ACHYUT TRIPATHI

17BCE0954

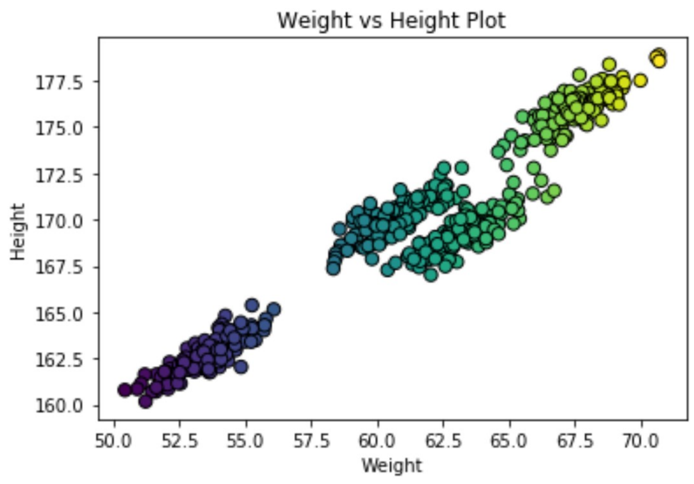
Clustering - Differentiate between K-Means and GMM

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [84]:
```
data = pd.read_csv("Book1.csv")
```

In [85]:
```
#plotting the dataset on matplotlib

plt.scatter(x = data["Weight"], y = data["Height"],c=data["Weight"], s= 55,edgecolo
rs='black')
plt.xlabel("Weight")
plt.ylabel("Height")
plt.title("Weight vs Height Plot")
plt.show()
```



K- Means

In [87]:
```
#import the libraries
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4)
kmeans.fit(data)
```

Out[87]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=4, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

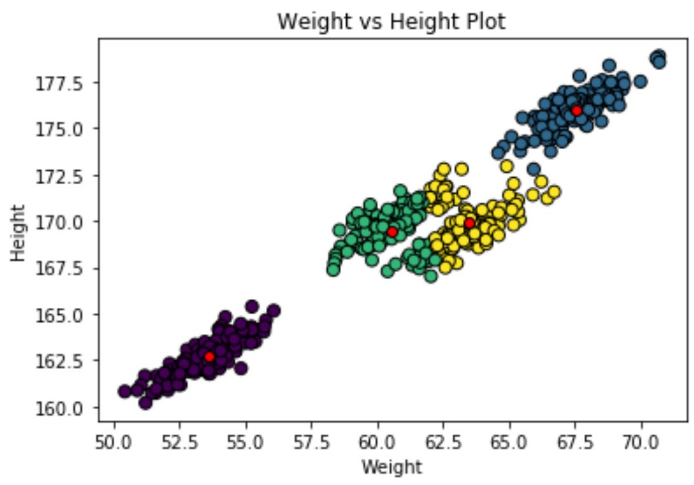
In [91]:
```
# here we store the predicted clusters for the data points
ypred  =kmeans.predict(data)
ypred
```

