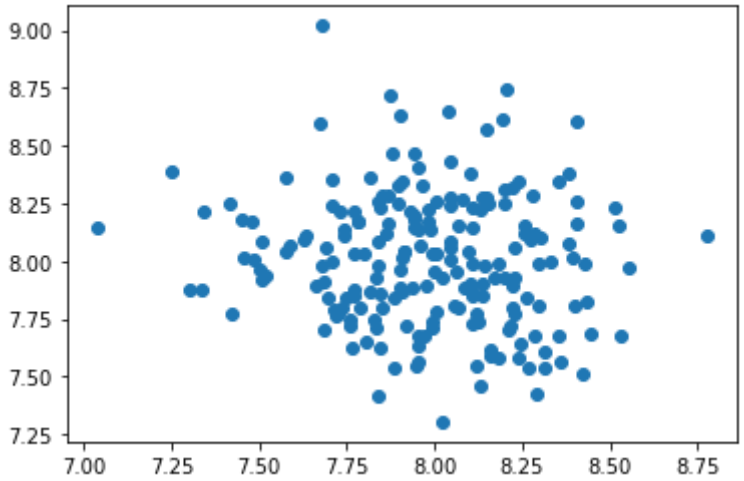
Anomaly Detection_One-Class SVM

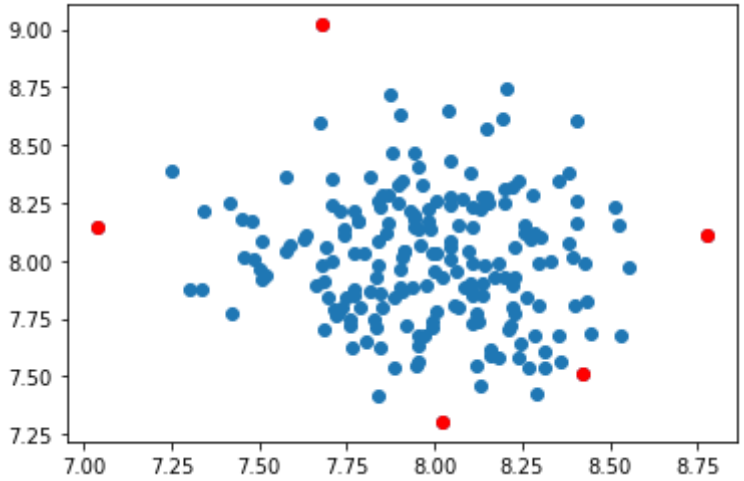
A One-class classification method is used to detect the outliers and anomalies in a dataset. Based on Support Vector Machines (SVM) evaluation, the One-class SVM applies a One-class classification method for novelty detection. The tutorial briefly explains how to detect anomaly in a dataset by using the One-class SVM method in Python.

In this part, we try to create a synthetic data and understand the theory behind One-class SVM for Anomaly detection.

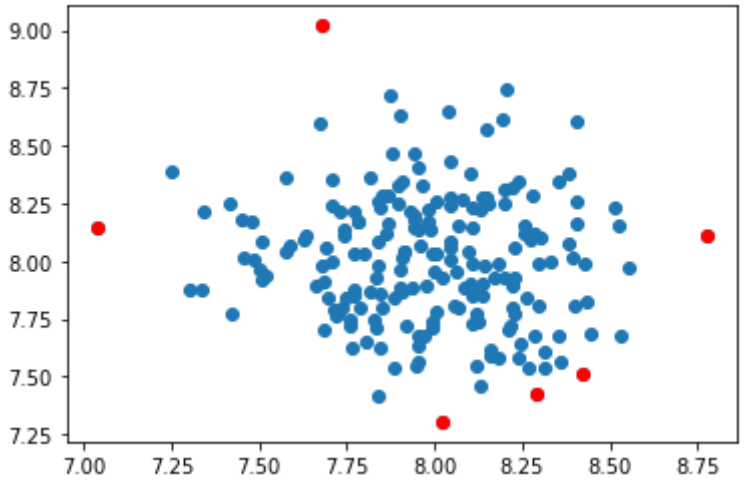
In [1]:
```
from sklearn.svm import OneClassSVM
from sklearn.datasets import make_blobs
from numpy import quantile, where, random
import matplotlib.pyplot as plt

random.seed(13)
x, _ = make_blobs(n_samples=200, centers=1, cluster_std=.3, center_box=(8, 8))

plt.scatter(x[:,0], x[:,1])
plt.show()

svm = OneClassSVM(kernel='rbf', gamma=0.001, nu=0.03)
print(svm)

svm.fit(x)
pred = svm.predict(x)
anom_index = where(pred==-1)
values = x[anom_index]

plt.scatter(x[:,0], x[:,1])
plt.scatter(values[:,0], values[:,1], color='r')
plt.show()

svm = OneClassSVM(kernel='rbf', gamma=0.001, nu=0.02)
print(svm)

pred = svm.fit_predict(x)
scores = svm.score_samples(x)

thresh = quantile(scores, 0.03)
print(thresh)
index = where(scores<=thresh)
values = x[index]

plt.scatter(x[:,0], x[:,1])
plt.scatter(values[:,0], values[:,1], color='r')
plt.show()
```




```
OneClassSVM(gamma=0.001, nu=0.03)
```




```
OneClassSVM(gamma=0.001, nu=0.02)
3.994389673293594
```



In [2]:
```
import pandas as pd
import numpy as np
from sklearn.svm import OneClassSVM
from sklearn.datasets import make_blobs
from numpy import quantile, where, random
import matplotlib.pyplot as plt
```

In [3]:
```
x=pd.read_csv('SOCR-HeightWeight.csv')
print(x)
```

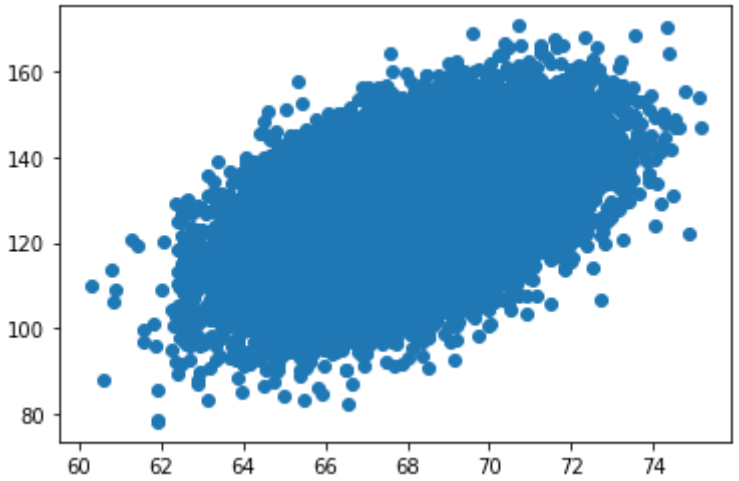
```
       Height  Weight
0       61.93   78.01
1       61.91   78.57
2       66.57   82.38
3       63.13   83.09
4       65.47   83.34
...       ...     ...
24995   72.32  168.23
24996   73.52  168.88
24997   69.57  169.13
24998   74.30  170.55
24999   70.71  170.92

[25000 rows x 2 columns]
```

In [4]:
```
x.shape
```

Out[4]:
```
(25000, 2)
```

In [5]:
```
x= x.values
```

In [6]:
```
plt.scatter(x[:,0], x[:,1])
plt.show()
```



In [7]:
```
svm = OneClassSVM(kernel='rbf', gamma=0.001, nu=0.02)
print(svm)
```

```
OneClassSVM(gamma=0.001, nu=0.02)
```

In [8]:
```
svm.fit(x)
pred = svm.predict(x)
anom_index = where(pred==-1)
values = x[anom_index]
```

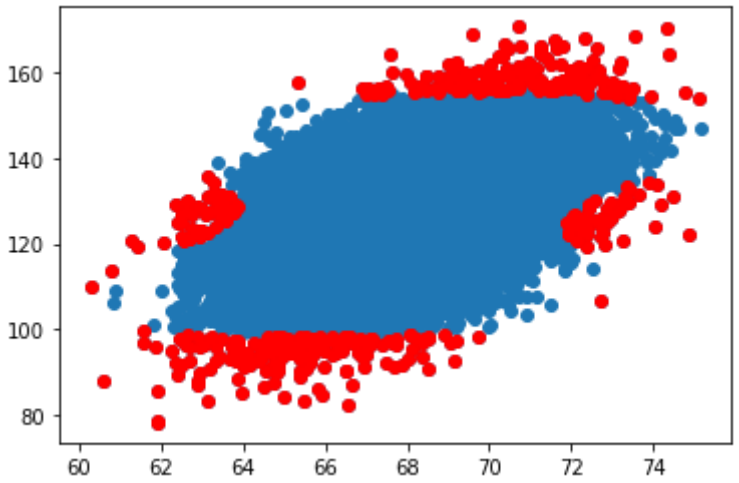
In [9]:
```
print(pred)
```

```
[-1 -1 -1 ... -1 -1 -1]
```

In [10]:
```
print(values)
```

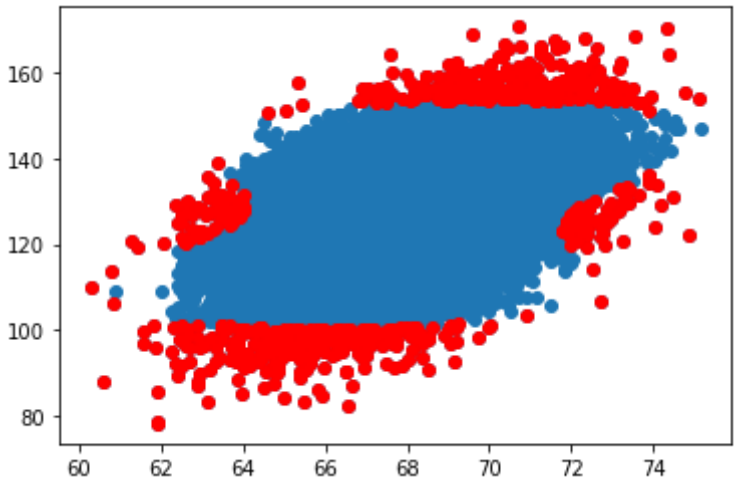
```
[[ 61.93  78.01]
 [ 61.91  78.57]
 [ 66.57  82.38]
 [ 63.13  83.09]
 [ 65.47  83.34]
 [ 65.01  84.26]
 [ 65.94  84.36]
 [ 63.97  84.86]
 [ 61.93  85.29]
 [ 65.84  85.99]
 [ 64.51  86.5 ]
 [ 62.88  86.67]
 [ 66.64  86.98]
 [ 64.74  87.38]
 [ 62.88  87.8 ]
 [ 60.61  88.05]
 [ 63.87  88.17]
 [ 65.36  88.81]
 [ 62.4   89.2 ]
```

In [11]:
```
plt.scatter(x[:,0], x[:,1])
plt.scatter(values[:,0], values[:,1], color='r')
plt.show()
```



In [12]:
```
svm = OneClassSVM(kernel='rbf', gamma=0.001, nu=0.03)
print(svm)

pred = svm.fit_predict(x)
scores = svm.score_samples(x)

thresh = quantile(scores, 0.03)
print(thresh)
index = where(scores<=thresh)
values = x[index]

plt.scatter(x[:,0], x[:,1])
plt.scatter(values[:,0], values[:,1], color='r')
plt.show()
```

```
OneClassSVM(gamma=0.001, nu=0.03)
374.9538840428008
```



In [ ]: