1. Split wine.csv into training data and test data.

我使用 pandas 來分類資料
先將資料讀取後，再將所有的資料先分成三種，為了可以很好的擷取 train 和 test 資料

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

filename = 'wine.csv'
df = pd.read_csv(filename)
#build up dataset
df_0 = pd.DataFrame(columns=list(df.columns))
df_1 = pd.DataFrame(columns=list(df.columns))
df_2 = pd.DataFrame(columns=list(df.columns))
# take all the different target into the DataFrame
df_0 = df.loc[df['target'] == 0]
df_1 = df.loc[df['target'] == 1]
df_2 = df.loc[df['target'] == 2]
# random the dataset
df_0 = df_0.reindex(np.random.permutation(df_0.index))
df_1 = df_1.reindex(np.random.permutation(df_1.index))
df_2 = df_2.reindex(np.random.permutation(df_2.index))
# set all the dataset with 1 to last index
df_0.reset_index(drop=True, inplace=True)
df_1.reset_index(drop=True, inplace=True)
df_2.reset_index(drop=True, inplace=True)

# build up train and test
train = pd.DataFrame(columns=list(df.columns))
test = pd.DataFrame(columns=list(df.columns))

test = pd.concat([df_0.loc[0:19,:],df_1.loc[0:19,:],df_2.loc[0:19,:]],axis = 0)
test.reset_index(drop=True, inplace=True)

train = pd.concat([df_0.loc[20:len(df_0),:],df_1.loc[20:len(df_1),:],df_2.loc[20:len(df_2),:]],axis = 0)
train.reset_index(drop=True, inplace=True)

train.to_csv('train.csv')
test.to_csv('test.csv')
```

Source: https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html
所有的 function 都是官網所提供的

2. Evaluate the posterior probabilities.

如果要算後驗機率，則必須先將先驗機率和概似機率都先算出來，但在算之前，先將訓練資料的所有特徵的平均與變異數算出來，這個可以透過 pandas 的 function: describe() 處理，只要能將事先將所有的資料分門別類，就可以幫其自動計算所有的參數。

將所有的資料的平均與變異數透過 pandas 算出來

```
# problem 2
data_0 = train.loc[train['target'] == 0]
data_1 = train.loc[train['target'] == 1]
data_2 = train.loc[train['target'] == 2]
data_0 = np.array(data_0.describe())[:,1:]
data_1 = np.array(data_1.describe())[:,1:]
data_2 = np.array(data_2.describe())[:,1:]
mean_0 = data_0[1,:]
mean_1 = data_1[1,:]
mean_2 = data_2[1,:]
std_0 = data_0[2,:]
std_1 = data_1[2,:]
std_2 = data_2[2,:]
```

先驗機率為整體資料的數據比例，然後後驗機率為先驗機率(酒的品種)乘上每一瓶酒的特徵在已知某一種酒的機率下(條件機率或稱為概似機率)，最後除與所有特徵的機率(又可以轉乘條件機率乘上先驗機率)，

```
import scipy
testd = np.array(test)
#scipy.stats.norm(distance, R).pdf(z[i])
p_0 = 175/483
p_1 = 205/483
p_2 = 103/483
ans = []
j = 0
while(j < np.shape(testd)[0]):

    posteroir_0 = p_0
    posteroir_1 = p_1
    posteroir_2 = p_2
    for i in range(0,13):
        posteroir_0 *= scipy.stats.norm(mean_0[i], std_0[i]).pdf(testd[j,i+1])
        #print(posteroir_0)
        posteroir_1 *= scipy.stats.norm(mean_1[i], std_1[i]).pdf(testd[j,i+1])
        posteroir_2 *= scipy.stats.norm(mean_2[i], std_2[i]).pdf(testd[j,i+1])
    evi =  posteroir_0+ posteroir_1+posteroir_2
    posteroir_0/=evi;posteroir_1/=evi;posteroir_2/=evi;

    if (posteroir_0 > posteroir_1) & (posteroir_0 > posteroir_2):
        ans.append(0)

    elif(posteroir_1 > posteroir_0) & (posteroir_1 > posteroir_2):
        ans.append(1)
    elif(posteroir_2 > posteroir_1) & (posteroir_2 > posteroir_0):
        ans.append(2)

    j+=1
```

然後做比較，找出最大的後驗機率，所以又可以稱為 MAP(最大後驗機率)
最後算其準確度，為了驗證是否總平均有達到 95%，所以自己有設定一個讓其跑 50 遍並算其平均(不會放入 coding 中，會影響整體美觀度)。

```
count = 0
for i in range(0,np.shape(testd)[0]):
    #print(int(test[i,0]),int(ans[i]))
    if(int(testd[i,0]) == int(ans[i])):
        count+=1

print("accuracy: ",count/60)
```

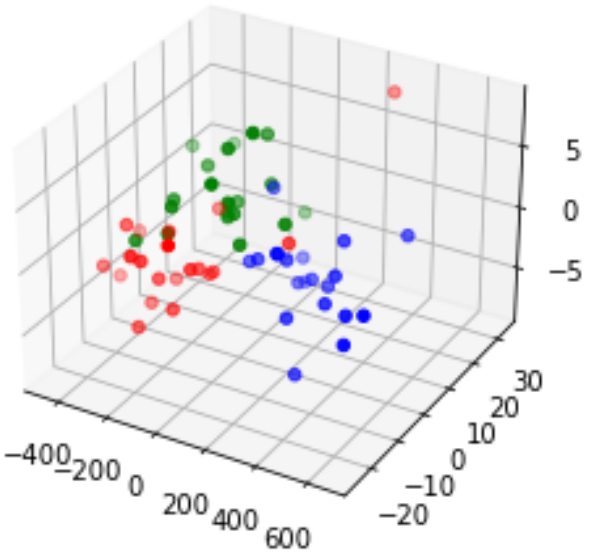
一部分的結果

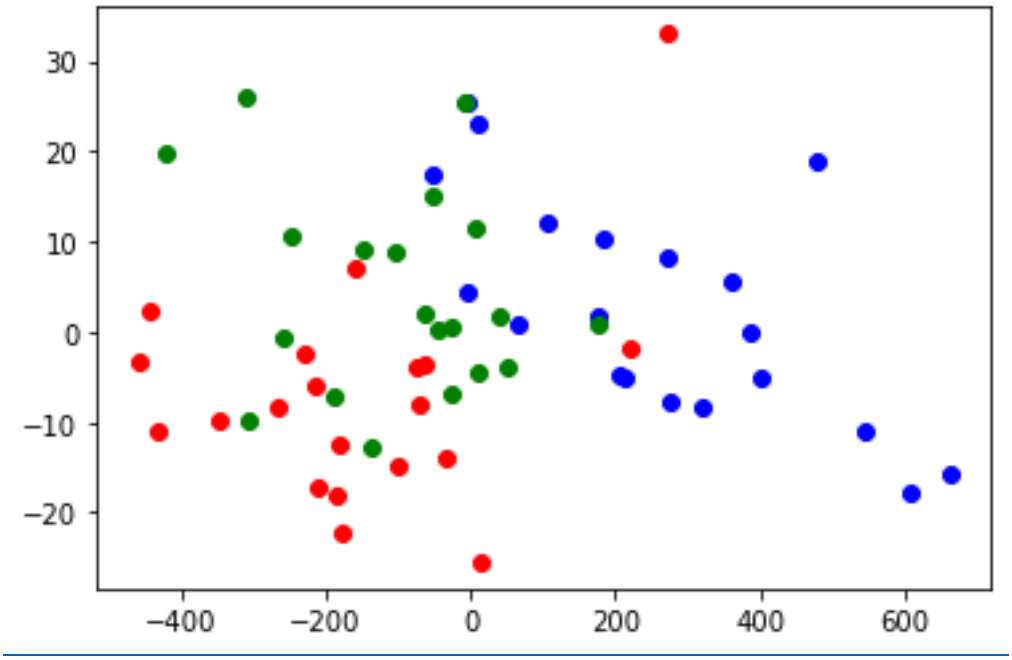
```
accuracy:  0.9666666666666667
accuracy:  0.9333333333333333
accuracy:  0.9333333333333333
accuracy:  0.9333333333333333
accuracy:  0.95
accuracy:  1.0
accuracy:  0.9333333333333333
accuracy:  0.9833333333333333
accuracy:  0.9666666666666667
accuracy:  0.9833333333333333
accuracy:  0.95
accuracy:  0.8833333333333333
accuracy:  0.9333333333333333
accuracy:  0.9833333333333333
accuracy:  0.9833333333333333
accuracy:  0.9666666666666667
accuracy:  0.9666666666666667
accuracy:  0.9333333333333333
accuracy:  0.9833333333333333
accuracy:  0.9333333333333333
accuracy:  0.9666666666666667
accuracy:  0.9666666666666667
accuracy:  0.9333333333333333
accuracy:  1.0
accuracy:  0.9666666666666667
```

```
All accuracy in 50 times:  [0.95, 0.95, 0.9666666666666667, 0.9333333333333333, 1.0, 0.9666666666666667,
0.9666666666666667, 0.9333333333333333, 1.0, 0.9333333333333333, 0.9666666666666667, 1.0, 0.9833333333333333,
0.9333333333333333, 0.9666666666666667, 0.9333333333333333, 0.9333333333333333, 0.9333333333333333, 0.95, 1.0,
0.9333333333333333, 0.9833333333333333, 0.9666666666666667, 0.9833333333333333, 0.95, 0.8833333333333333,
0.9333333333333333, 0.9833333333333333, 0.9833333333333333, 0.9666666666666667, 0.9666666666666667,
0.9333333333333333, 0.9833333333333333, 0.9333333333333333, 0.9666666666666667, 0.9666666666666667,
0.9333333333333333, 1.0, 0.9666666666666667, 0.95, 0.9166666666666666, 0.9666666666666667, 0.9833333333333333,
0.9666666666666667, 0.9333333333333333, 0.95, 0.9833333333333333, 0.9666666666666667, 0.9666666666666667, 0.95]
Average accuracy:  0.959000000000002
```

可以看出五十次的平均結果達到 0.959

3. Plot the visualized result of testing data.





輸出兩張圖，一張為 PCA 到三維，一個為 PCA 到二維。
PCA 主要的用處，其實就是找到所有資料中，可以保持最多變異性的線，並且將資料投影到上面，也可以看出每筆資料的變異性，而為何要做三維與二維，其實可以從上圖看出，二維的界線較不清楚，但三維(多納入一個成分分析)可以較好的區別出差別，有了 PCA，可以使用低維的資料，來進行分類，減少計算量，還可以進行可視化(畫圖)，但以這次作業來說，如果要到更好的區別，可能需要更多資料來處理，或是要 tuning 更好的隨機分配，有的界線會比較明顯，或是用其他方法。

4. Please discuss the effect of prior distribution on the posterior probabilities in your report.

先驗機率，代表我們有事先對這些資料有一定的理解，比如說，沙漠下雨的機率，一般人都知道，沙漠幾乎不下雨，所以如果今天問題是問沙漠是否下雨的機率，給予很多天氣特徵，其先驗機率就是是否下雨的機率，那我一定猜有九成的機會不下雨，因為這是先備知識，它可以讓問題變得更精確，但也有可能導致分類器變不準，比如說資料量差太多，可能會導致資料量較小的，需要更大的概似機率來使其靠向其他可能性，或許會跟真實世界的結果不一樣，所以資料最好能收集平均的資料，然後先驗機率希望是有科學根據，或是符合常理的分配，這樣就可以大大的增加資料的準確度。

以這題來說，因為事前機率我是透過資料量的比例，其結果相當好，主要是概似機率主導了後驗機率，而且教授所提供的資料算是好分辨的，但如果今天我

故意將先驗機率改成 0.98, 0.01, 0.01，其平均 30 次的平均準確率為

```
average accuracy:  0.936111111111111
```

可以看出極大的先驗機率，可能會導致準確度降低，所以先驗機率的大小，如果資料都很相近的話，可能會大大的影響準確度，因為概似機率可能沒辦法很清楚的區分，反之，資料差距大，則先驗機率的大小，影響力則會較低。