Out[91]:
```
array([1, 1, 2, 2, 1, 2, 3, 2, 3, 0, 3, 1, 3, 0, 2, 2, 1, 2, 0, 2, 2, 1,
       3, 0, 2, 0, 1, 3, 0, 3, 3, 3, 1, 3, 3, 1, 0, 3, 0, 1, 0, 3, 2, 1,
       3, 1, 3, 2, 1, 1, 0, 0, 3, 3, 2, 3, 3, 0, 0, 1, 1, 0, 1, 1, 1, 3,
       2, 1, 2, 3, 0, 2, 0, 1, 0, 1, 3, 3, 2, 2, 0, 2, 0, 3, 2, 3, 0, 0,
       1, 0, 0, 0, 1, 0, 2, 2, 1, 1, 1, 2, 0, 0, 1, 0, 2, 2, 2, 1, 2, 0,
       0, 3, 0, 0, 1, 0, 1, 2, 3, 1, 3, 0, 1, 1, 0, 1, 1, 1, 1, 0, 2, 0,
       3, 0, 0, 0, 2, 1, 2, 3, 1, 1, 1, 1, 3, 0, 3, 2, 3, 0, 0, 1, 2, 2,
       2, 3, 3, 1, 2, 2, 1, 2, 1, 1, 3, 3, 1, 2, 0, 3, 1, 0, 1, 3, 3, 0,
       3, 0, 0, 0, 1, 1, 3, 2, 2, 0, 3, 1, 3, 0, 1, 3, 0, 3, 3, 0, 2, 3,
       2, 0, 0, 2, 2, 3, 2, 3, 3, 2, 3, 3, 2, 1, 1, 1, 2, 1, 3, 0, 0, 0,
       3, 1, 3, 0, 3, 1, 0, 2, 0, 2, 3, 3, 2, 3, 2, 3, 2, 3, 0, 1, 1, 3,
       2, 1, 0, 0, 2, 1, 2, 3, 2, 3, 1, 0, 2, 0, 0, 3, 0, 3, 2, 2, 1, 0,
       1, 1, 2, 2, 0, 1, 2, 0, 1, 2, 1, 0, 3, 3, 2, 1, 1, 2, 1, 3, 1, 2,
       3, 1, 2, 2, 2, 1, 0, 0, 1, 0, 0, 0, 2, 1, 1, 0, 3, 0, 1, 1, 3, 0,
       0, 2, 0, 3, 1, 3, 0, 3, 0, 2, 3, 2, 2, 2, 3, 3, 0, 1, 2, 3, 0, 1,
       2, 2, 1, 1, 3, 1, 3, 2, 2, 3, 0, 1, 1, 3, 1, 2, 1, 1, 1, 1, 2, 1,
       2, 1, 2, 2, 0, 1, 2, 0, 0, 2, 2, 1, 1, 3, 3, 1, 2, 0, 2, 2, 2, 0,
       2, 2, 2, 0, 0, 2, 0, 0, 2, 0, 0, 3, 2, 3, 3, 0, 0, 0, 0, 1, 0, 1,
       2, 1, 0, 2, 1, 2, 1, 2, 2, 3, 0, 2, 0, 3, 3, 1, 0, 3, 2, 1, 0, 3,
       3, 2, 1, 2, 1, 3, 1, 3, 1, 0, 2, 1, 0, 1, 2, 3, 3, 3, 1, 0, 2, 2,
       3, 1, 1, 2, 2, 3, 2, 2, 0, 1, 3, 1, 0, 2, 3, 3, 0, 3, 2, 0, 3, 3,
       3, 3, 3, 2, 0, 1, 3, 1, 2, 1, 3, 0, 2, 0, 3, 0, 2, 0, 2, 3, 3, 0,
       3, 3, 2, 0, 1, 3, 3, 0, 3, 2, 2, 2, 1, 0, 0, 2])
```

In [92]:
```
#here we get the centroids of the clusters
centroids = kmeans.cluster_centers_
centroids
```

Out[92]:
```
array([[ 53.60029816, 162.76478   ],
       [ 67.52524584, 175.9848496 ],
       [ 60.54594217, 169.41528605],
       [ 63.50413331, 169.9141719 ]])
```

In [93]:
```
#plotting the clusters and the red points denote the centroid
plt.scatter(x = data["Weight"], y = data["Height"], c=ypred,s=50, edgecolors='black
')
for i in centroids:
    plt.scatter(x= i[0], y=i[1], c='red', edgecolors='black')
plt.xlabel("Weight")
plt.ylabel("Height")
plt.title("Weight vs Height Plot")
plt.show()
```



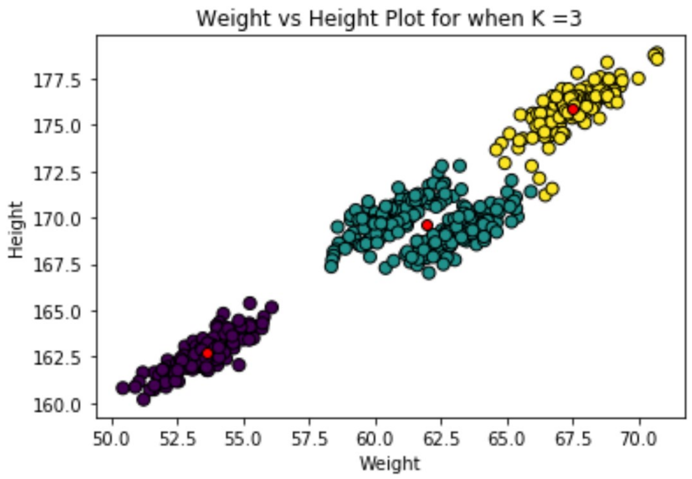
In [104]:
```
kmeans_3 = KMeans(n_clusters=3)
kmeans_3.fit(data)
ypred_2 = kmeans_5.predict(data)
ypred_2
```

Out[104]:
```
array([2, 2, 1, 1, 2, 1, 1, 1, 1, 0, 1, 2, 1, 0, 1, 1, 2, 1, 0, 1, 1, 2,
       1, 0, 1, 0, 2, 1, 0, 1, 1, 1, 2, 1, 1, 2, 0, 1, 0, 2, 0, 1, 1, 2,
       1, 2, 1, 1, 2, 2, 0, 0, 2, 1, 1, 1, 1, 0, 0, 2, 2, 0, 2, 2, 2, 1,
       1, 2, 1, 1, 0, 1, 0, 2, 0, 2, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0,
       2, 0, 0, 0, 2, 0, 1, 1, 2, 2, 2, 1, 0, 0, 2, 0, 1, 1, 1, 2, 1, 0,
       0, 1, 0, 0, 2, 0, 2, 1, 1, 2, 1, 0, 2, 2, 0, 2, 2, 2, 2, 0, 1, 0,
       1, 0, 0, 0, 1, 2, 1, 1, 2, 2, 2, 2, 1, 0, 1, 1, 1, 0, 0, 2, 1, 1,
       1, 1, 1, 2, 1, 1, 2, 1, 2, 2, 1, 1, 2, 1, 0, 1, 2, 0, 2, 1, 1, 0,
       1, 0, 0, 0, 2, 2, 2, 1, 1, 0, 1, 2, 1, 0, 2, 1, 0, 1, 1, 0, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 1, 2, 1, 0, 0, 0,
       1, 2, 1, 0, 1, 2, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 2, 2, 1,
       1, 2, 0, 0, 1, 2, 1, 1, 1, 1, 2, 0, 1, 0, 0, 1, 0, 1, 1, 1, 2, 0,
       2, 2, 1, 1, 0, 2, 1, 0, 2, 1, 2, 0, 1, 1, 1, 2, 2, 1, 2, 1, 2, 1,
       1, 2, 1, 1, 1, 2, 0, 0, 2, 0, 0, 0, 1, 2, 2, 0, 1, 0, 2, 2, 1, 0,
       0, 1, 0, 1, 2, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 2, 1, 1, 0, 2,
       1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 0, 2, 2, 1, 2, 1, 2, 2, 2, 2, 1, 2,
       1, 2, 1, 1, 0, 2, 1, 0, 0, 1, 1, 2, 2, 1, 1, 2, 1, 0, 1, 1, 1, 0,
       1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 2, 0, 2,
       1, 2, 0, 1, 2, 1, 2, 1, 1, 1, 0, 1, 0, 1, 1, 2, 0, 1, 1, 2, 0, 1,
       1, 1, 2, 1, 2, 1, 2, 1, 2, 0, 1, 2, 0, 2, 1, 1, 1, 1, 2, 0, 1, 1,
       1, 2, 2, 1, 1, 1, 1, 1, 0, 2, 1, 2, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 2, 1, 0, 2, 1, 2, 1, 2, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0,
       1, 1, 1, 0, 2, 1, 1, 0, 1, 1, 1, 1, 2, 0, 0, 1])
```

In [105]:
```
centroids_2 = kmeans_3.cluster_centers_
centroids_2
```

Out[105]:
```
array([[ 61.91101512, 169.61922805],
       [ 67.48040837, 175.86017674],
       [ 53.60029816, 162.76478   ]])
```

In [106]:
```
#plotting the clusters and the red points denote the centroid for 3 centroid
plt.scatter(x = data["Weight"], y = data["Height"], c=ypred_2,s=50, edgecolors='bl
ack')
for i in centroids_2:
    plt.scatter(x= i[0], y=i[1], c='red', edgecolors='black')
plt.xlabel("Weight")
plt.ylabel("Height")
plt.title("Weight vs Height Plot for when K =3")
plt.show()
```



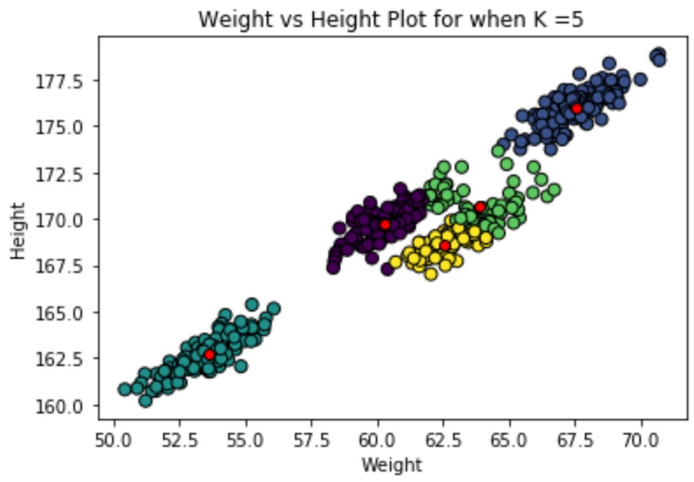
In [107]:
```
# for when number of centroids is 5
kmeans_5 = KMeans(n_clusters=5)
kmeans_5.fit(data)
ypred_3 = kmeans_5.predict(data)
ypred_3
```

Out[107]:
```
array([1, 1, 0, 0, 1, 4, 3, 0, 3, 2, 4, 1, 3, 2, 0, 0, 1, 0, 2, 0, 0, 1,
       3, 2, 0, 2, 1, 4, 2, 4, 3, 3, 1, 3, 4, 1, 2, 3, 2, 1, 2, 3, 0, 1,
       3, 1, 3, 0, 1, 1, 2, 2, 3, 3, 0, 4, 3, 2, 2, 1, 1, 2, 1, 1, 1, 4,
       0, 1, 0, 4, 2, 0, 2, 1, 2, 1, 3, 3, 0, 0, 2, 0, 2, 3, 0, 3, 2, 2,
       1, 2, 2, 2, 1, 2, 0, 0, 1, 1, 1, 0, 2, 2, 1, 2, 0, 0, 4, 1, 0, 2,
       2, 4, 2, 2, 1, 2, 1, 0, 3, 1, 3, 2, 1, 1, 2, 1, 1, 1, 1, 2, 0, 2,
       4, 2, 2, 2, 0, 1, 0, 3, 1, 1, 1, 1, 4, 2, 3, 0, 3, 2, 2, 1, 0, 0,
       0, 4, 4, 1, 4, 0, 1, 0, 1, 1, 4, 3, 1, 0, 2, 3, 1, 2, 1, 4, 3, 2,
       3, 2, 2, 2, 1, 1, 3, 0, 4, 2, 4, 1, 4, 2, 1, 4, 2, 4, 3, 2, 0, 4,
       0, 2, 2, 0, 4, 3, 0, 3, 3, 0, 4, 3, 4, 1, 1, 1, 0, 1, 3, 2, 2, 2,
       3, 1, 4, 2, 4, 1, 2, 0, 2, 4, 3, 4, 4, 4, 0, 3, 0, 3, 2, 1, 1, 3,
       4, 1, 2, 2, 0, 1, 0, 3, 0, 4, 1, 2, 0, 2, 2, 4, 2, 4, 0, 0, 1, 2,
       1, 1, 0, 0, 2, 1, 0, 2, 1, 4, 1, 2, 4, 4, 0, 1, 1, 0, 1, 4, 1, 0,
       3, 1, 0, 0, 0, 1, 2, 2, 1, 2, 2, 2, 0, 1, 1, 2, 3, 2, 1, 1, 3, 2,
       2, 0, 2, 3, 1, 3, 2, 3, 2, 4, 4, 0, 0, 0, 3, 4, 2, 1, 0, 4, 2, 1,
       0, 0, 1, 1, 3, 1, 3, 4, 0, 4, 2, 1, 1, 3, 3, 0, 1, 1, 1, 1, 0, 1,
       0, 1, 0, 0, 2, 3, 0, 2, 2, 4, 0, 1, 1, 3, 3, 1, 4, 2, 0, 0, 0, 2,
       0, 4, 0, 2, 2, 0, 2, 2, 4, 2, 2, 4, 4, 3, 3, 2, 2, 2, 2, 1, 2, 1,
       0, 1, 2, 0, 1, 0, 1, 0, 4, 4, 2, 0, 2, 3, 3, 1, 2, 3, 0, 1, 2, 3,
       4, 0, 1, 0, 1, 3, 1, 3, 1, 2, 4, 1, 2, 1, 0, 3, 3, 4, 1, 2, 0, 0,
       4, 1, 1, 0, 0, 3, 0, 4, 2, 1, 4, 1, 2, 4, 3, 4, 2, 3, 0, 2, 3, 4,
       3, 4, 3, 0, 2, 1, 4, 1, 0, 1, 4, 2, 0, 2, 3, 2, 0, 2, 0, 4, 3, 2,
       4, 4, 4, 2, 1, 3, 4, 2, 3, 4, 4, 0, 1, 2, 2, 0])
```

```
In [108]: centroids_5 = kmeans_5.cluster_centers_
          centroids_5

Out[108]: array([[ 60.29094962, 169.72307333],
                 [ 67.56197073, 176.02969431],
                 [ 53.60029816, 162.76478   ],
                 [ 63.87843919, 170.71679595],
                 [ 62.5672026 , 168.58458767]])
```

```
In [130]: #plotting the clusters and the red points denote the centroid for 3 centroid
          plt.scatter(x = data["Weight"], y = data["Height"], c=ypred_3,s=50, edgecolors='bl
          ack')
          for i in centroids_5:
              plt.scatter(x= i[0], y=i[1], c='red', edgecolors='black')
          plt.xlabel("Weight")
          plt.ylabel("Height")
          plt.title("Weight vs Height Plot for when K =5")
          plt.show()
```



GMM model

```
In [68]: from sklearn.mixture import GaussianMixture
```

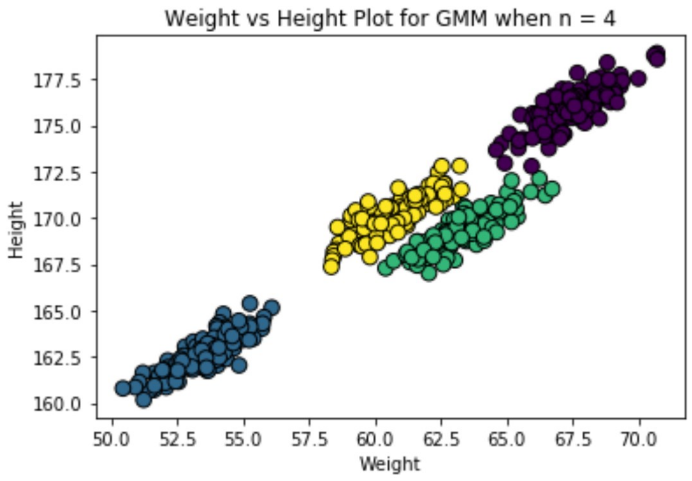
```
In [113]: gmm4 = GaussianMixture(n_components=4)
          gmm4.fit(data)

Out[113]: GaussianMixture(covariance_type='full', init_params='kmeans', max_iter=100,
                          means_init=None, n_components=4, n_init=1, precisions_init=None,
                          random_state=None, reg_covar=1e-06, tol=0.001, verbose=0,
                          verbose_interval=10, warm_start=False, weights_init=None)
```

```
In [115]: y1 = gmm4.predict(data)
          y1
```

```
Out[115]: array([0, 0, 3, 3, 0, 2, 2, 3, 3, 1, 2, 0, 2, 1, 3, 3, 0, 3, 1, 3, 3, 0,
                 2, 1, 3, 1, 0, 2, 1, 2, 2, 2, 0, 2, 2, 0, 1, 3, 1, 0, 1, 3, 3, 0,
                 2, 0, 3, 3, 0, 0, 1, 1, 2, 3, 3, 2, 2, 1, 1, 0, 0, 1, 0, 0, 0, 2,
                 3, 0, 3, 2, 1, 3, 1, 0, 1, 0, 2, 2, 3, 3, 1, 3, 1, 3, 3, 2, 1, 1,
                 0, 1, 1, 1, 0, 1, 3, 3, 0, 0, 0, 3, 1, 1, 0, 1, 3, 3, 2, 0, 3, 1,
                 1, 2, 1, 1, 0, 1, 0, 3, 2, 0, 3, 1, 0, 0, 1, 0, 0, 0, 0, 1, 3, 1,
                 2, 1, 1, 1, 3, 0, 3, 2, 0, 0, 0, 0, 2, 1, 3, 3, 2, 1, 1, 0, 3, 3,
                 3, 2, 2, 0, 2, 3, 0, 3, 0, 0, 2, 2, 0, 3, 1, 2, 0, 1, 0, 2, 3, 1,
                 3, 1, 1, 1, 0, 0, 0, 3, 2, 1, 2, 0, 2, 1, 0, 2, 1, 2, 2, 1, 3, 2,
                 3, 1, 1, 3, 2, 2, 3, 2, 2, 3, 2, 2, 2, 0, 0, 0, 3, 0, 2, 1, 1, 1,
                 3, 0, 2, 1, 2, 0, 1, 3, 1, 2, 2, 2, 2, 2, 3, 3, 3, 2, 1, 0, 0, 2,
                 2, 0, 1, 1, 3, 0, 3, 3, 3, 2, 0, 1, 3, 1, 1, 2, 1, 2, 3, 3, 0, 1,
                 0, 0, 3, 3, 1, 0, 3, 1, 0, 2, 0, 1, 2, 2, 3, 0, 0, 3, 0, 2, 0, 2,
                 2, 0, 3, 3, 3, 0, 1, 1, 0, 1, 1, 1, 3, 0, 0, 1, 3, 1, 0, 0, 2, 1,
                 1, 3, 1, 3, 0, 3, 1, 2, 1, 2, 2, 3, 3, 3, 2, 2, 1, 0, 3, 2, 1, 0,
                 3, 3, 0, 0, 2, 0, 2, 2, 3, 2, 1, 0, 0, 3, 0, 3, 0, 0, 0, 0, 3, 0,
                 3, 0, 3, 3, 1, 0, 3, 1, 1, 2, 3, 0, 0, 2, 2, 0, 2, 1, 3, 3, 3, 1,
                 3, 2, 3, 1, 1, 3, 1, 1, 2, 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 0, 1, 0,
                 3, 0, 1, 3, 0, 3, 0, 3, 2, 2, 1, 3, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2,
                 2, 3, 0, 3, 0, 2, 0, 3, 0, 1, 2, 0, 1, 0, 3, 2, 3, 2, 0, 1, 3, 3,
                 2, 0, 0, 3, 3, 2, 3, 2, 1, 0, 2, 0, 1, 2, 2, 2, 1, 2, 3, 1, 2, 2,
                 2, 2, 2, 3, 1, 0, 2, 0, 3, 0, 2, 1, 3, 1, 2, 1, 3, 1, 3, 2, 2, 1,
                 2, 2, 2, 1, 0, 2, 2, 1, 3, 2, 2, 3, 0, 1, 1, 3], dtype=int64)
```

```
In [134]: plt.scatter(x = data["Weight"], y = data["Height"], c=y1,s=75, edgecolors='black')
          plt.xlabel("Weight")
          plt.ylabel("Height")
          plt.title("Weight vs Height Plot for GMM when n = 4")
          plt.show()
```



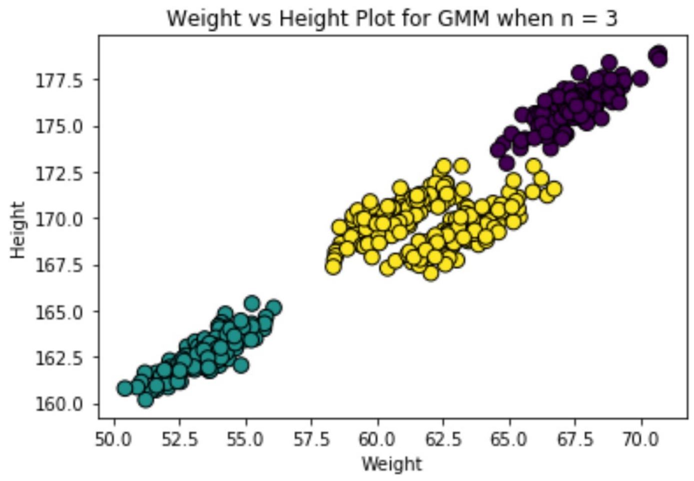
```
In [120]: gmm3 = GaussianMixture(n_components=3)
          gmm3.fit(data)
```

```
Out[120]: GaussianMixture(covariance_type='full', init_params='kmeans', max_iter=100,
                          means_init=None, n_components=3, n_init=1, precisions_init=None,
                          random_state=None, reg_covar=1e-06, tol=0.001, verbose=0,
                          verbose_interval=10, warm_start=False, weights_init=None)
```

```
In [121]: y2 = gmm3.predict(data)
          y2
```

```
Out[121]: array([0, 0, 2, 2, 0, 2, 2, 2, 2, 1, 2, 0, 2, 1, 2, 2, 0, 2, 1, 2, 2, 0,
                 2, 1, 2, 1, 0, 2, 1, 2, 2, 2, 0, 2, 2, 0, 1, 2, 1, 0, 1, 2, 2, 0,
                 2, 0, 2, 2, 0, 0, 1, 1, 2, 2, 2, 2, 2, 1, 1, 0, 0, 1, 0, 0, 0, 2,
                 2, 0, 2, 2, 1, 2, 1, 0, 1, 0, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 1, 1,
                 0, 1, 1, 1, 0, 1, 2, 2, 0, 0, 0, 2, 1, 1, 0, 1, 2, 2, 2, 0, 2, 1,
                 1, 2, 1, 1, 0, 1, 0, 2, 2, 0, 2, 1, 0, 0, 1, 0, 0, 0, 0, 1, 2, 1,
                 2, 1, 1, 1, 2, 0, 2, 2, 0, 0, 0, 0, 2, 1, 2, 2, 2, 1, 1, 0, 2, 2,
                 2, 2, 2, 0, 2, 2, 0, 2, 0, 0, 2, 2, 0, 2, 1, 2, 0, 1, 0, 2, 2, 1,
                 2, 1, 1, 1, 0, 0, 0, 2, 2, 1, 2, 0, 2, 1, 0, 2, 1, 2, 2, 1, 2, 2,
                 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 2, 0, 2, 1, 1, 1,
                 2, 0, 2, 1, 2, 0, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 0, 0, 2,
                 2, 0, 1, 1, 2, 0, 2, 2, 2, 2, 0, 1, 2, 1, 1, 2, 1, 2, 2, 2, 0, 1,
                 0, 0, 2, 2, 1, 0, 2, 1, 0, 2, 0, 1, 2, 2, 2, 0, 0, 2, 0, 2, 0, 2,
                 2, 0, 2, 2, 2, 0, 1, 1, 0, 1, 1, 1, 2, 0, 0, 1, 2, 1, 0, 0, 2, 1,
                 1, 2, 1, 2, 0, 2, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 1, 0, 2, 2, 1, 0,
                 2, 2, 0, 0, 2, 0, 2, 2, 2, 2, 1, 0, 0, 2, 2, 2, 0, 0, 0, 0, 2, 0,
                 2, 0, 2, 2, 1, 0, 2, 1, 1, 2, 2, 0, 0, 2, 2, 0, 2, 1, 2, 2, 2, 1,
                 2, 2, 2, 1, 1, 2, 1, 1, 2, 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 0, 1, 0,
                 2, 0, 1, 2, 0, 2, 0, 2, 2, 2, 1, 2, 1, 2, 2, 0, 1, 2, 2, 0, 1, 2,
                 2, 2, 0, 2, 0, 2, 0, 2, 0, 1, 2, 0, 1, 0, 2, 2, 2, 2, 0, 1, 2, 2,
                 2, 0, 0, 2, 2, 2, 2, 2, 1, 0, 2, 0, 1, 2, 2, 2, 1, 2, 2, 1, 2, 2,
                 2, 2, 2, 2, 1, 0, 2, 0, 2, 0, 2, 1, 2, 1, 2, 1, 2, 1, 2, 2, 2, 1,
                 2, 2, 2, 1, 0, 2, 2, 1, 2, 2, 2, 2, 0, 1, 1, 2], dtype=int64)
```

```
In [132]: plt.scatter(x = data["Weight"], y = data["Height"], c=y2,s=75, edgecolors='black')
          plt.xlabel("Weight")
          plt.ylabel("Height")
          plt.title("Weight vs Height Plot for GMM when n = 3")
          plt.show()
```



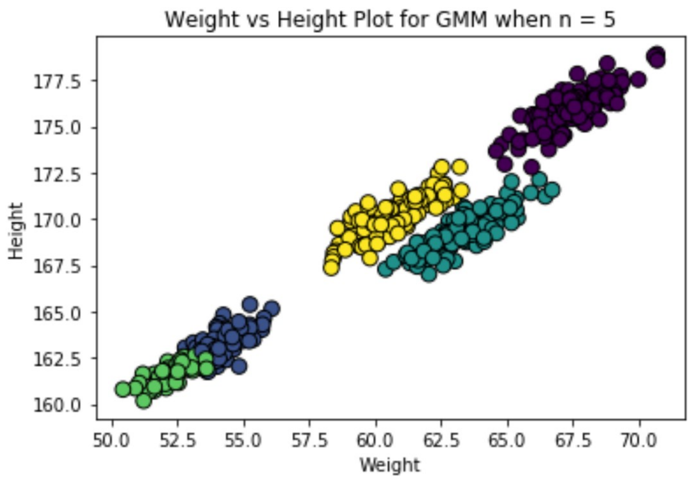
```
In [125]: gmm5 = GaussianMixture(n_components=5)
          gmm5.fit(data)
```

```
Out[125]: GaussianMixture(covariance_type='full', init_params='kmeans', max_iter=100,
                          means_init=None, n_components=5, n_init=1, precisions_init=None,
                          random_state=None, reg_covar=1e-06, tol=0.001, verbose=0,
                          verbose_interval=10, warm_start=False, weights_init=None)
```

In [126]:
```
y3 = gmm5.predict(data)
y3
```

Out[126]:
```
array([0, 0, 4, 4, 0, 2, 2, 4, 4, 3, 2, 0, 2, 1, 4, 4, 0, 4, 1, 4, 4, 0,
       2, 3, 4, 1, 0, 2, 1, 2, 2, 2, 0, 2, 2, 0, 3, 4, 1, 0, 3, 4, 4, 0,
       2, 0, 4, 4, 0, 0, 3, 3, 2, 4, 4, 2, 2, 3, 1, 0, 0, 1, 0, 0, 0, 2,
       4, 0, 4, 2, 3, 4, 3, 0, 3, 0, 2, 2, 4, 4, 3, 4, 1, 4, 4, 2, 3, 1,
       0, 1, 1, 1, 0, 1, 4, 4, 0, 0, 0, 4, 3, 1, 0, 1, 4, 4, 2, 0, 4, 1,
       1, 2, 1, 1, 0, 3, 0, 4, 2, 0, 4, 3, 0, 0, 1, 0, 0, 0, 0, 1, 4, 1,
       2, 1, 1, 1, 4, 0, 4, 2, 0, 0, 0, 0, 2, 1, 4, 4, 2, 3, 1, 0, 4, 4,
       4, 2, 2, 0, 2, 4, 0, 4, 0, 0, 2, 2, 0, 4, 3, 2, 0, 3, 0, 2, 4, 3,
       4, 3, 3, 1, 0, 0, 0, 4, 2, 1, 2, 0, 2, 1, 0, 2, 3, 2, 2, 3, 4, 2,
       4, 1, 1, 4, 2, 2, 4, 2, 2, 4, 2, 2, 2, 0, 0, 0, 4, 0, 2, 3, 1, 3,
       4, 0, 2, 1, 2, 0, 1, 4, 1, 2, 2, 2, 2, 2, 4, 4, 4, 2, 3, 0, 0, 2,
       2, 0, 3, 1, 4, 0, 4, 4, 4, 2, 0, 3, 4, 1, 3, 2, 1, 2, 4, 4, 0, 3,
       0, 0, 4, 4, 1, 0, 4, 3, 0, 2, 0, 3, 2, 2, 4, 0, 0, 4, 0, 2, 0, 2,
       2, 0, 4, 4, 4, 0, 3, 1, 0, 3, 3, 1, 4, 0, 0, 1, 4, 3, 0, 0, 2, 1,
       1, 4, 1, 4, 0, 4, 1, 2, 3, 2, 2, 4, 4, 4, 2, 2, 3, 0, 4, 2, 3, 0,
       4, 4, 0, 0, 2, 0, 2, 2, 4, 2, 1, 0, 0, 4, 0, 4, 0, 0, 0, 0, 4, 0,
       4, 0, 4, 4, 1, 0, 4, 1, 1, 2, 4, 0, 0, 2, 2, 0, 2, 1, 4, 4, 4, 3,
       4, 2, 4, 1, 3, 4, 1, 1, 2, 3, 1, 2, 2, 2, 2, 3, 3, 1, 3, 0, 1, 0,
       4, 0, 3, 4, 0, 4, 0, 4, 2, 2, 3, 4, 1, 2, 4, 0, 1, 2, 4, 0, 1, 2,
       2, 4, 0, 4, 0, 2, 0, 4, 0, 1, 2, 0, 1, 0, 4, 2, 4, 2, 0, 1, 4, 4,
       2, 0, 0, 4, 4, 2, 4, 2, 1, 0, 2, 0, 1, 2, 2, 2, 1, 2, 4, 3, 2, 2,
       2, 2, 2, 4, 3, 0, 2, 0, 4, 0, 2, 1, 4, 3, 2, 3, 4, 1, 4, 2, 2, 3,
       2, 2, 2, 1, 0, 2, 2, 1, 4, 2, 2, 4, 0, 3, 3, 4], dtype=int64)
```

In [131]:
```
plt.scatter(x = data["Weight"], y = data["Height"], c=y3,s=75, edgecolors='black')
plt.xlabel("Weight")
plt.ylabel("Height")
plt.title("Weight vs Height Plot for GMM when n = 5")
plt.show()
```



Working

How does K Means Work ?

The k-means clustering algorithm attempts to split a given anonymous data set (a set containing no information as to class identity) into a fixed number (k) of clusters. Initially k number of so called centroids are chosen. A centroid is a data point (imaginary or real) at the center of a cluster. In Praat each centroid is an existing data point in the given input data set, picked at random, such that all centroids are unique (that is, for all centroids ci and cj, ci ≠ cj). These centroids are used to train a kNN classifier. The resulting classifier is used to classify (using k = 1) the data and thereby produce an initial randomized set of clusters. Each centroid is thereafter set to the arithmetic mean of the cluster it defines. The process of classification and centroid adjustment is repeated until the values of the centroids stabilize. The final centroids will be used to produce the final classification/clustering of the input data, effectively turning the set of initially anonymous data points into a set of data points, each with a class identity.

How does GMM work ?

A Gaussian Mixture is a function that is comprised of several Gaussians, each identified by k ∈ {1,…, K}, where K is the number of clusters of our dataset. Each Gaussian k in the mixture is comprised of the following parameters: A mean μ that defines its centre. A covariance Σ that defines its width. This would be equivalent to the dimensions of an ellipsoid in a multivariate scenario. A mixing probability π that defines how big or small the Gaussian function will be.

Result and Conclusion

we were successfully able to plot Kmeans plot for 3,4,5 clusters and similarly a GMM plot as well for 3,4,5 clusters

```
In [ ]:
```