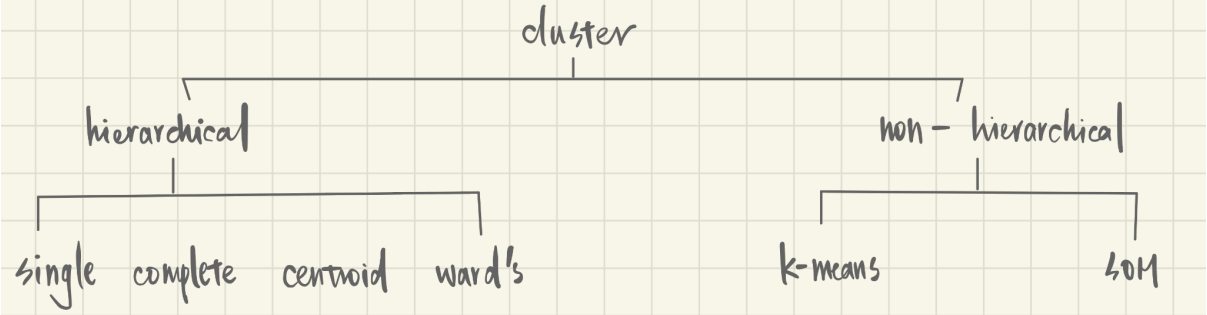
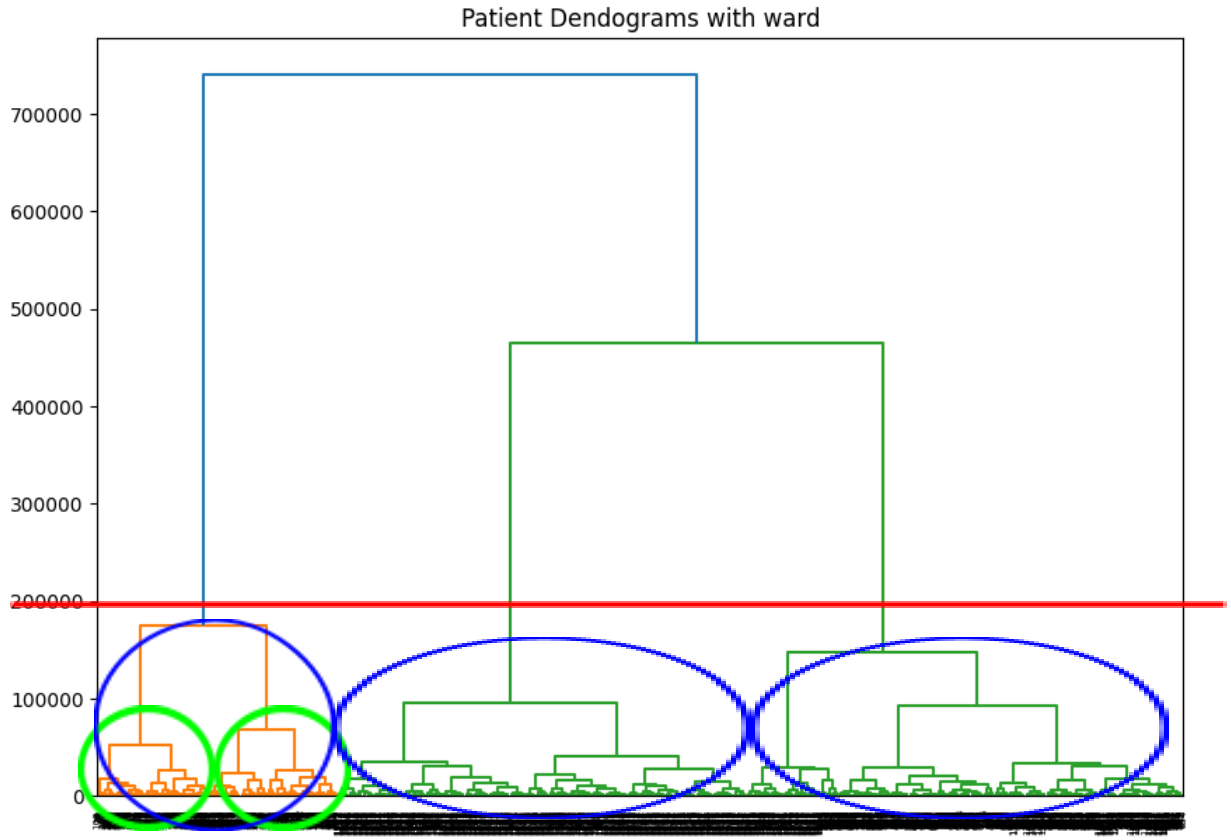
至期中以前，目前學習到的分方法有以下方法：



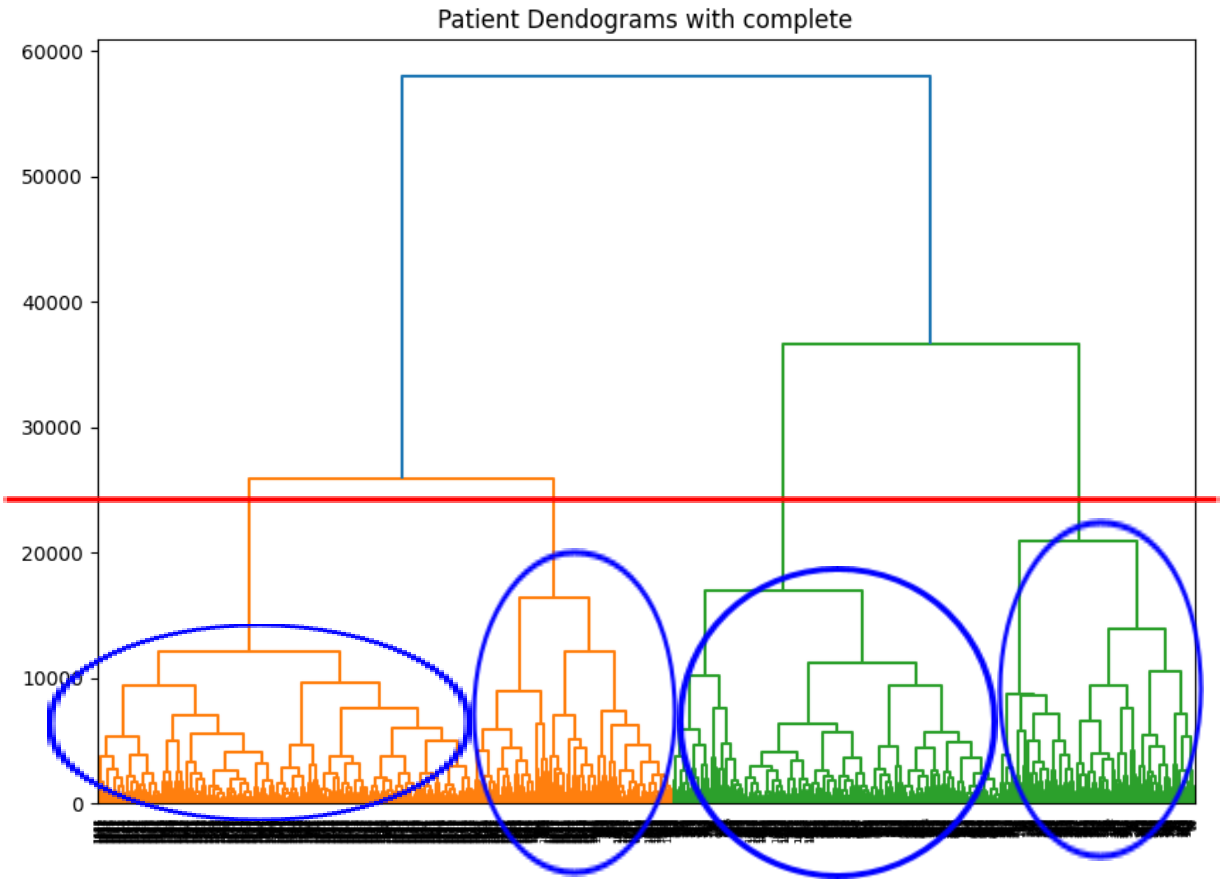
以及Classification中的Discriminant

首先我們就先將資料以hierarchical進行查看



本張圖先採用hierarchical中的ward法，發現若要將資料大致分群，譬如以我繪製的紅色線條以下，可以根據藍色圓圈分為三大群，然而若要再分得更細，可以將最左邊的藍色圓圈再細分為兩個亮綠色圓圈內的群。(甚至中間與右邊的藍色圓圈分別還可以再將群細分)

考量到不知道應該分幾群好，我想用更嚴格的complete法去看看會產生怎麼樣的結果



從complete法當中可見得在紅線之下後可清晰的分為4大群，若再將此大群細分為數個小群又過於細小，我認為4大群是較剛好的分群數。

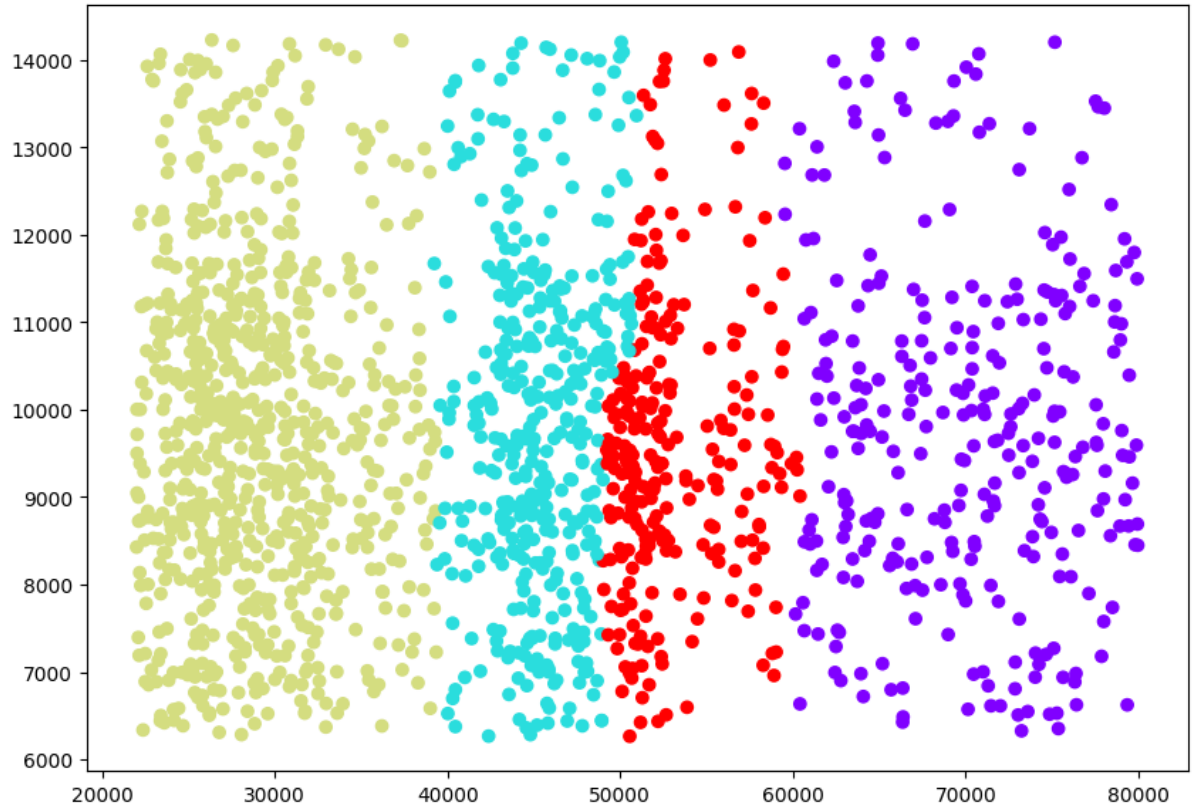
接下來透過散佈圖查看資料實際的分佈情況

```
# 選取分析欄位：buying_price, maint_price, persons
patient_data = data.iloc[:, 1:4].values
```

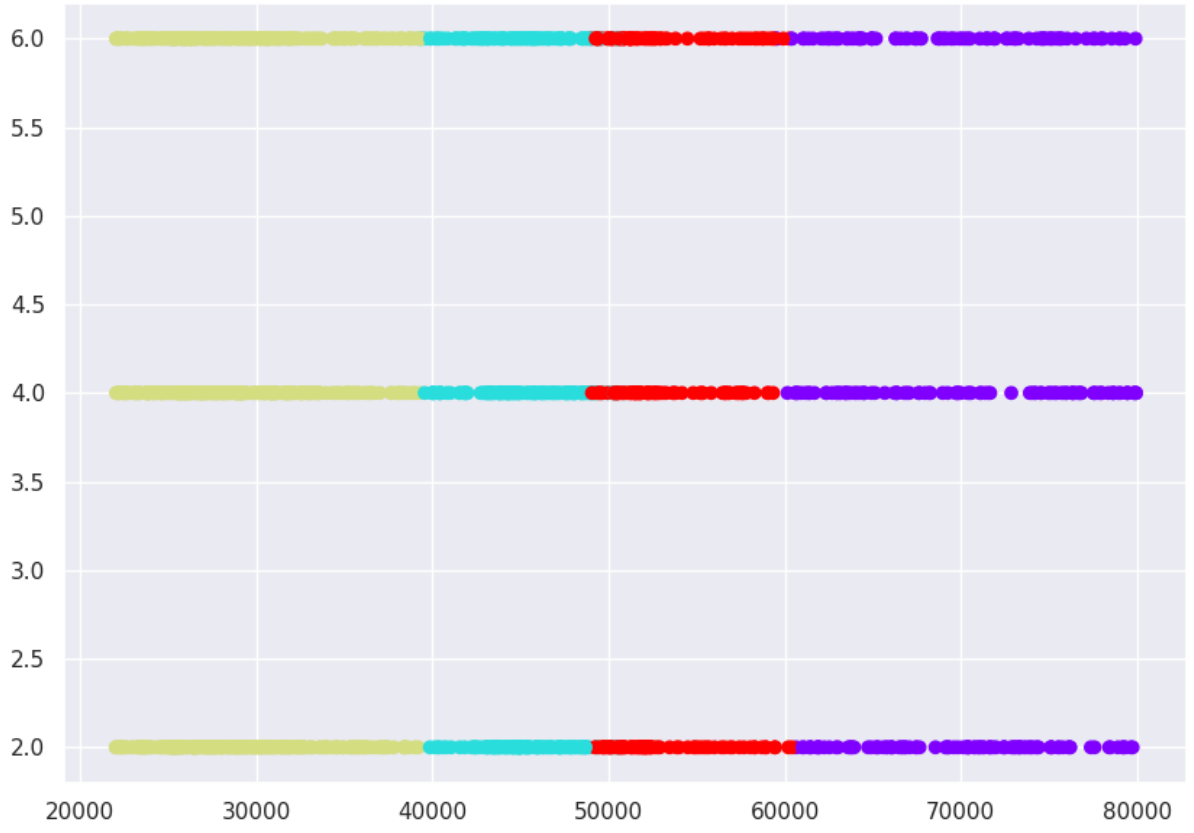
```
# 繪製散佈圖查看分群情況
from sklearn.cluster import AgglomerativeClustering

cluster = AgglomerativeClustering(n_clusters=4, affinity='euclidean', linkage='complete')
cluster.fit_predict(patient_data)
plt.figure(figsize=(10, 7))
plt.scatter(patient_data[:,0], patient_data[:,1], c=cluster.labels_, cmap='rainbow')
```

我選用buying_price, maint_price, persons欄位來進行分析，並一樣使用complete法來作為linkage，之後x座標使用buying_price, y座標使用maint_price，繪製出的散布圖結果如下：

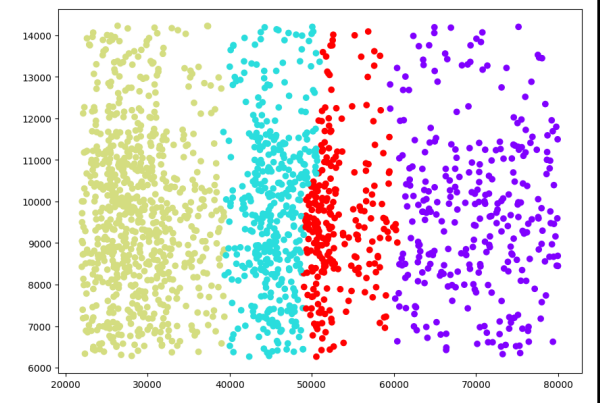


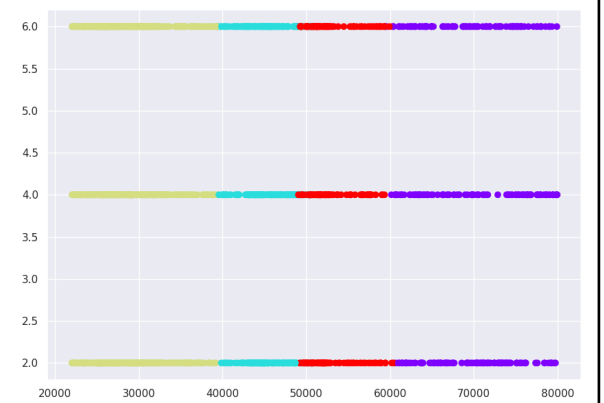
可看見是依buying_price售出價格的高低來進行分群的，至於與承載人數的相關性我們再以x軸為buying_price，y軸為persons進行查看：



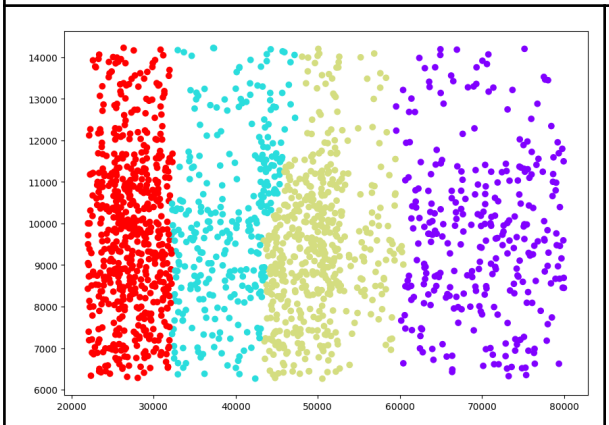
發現不管在何種售價區間分群下，都有不同種類的承載人數，因此確實分群結果還是與售出價格有較直接關係。

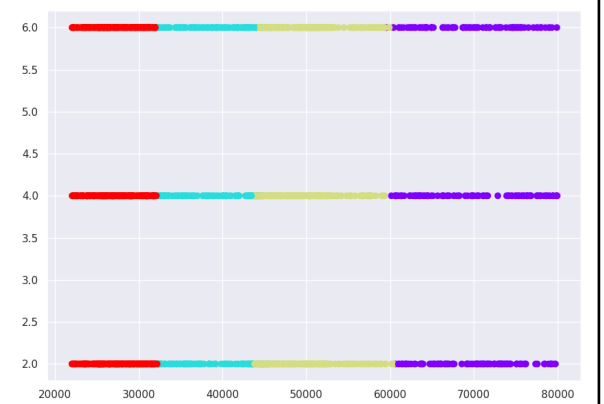
然而我若將ID, Suggestion以外的所有欄位納入進行分析，又會有些微不同的結果，下用表格進行比較：





選取欄位3欄的散佈圖結果

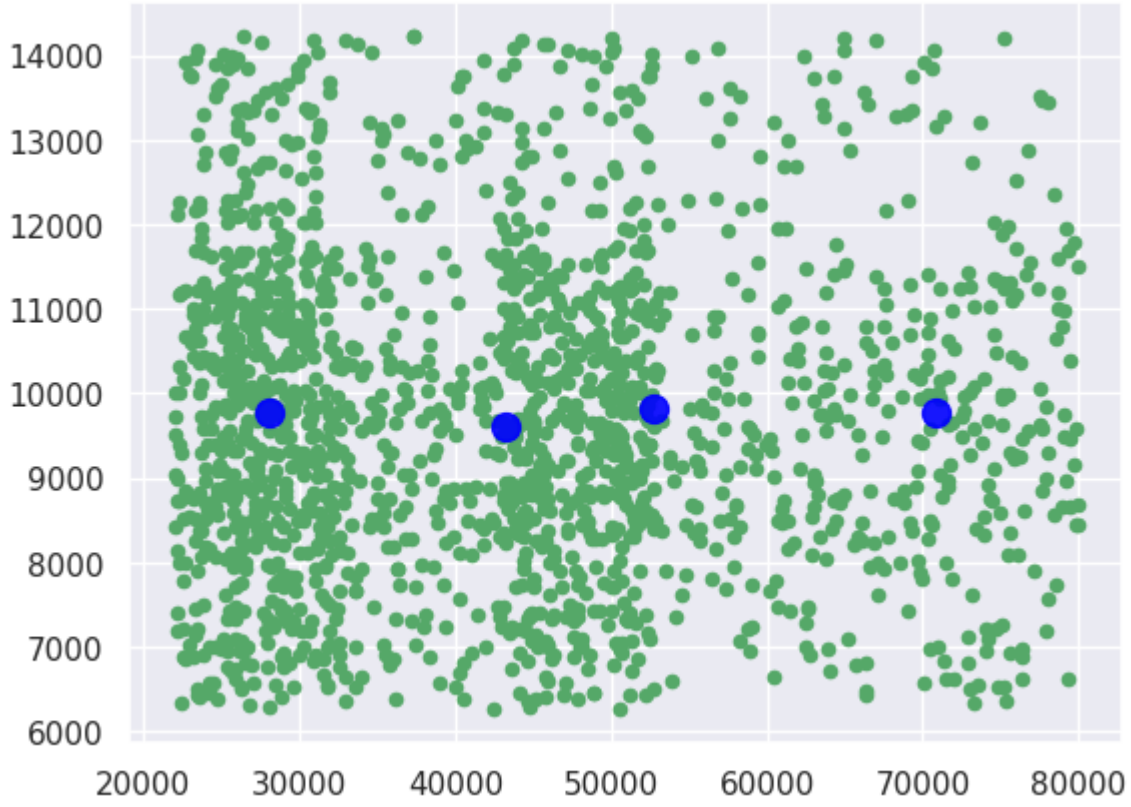




選取ID, Suggestion以外的所有欄位結果

會發現在buying_price的劃上有了細微的不同，而高價位的車(60000以上)不管在何種欄位選取下幾乎無差異，僅有中低價位的車量分群有變化。

接下來也看看透過k-means得知群中心的狀況(在此劃分為4群，且將ID, Suggestion以外的所有欄位納入進行分析)




```
centers
```

```
array([[2.80940395e+04, 9.78262794e+03, 3.98240469e+00, 3.49853372e+00,
        3.93135133e+01, 4.36205287e+01],
       [7.07827631e+04, 9.77594676e+03, 3.97857143e+00, 3.48571429e+00,
        3.97245074e+01, 4.41514921e+01],
       [5.27029872e+04, 9.81446075e+03, 4.06010929e+00, 3.48907104e+00,
        4.03041620e+01, 4.42008743e+01],
       [4.31976828e+04, 9.61578976e+03, 3.99000000e+00, 3.52250000e+00,
        4.02036043e+01, 4.36403252e+01]])
```

可知群中心的buying_price大致為28094, 43198, 52703, 70783，至於其他欄位的群中心則相差無幾，由此更加確認了本車輛的分析確是以售出的價格進行分群。

此外，也想使用SOM看看會對這筆car資料集進行怎樣的分群

```
samples = []

for i in range(len(df)):
    sample = temp[i]
    sample = sample.reshape(1, 6)
    samples.append(sample)
```

首先需要將資料進行reshape，然而結果發現dtype為object，無法進行分群，所以需要將資料轉成float

```
# 將dtype=object改為float
samples = np.array(samples).astype(dtype=float).tolist()
```

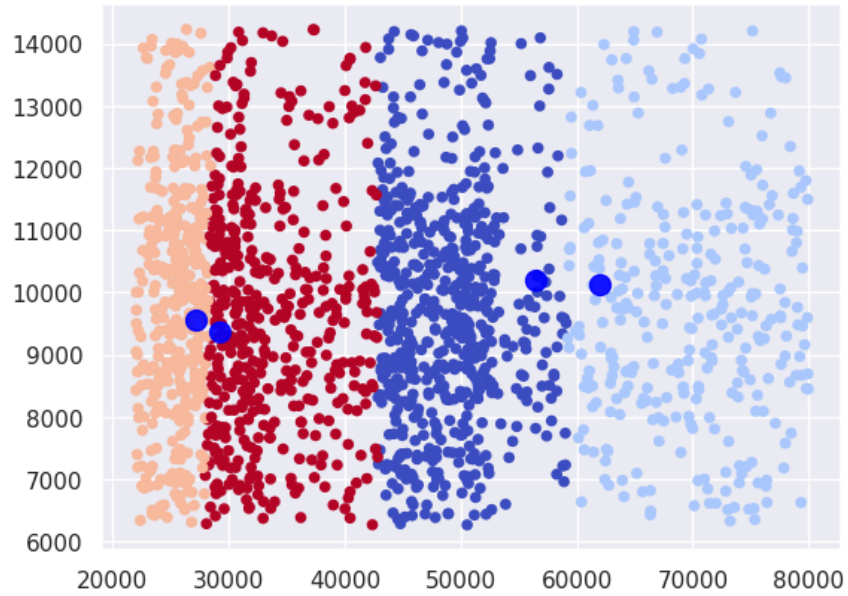
之後便開始進行模型訓練的迭代，一樣為了與其他分群方法進行比較，我們分為4群，因此neurons選用1*4=4

```
s = som.SOM(neurons=(1, 4), dimentions=6 ,n_iter=500, learning_rate=0.2)
s.train(samples)
print("Cluster centres:", s.weights_)
print("labels:", s.labels_)
result = s.predict(samples)
```

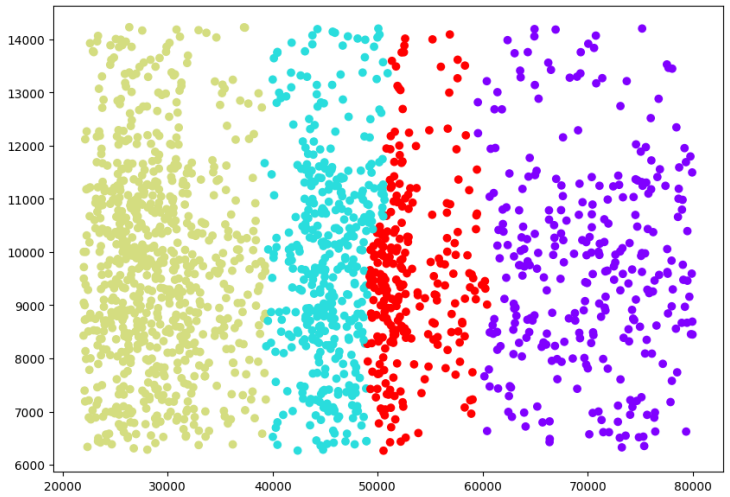
以下顯示群中心的座標結果，以及分群狀況

```
Cluster centres: [[5.64002269e+04 1.02062707e+04 3.66786815e+00 3.23439523e+00
  4.[illegible]+01 4.66311699e+01]
 [6.19362884e+04 1.01209466e+04 3.73924637e+00 3.21455762e+00
  4.19234193e+01 4.75460992e+01]
 [2.71955921e+04 9.56411188e+03 3.58922587e+00 3.70689768e+00
  3.87700331e+01 4.47658759e+01]
 [2.92162669e+04 9.35757810e+03 3.62839865e+00 3.67307074e+00
  3.98690104e+01 4.45812259e+01]]
labels: [1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1,
```

