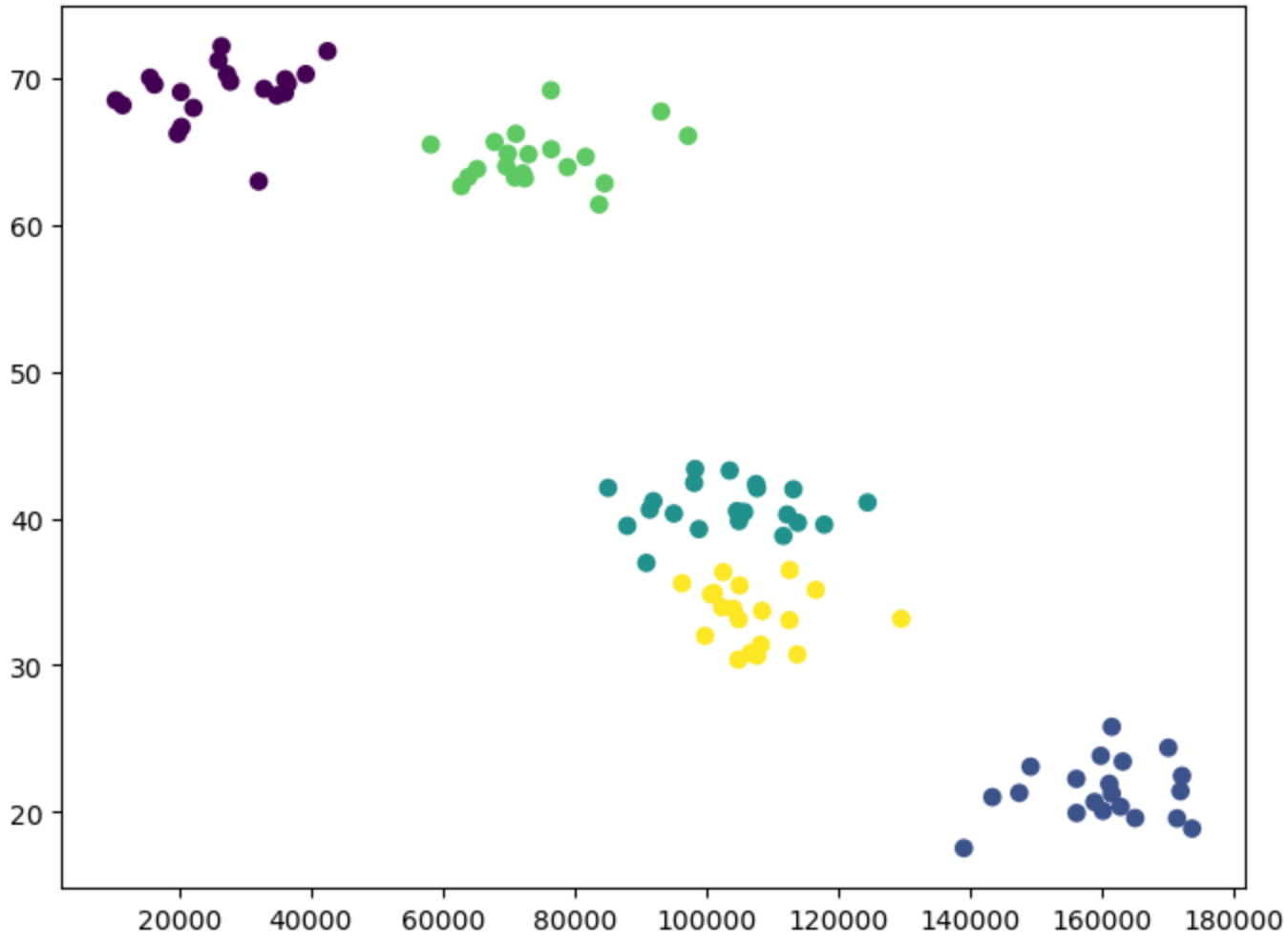
In [1]:
```
from numpy import random, array
def createClusteredData(N, k):
    random.seed(10)
    pointsPerCluster = float(N)/k
    X = []
    for i in range (k):
        incomeCentroid = random.uniform(20000.0, 200000.0)
        ageCentroid = random.uniform(20.0, 70.0)
        for j in range(int(pointsPerCluster)):
            X.append([random.normal(incomeCentroid, 10000.0), random.normal(ageCentroid,
    X = array(X)
    return X
```

In [3]:
```
%matplotlib inline
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale
data = createClusteredData(100, 5)
model = KMeans(n_clusters=5)
model = model.fit(scale(data))
print(model.labels_)
plt.figure(figsize=(8, 6))
plt.scatter(data[:,0], data[:,1], c=model.labels_.astype(float))
plt.show()
```

```
C:\Users\SAKTHI\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: FutureWarni
ng: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of
`n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\SAKTHI\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarnin
g: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks
than available threads. You can avoid it by setting the environment variable OMP_NUM_THR
EADS=1.
  warnings.warn(
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```



In [ ]: