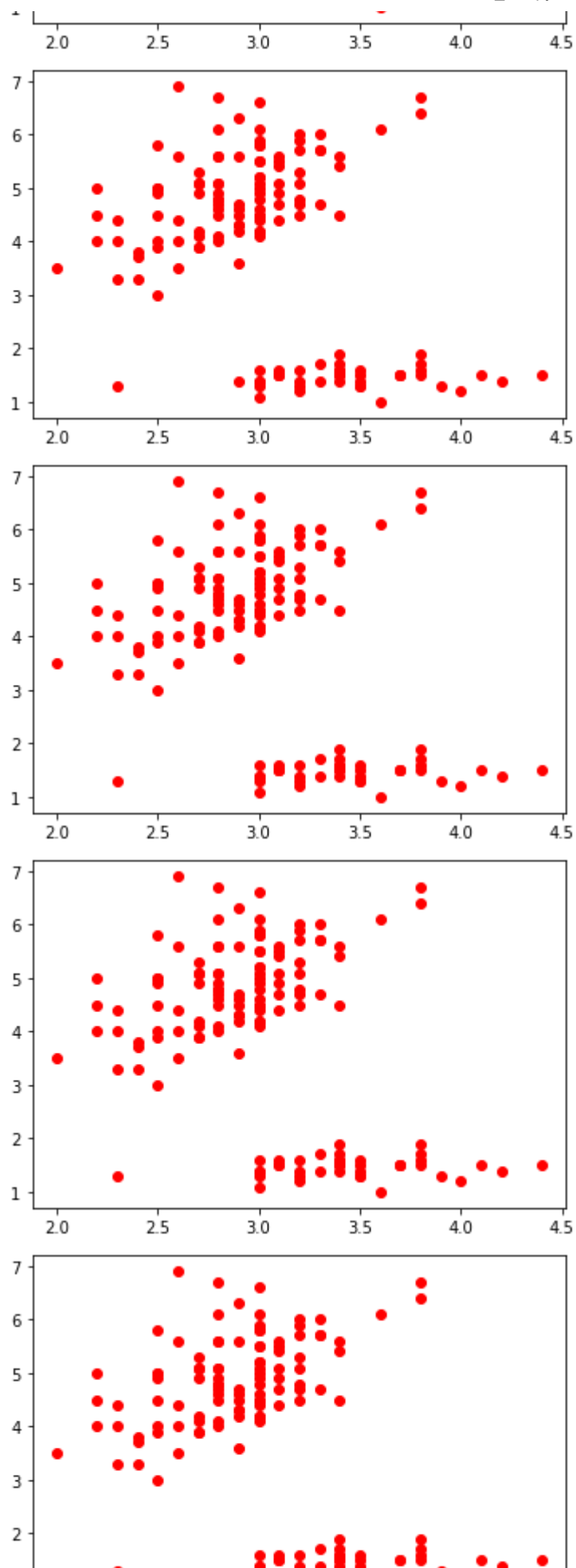


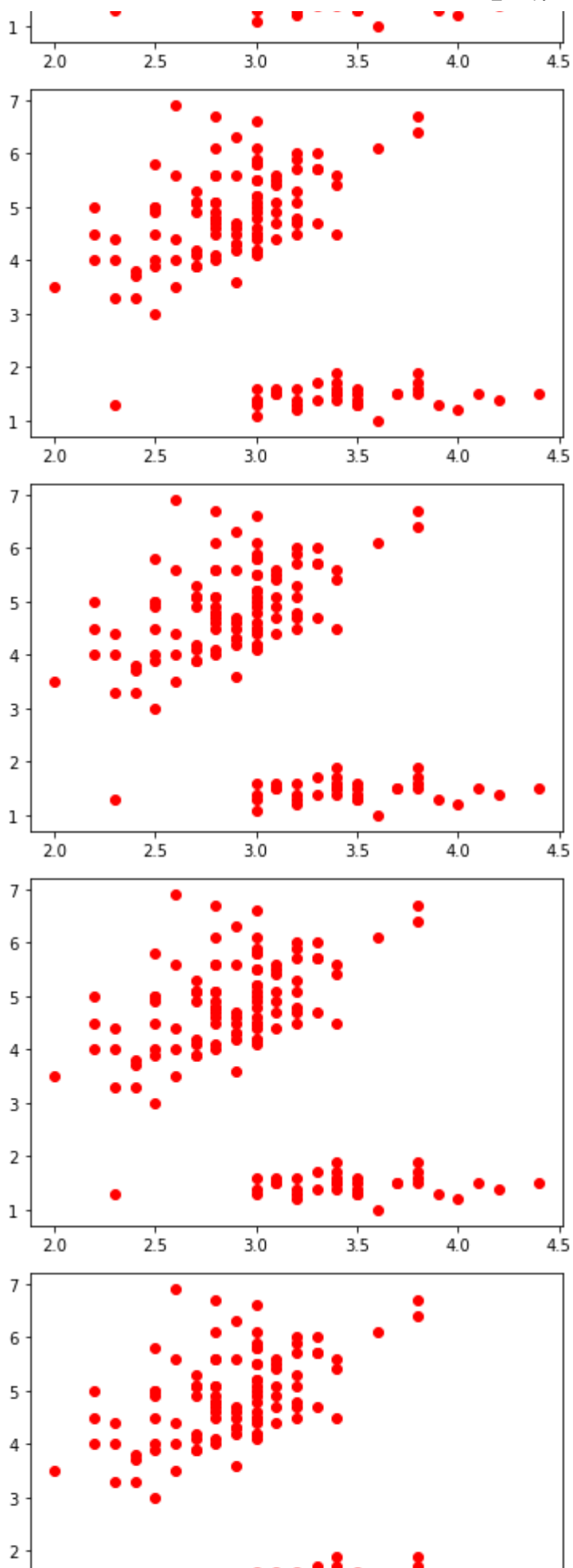
Taking Sepal Width and Petal Length as our two features for clustering

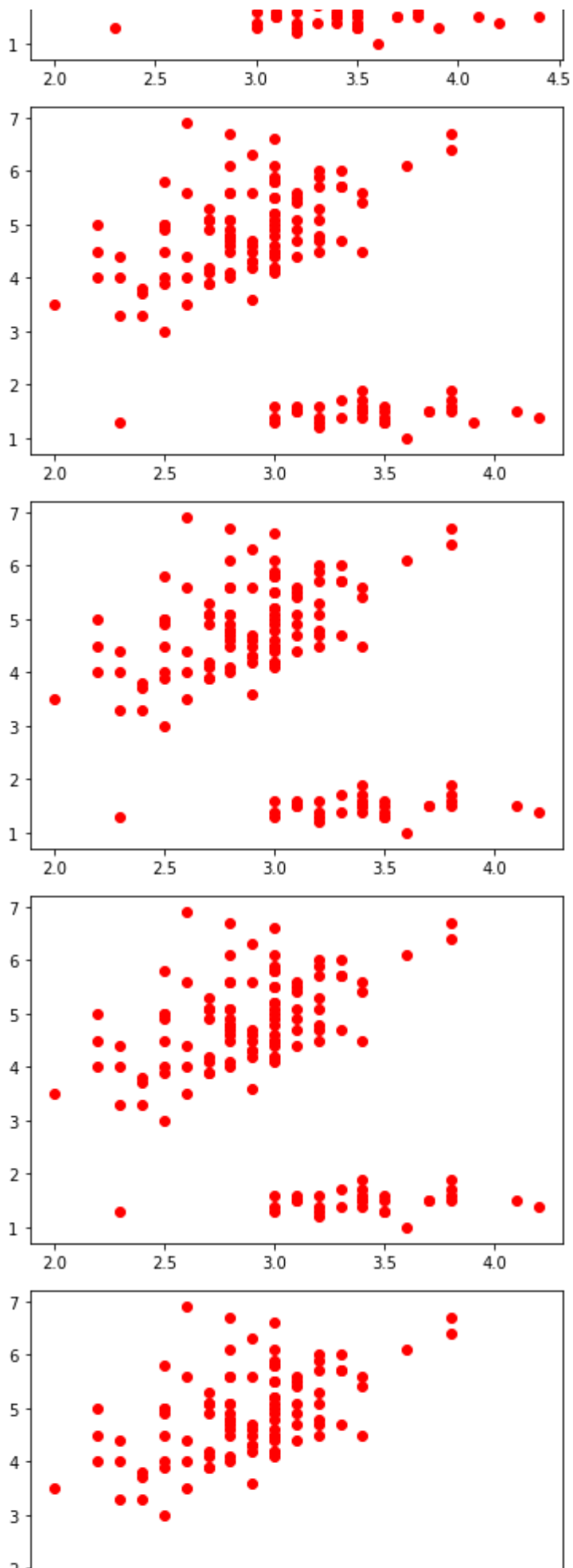
```
for i in range(150):  
    if i<=49:  
        plt.plot(iris.values[i:,1],iris.values[i:,2], 'ro')  
    if i>49 and i<=99:  
        plt.plot(iris.values[i:,1],iris.values[i:,2], 'bo')  
    if i>99:  
        plt.plot(iris.values[i:,1],iris.values[i:,2], 'go')  
plt.show()
```

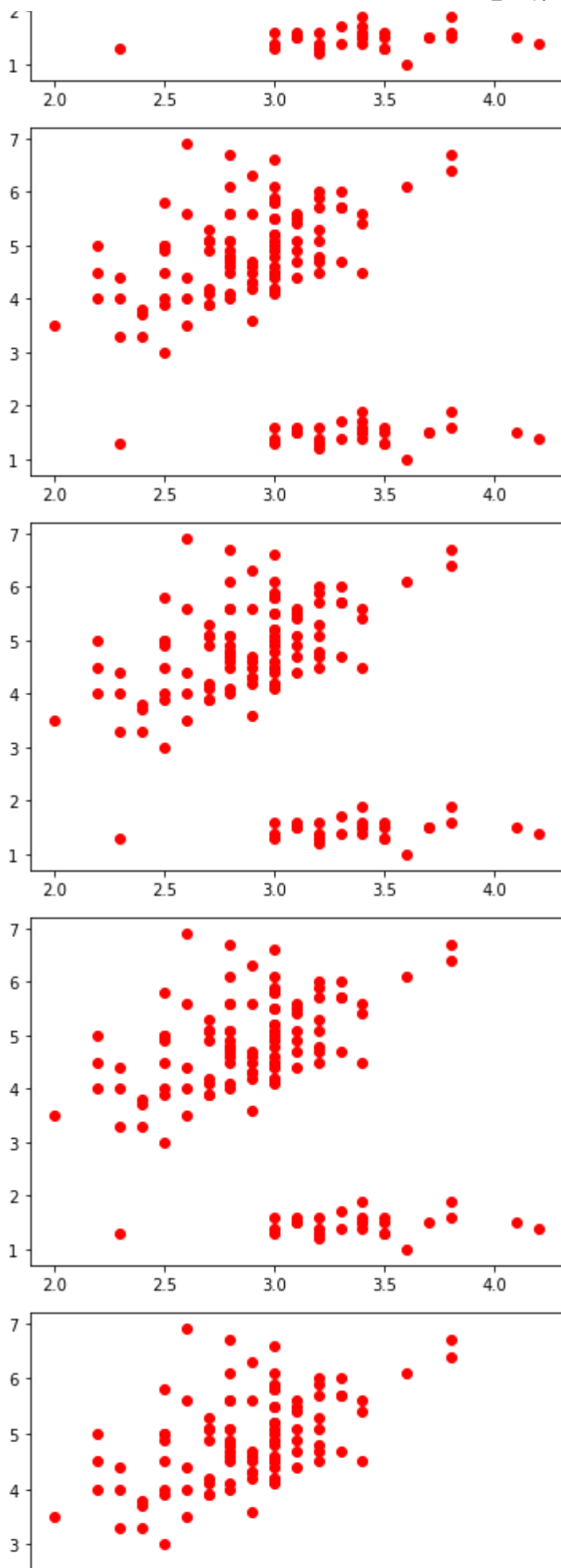


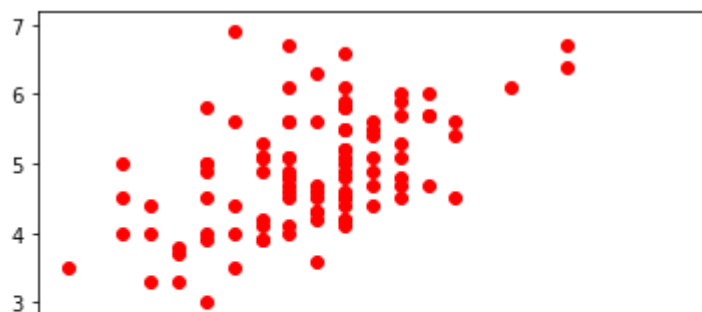
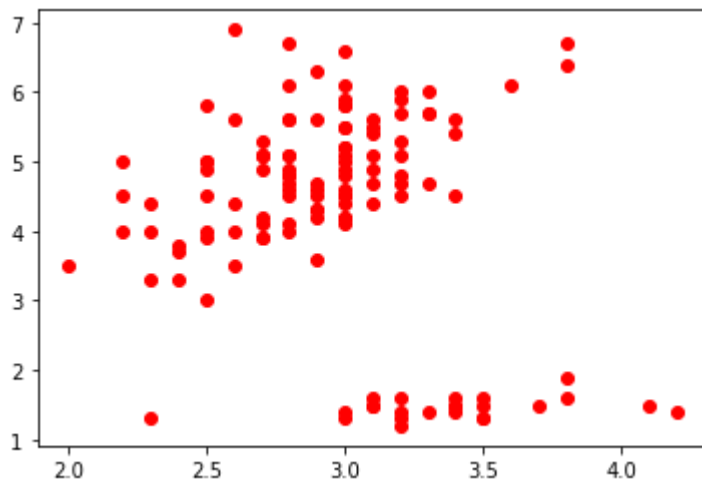
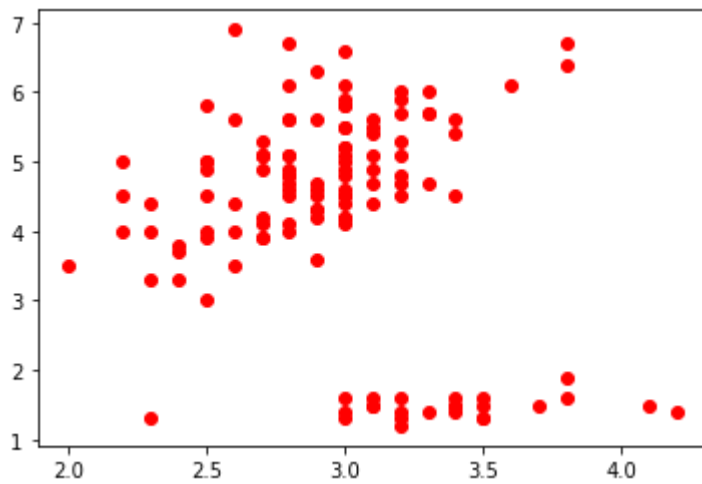
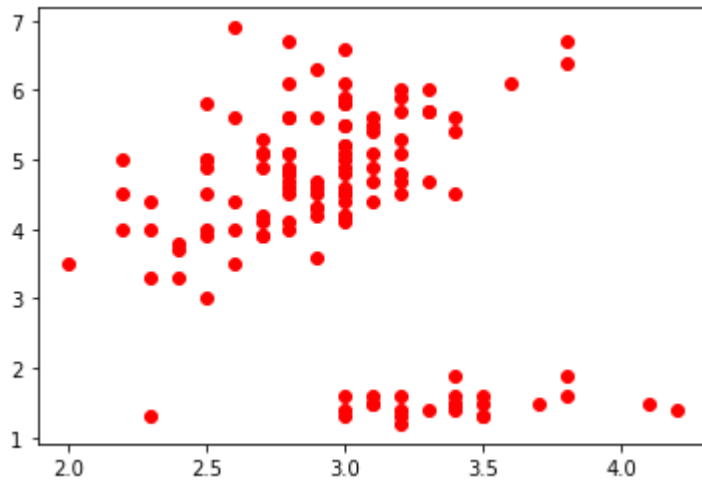
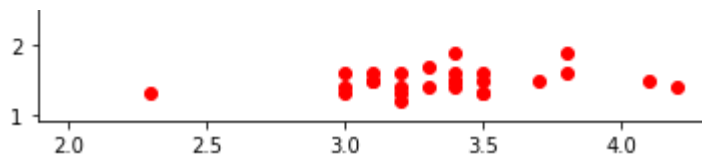


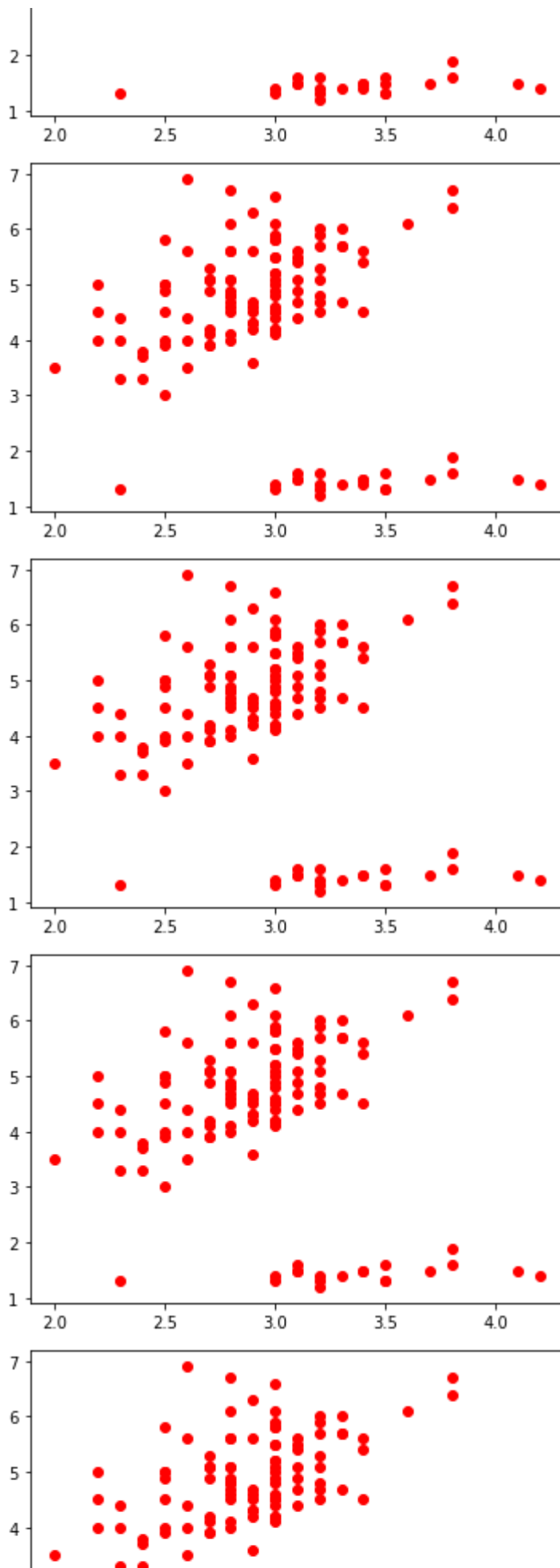


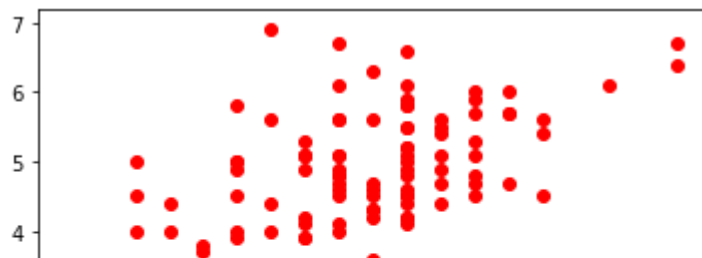
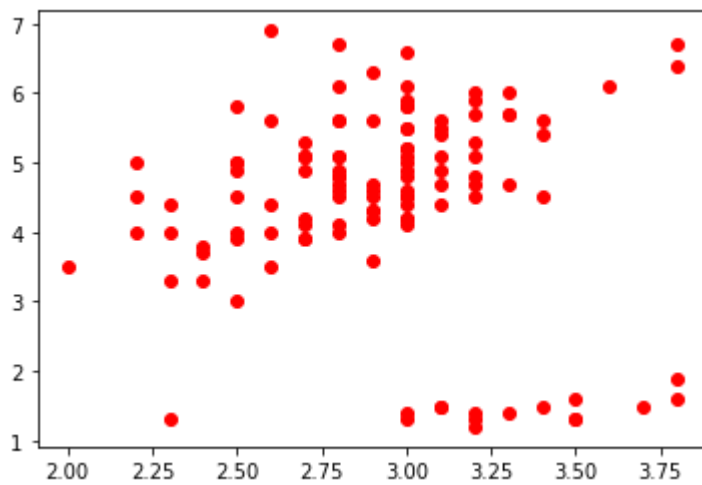
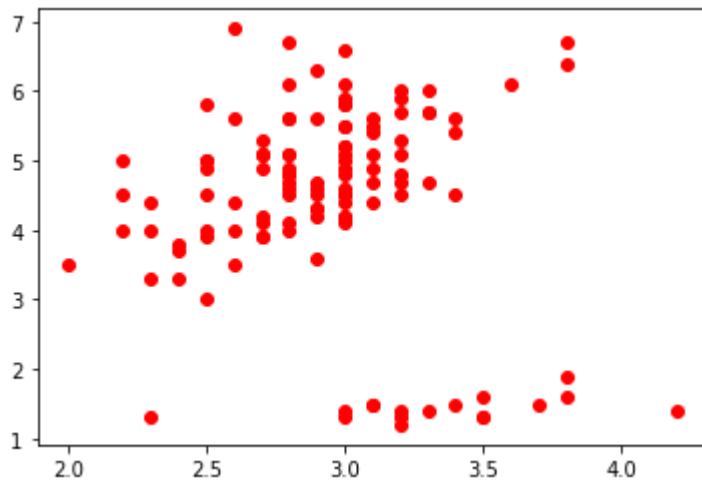
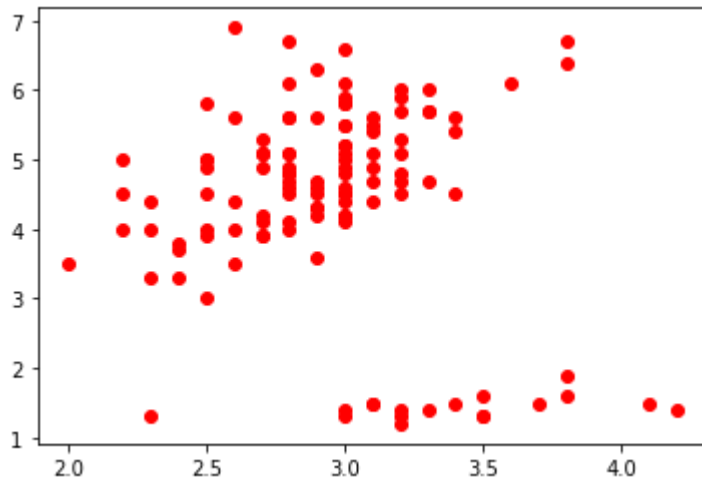
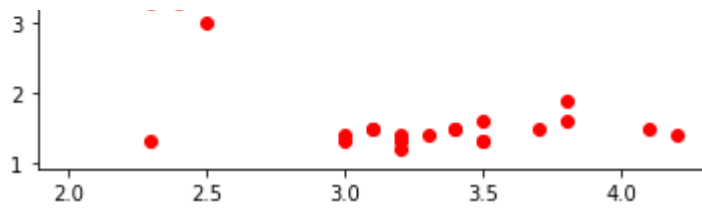


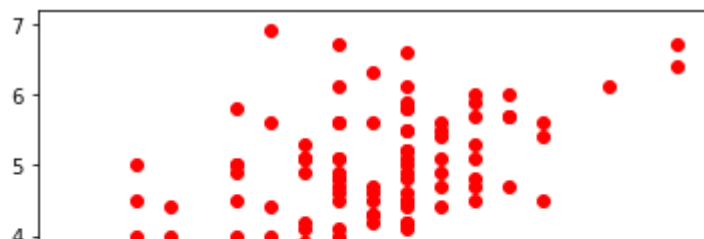
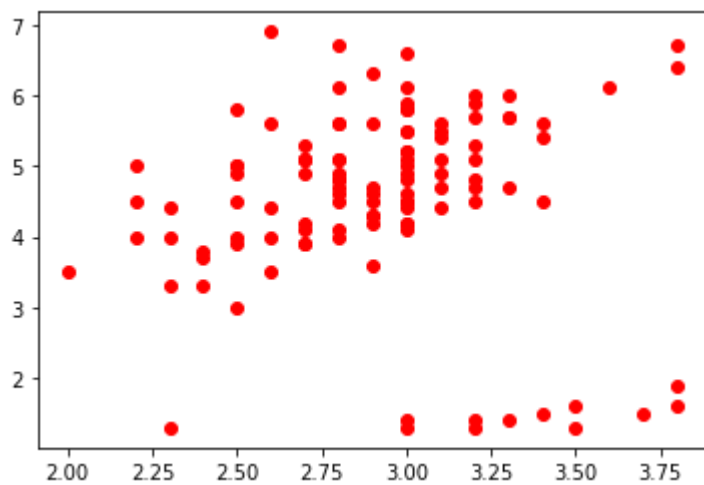
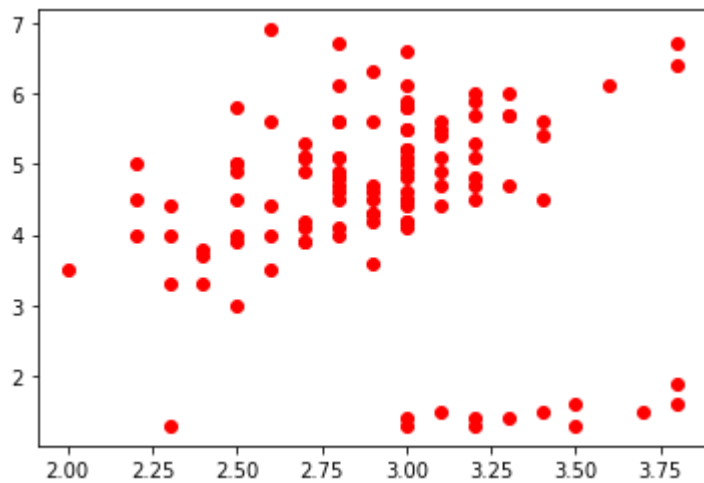
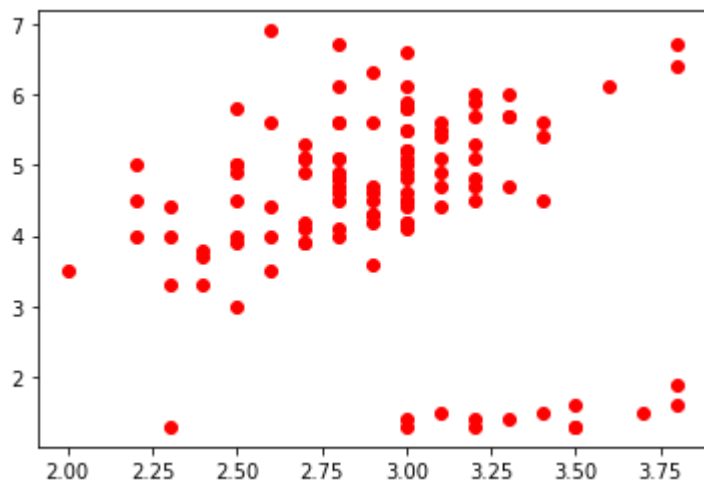
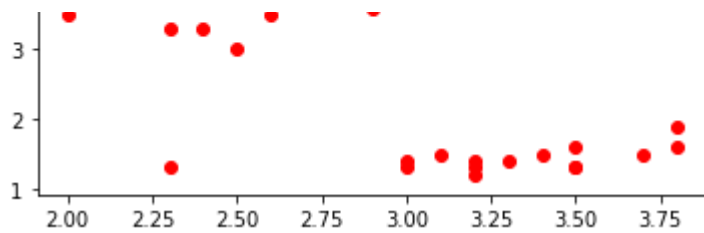


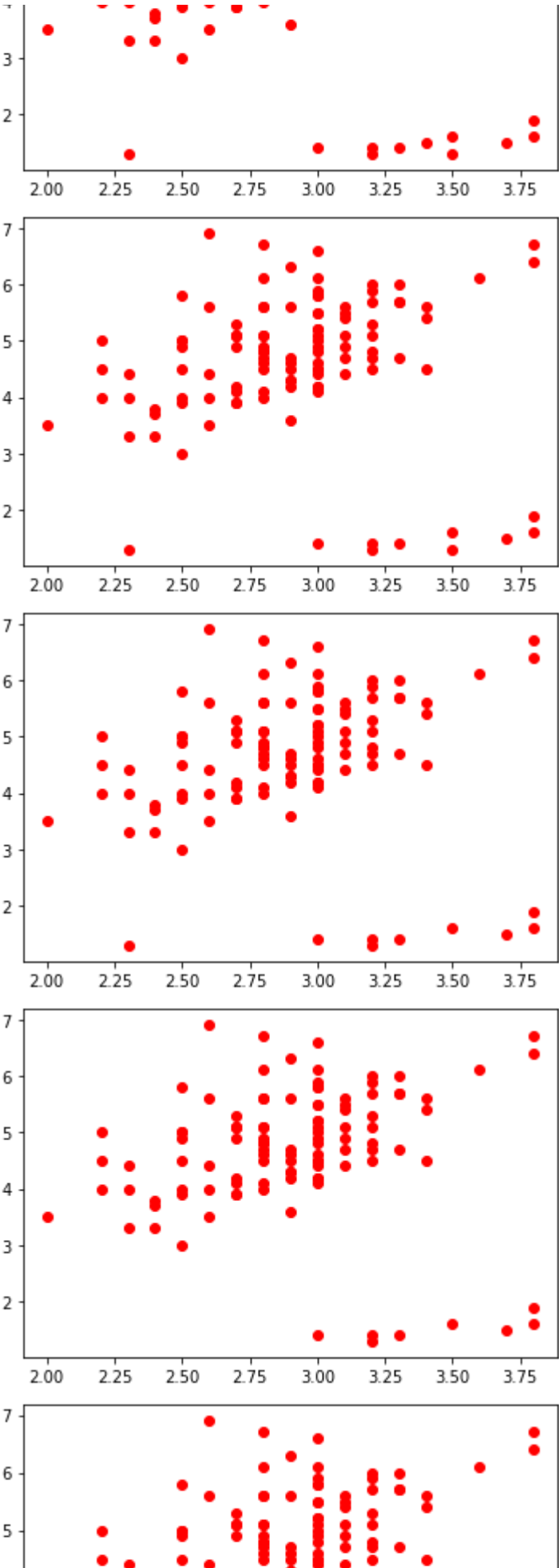


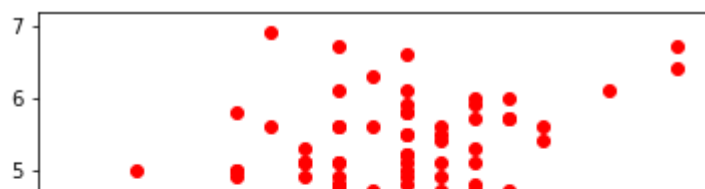
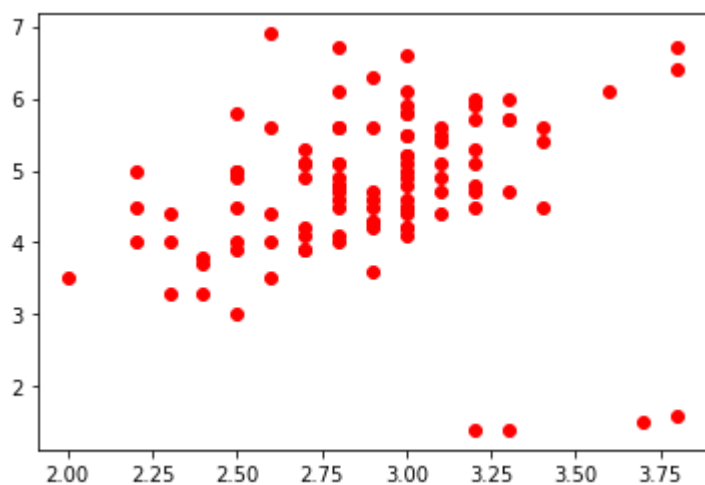
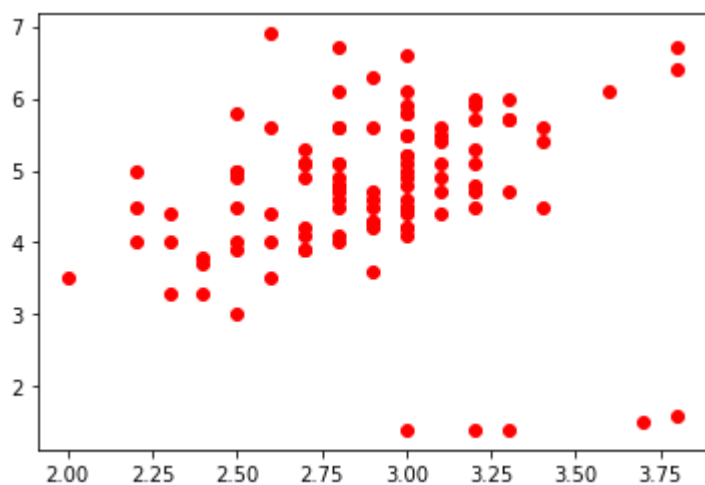
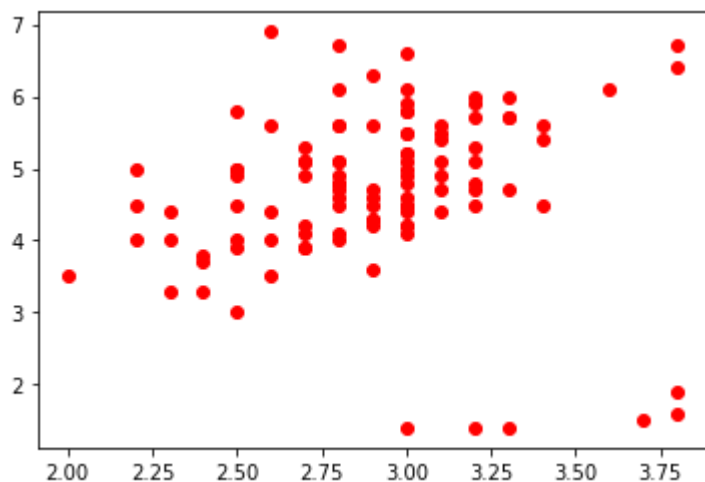
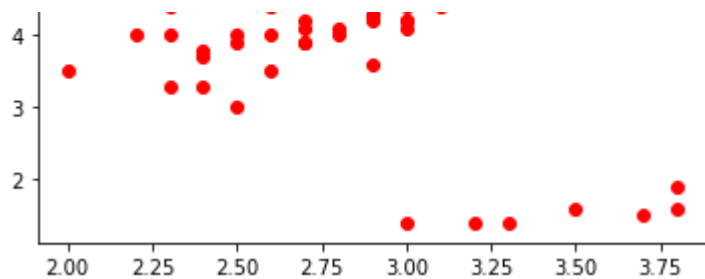


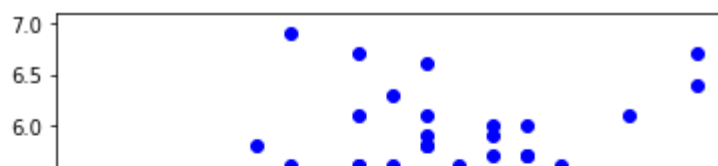
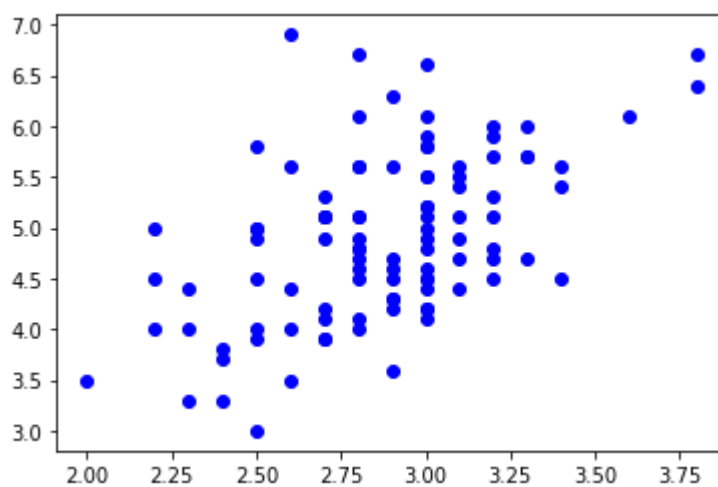
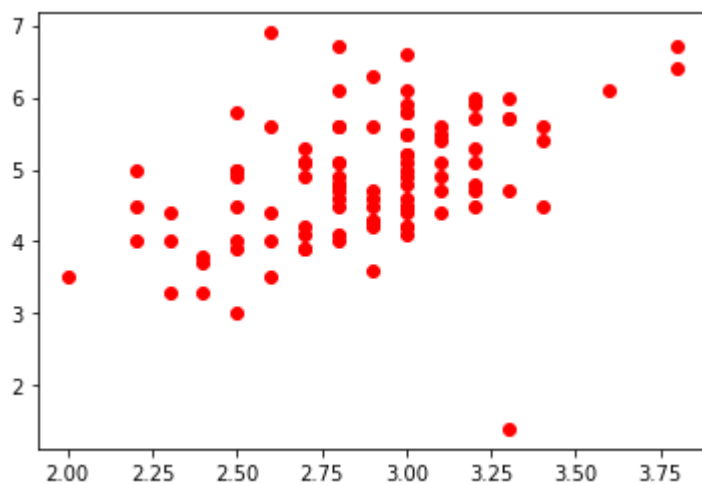
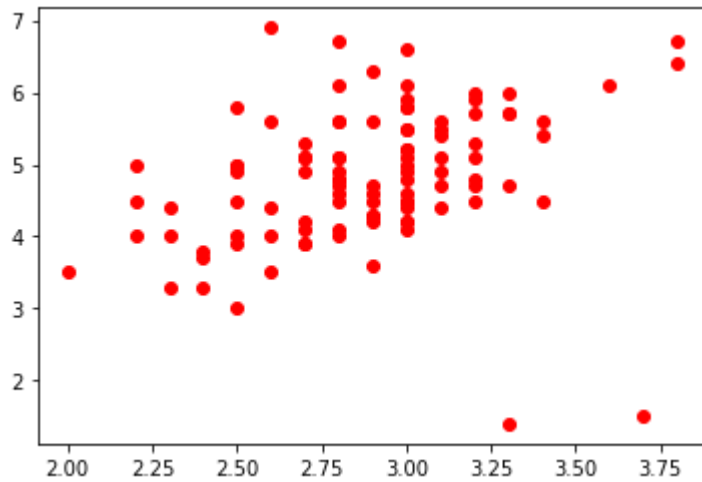
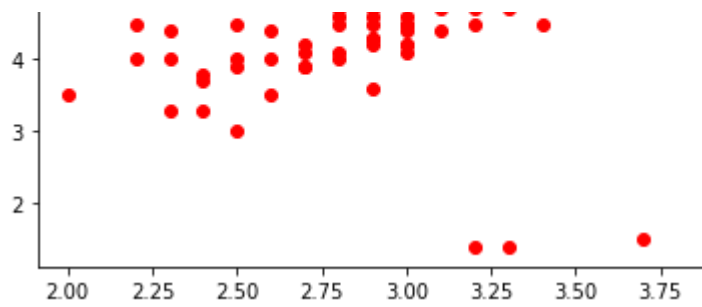


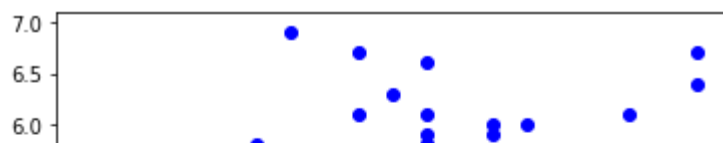
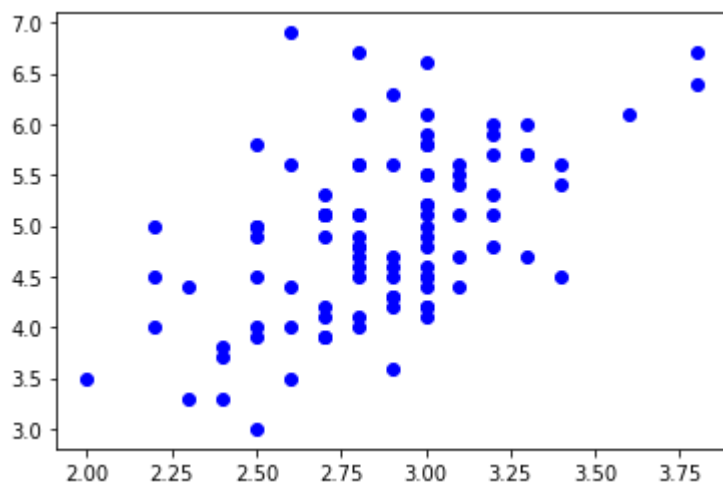
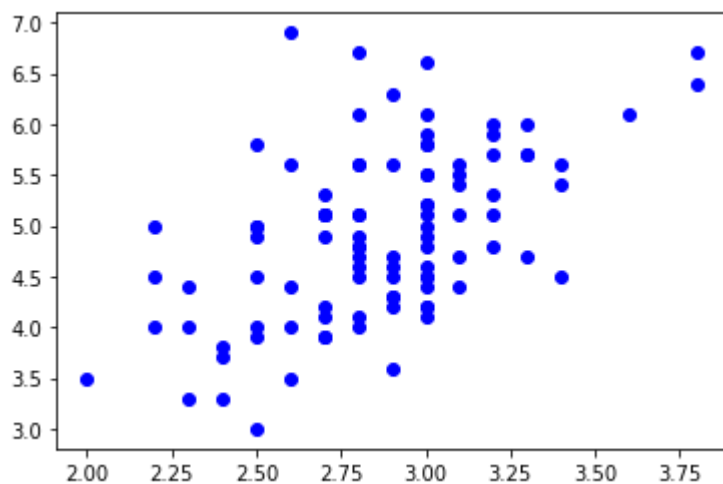
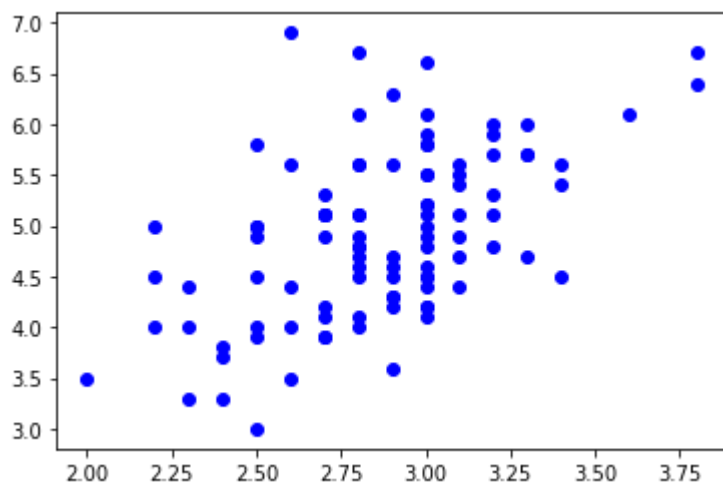
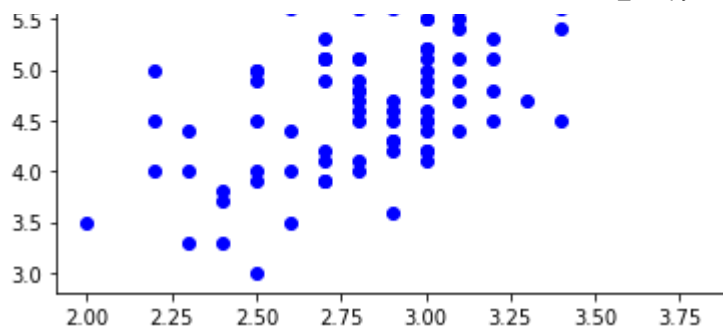


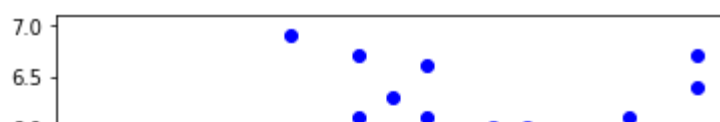
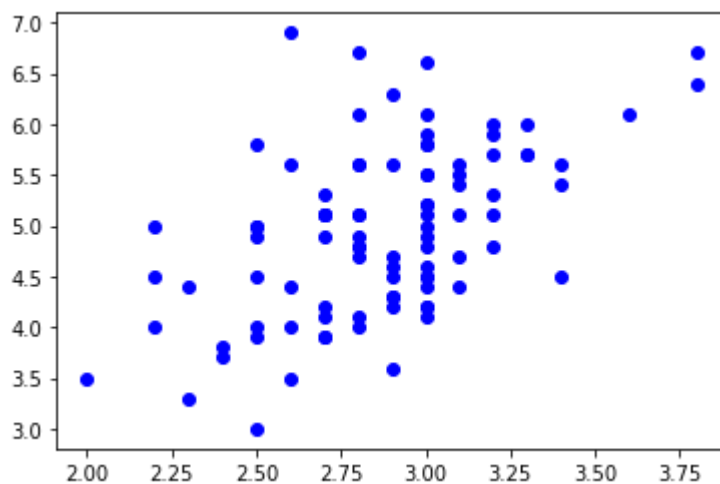
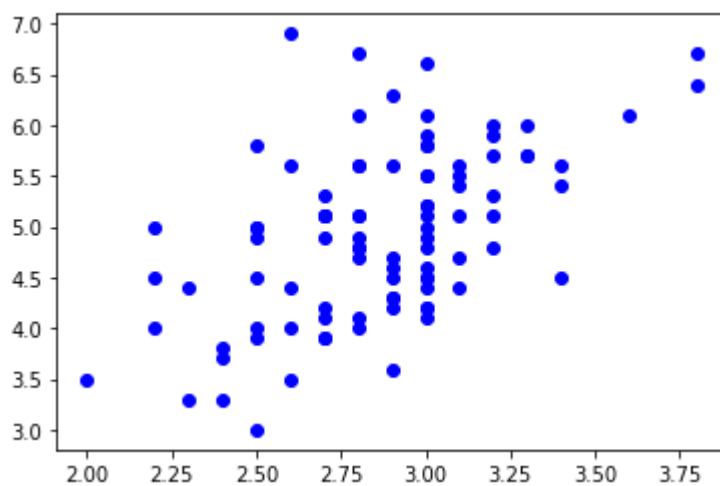
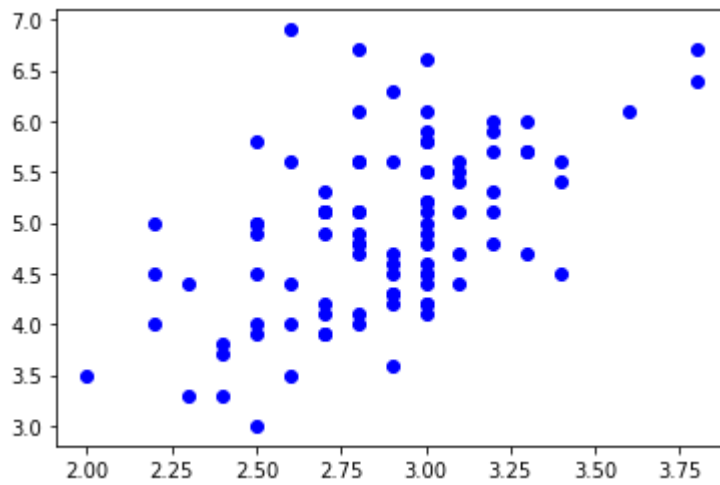
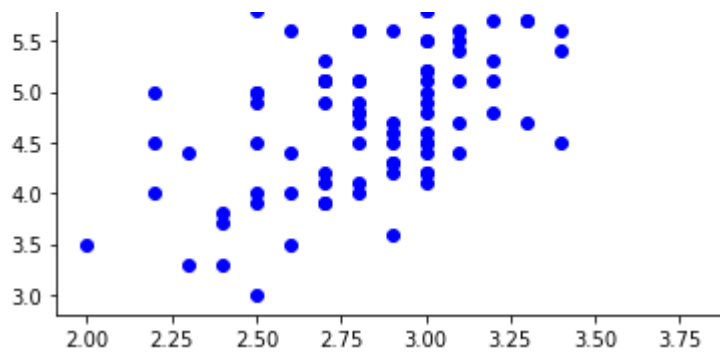


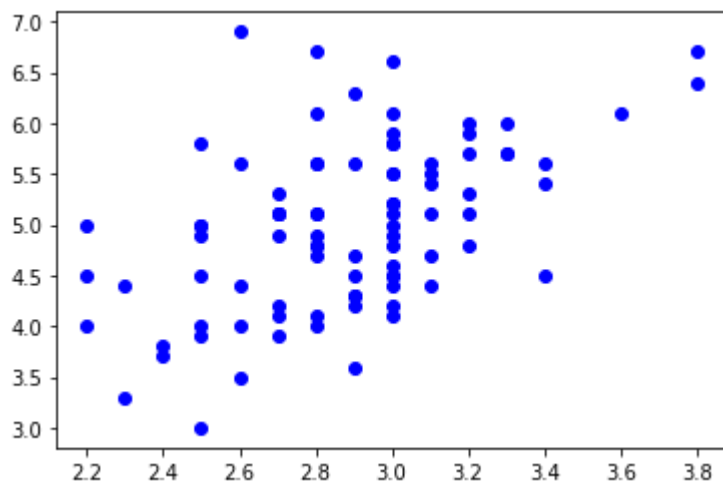
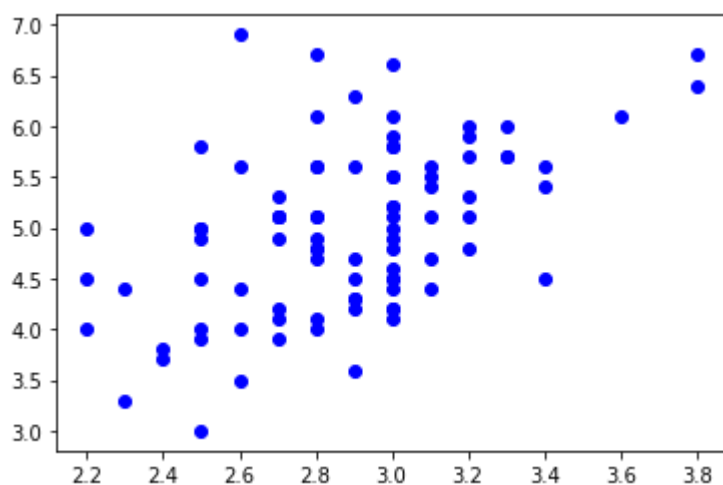
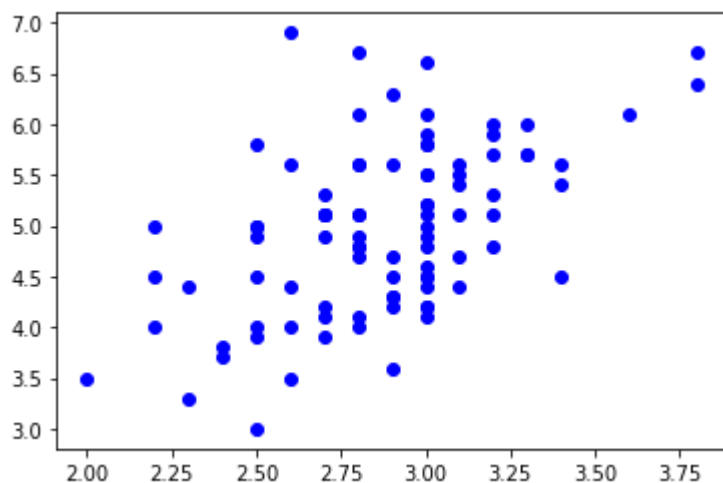
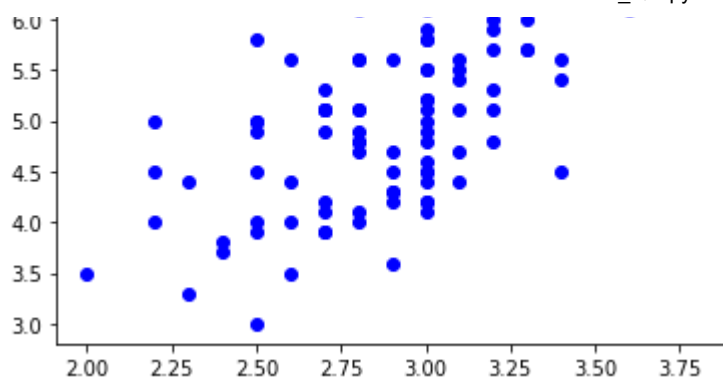


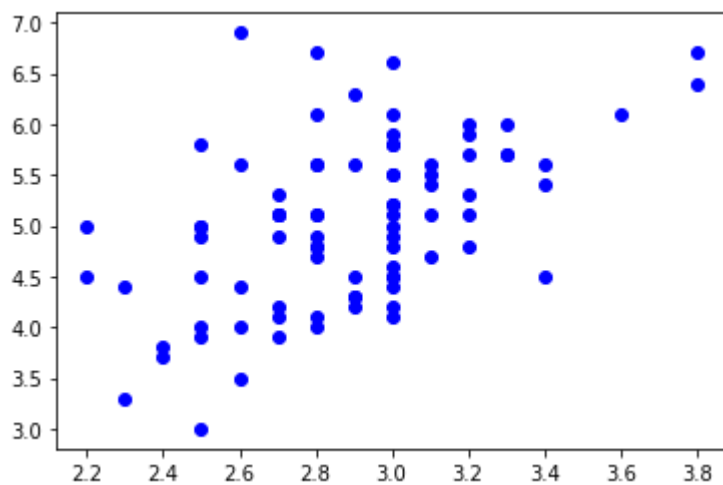
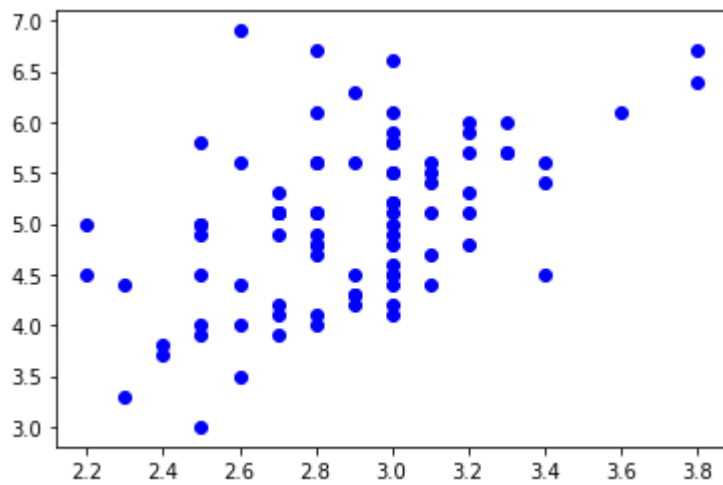
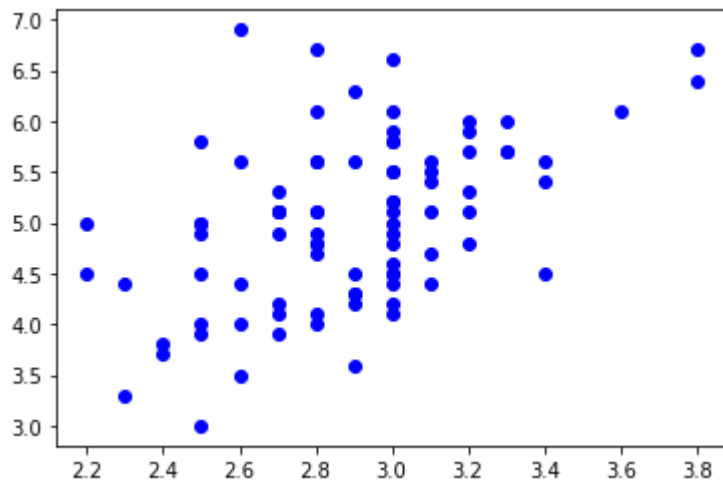
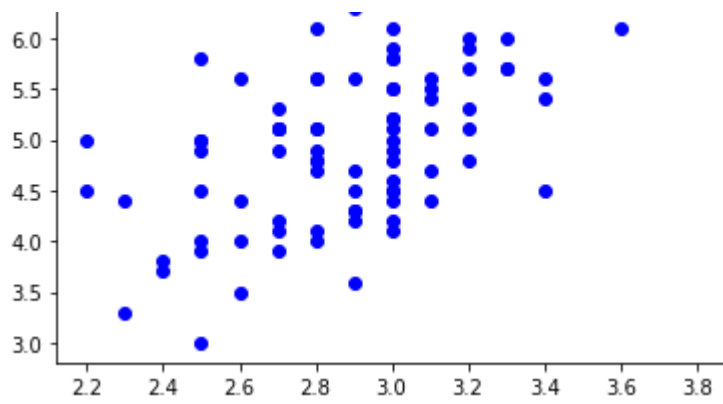


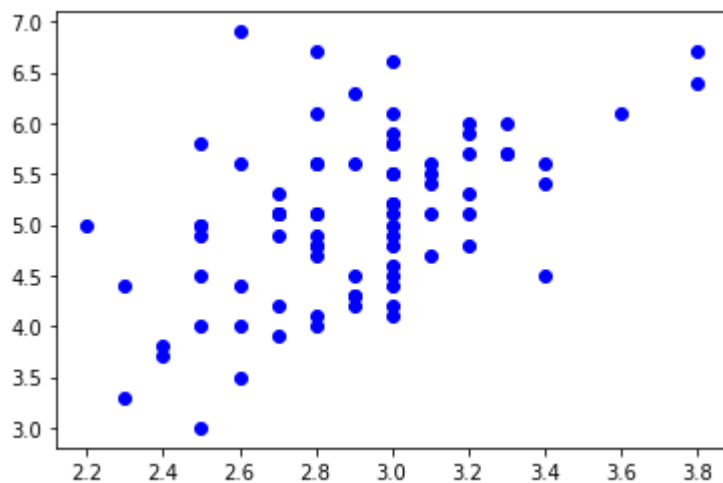
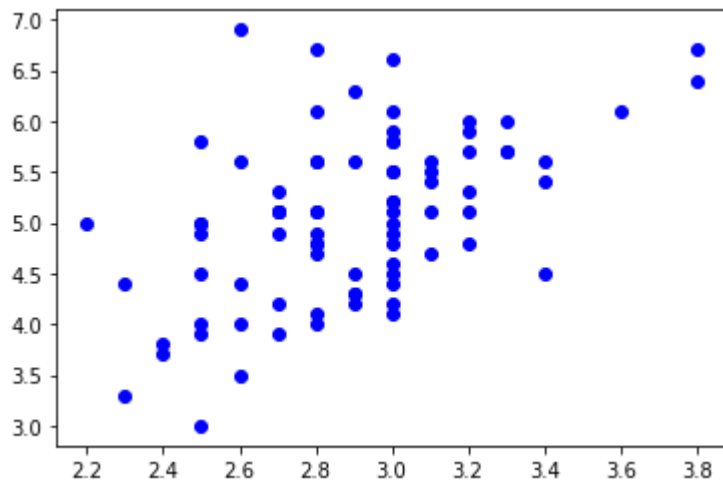
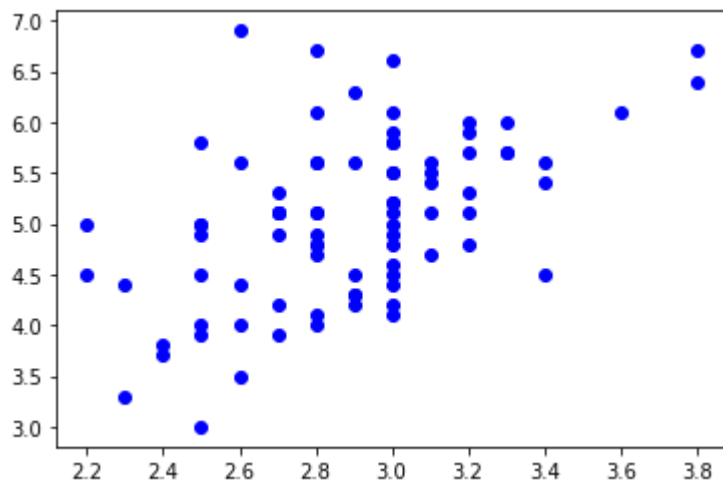
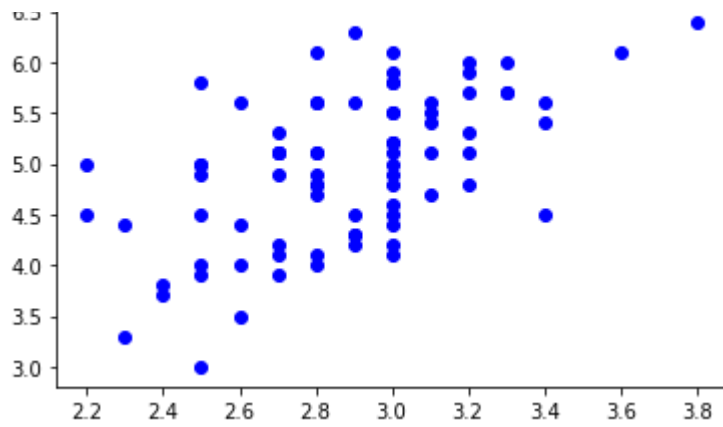


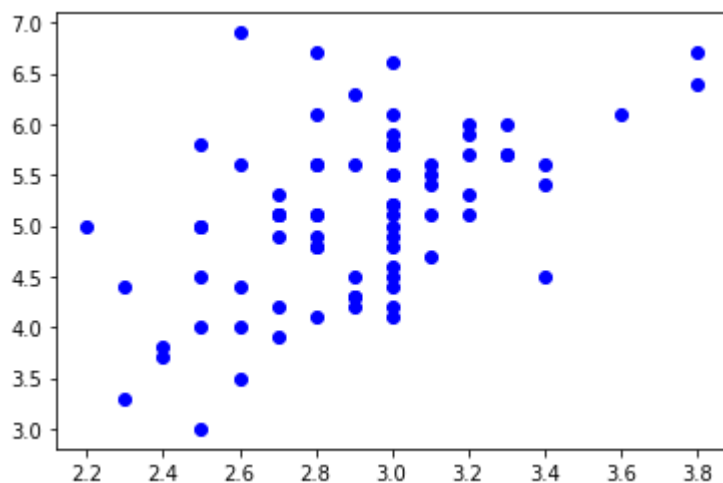
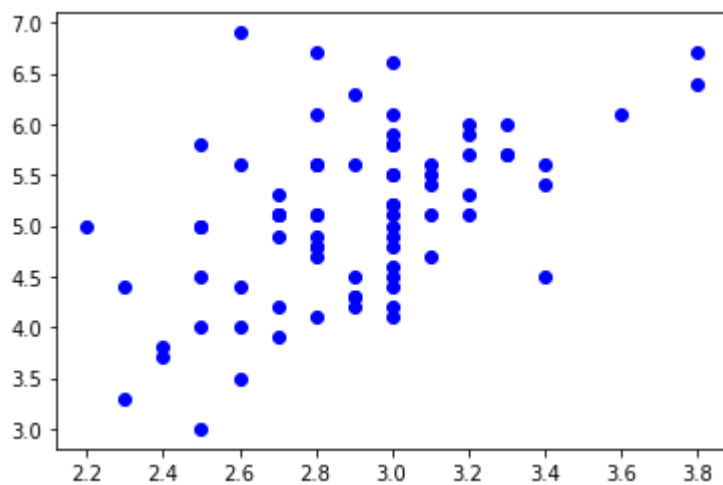
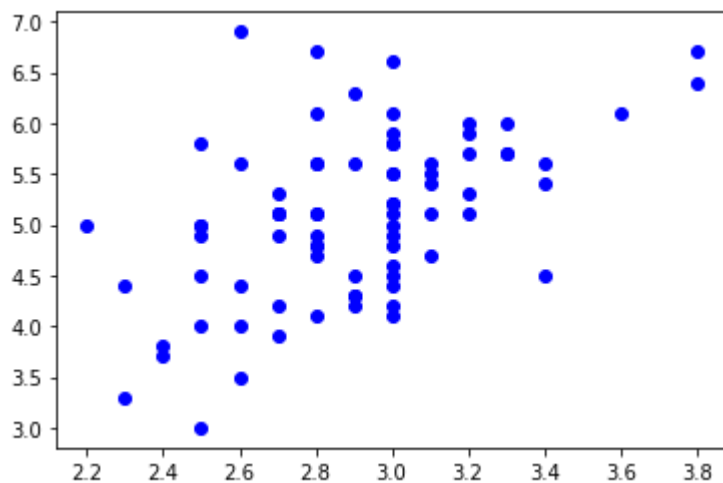
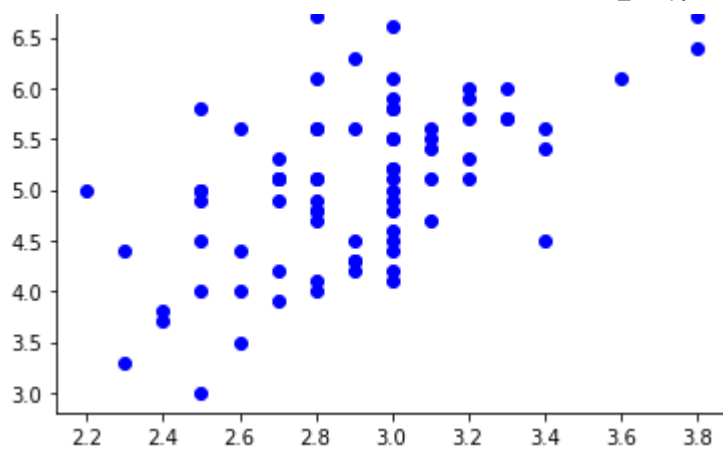


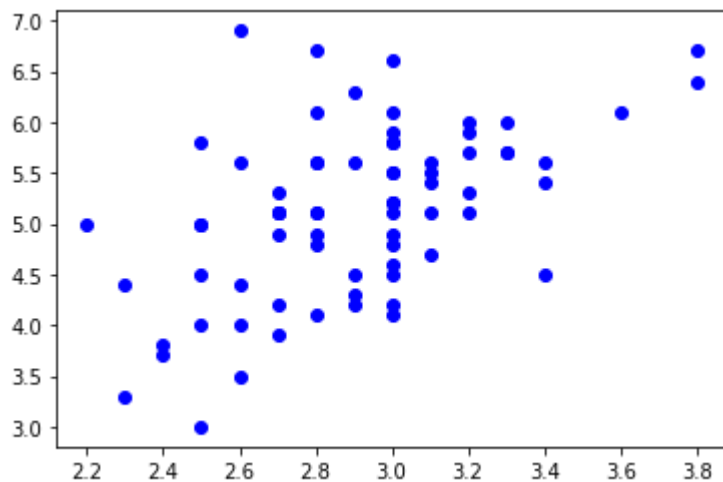
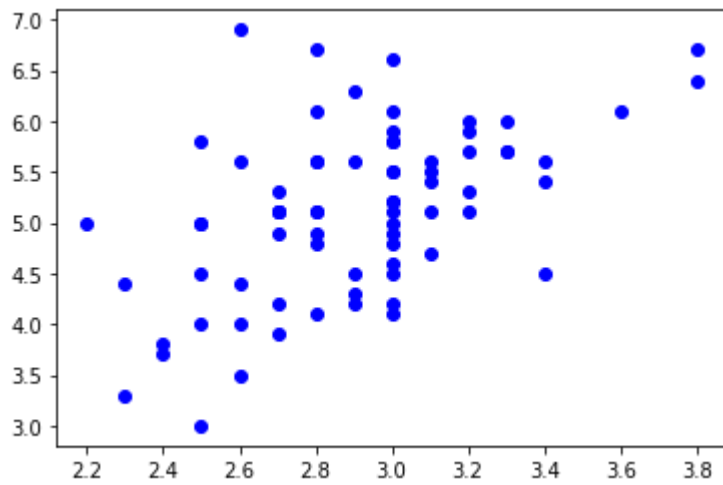
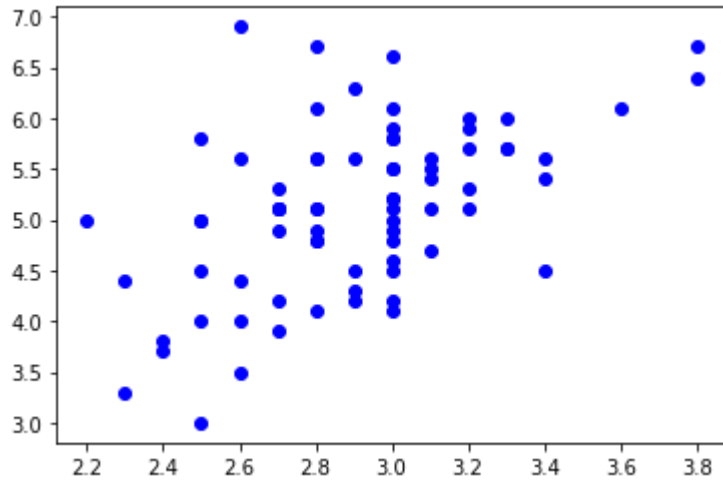
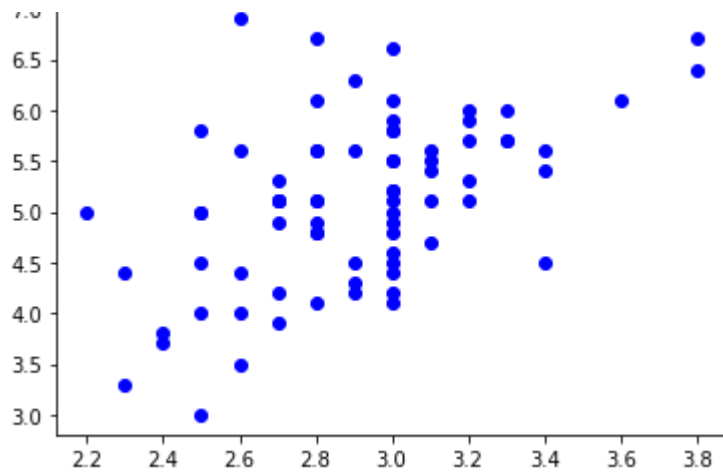


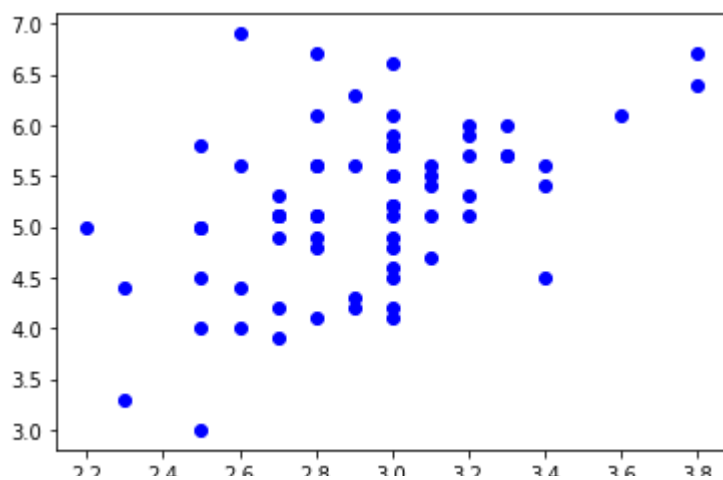
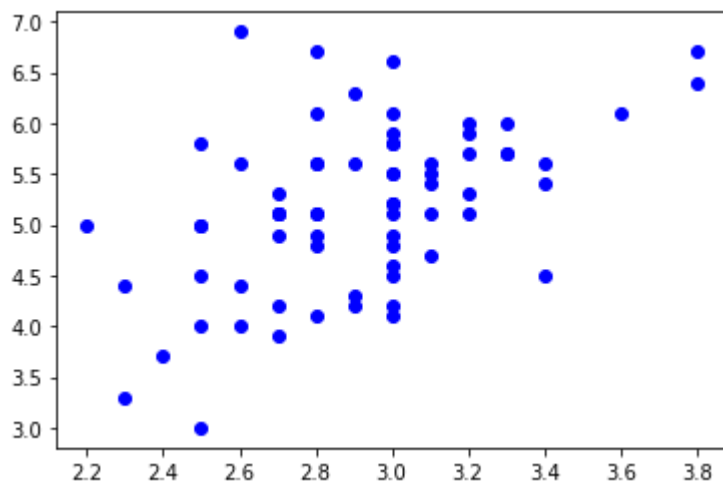
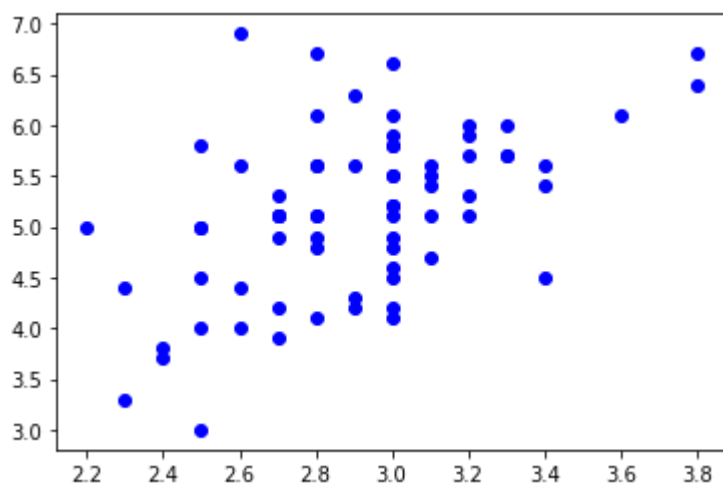
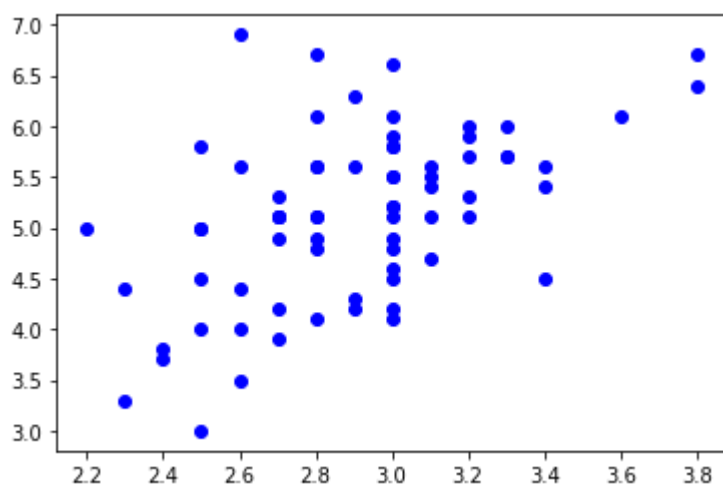


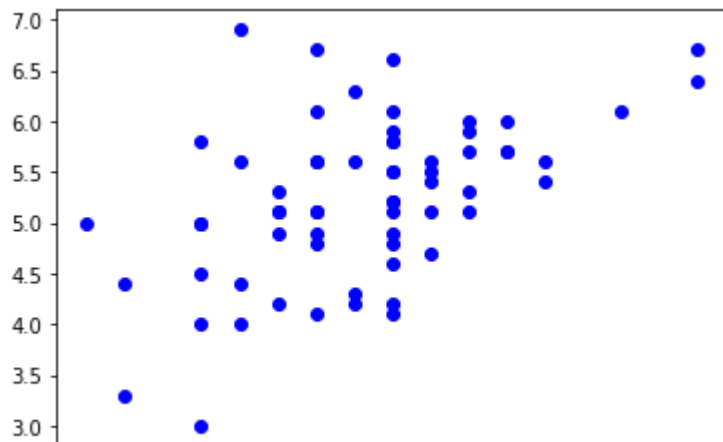
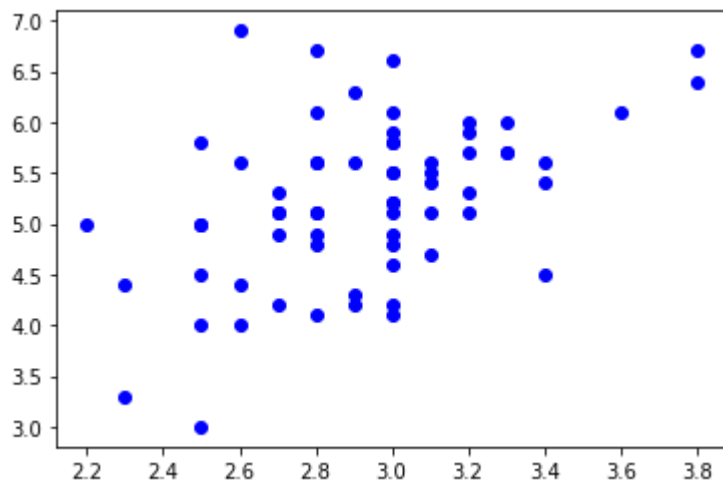
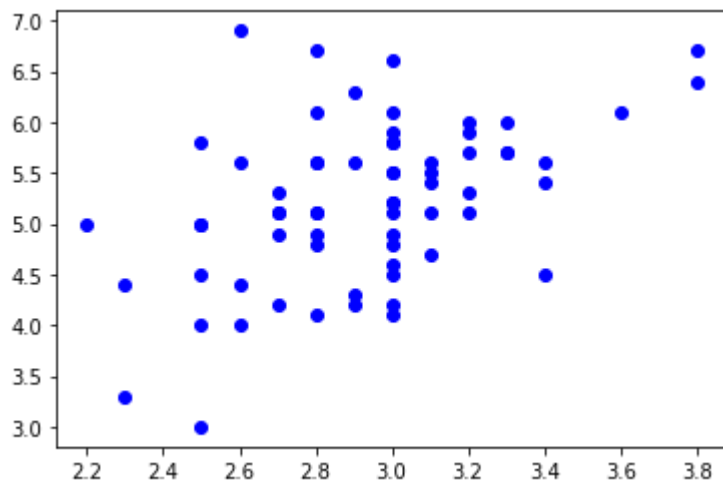
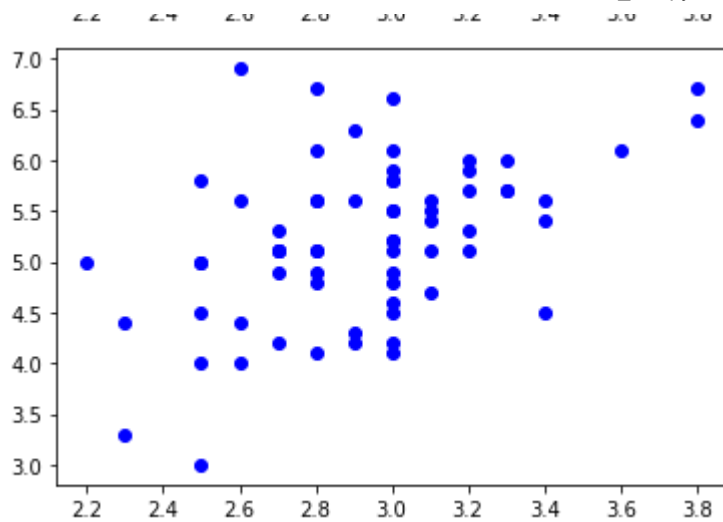


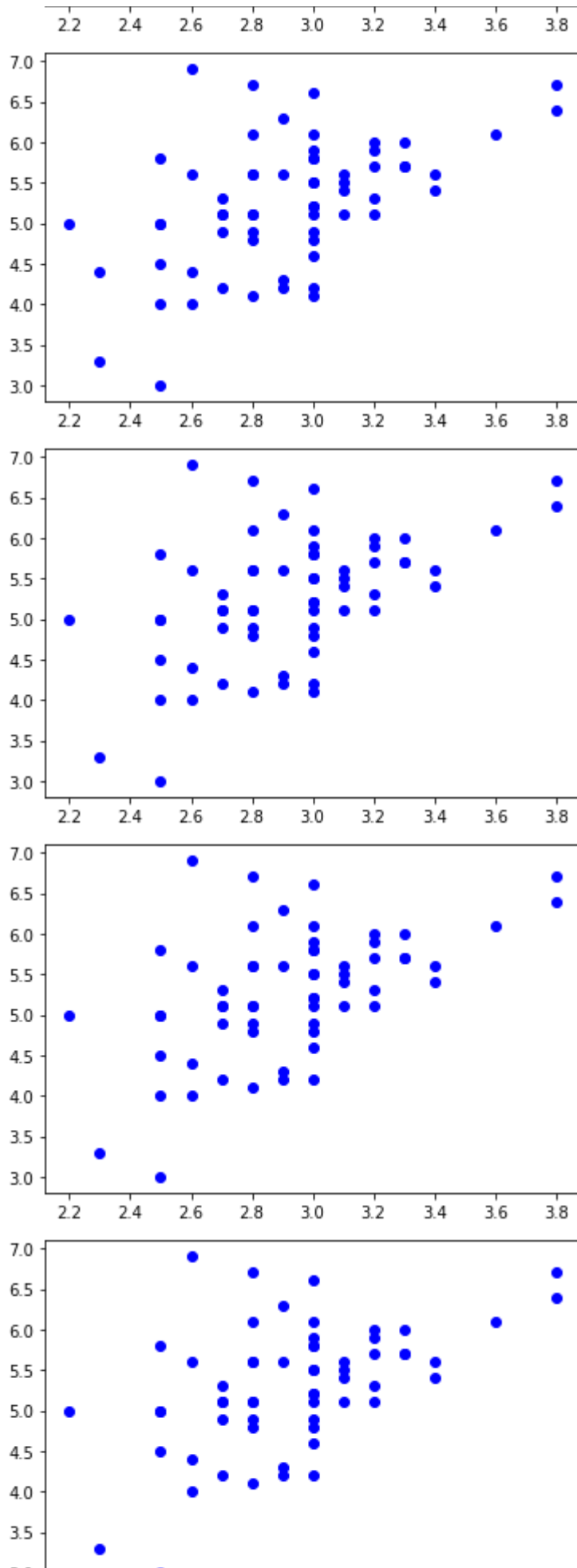


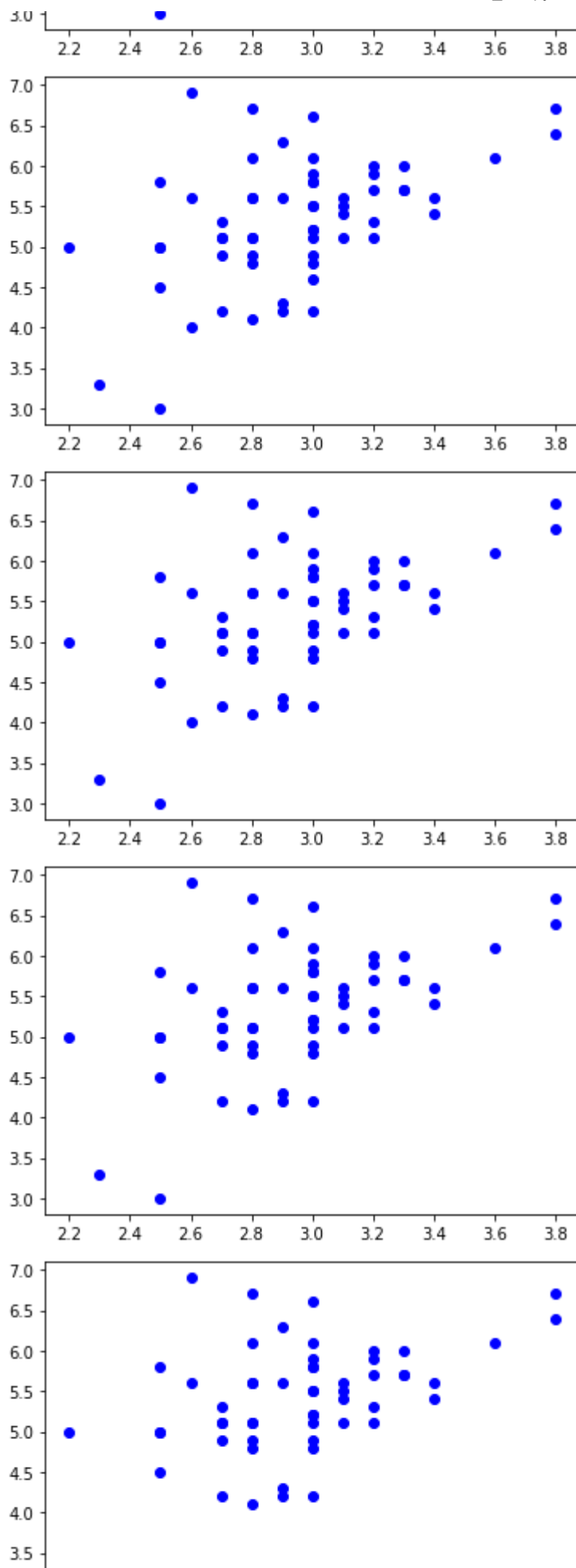


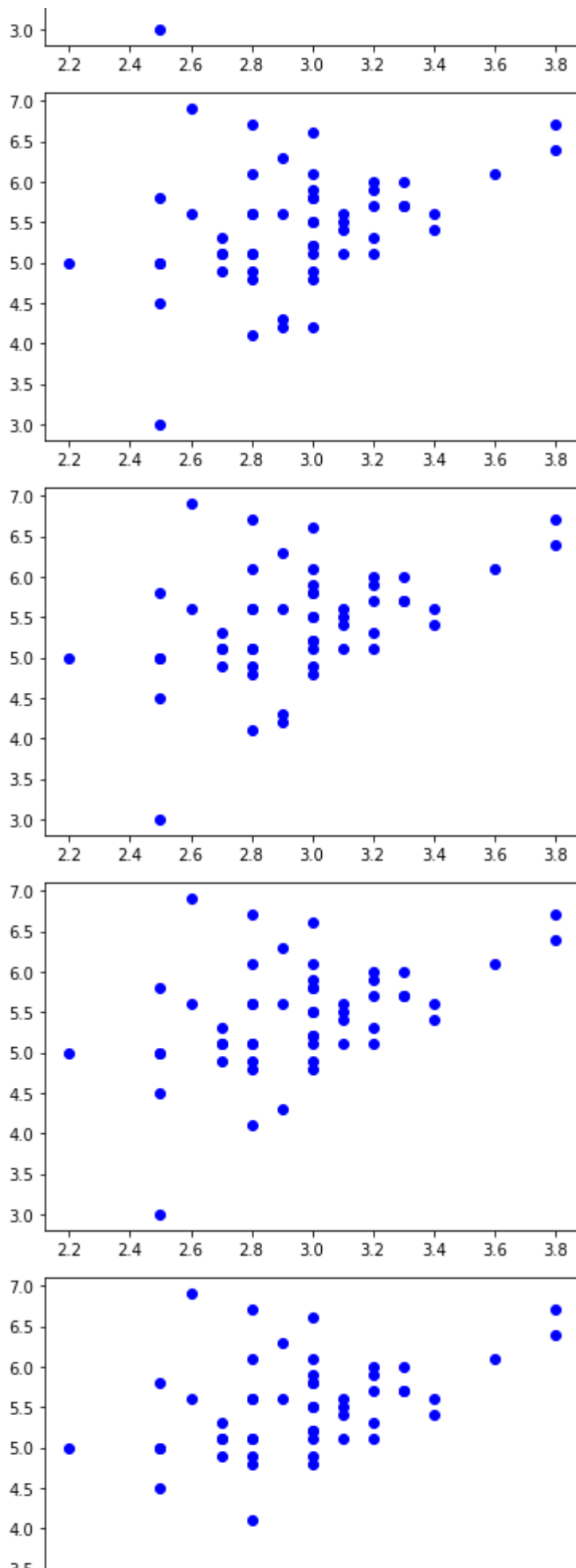


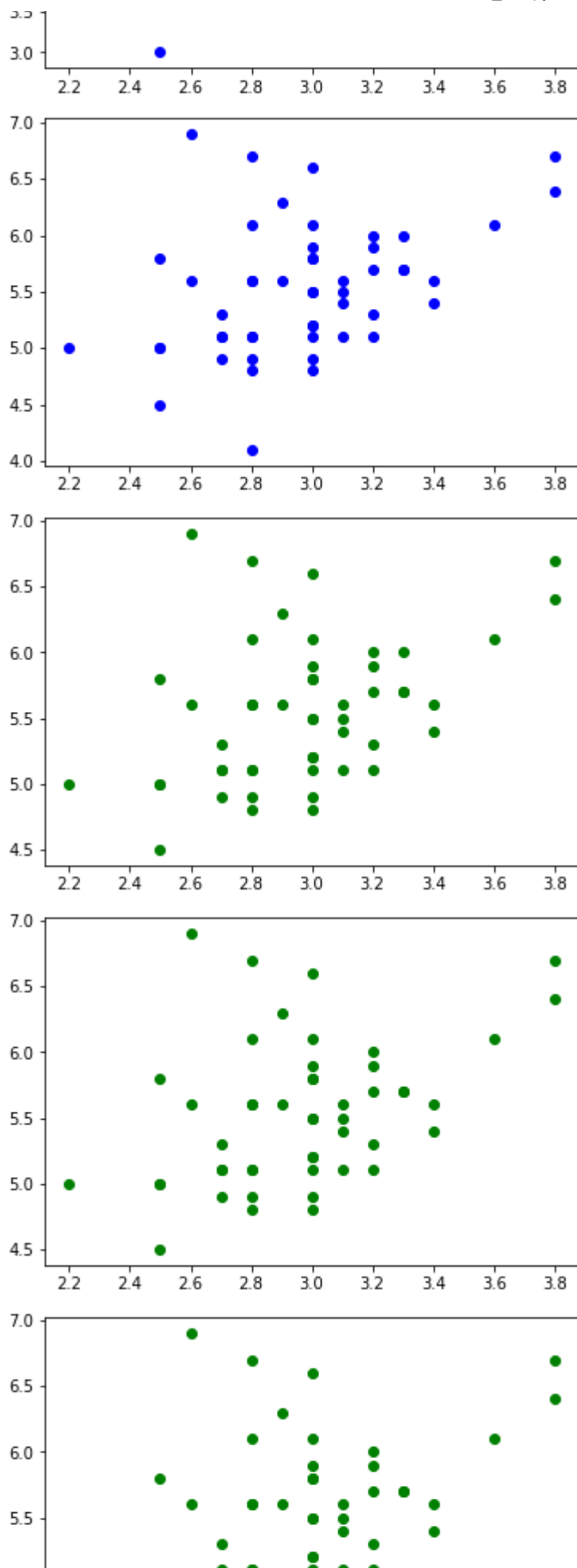


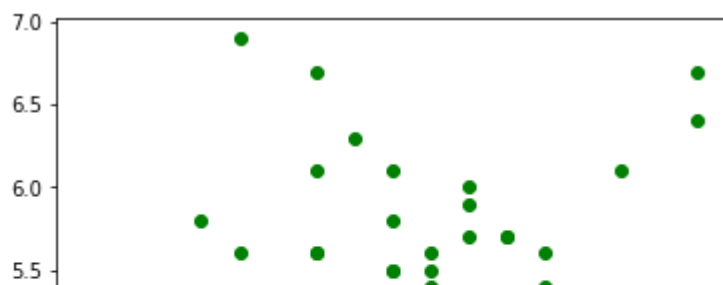
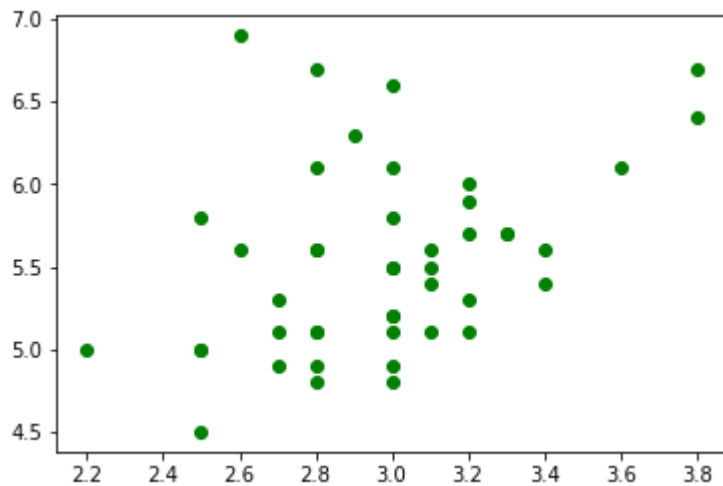
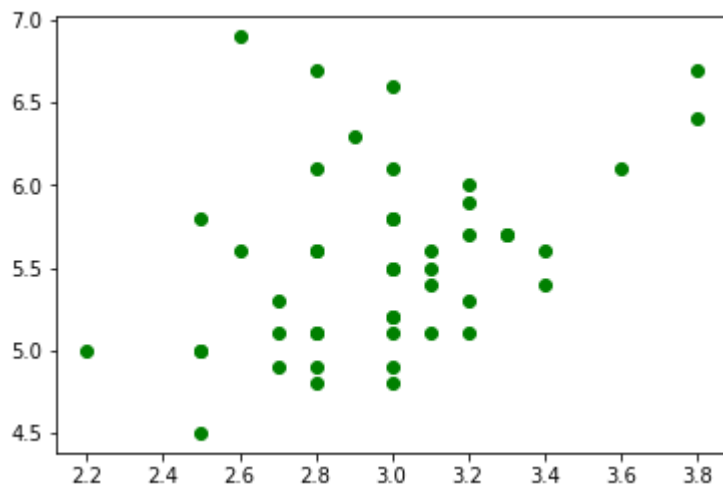
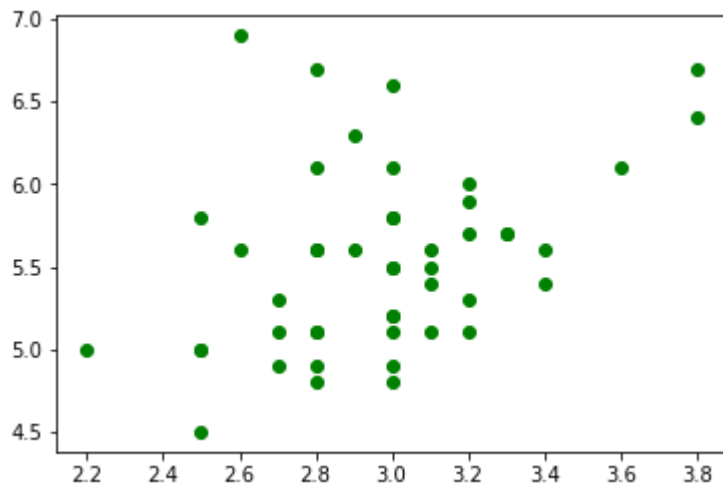
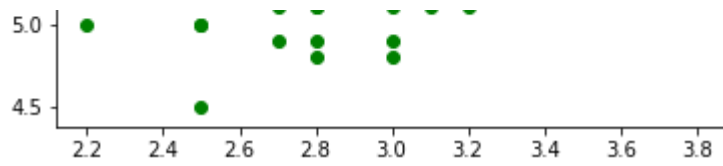


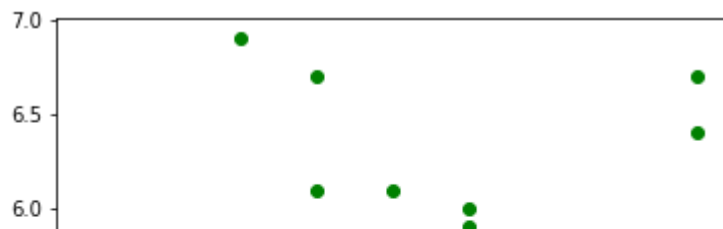
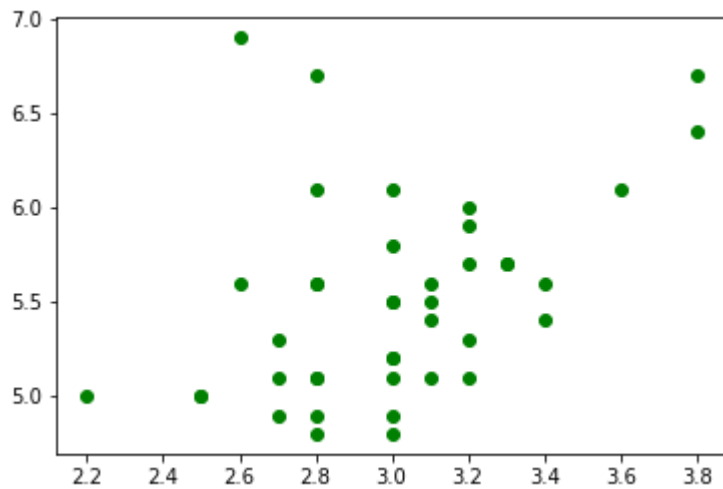
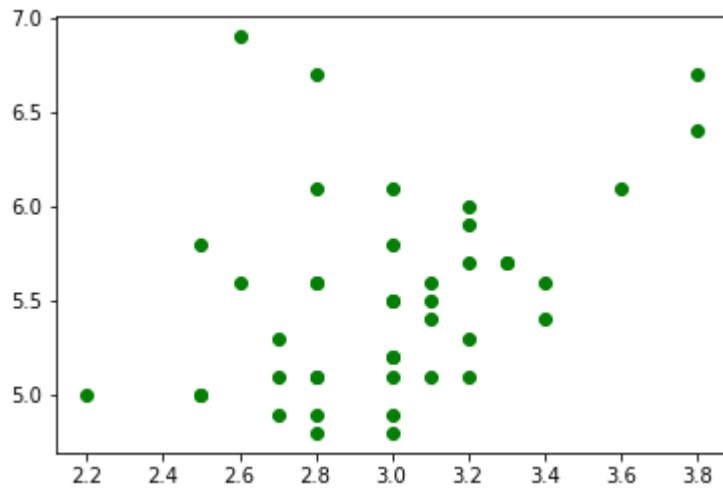
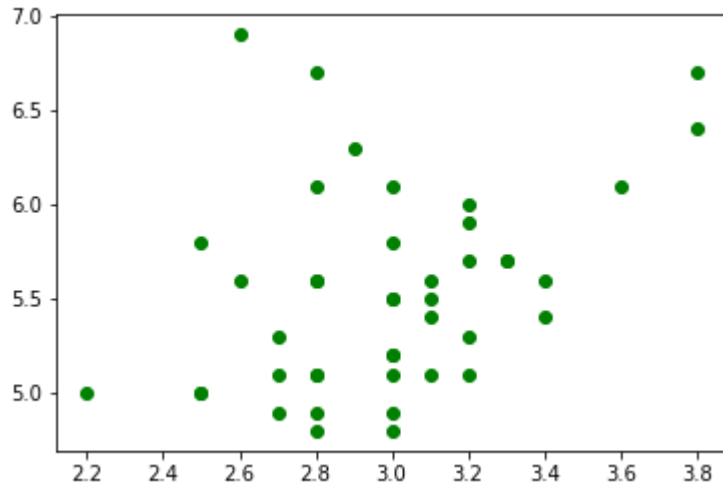
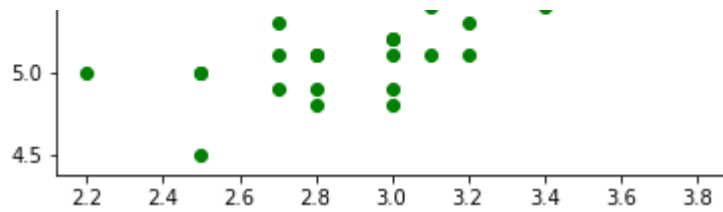


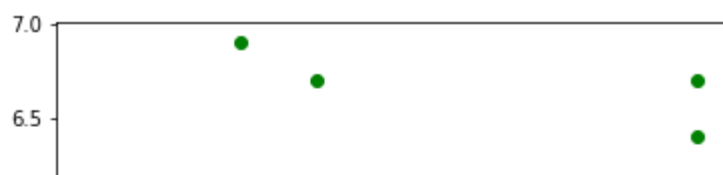
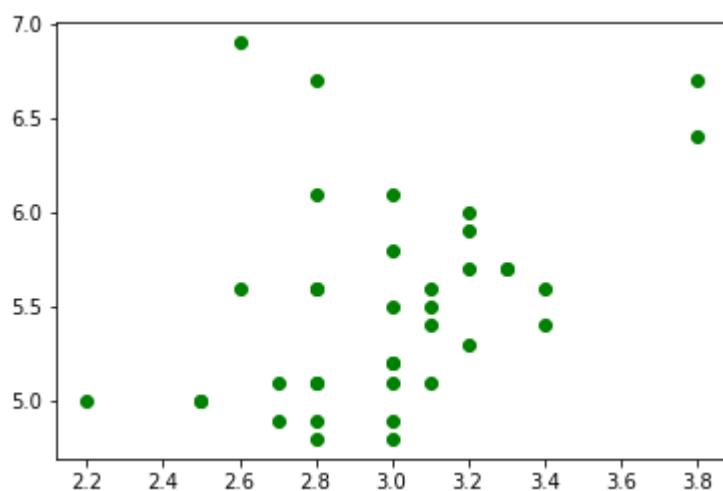
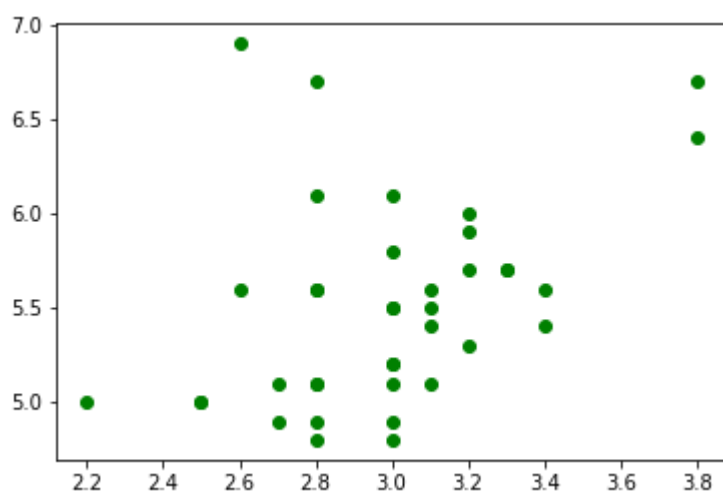
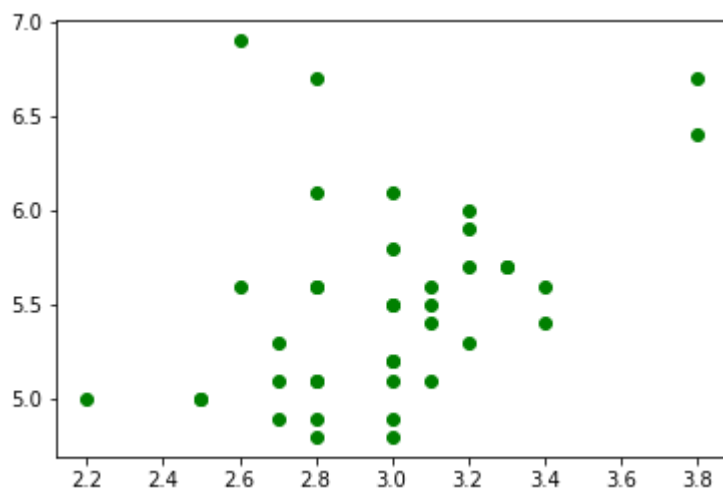
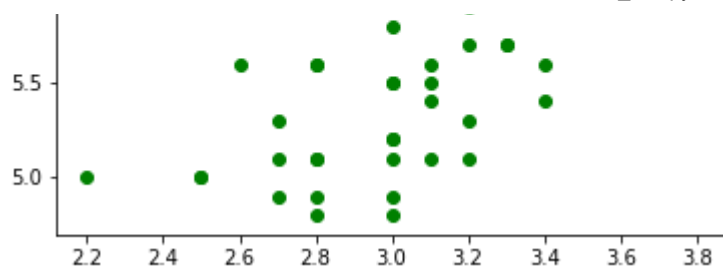


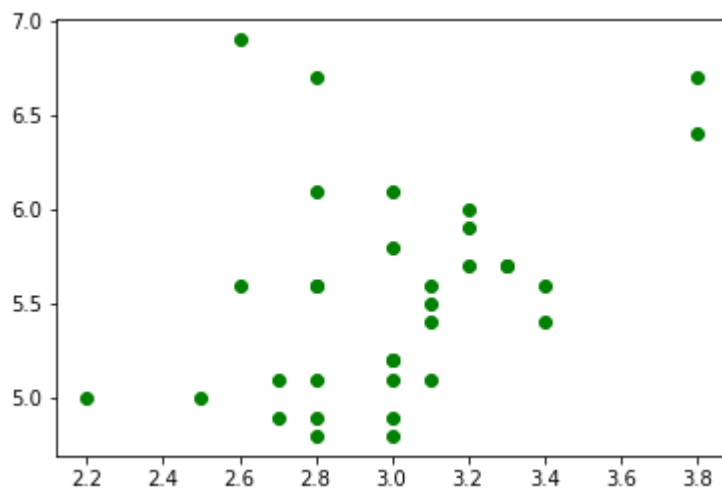
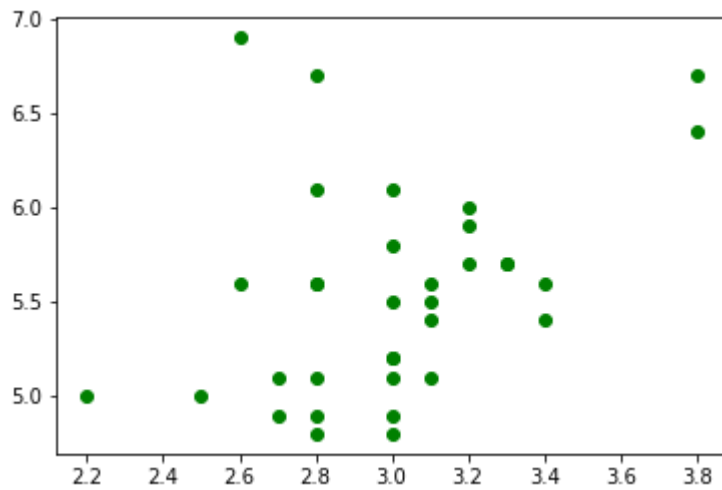
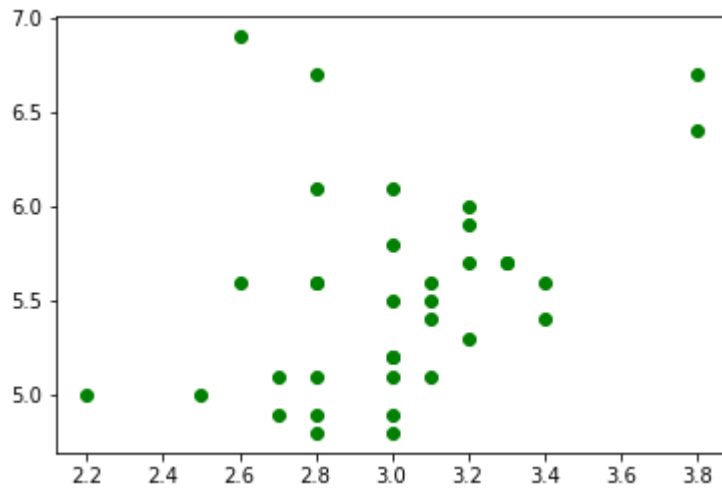
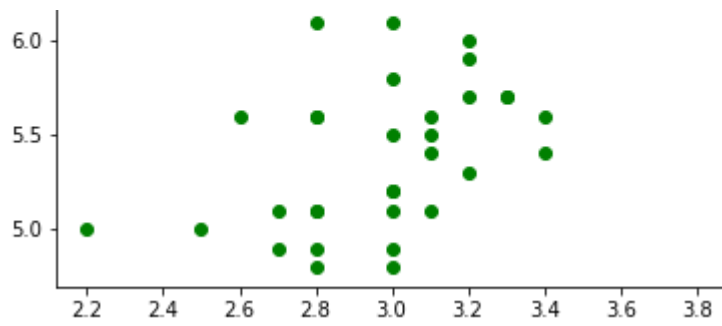


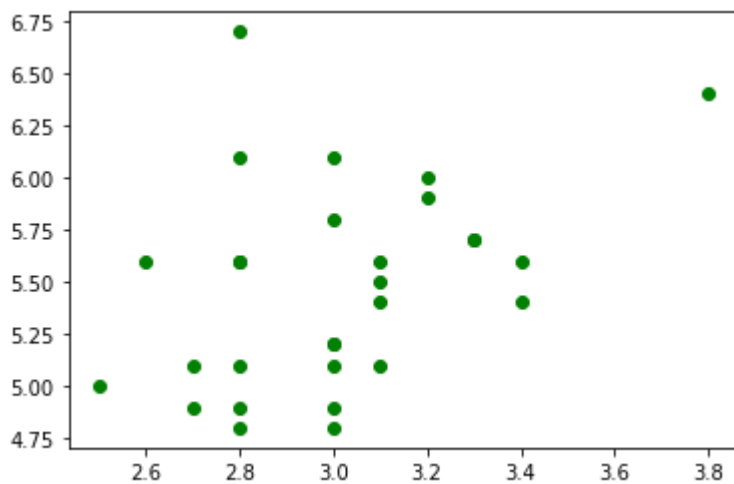
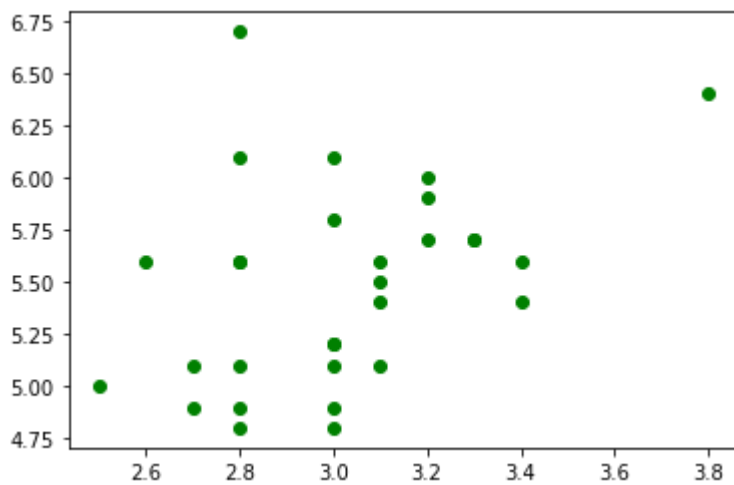
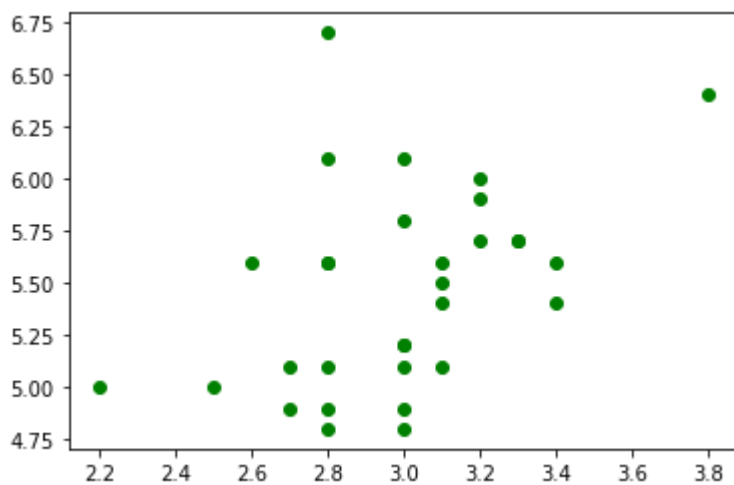
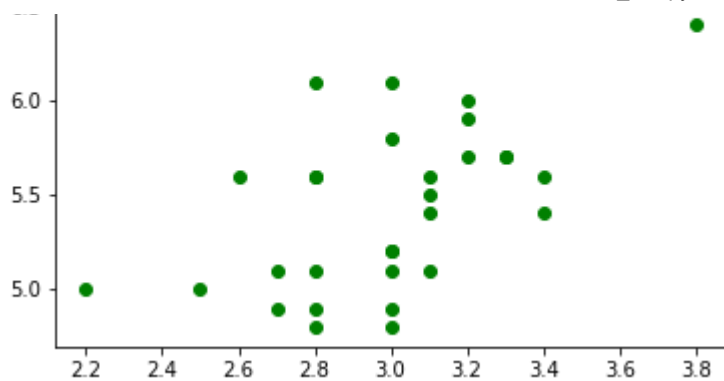


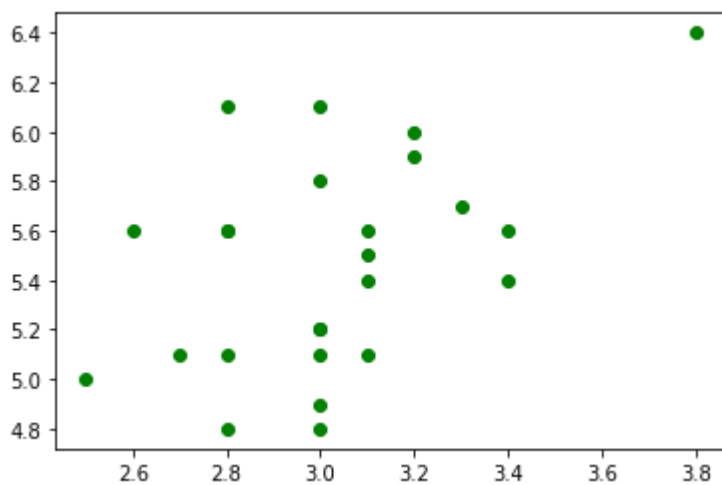
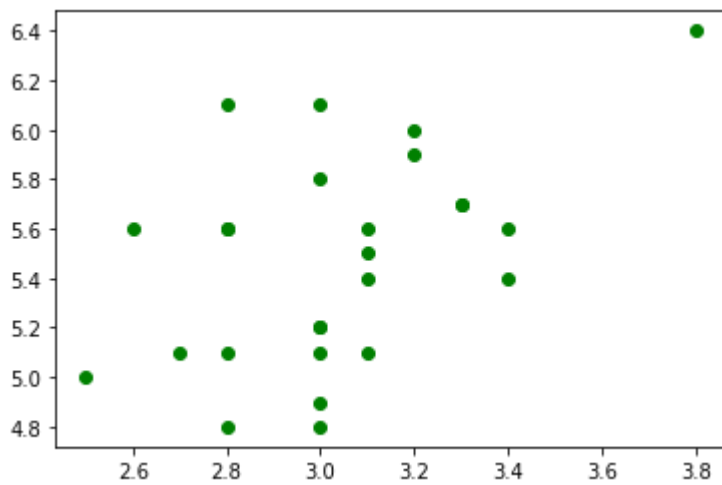
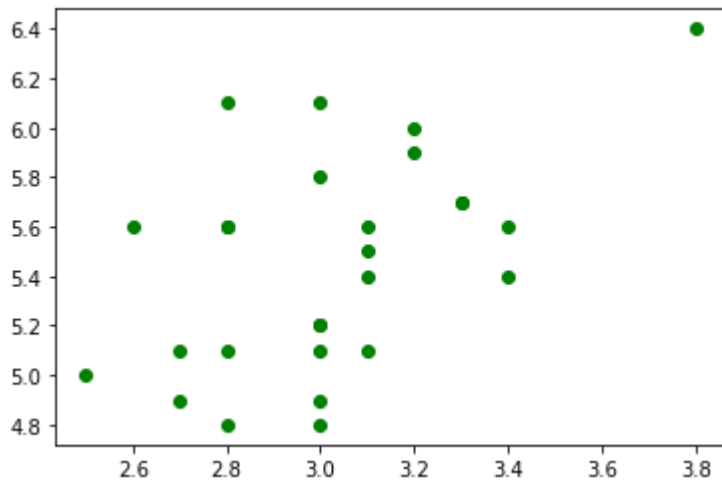
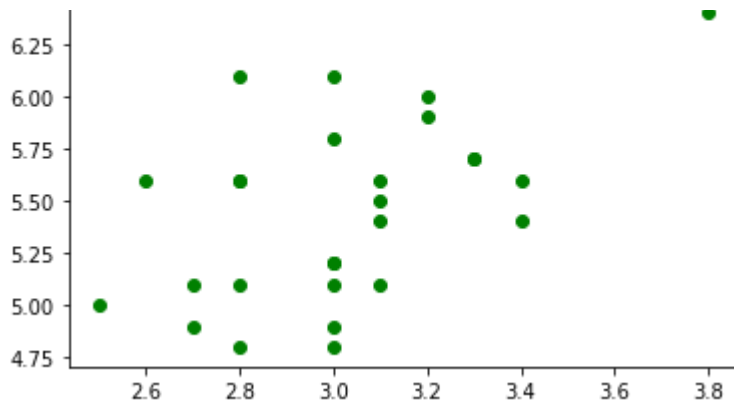


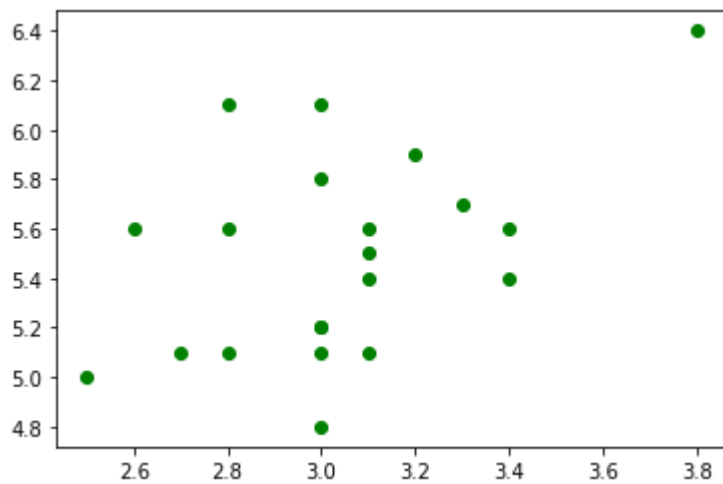
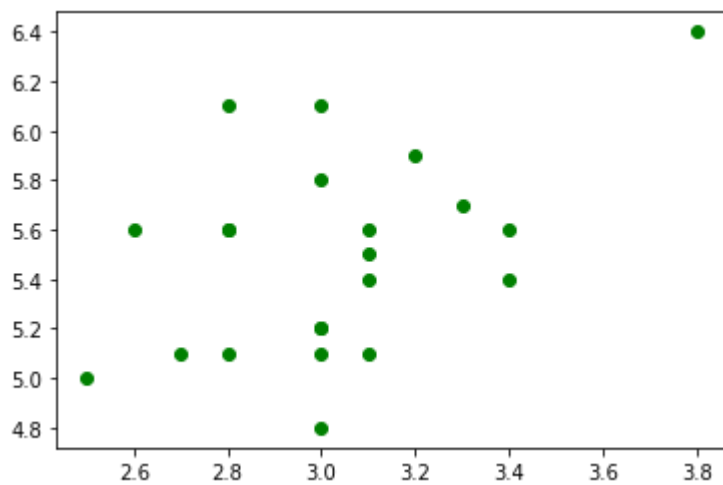
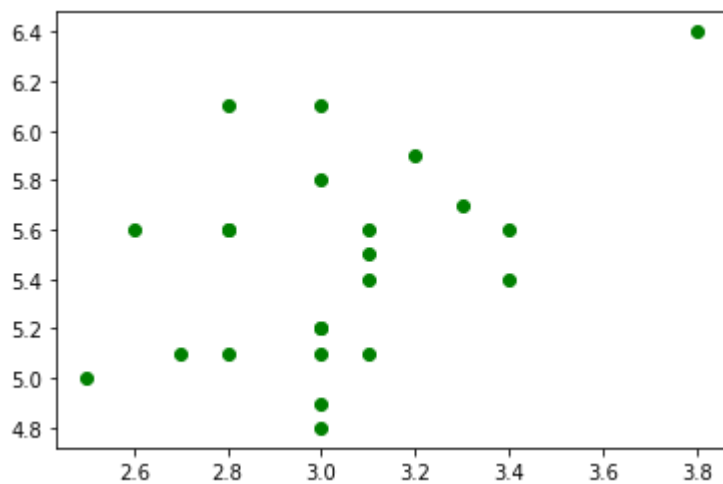
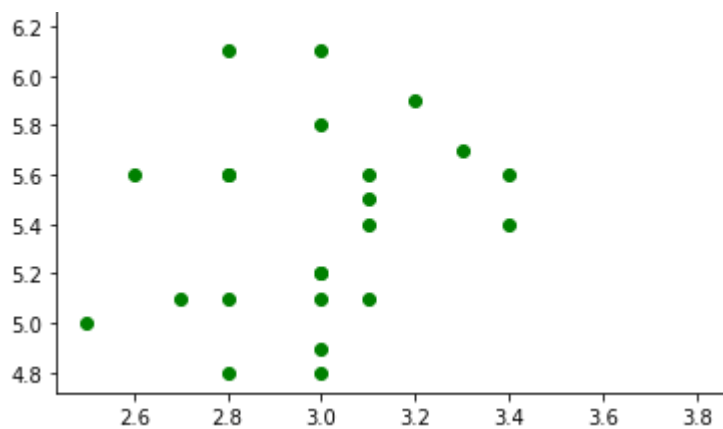


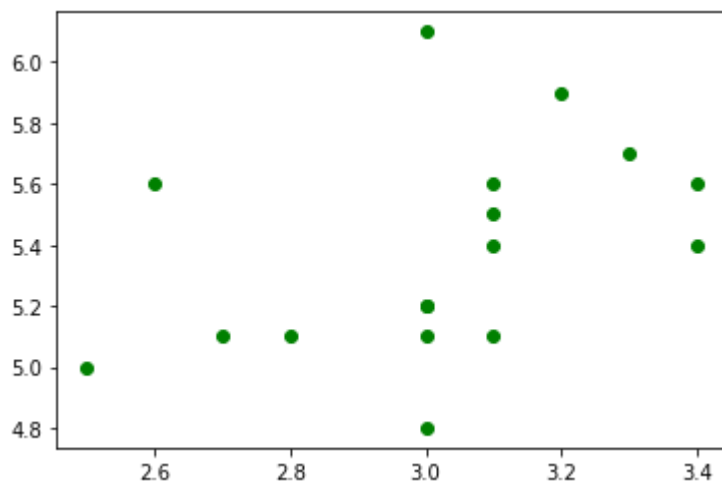
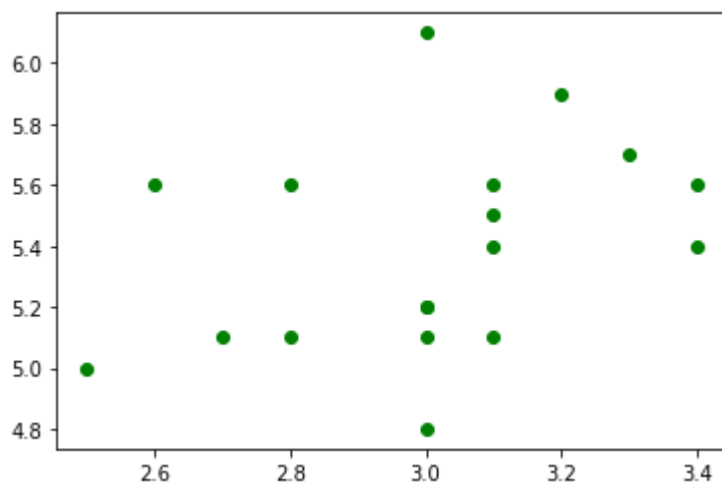
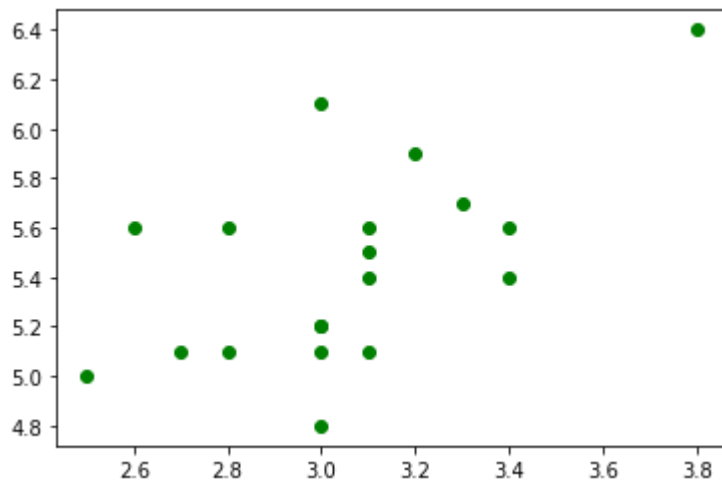
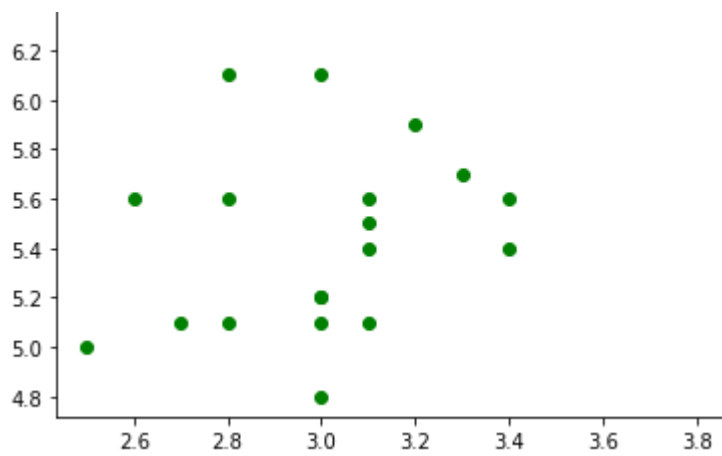


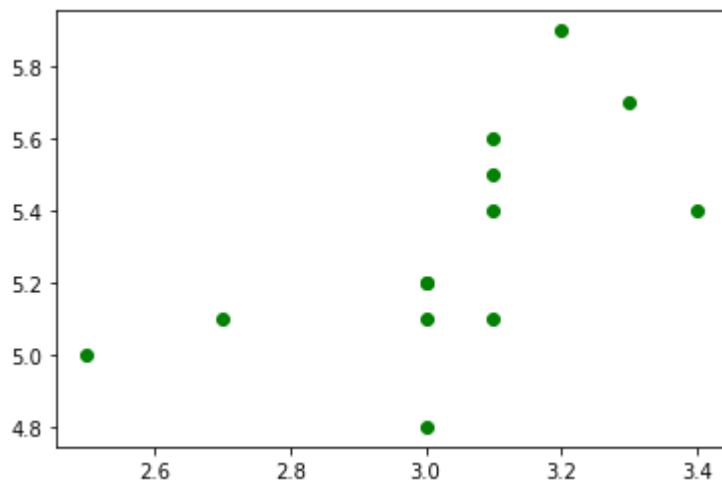
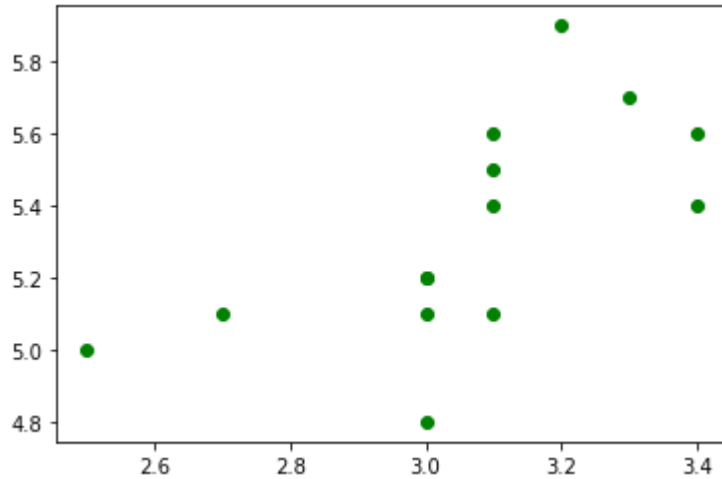
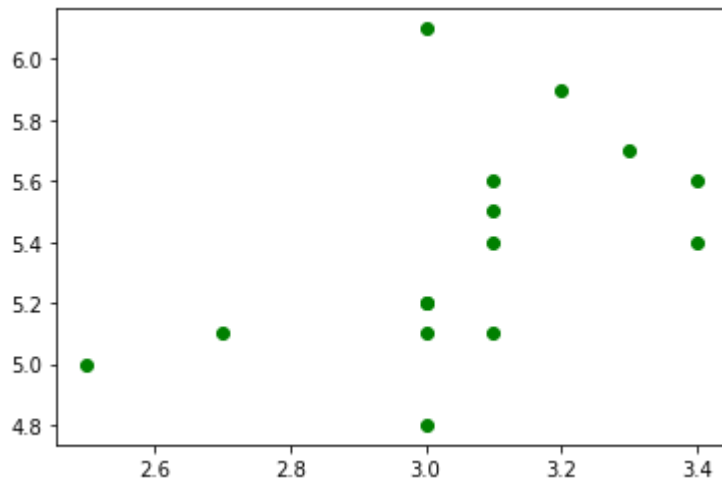
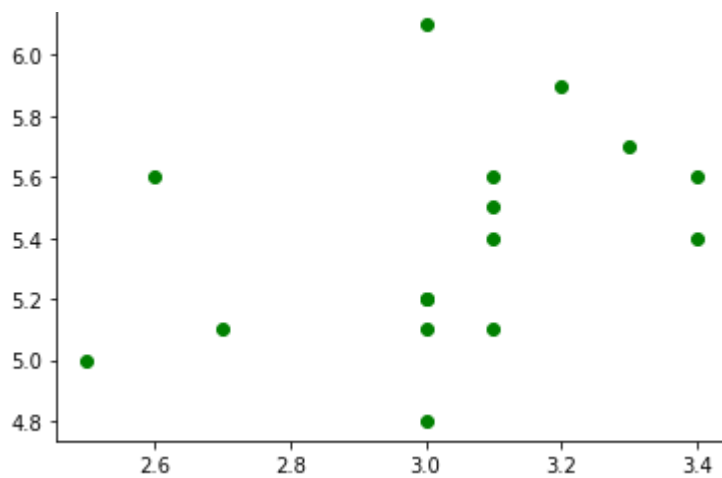


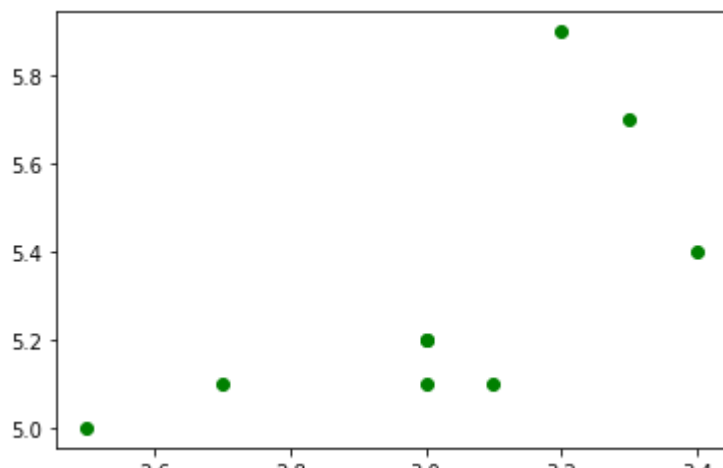
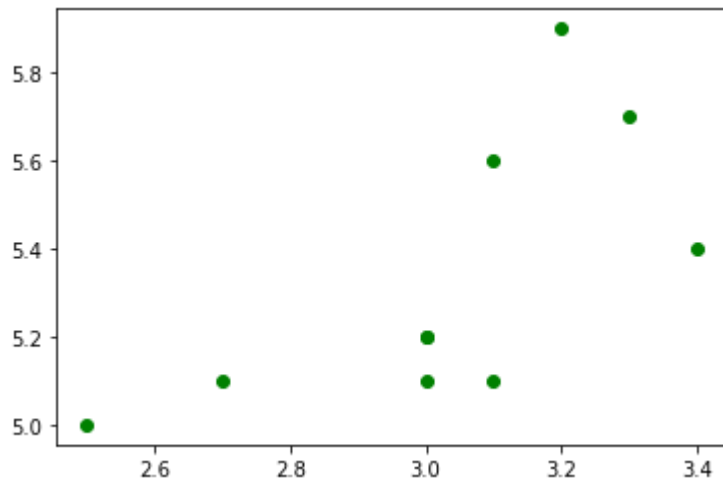
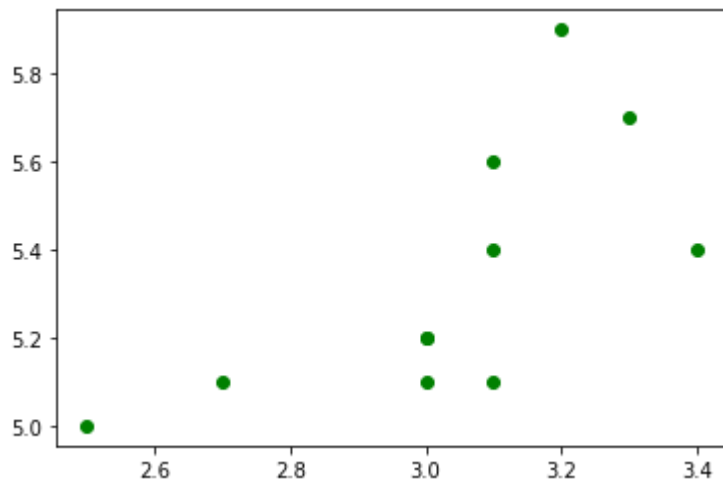
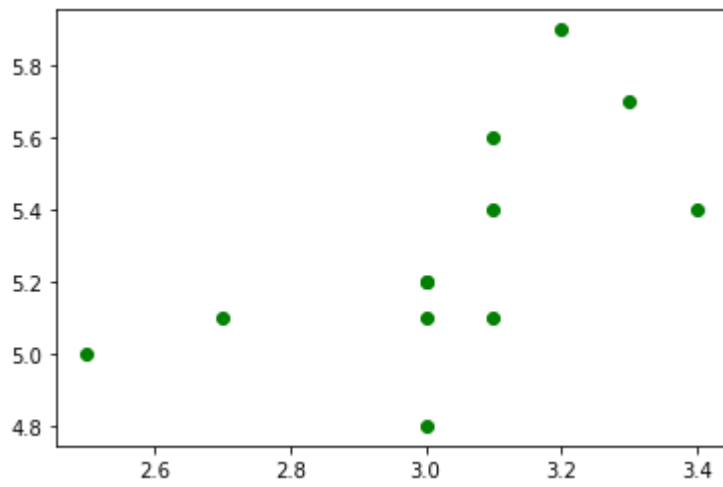


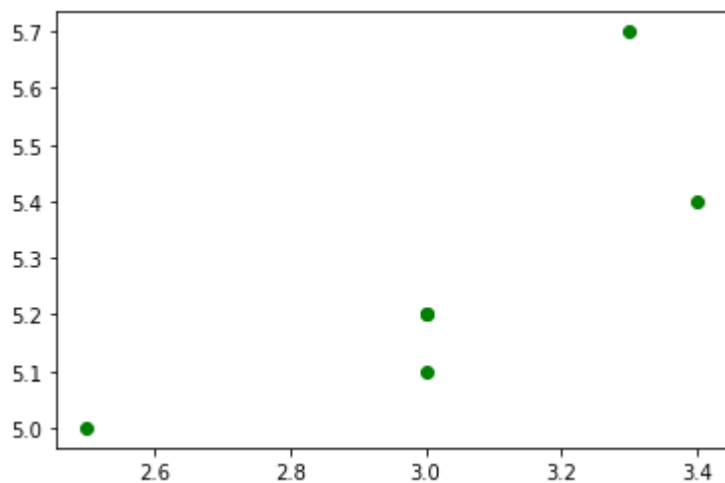
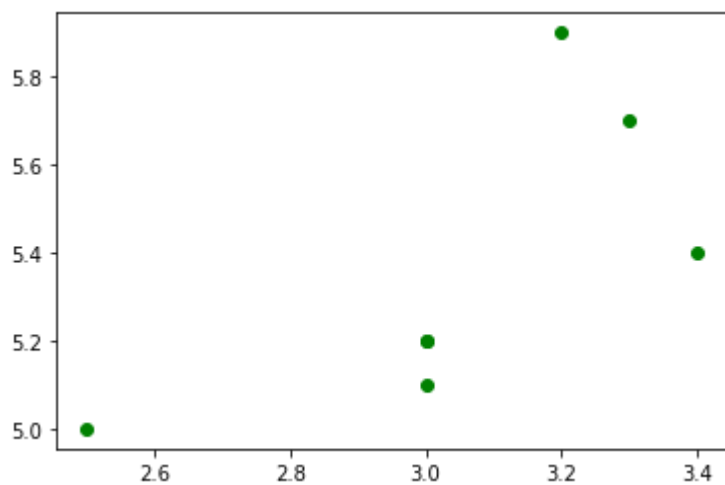
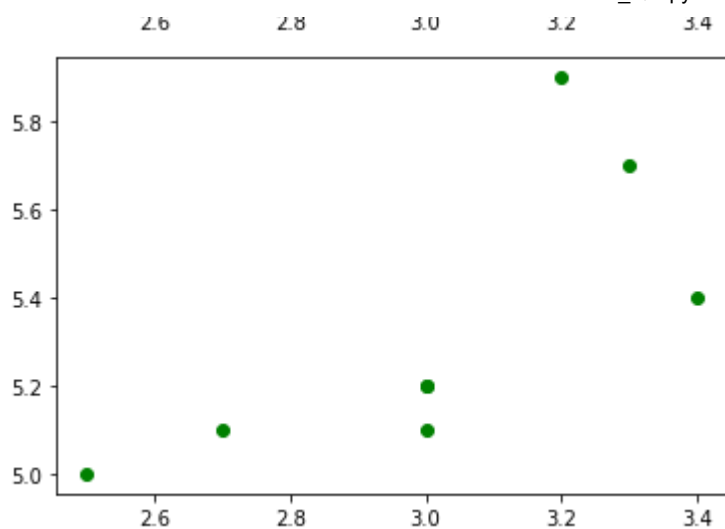


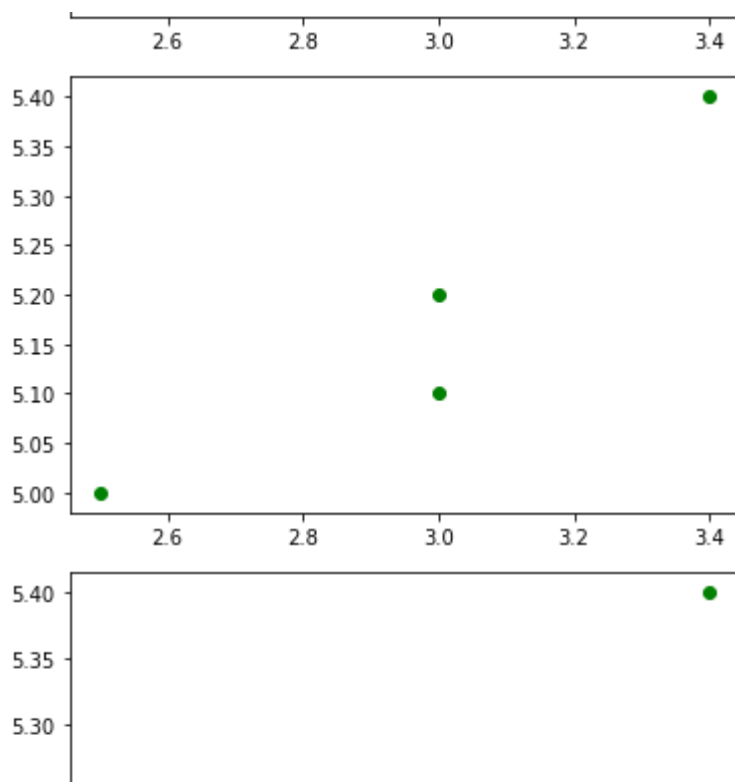






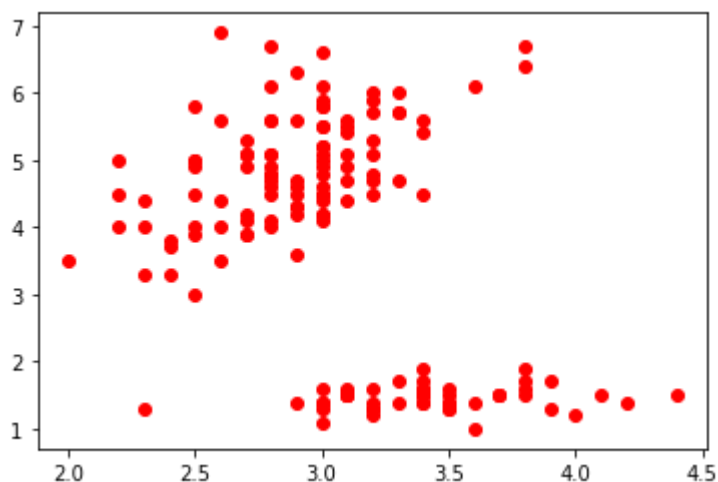






Plotting unlabeled iris data set

```
plt.plot(iris.values[:,1],iris.values[:,2],'ro')
plt.show()
```



Clustering using KMeans Clustering Algorithm

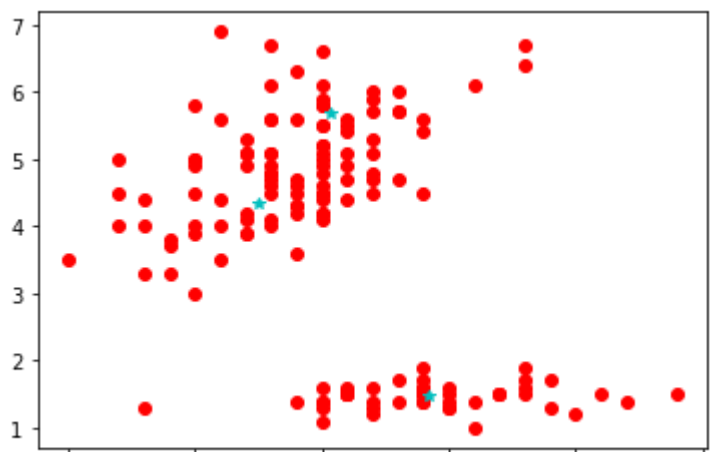
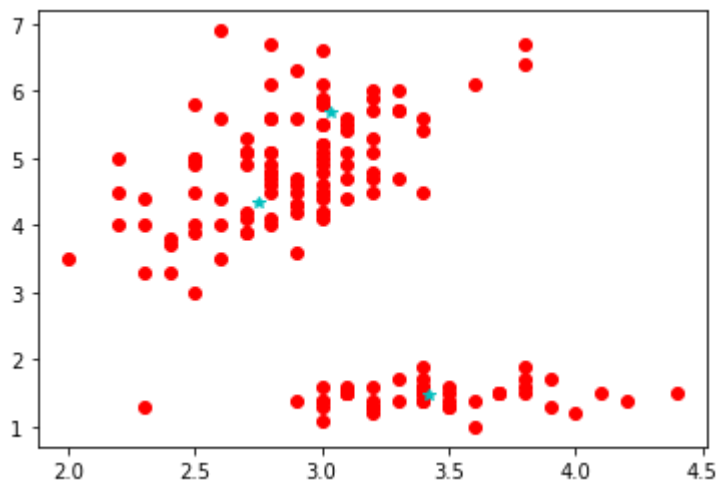
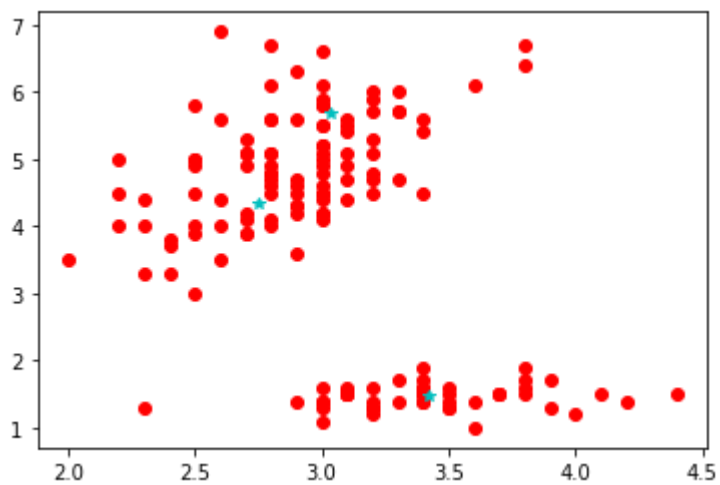
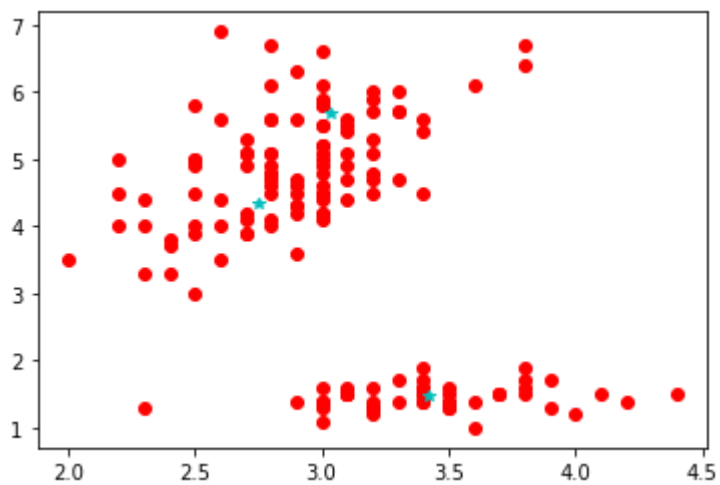
```
estimate1 = KMeans(n_clusters=3)
estimate1.fit(iris.values[:,1:3])
```

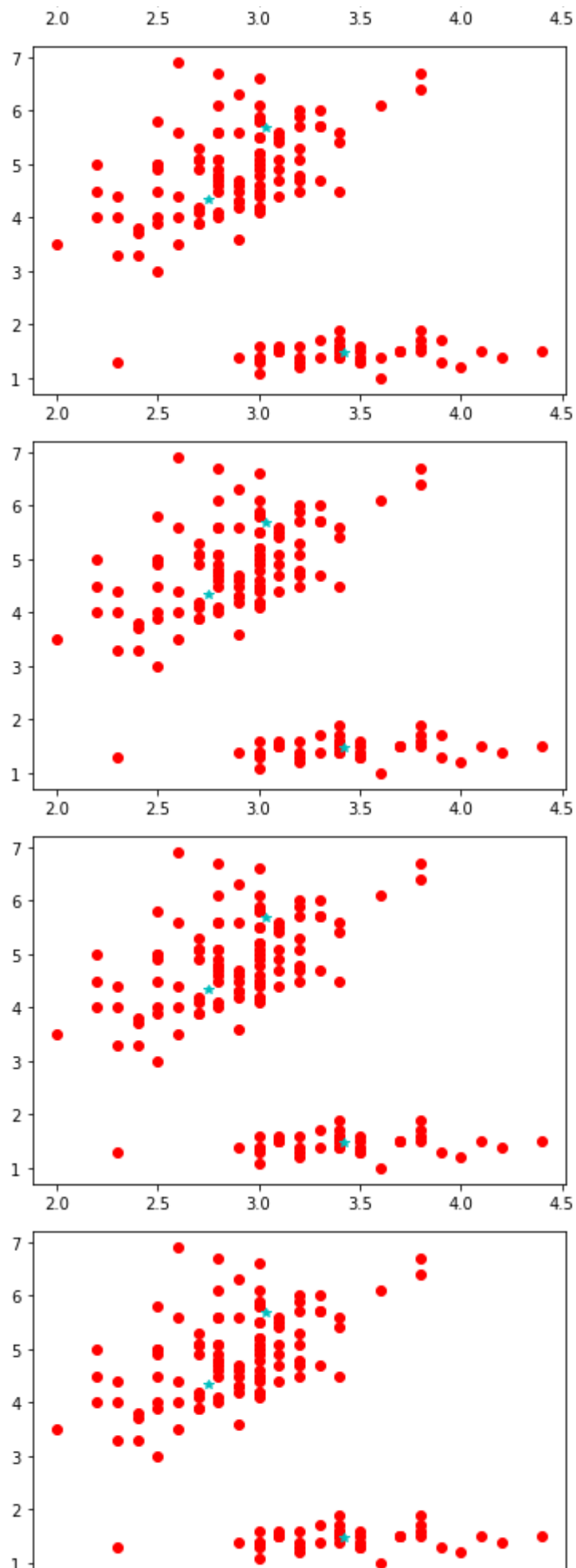
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
        n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
```

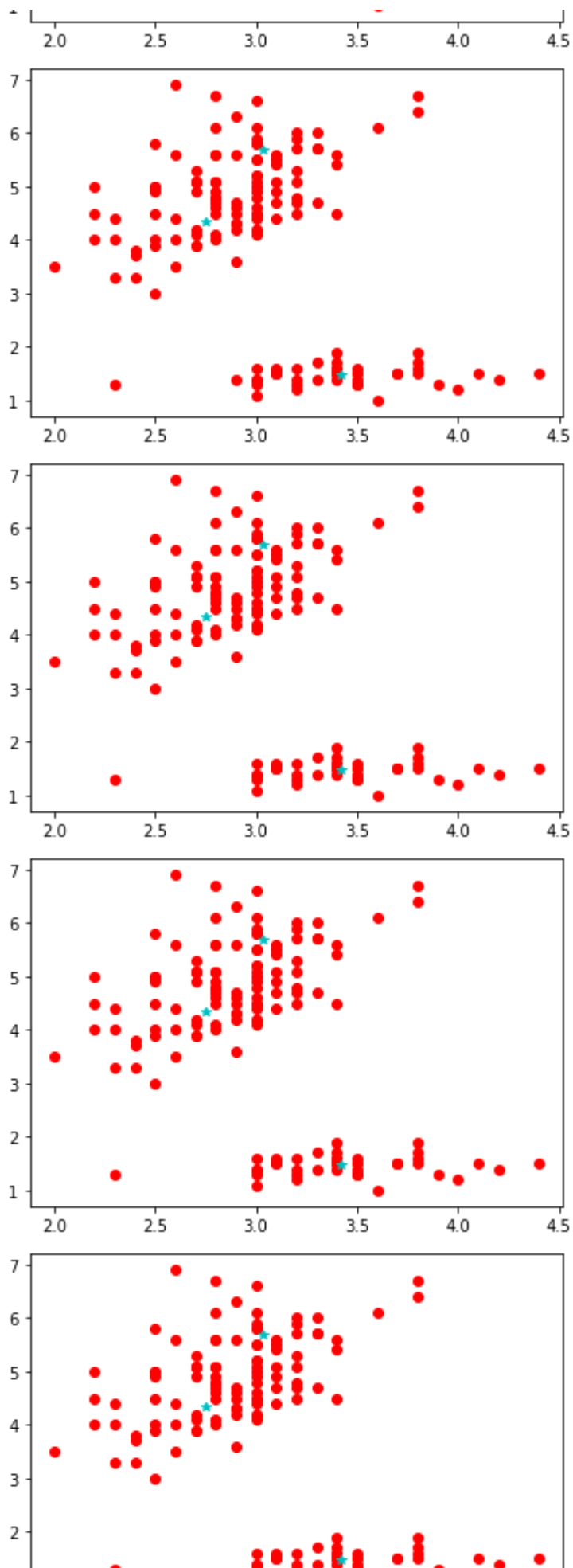
```
random_state=None, tol=0.0001, verbose=0)
2.85    2.90    2.95    3.00    3.05    3.10    3.15
```

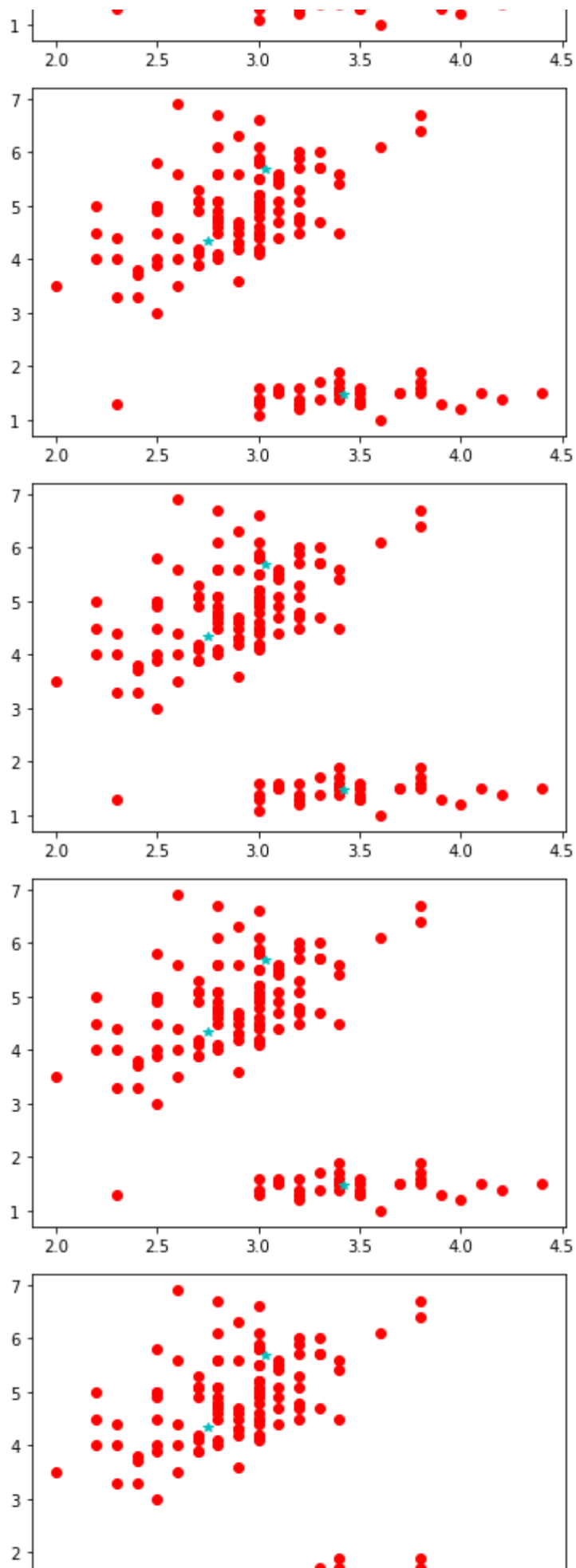
Plotting Clustered data points using K_Means with 3 clusters

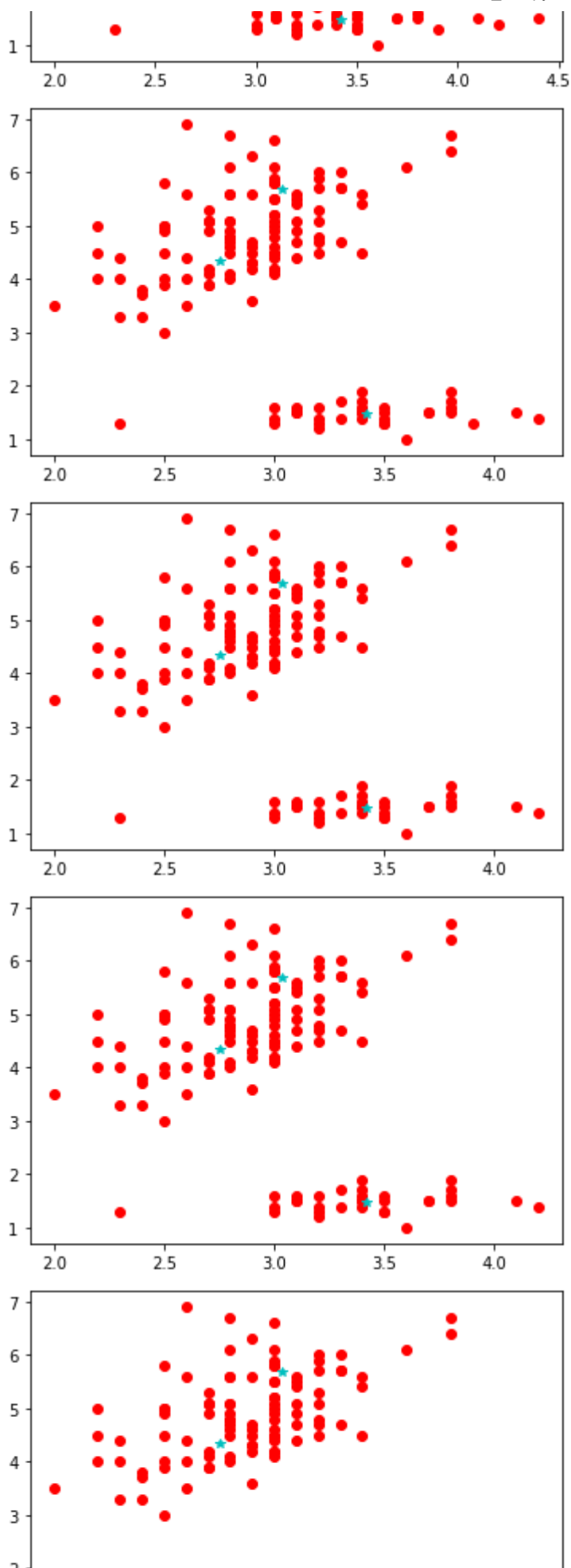
```
for i in range(150):
    if estimate1.labels_[i]==0:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'go')
        plt.plot(estimate1.cluster_centers_[:,0],estimate1.cluster_centers_[:,1],'c*')
    elif estimate1.labels_[i]==1:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'ro')
        plt.plot(estimate1.cluster_centers_[:,0],estimate1.cluster_centers_[:,1],'c*')
    elif estimate1.labels_[i]==2:
        plt.plot(iris.values[i:,1],iris.values[i:,2],'bo')
        plt.plot(estimate1.cluster_centers_[:,0],estimate1.cluster_centers_[:,1],'c*')
plt.show()
```

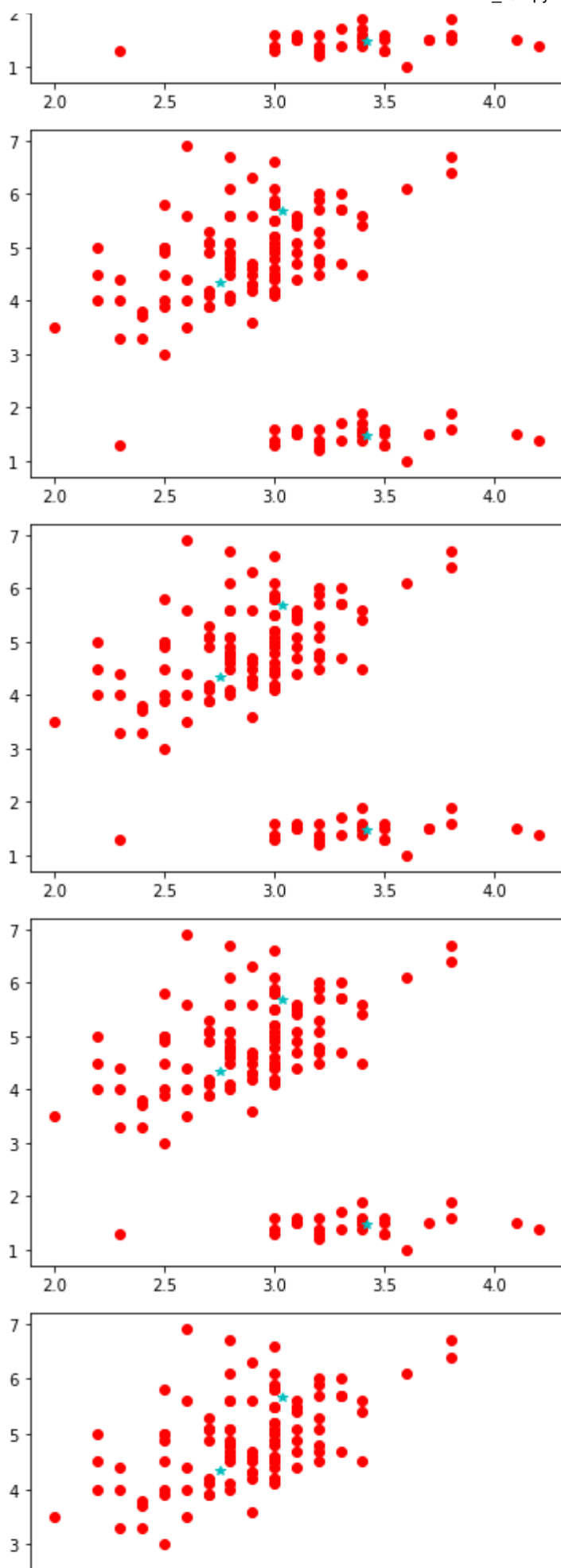



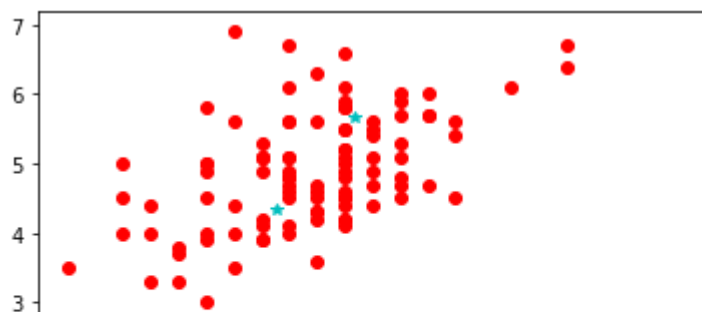
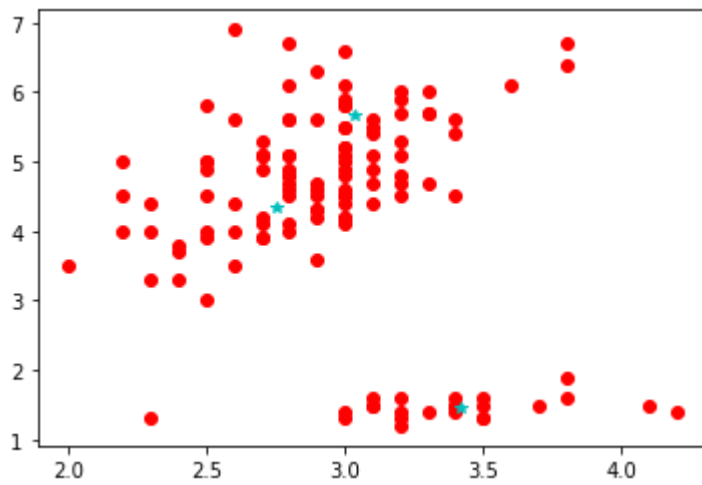
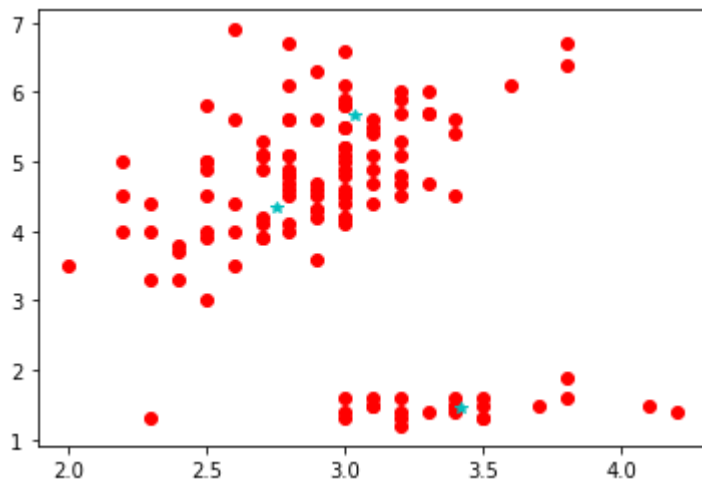
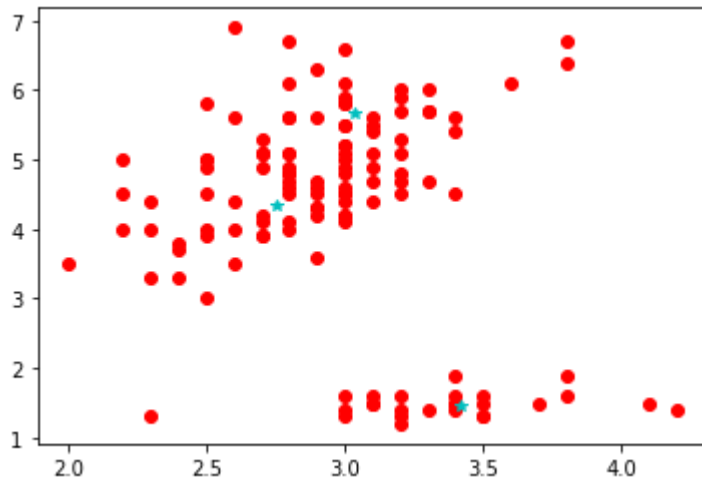
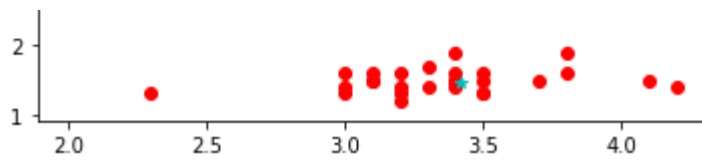


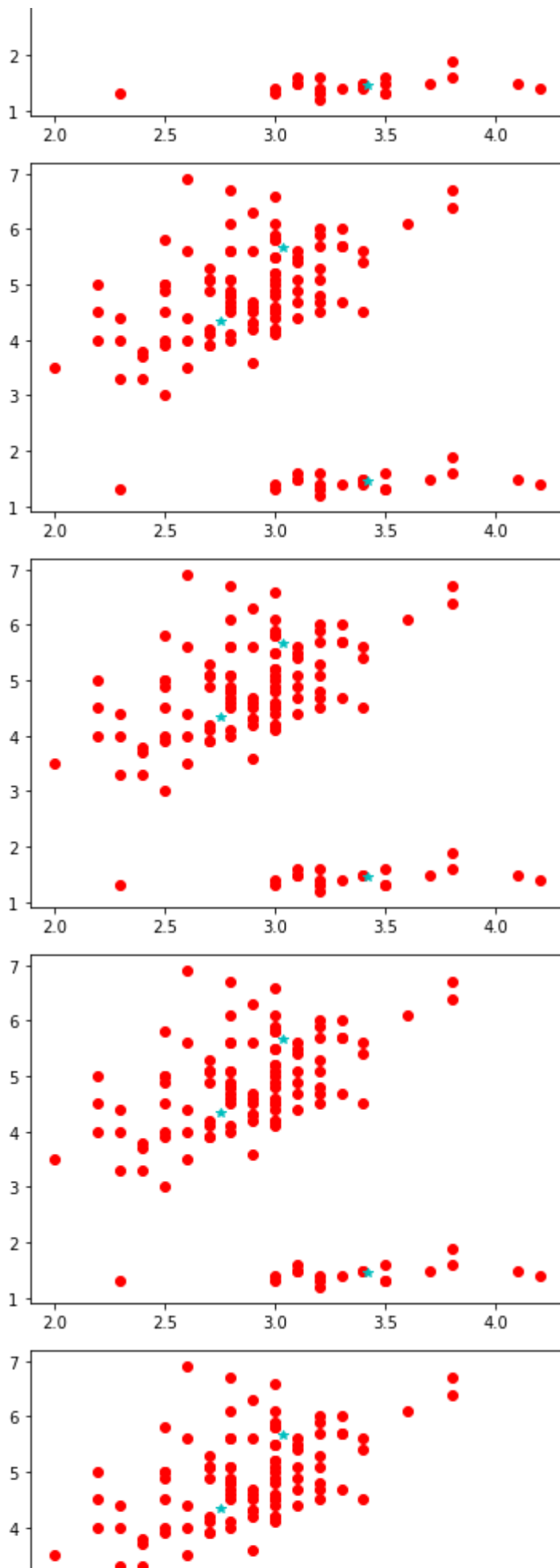


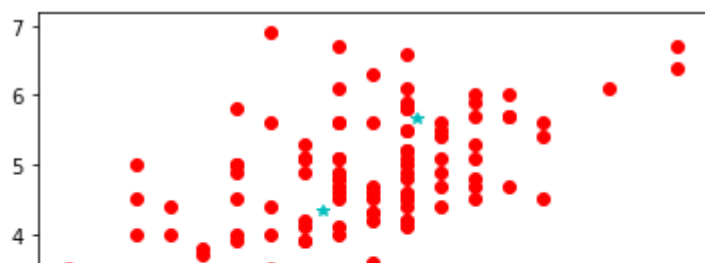
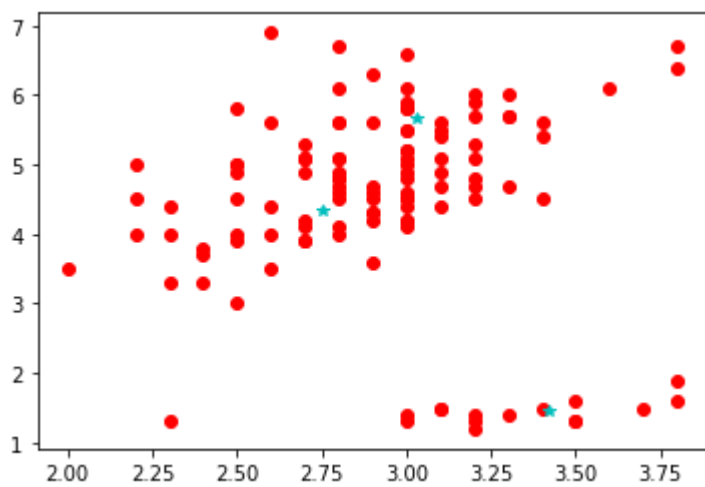
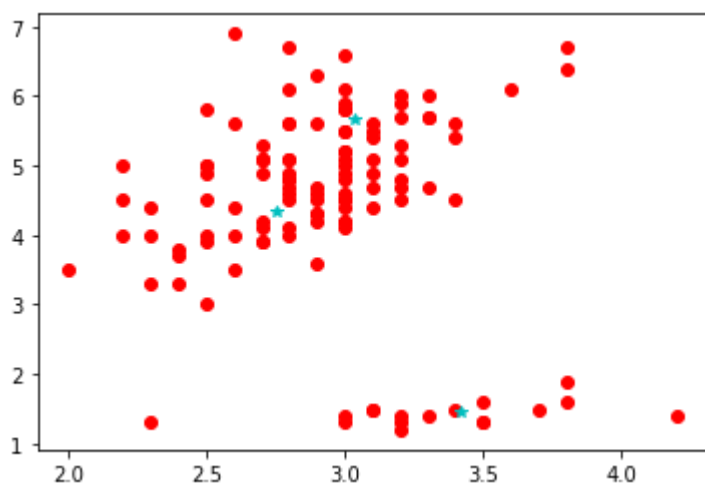
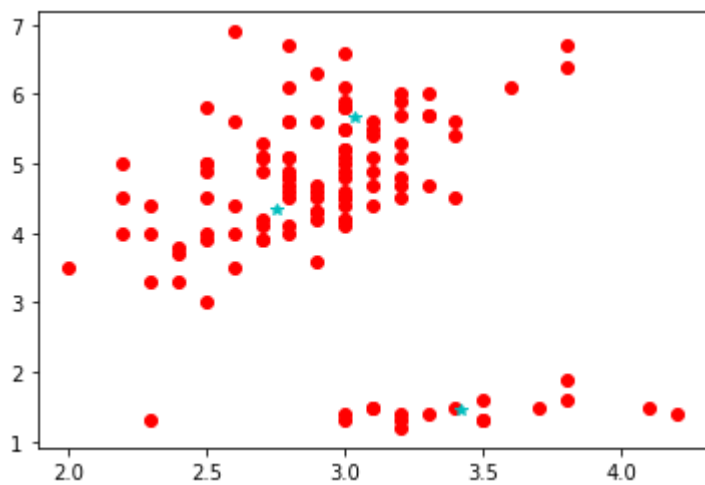
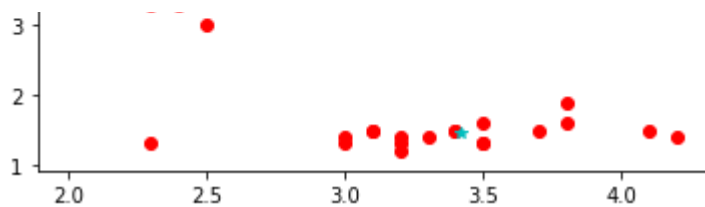


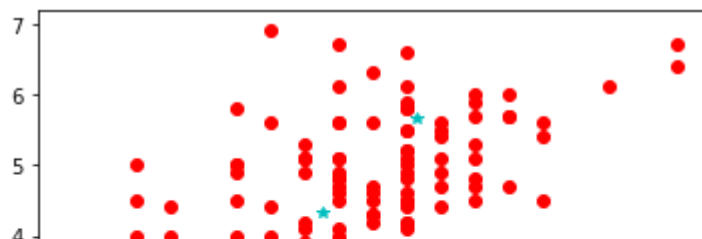
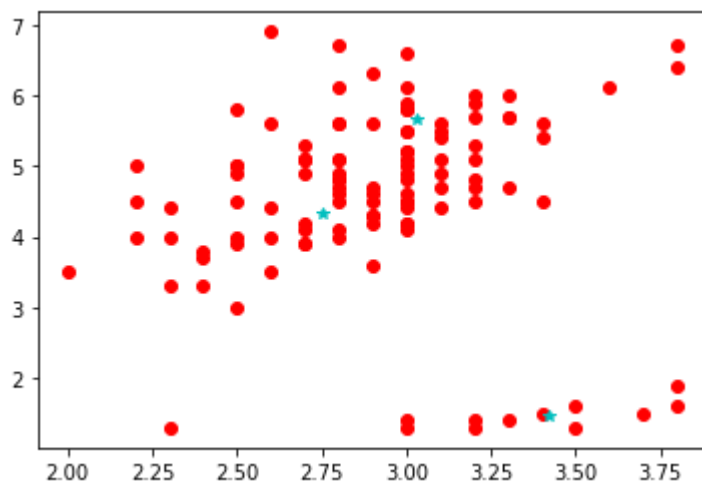
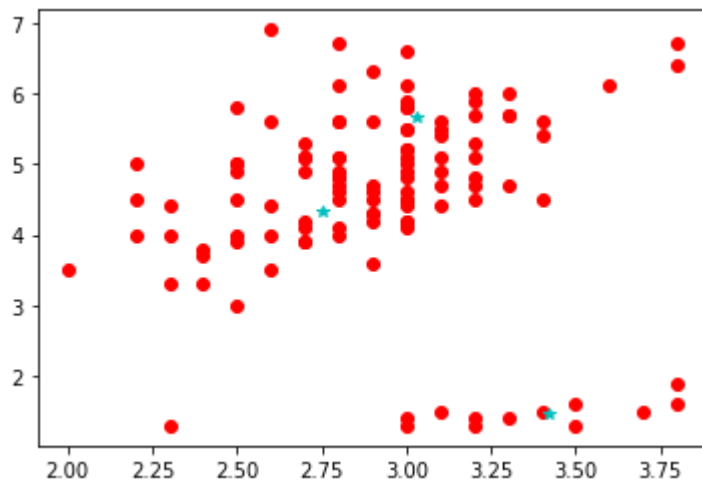
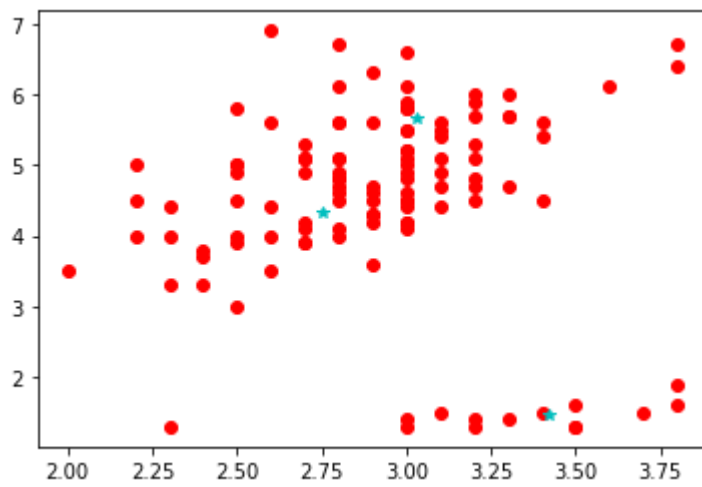
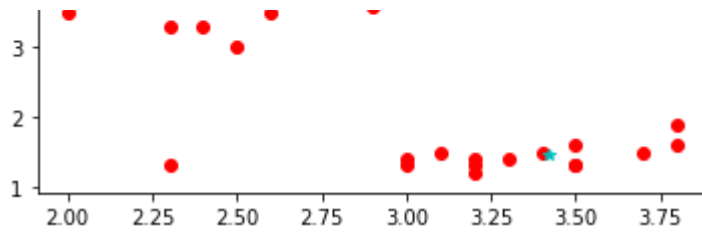


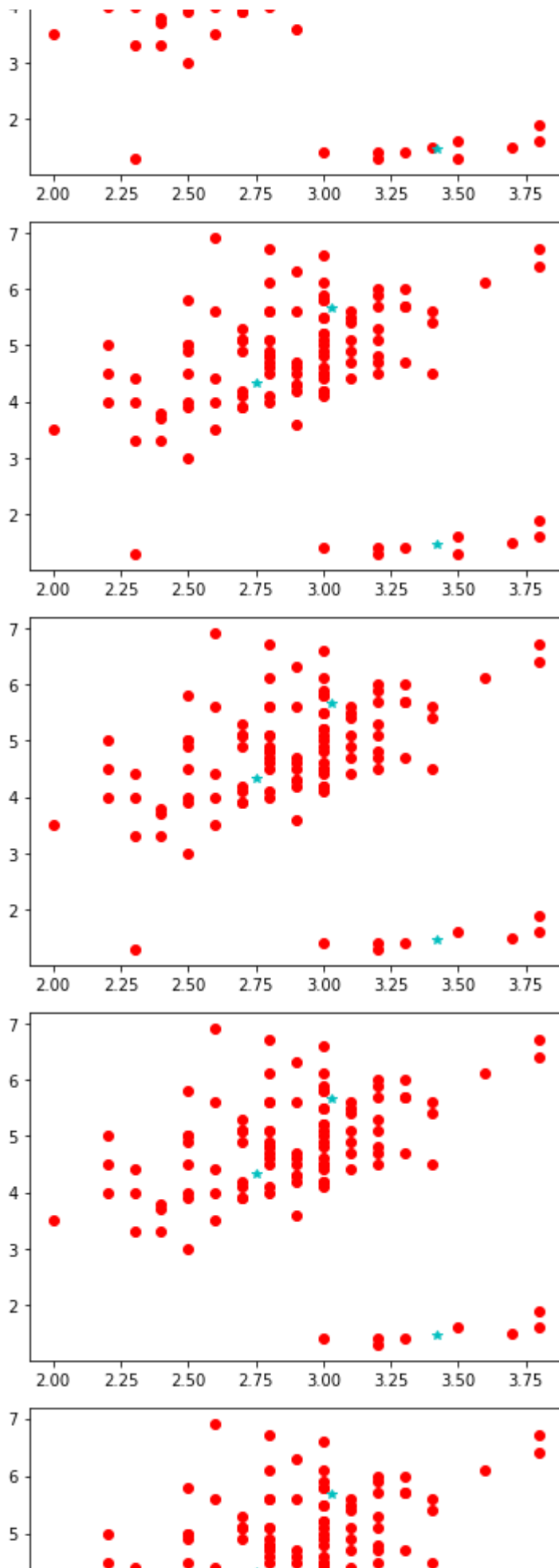


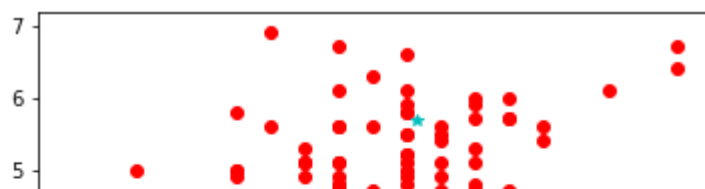
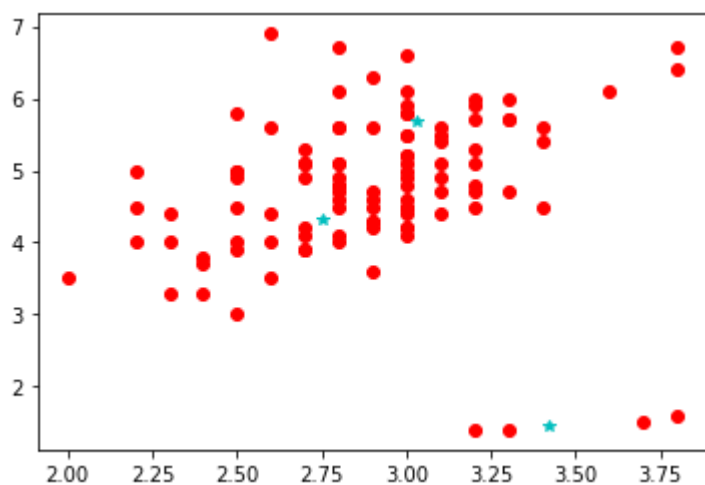
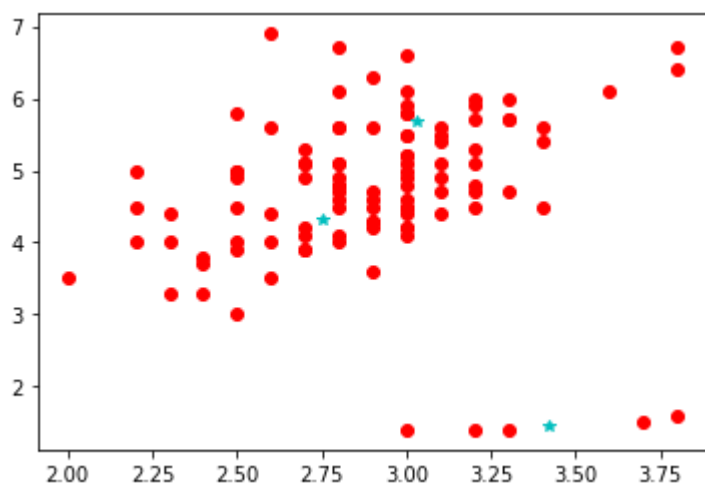
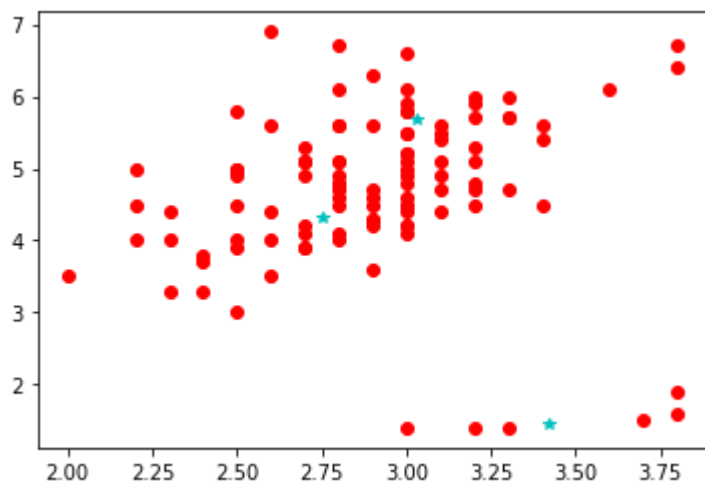
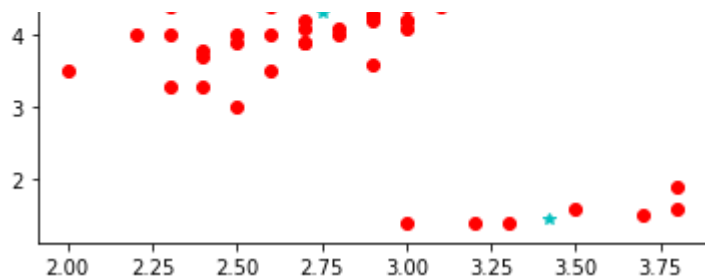


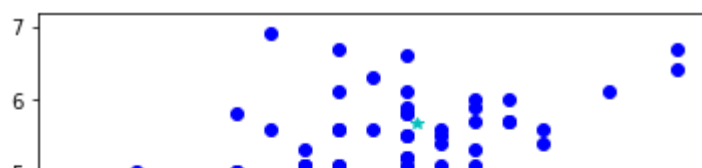
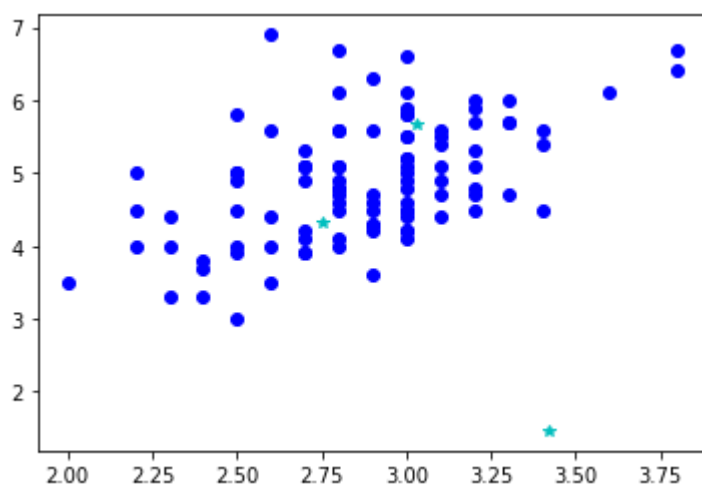
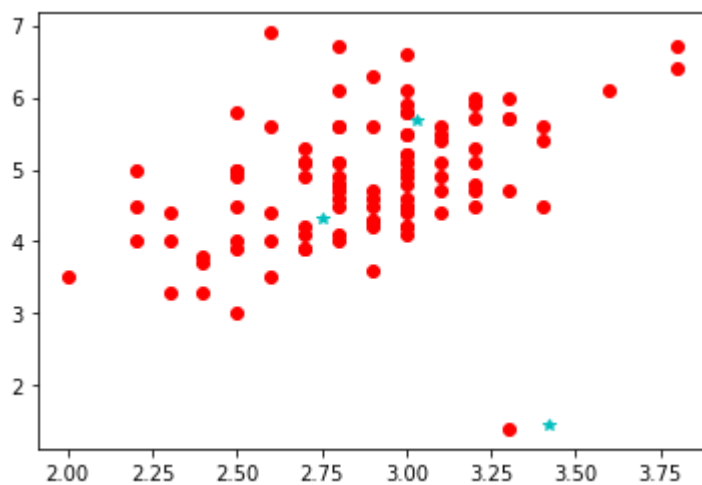
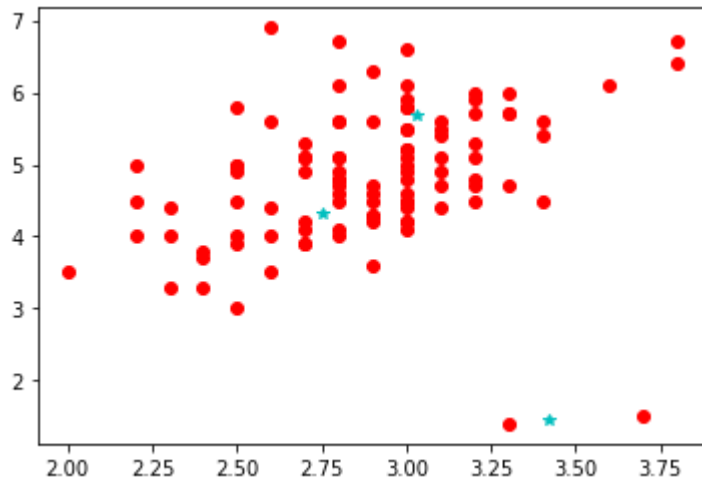
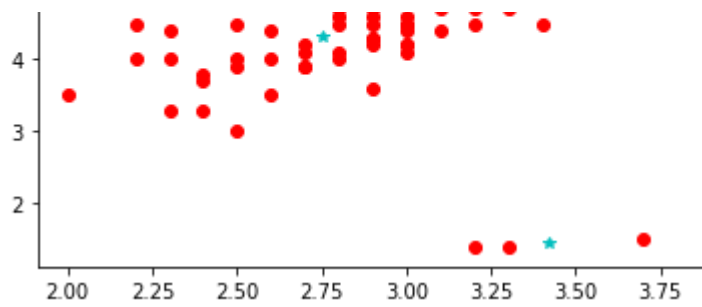


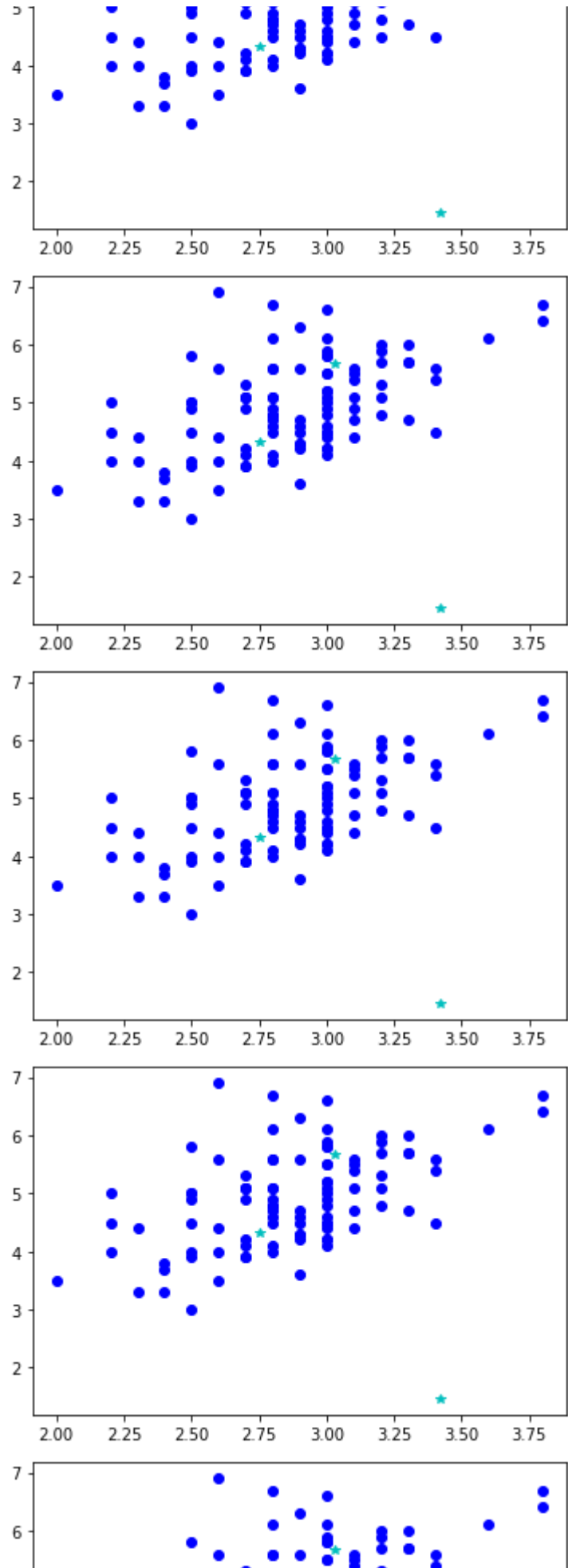


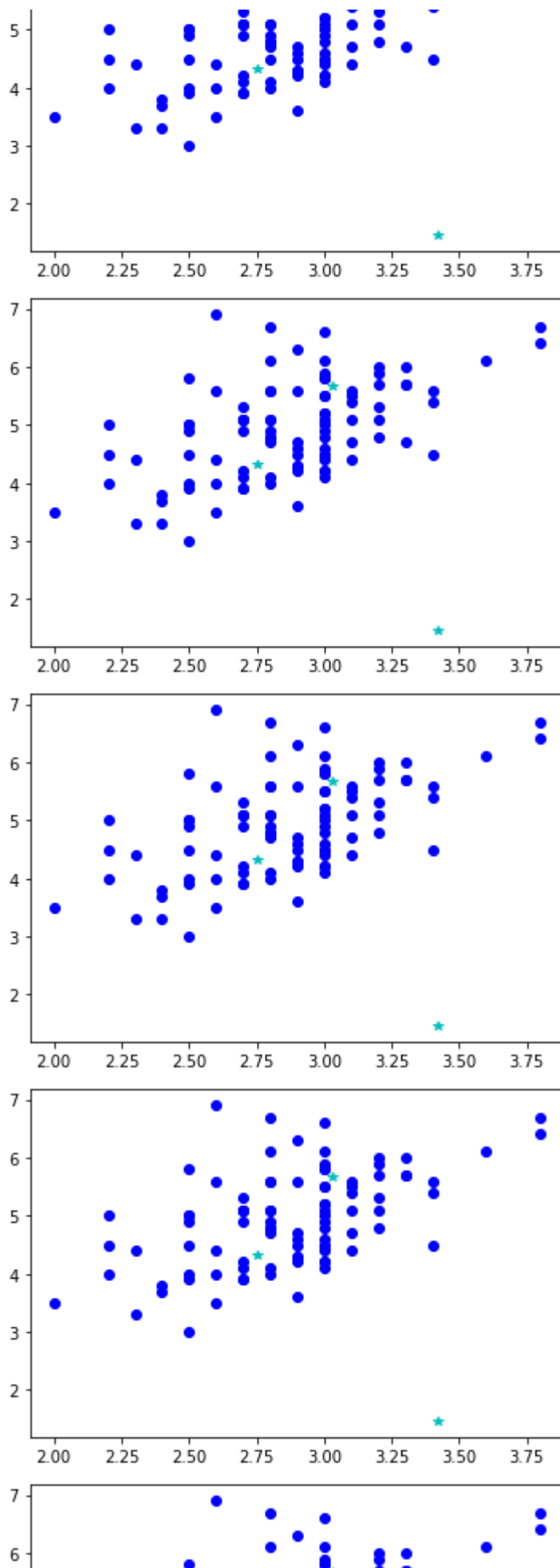


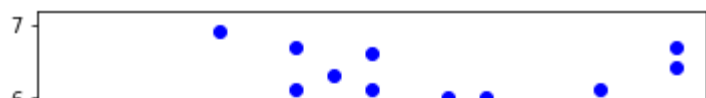
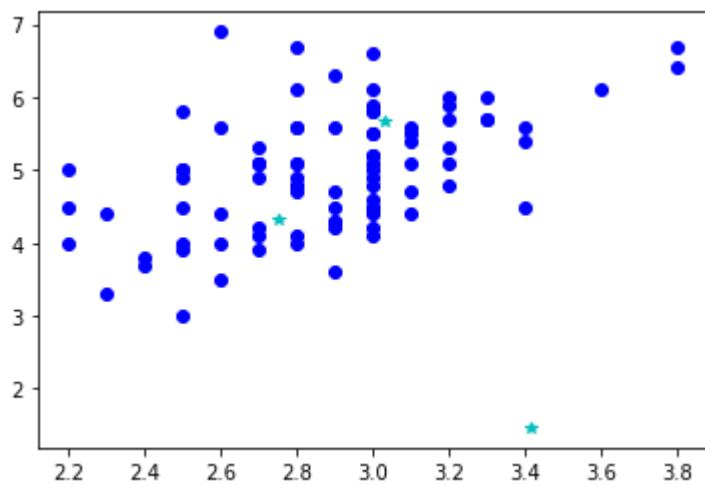
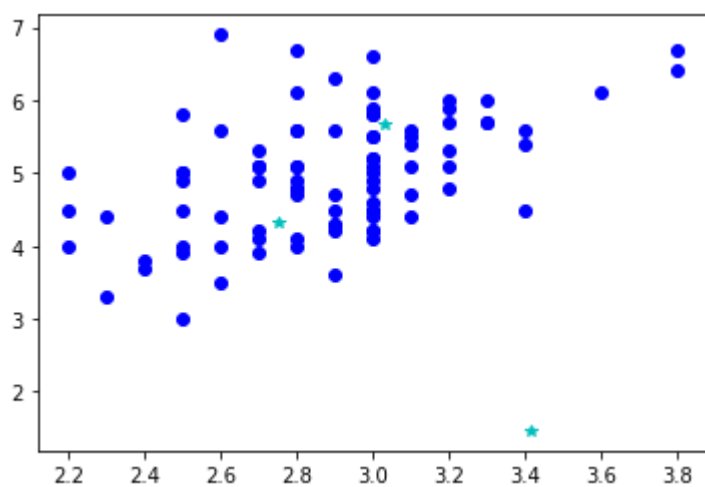
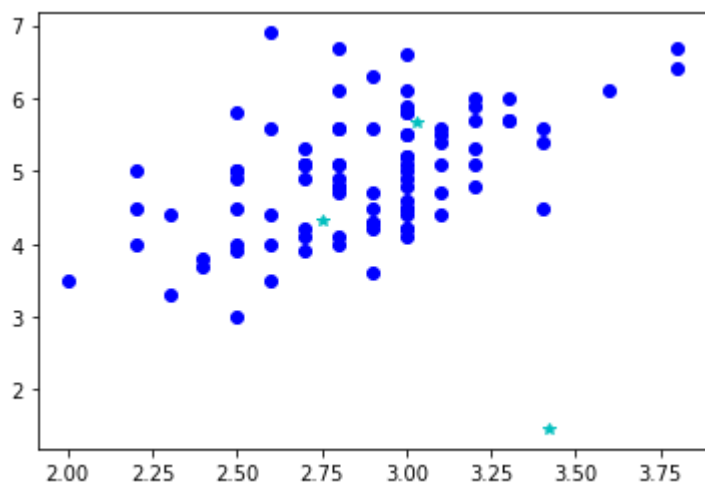
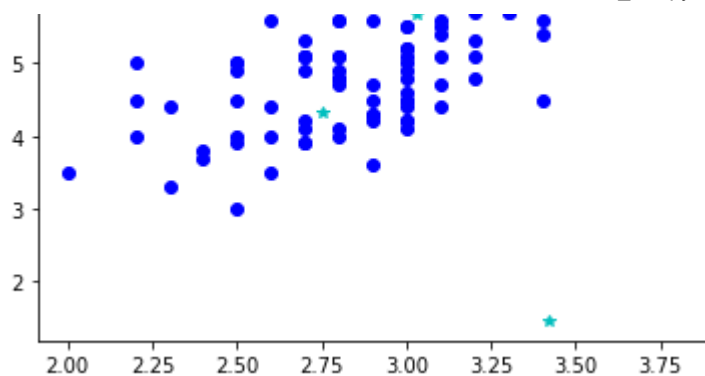


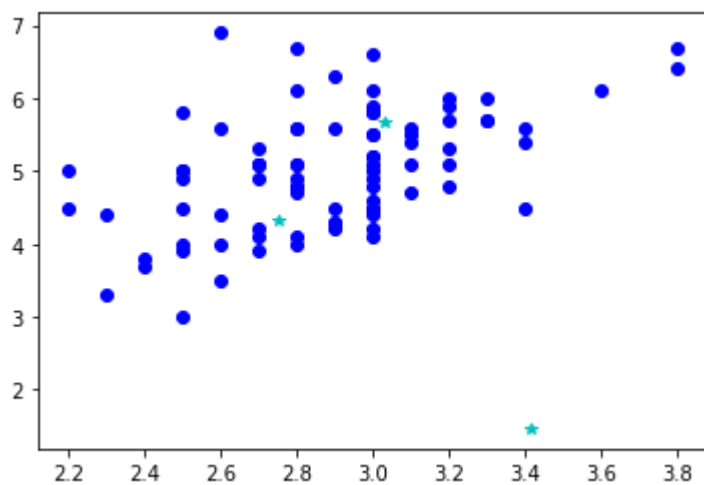
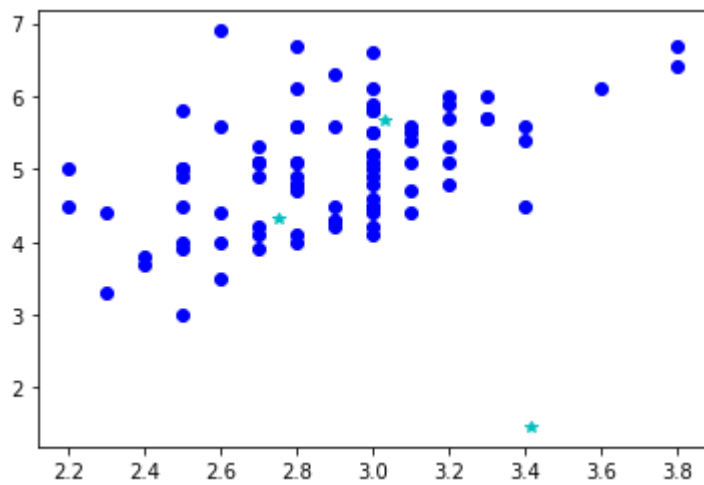
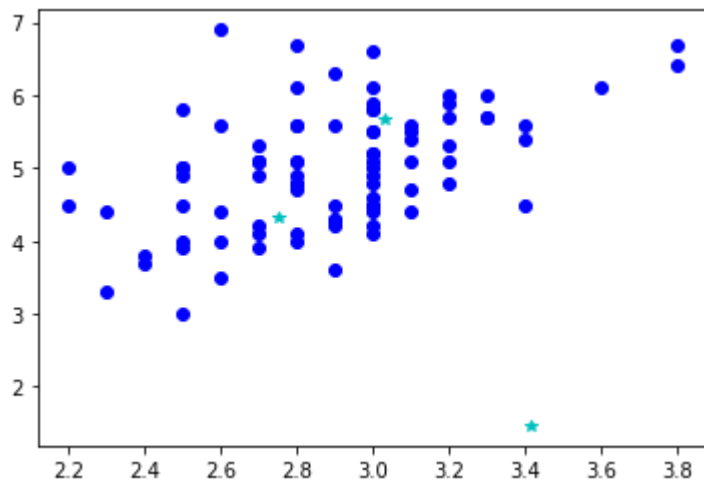
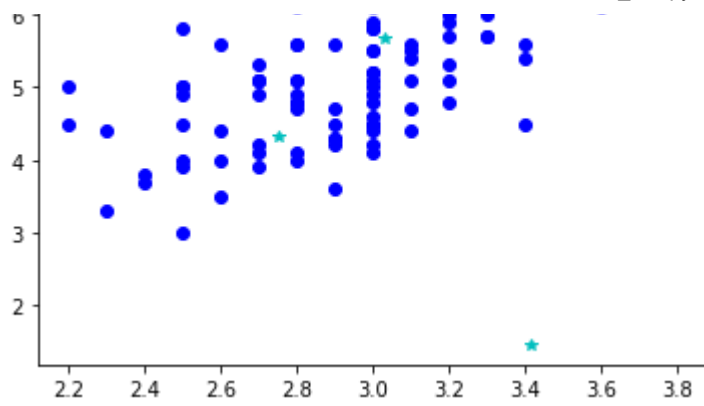


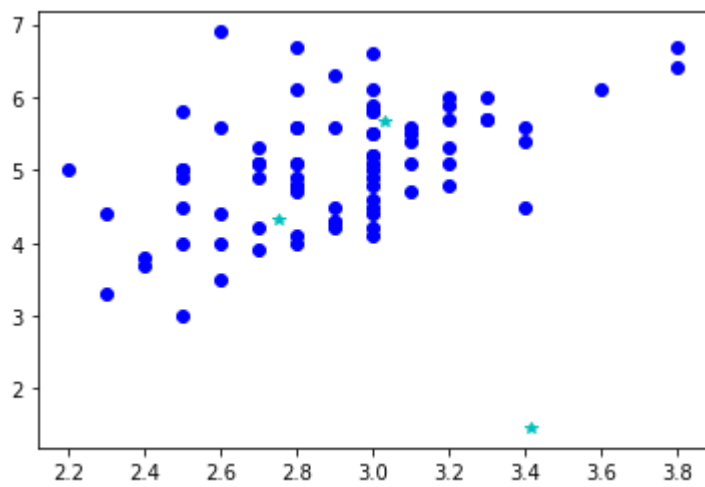
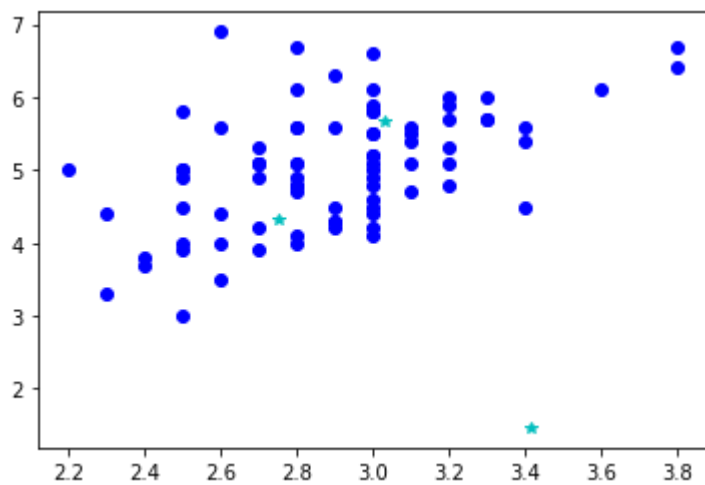
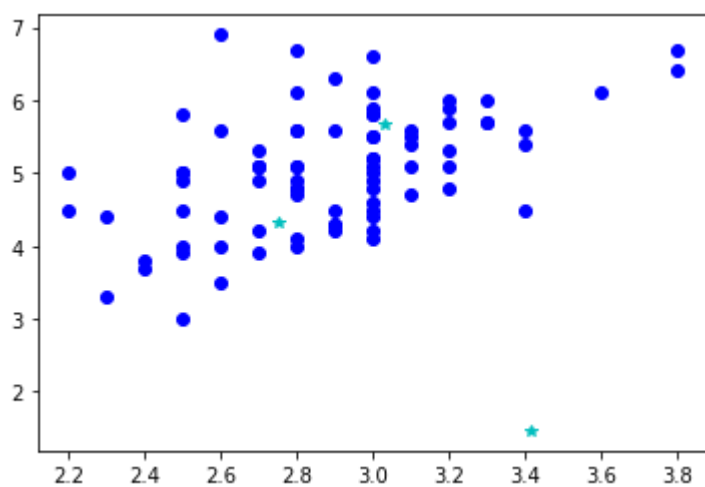
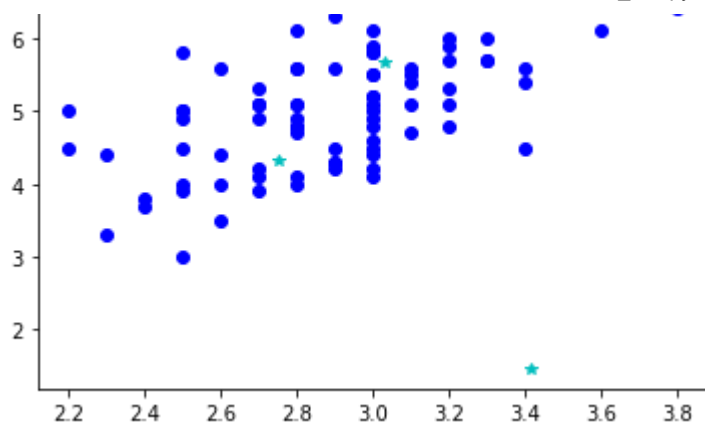


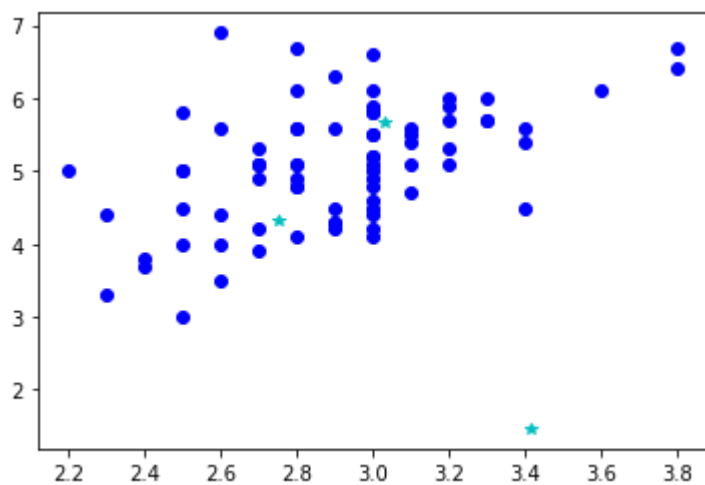
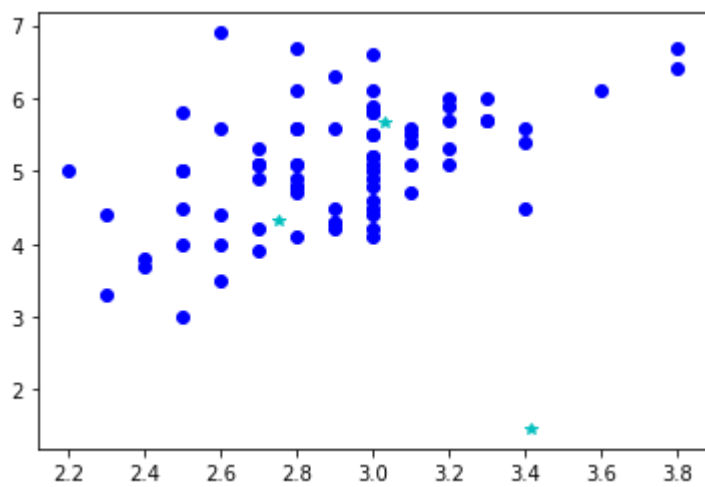