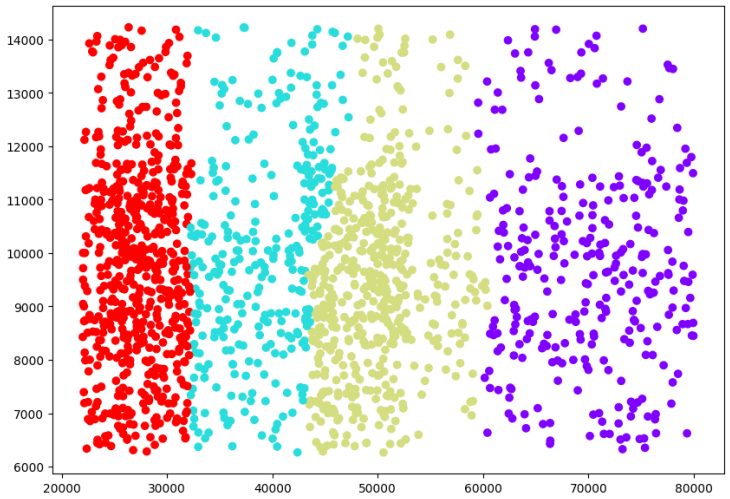
我們使用散布圖進行直觀的觀察



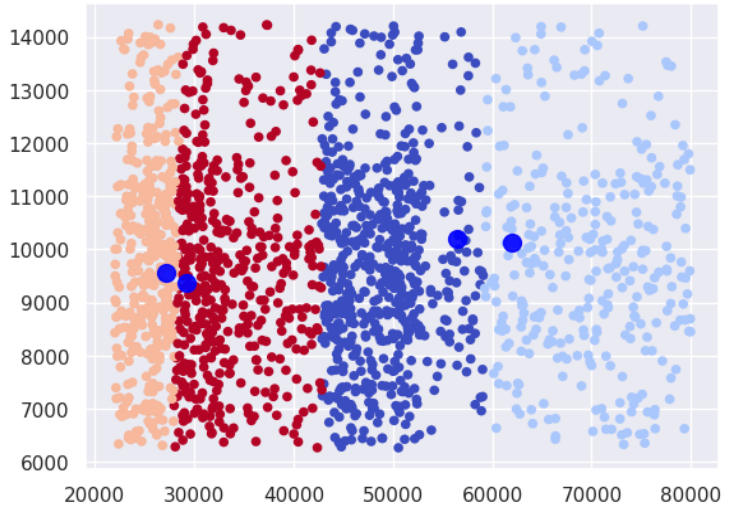
發現劃分的buying_price依據又與先前的結果有些許的不同，以下我們將過去幾種分出來的結果進行比較：



AgglomerativeClustering 選取欄位3欄的散佈圖結果



AgglomerativeClustering 選取ID, Suggestion以外的所有欄位結果



SOM 選取ID, Suggestion以外的所有欄位結果

經由比較可以發現三種結果在20000-30000之間的分界產生了歧異。

最後，我們再由classification的discriminant進行判斷模型的建置，看看如果未來要以這樣的模型進行其他筆資料的購買價值的判斷，是否足夠準確

```
features = ['buying_price', 'maint_price', 'persons', 'doors', 'lug_boot', 'safety']

# Separating out the features
x = df1.loc[:, features].values

# Separating out the target
y = df1.loc[:,['Suggestion']].values
```

```
X = np.array(StandardScaler().fit_transform(x))
y= np.array(y)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

一樣我們將ID, Suggestion以外的所有欄位納入進行分析，並將suggestion作為我們要預測判斷的y值，之後將特徵標準化，並且分為training資料與testing資料。

```
temp=y_train.flatten()
print(Counter(temp))
```

```
Counter({'N': 848, 'Y': 361})
```

```
from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state=42)
X_train_SM, y_train_SM = sm.fit_resample(X_train, y_train)
print(Counter(y_train_SM))
```

之後進行y資料的處理，發現N與Y有資料數量上的差異，需要進行SMOTE進行校正。

```
clf = LinearDiscriminantAnalysis()
clf.fit(X_train_SM, y_train_SM)
```

```
▾ LinearDiscriminantAnalysis
LinearDiscriminantAnalysis()
```

接著便進行模型的訓練

```
y_predicted = clf.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_predicted)
```

```
array([[278,  84],
       [ 41, 116]])
```

發現錯誤的總筆數高達84+41=125筆

結論:

在分群結果發現主要以buying_price作為特徵，並且可分為4群，然而hierarchical與SOM在進行劃分的時候不盡相同，我認為這是有待人為去進一步做決策的部分，或許可以取中間值作為適當的劃分。

而在classification當中，發現用discriminant方法作為模型訓練後，出來的預測結果錯誤率達125/519 = 0.24，即為20%的錯誤機率，相信此法對於本資料來說在預測方面尚未是最佳解(以往在進行其他資料分析或文獻的探討時，準確度皆有88%左右甚至是以上的水準)，希望在接下來的半個學期能夠學習到其餘的classification方法，再來適用於本資料集上，看看預測結果如何。