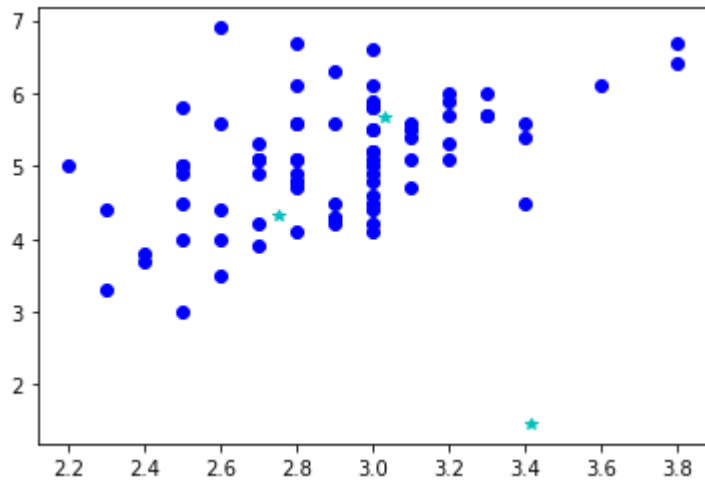
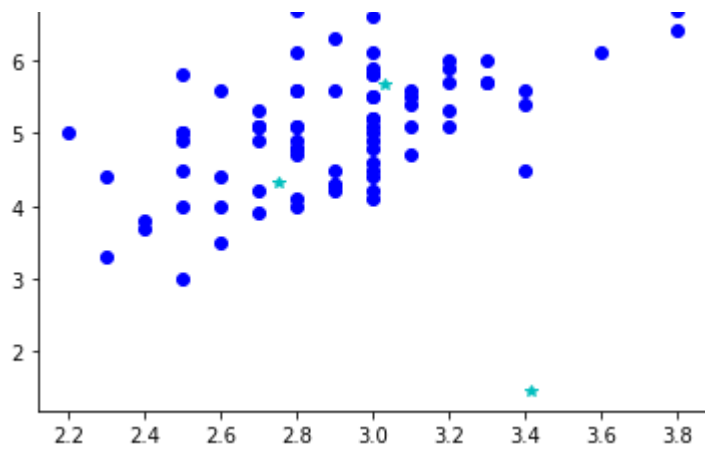


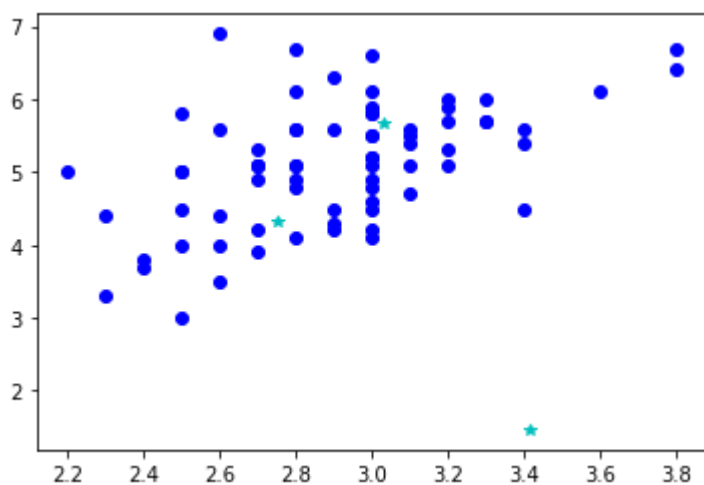
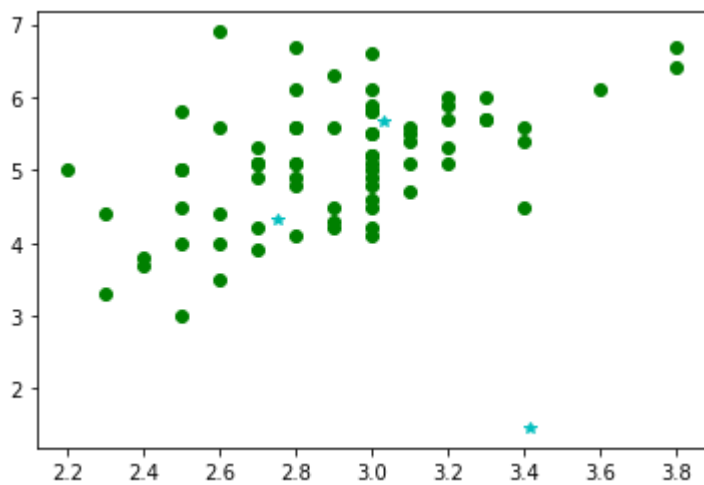
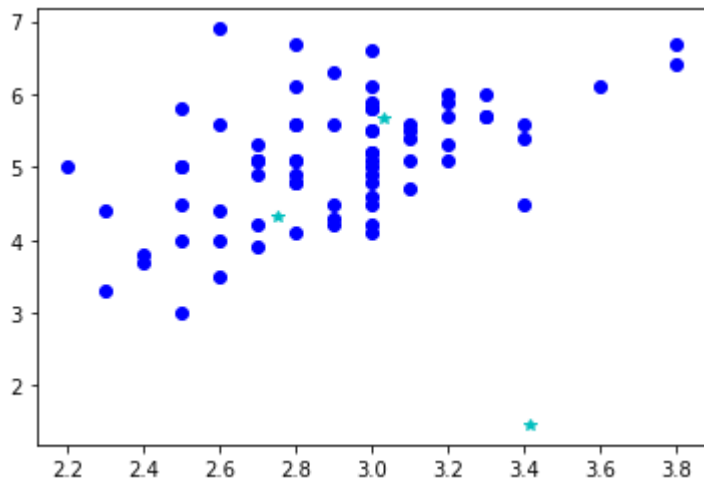
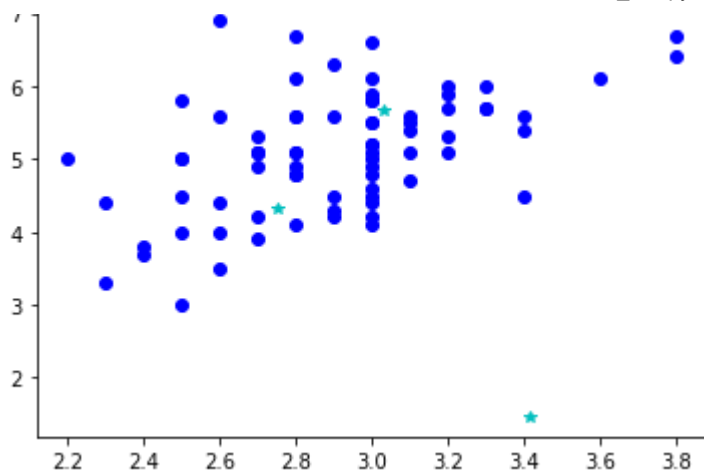


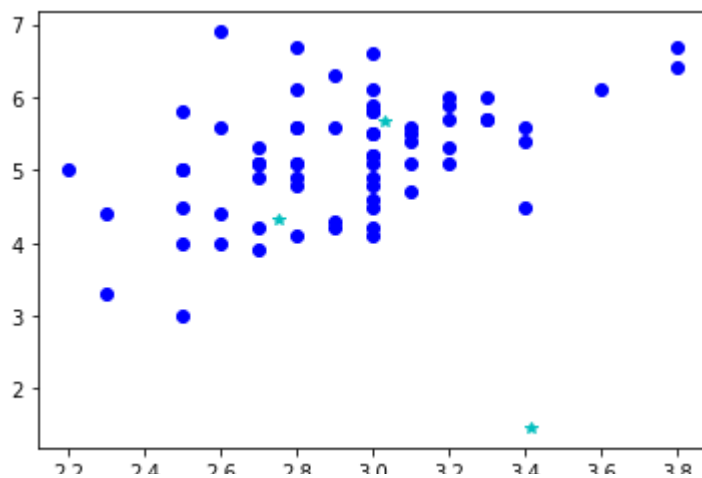
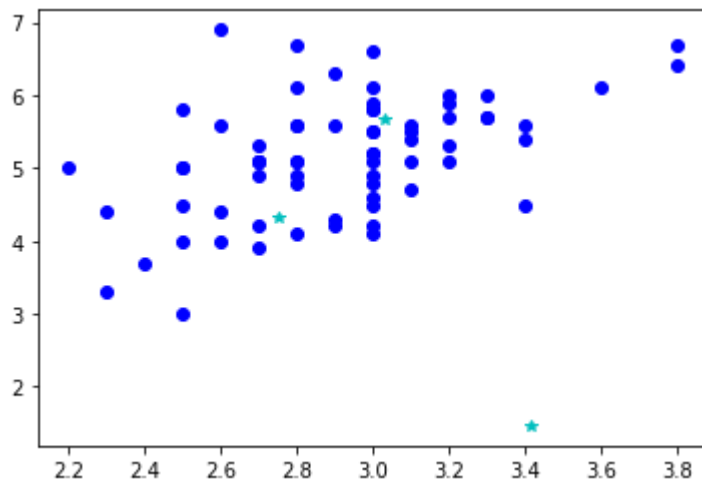
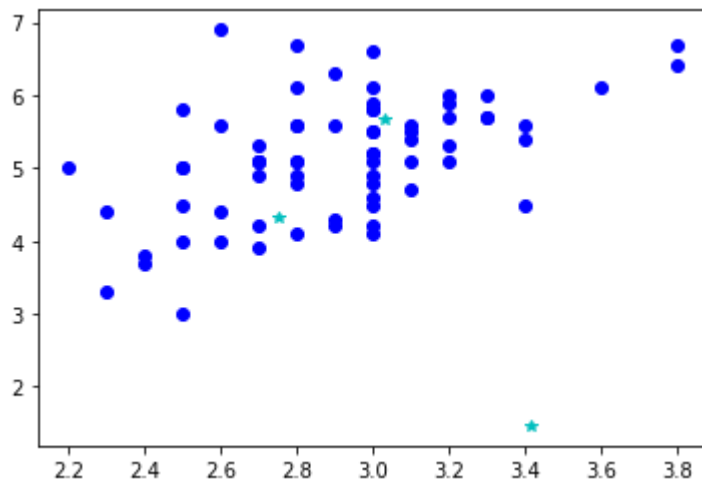
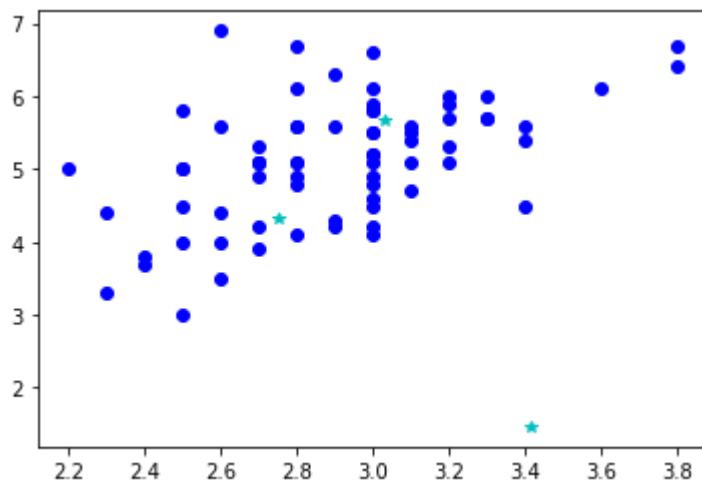


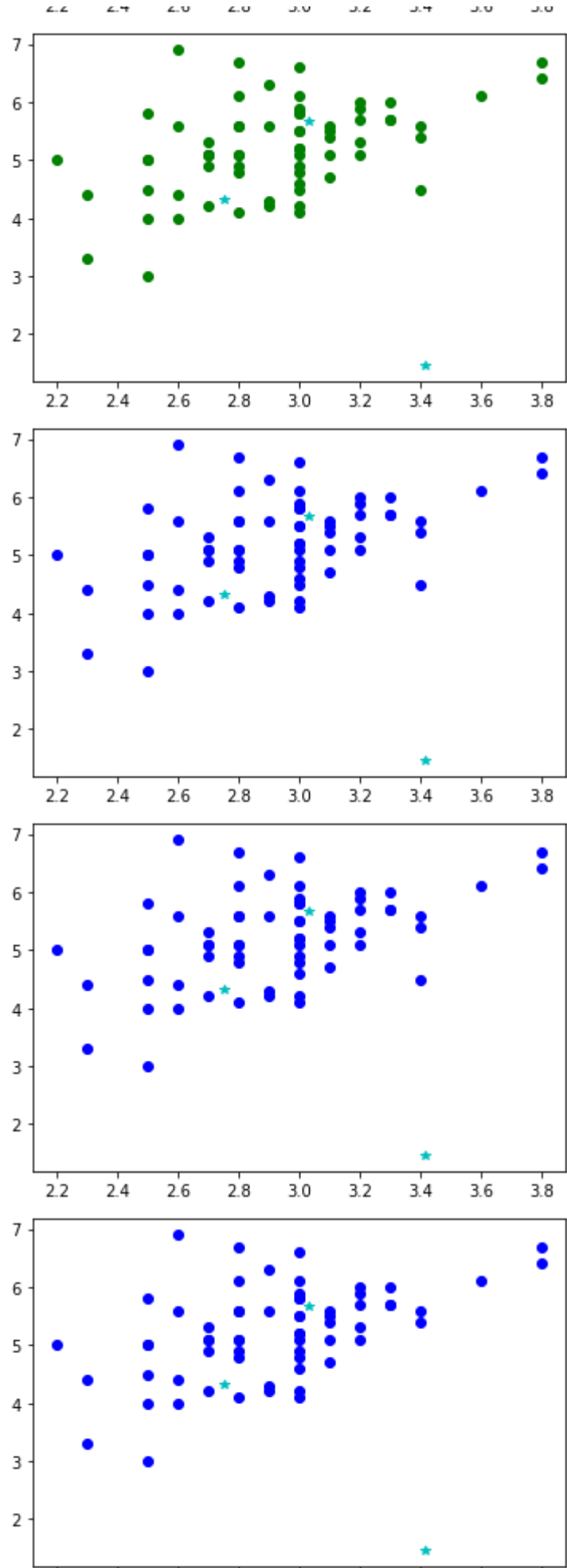


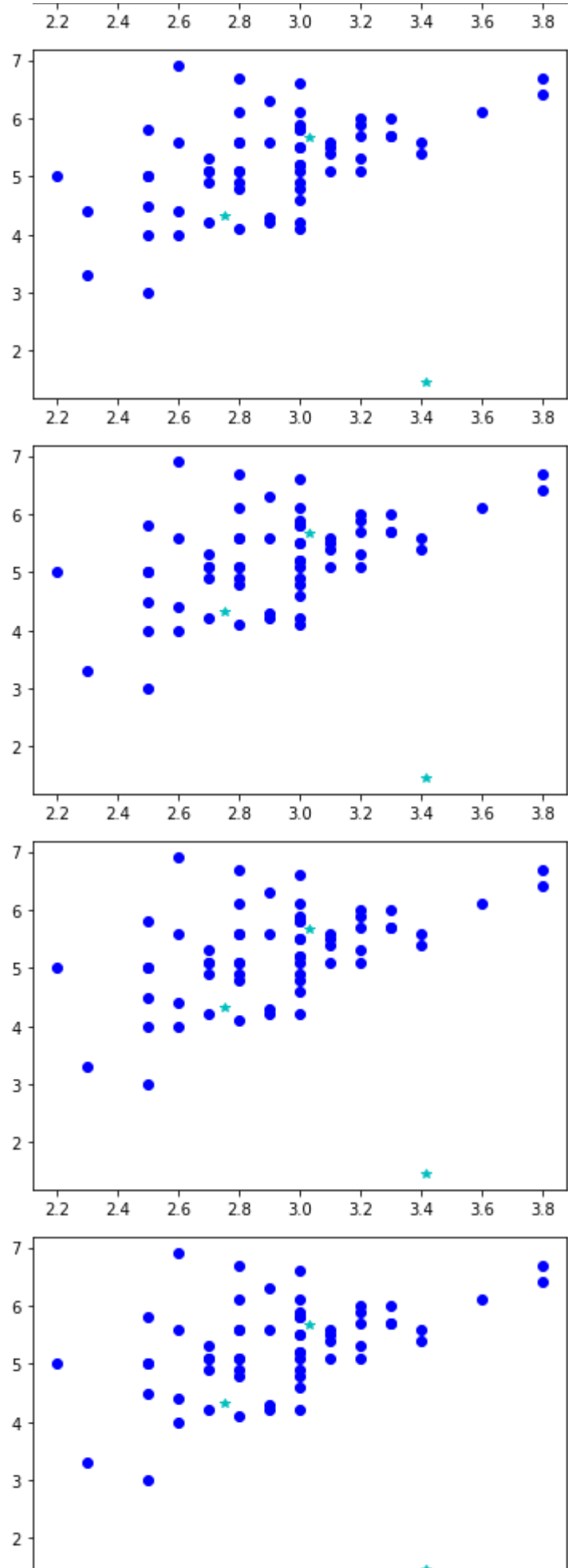


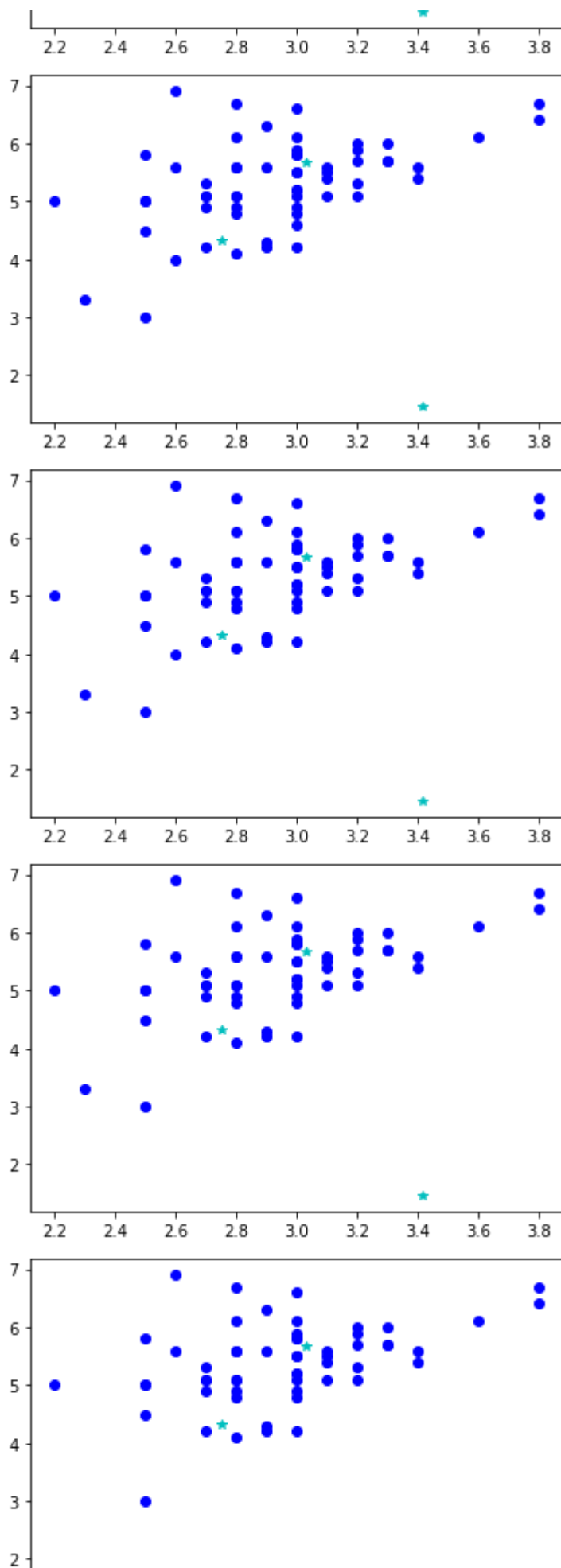


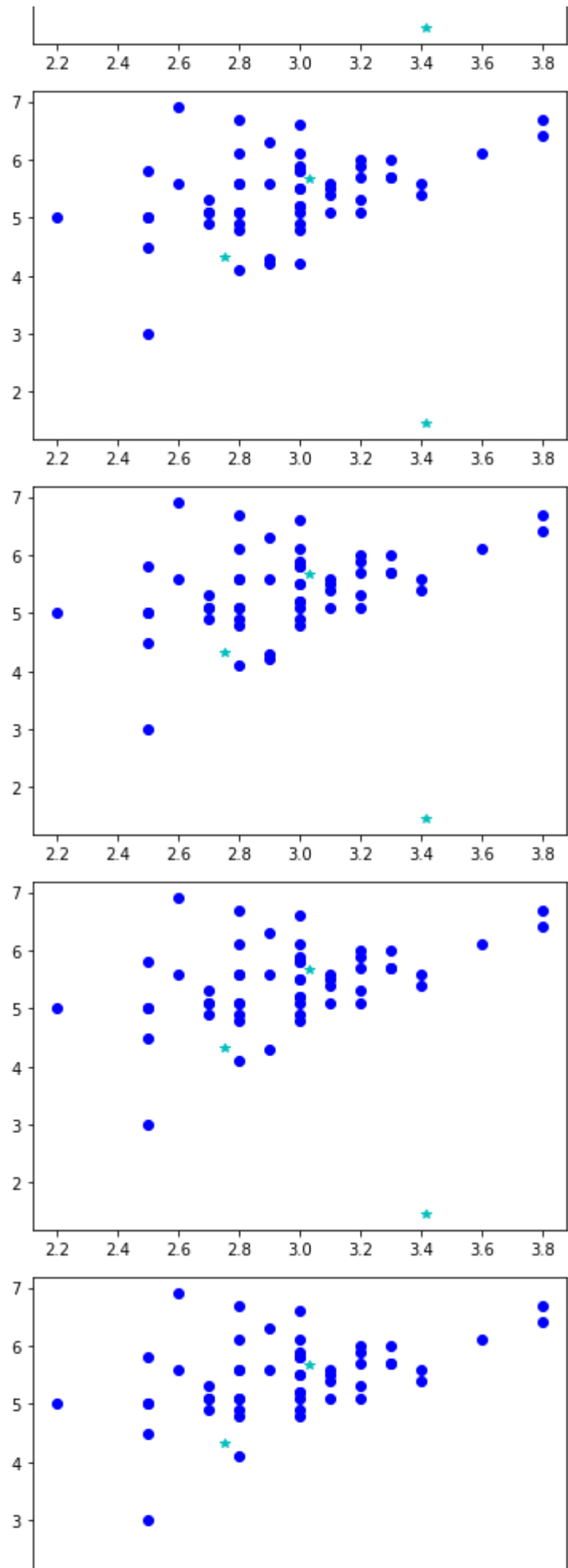


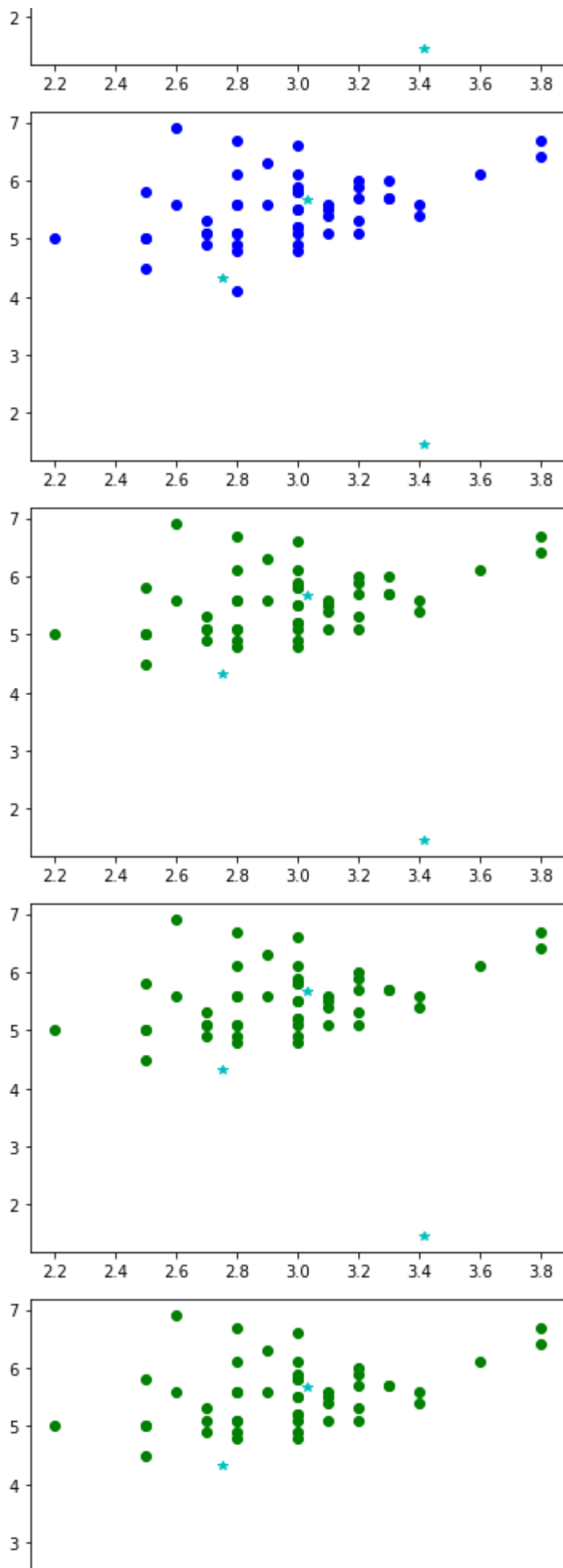


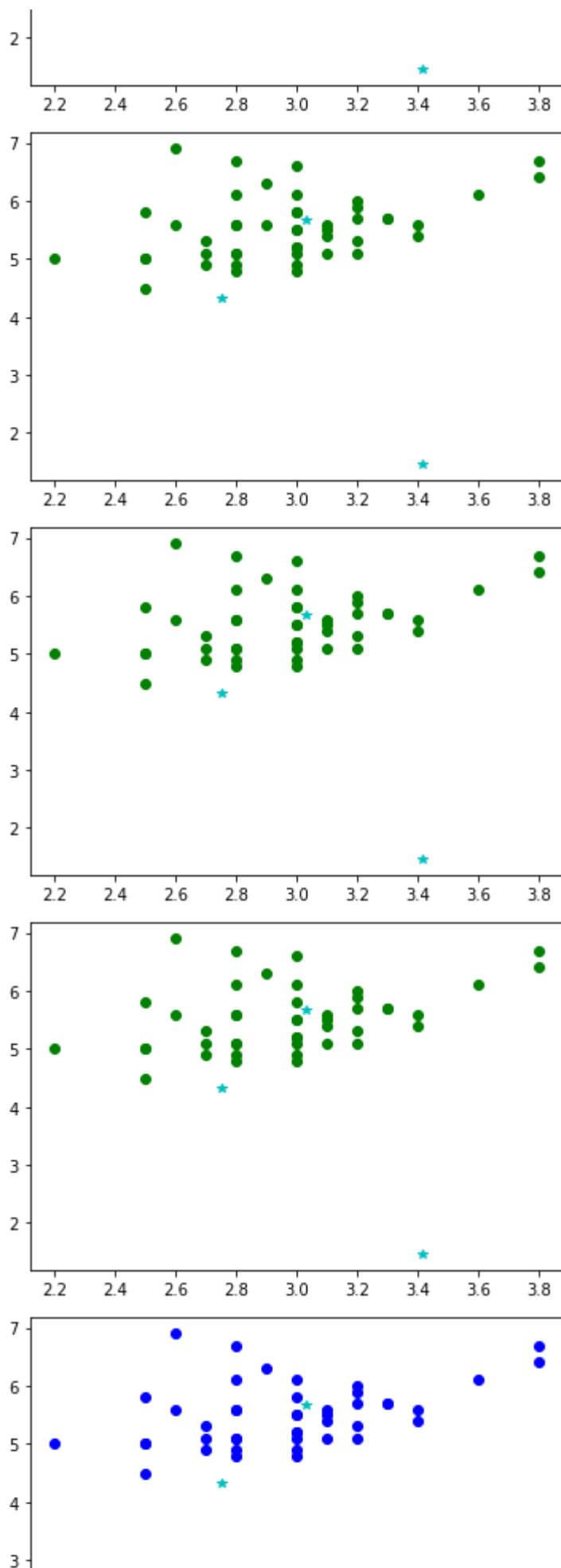


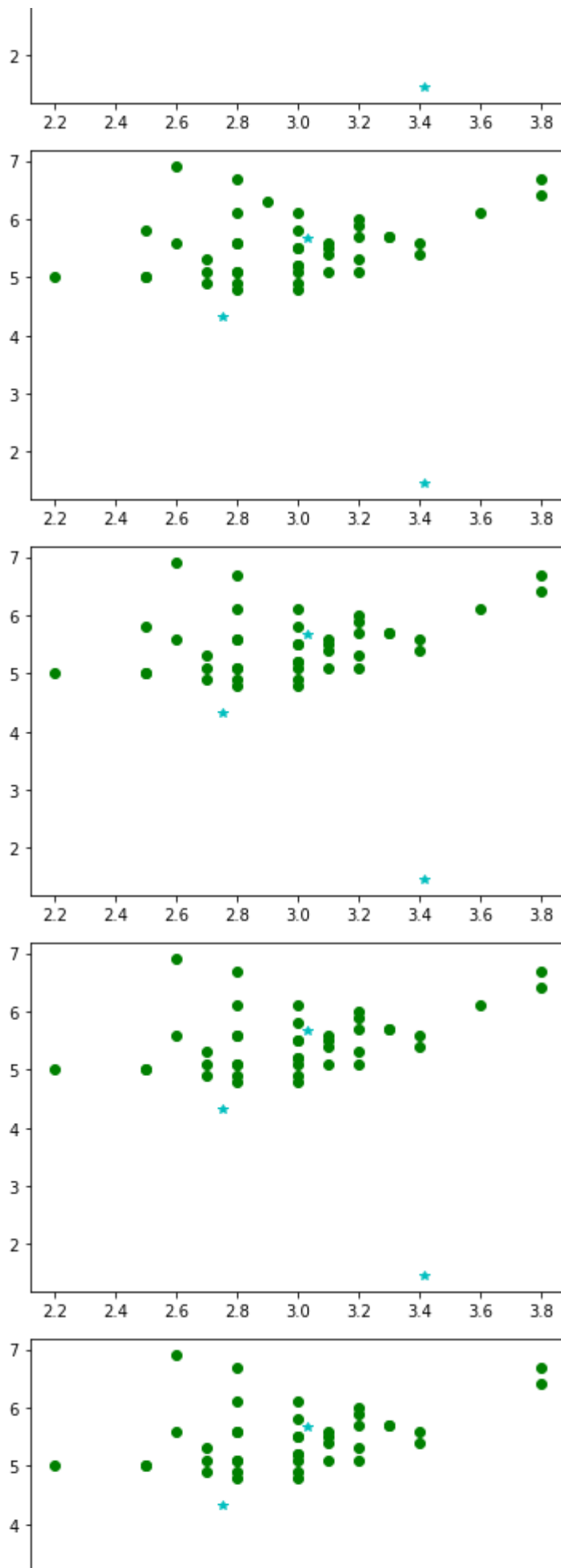


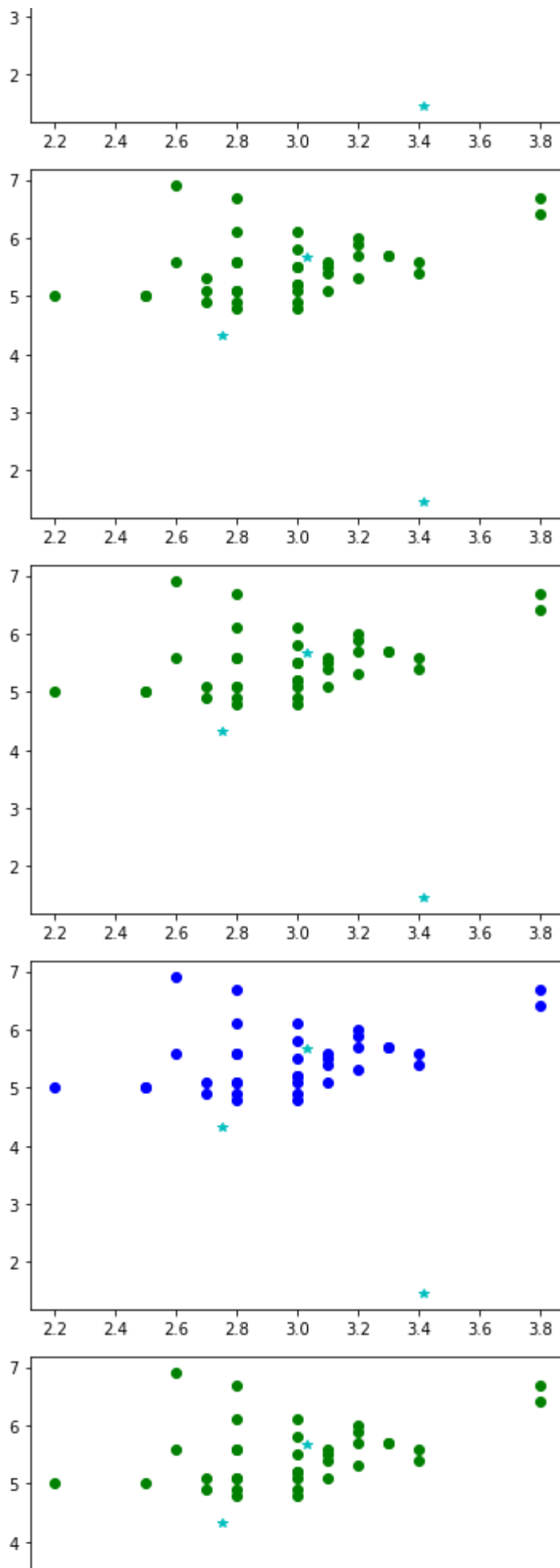


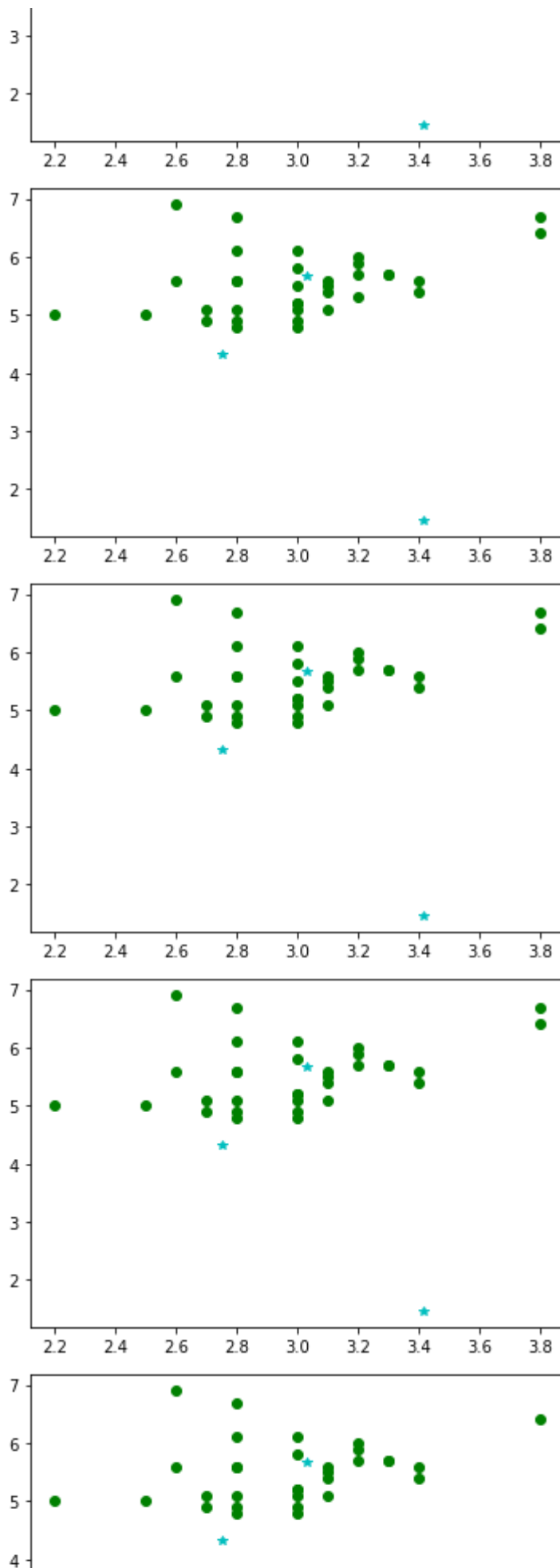


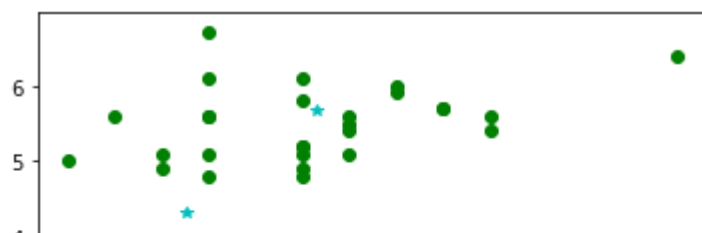
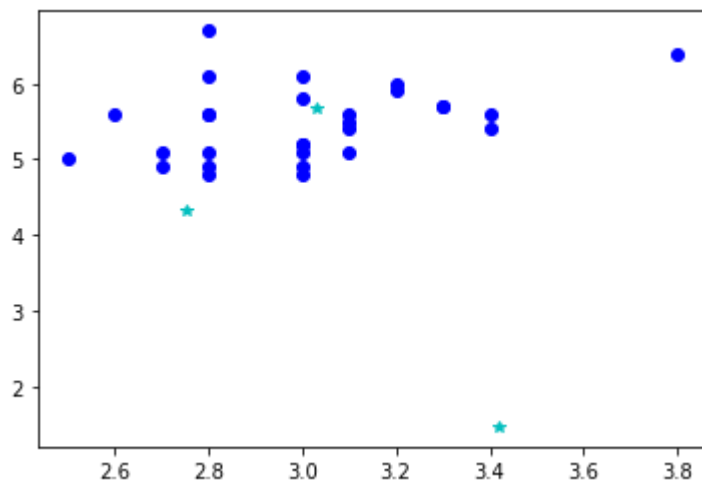
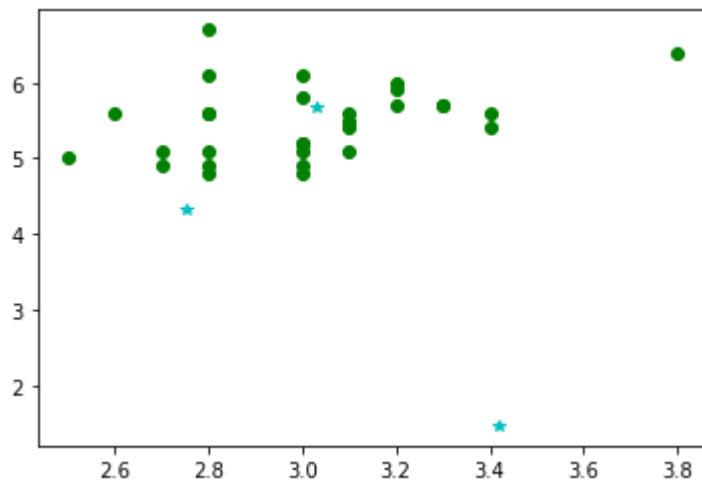
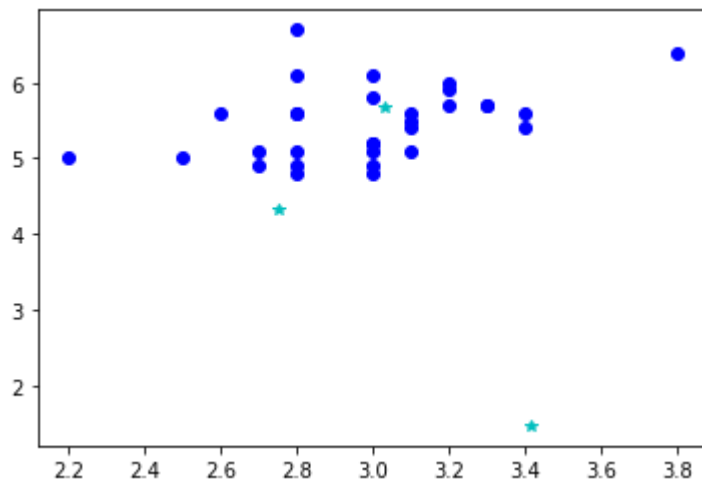
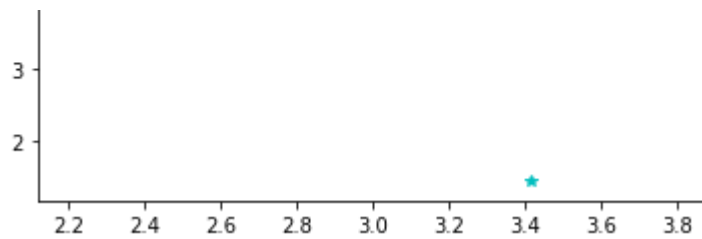


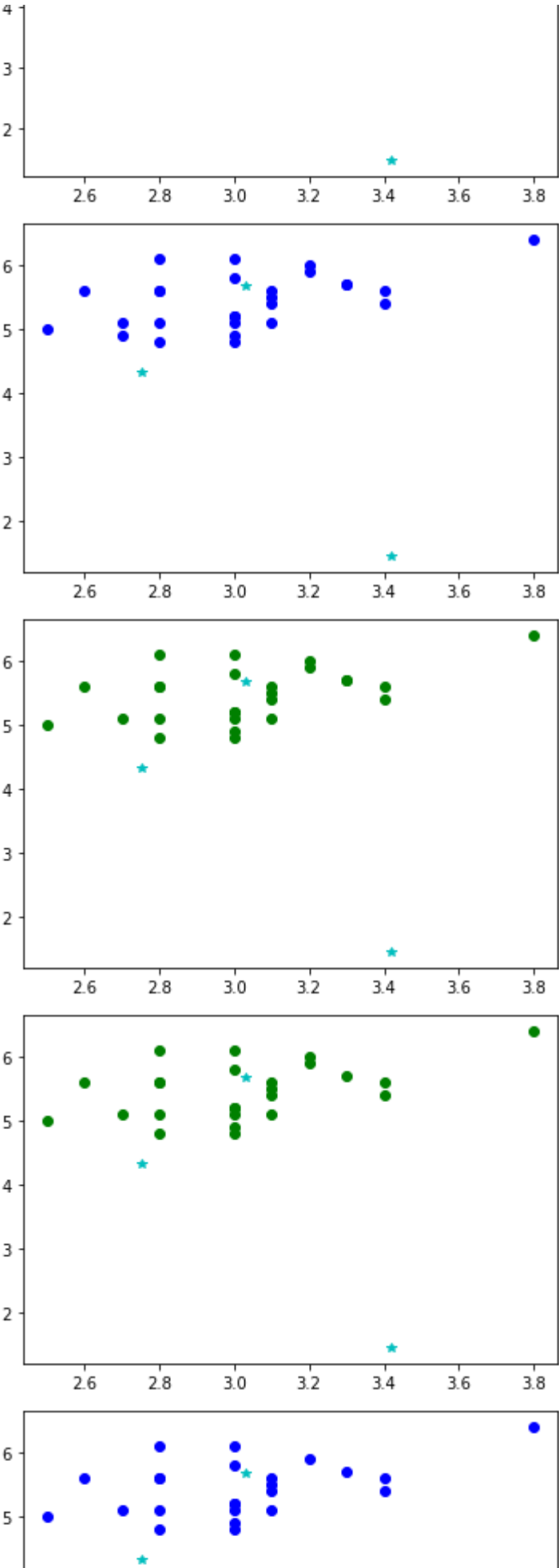


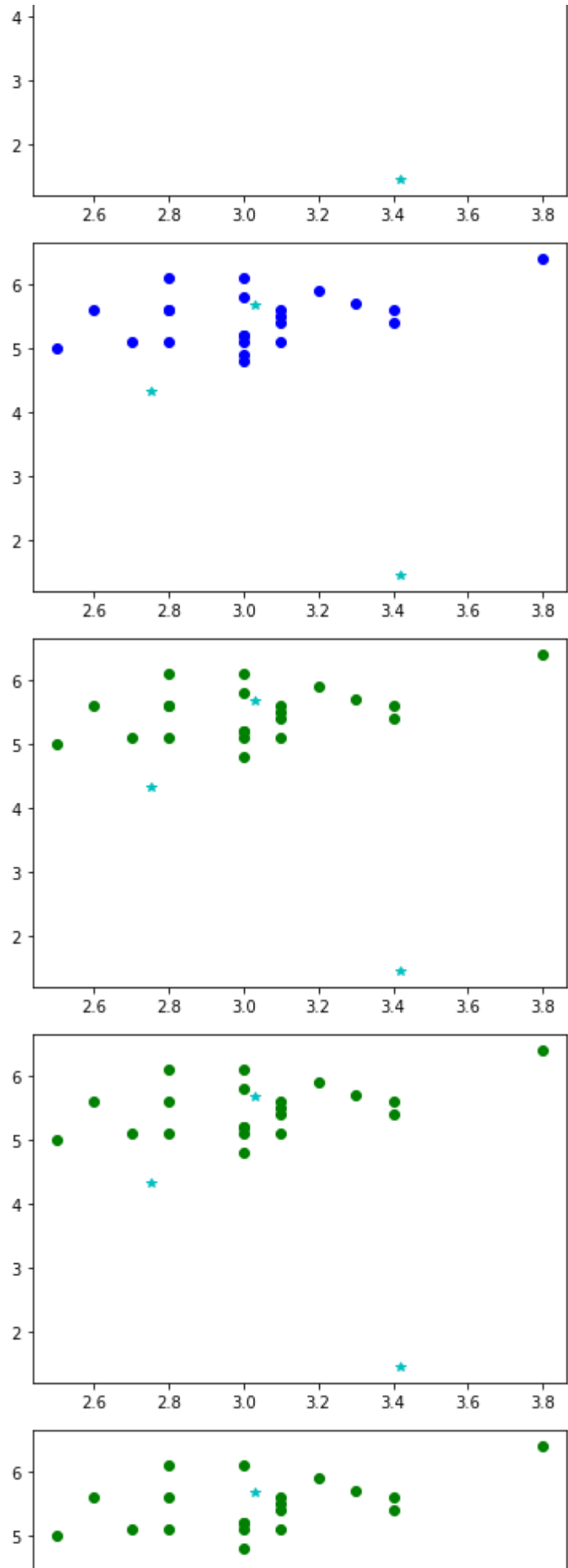


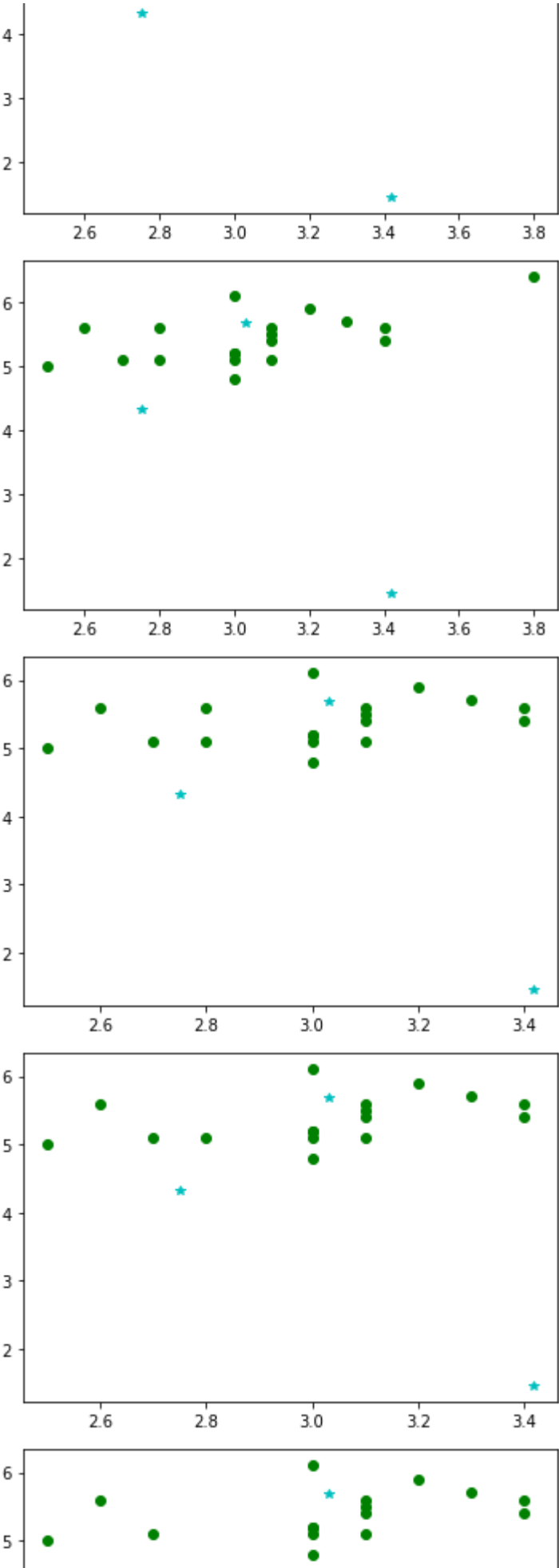


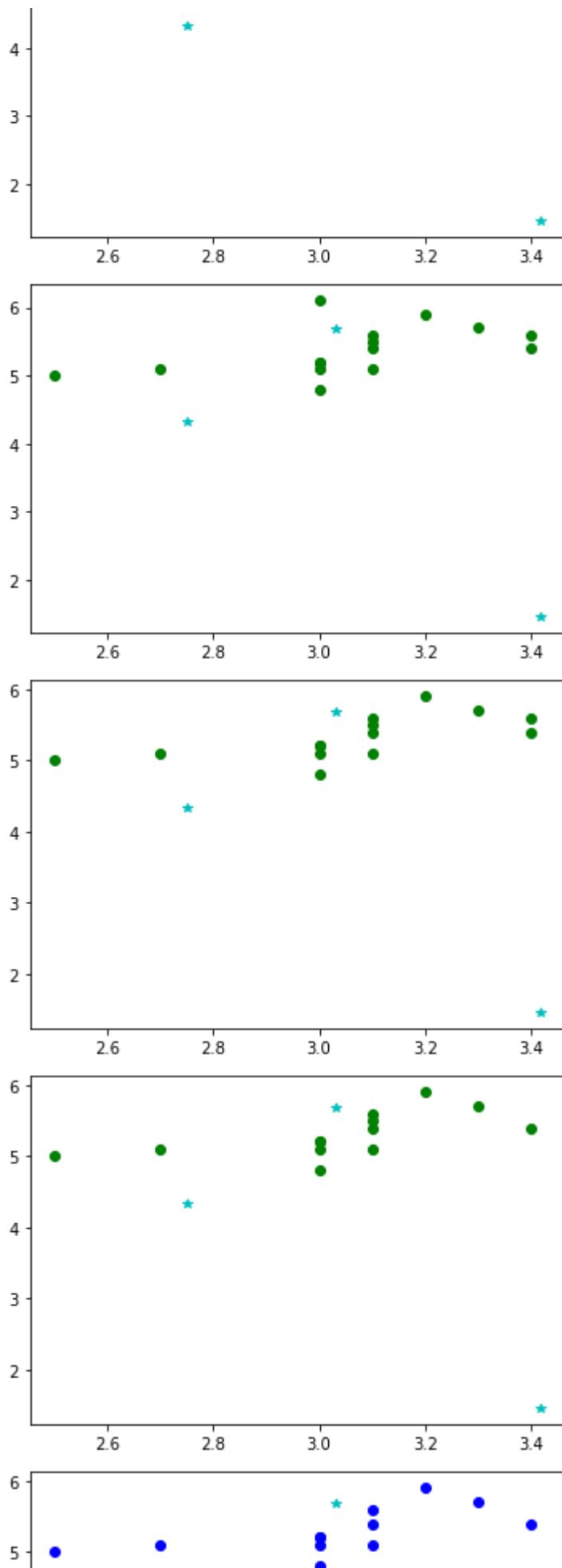


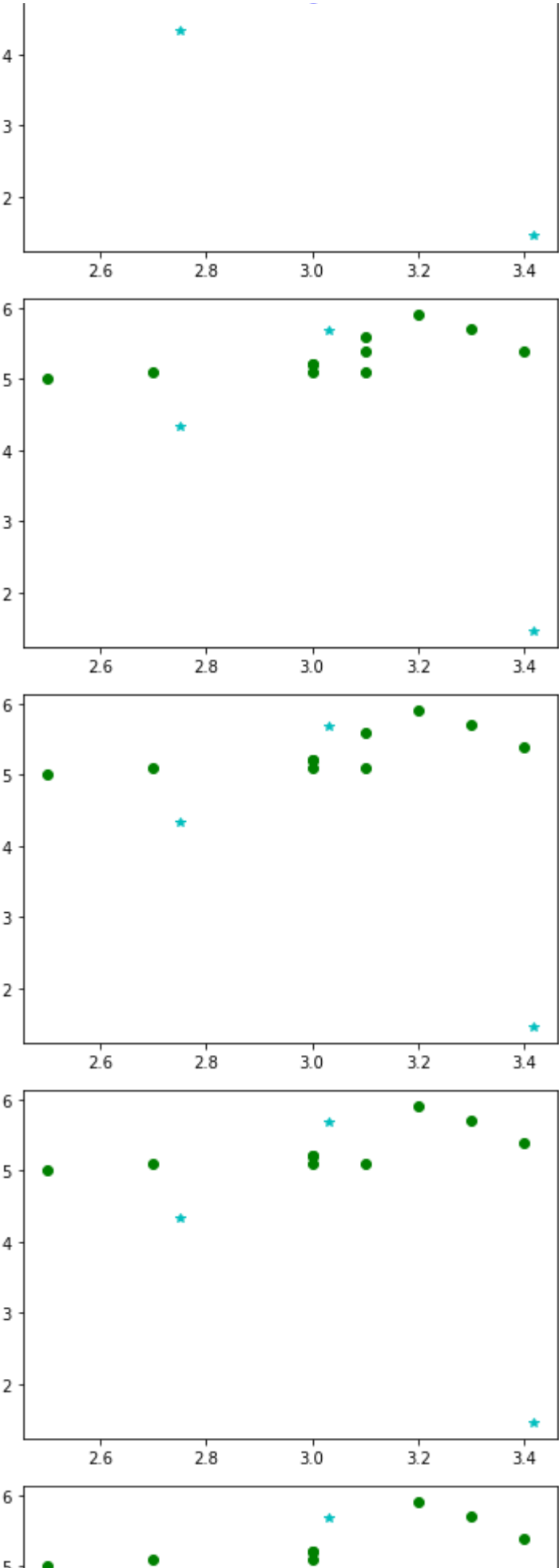


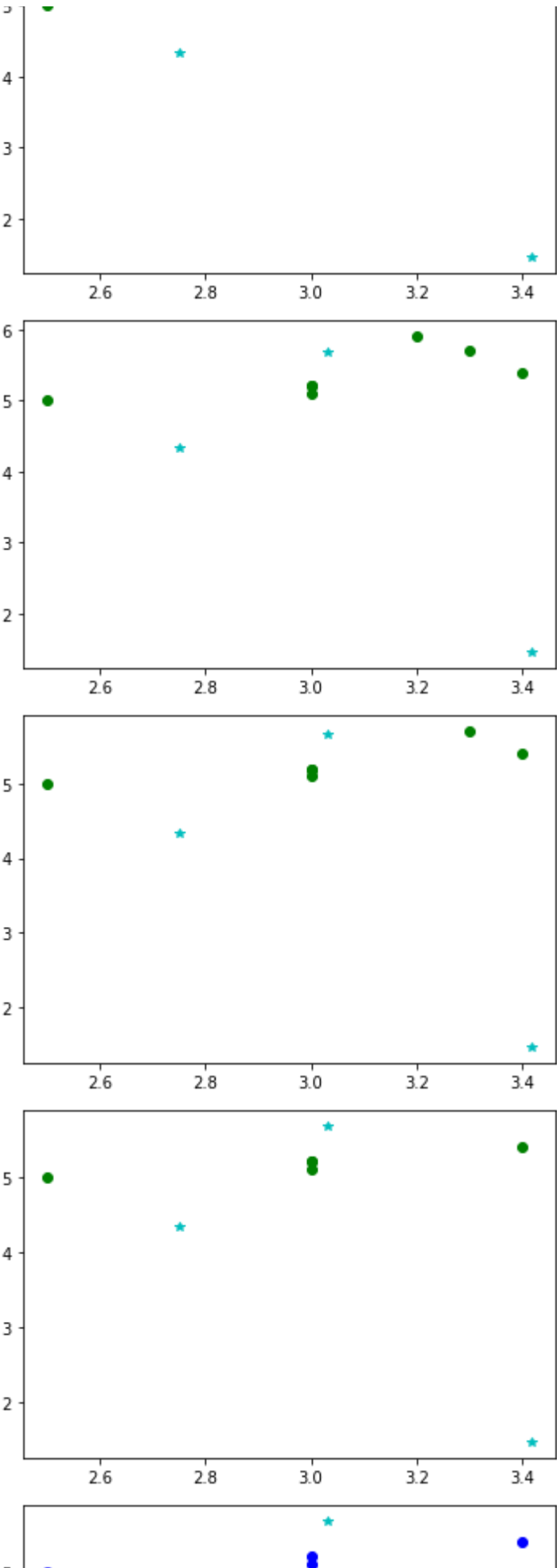


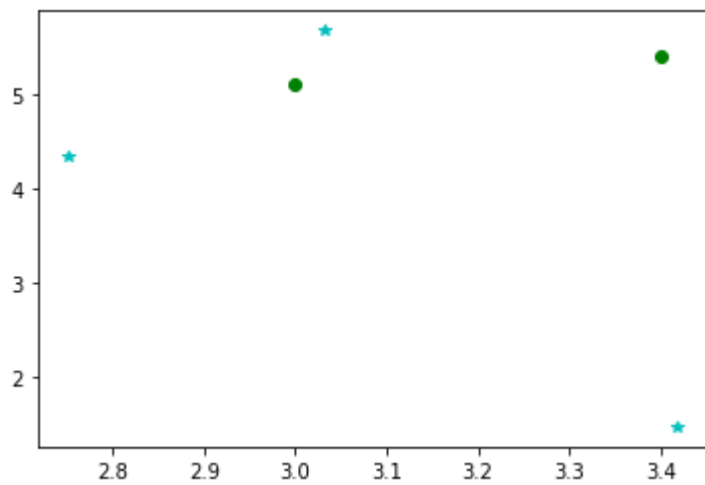
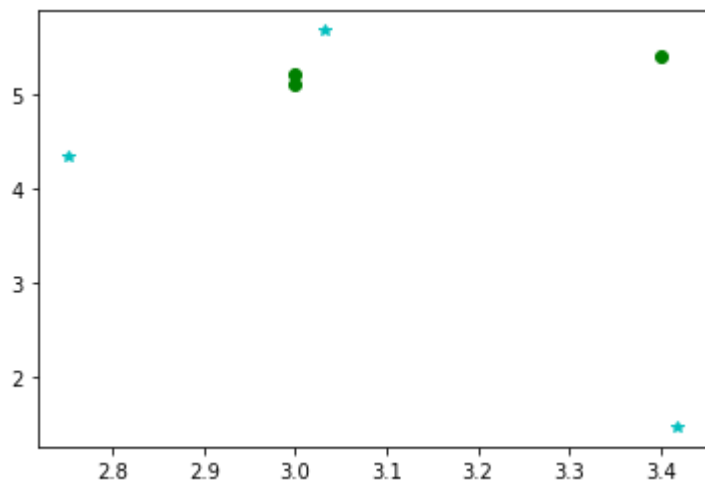
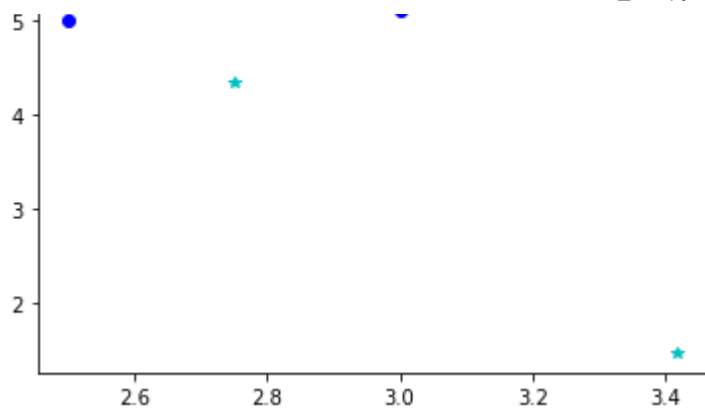












✓ 24s completed at 12:05 AM

● ✕