學號：R05921037 系級： 電機碩一 姓名：陳冠廷

1.請說明你實作的 generative model，其訓練方式和準確率為何？

答：

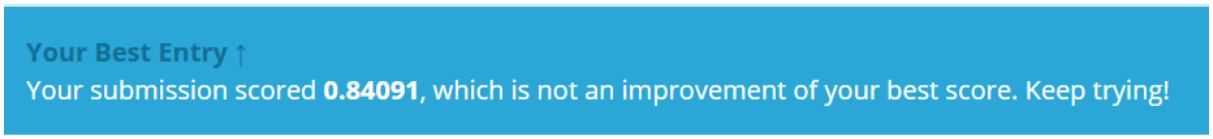
```
#data extraction
train_data,target_data,test_data,predict=data_extraction()

#mean,cov and splitted data
B_mu,S_mu,cov,B,S=mean_cov(train_data,target_data)

#P(B) P(S)
P_B=len(B)/(len(B)+len(S))
P_S=len(S)/(len(B)+len(S))
```

左圖這幾行是用來算出所需要的 MEAN 以及 COV。

```
#Gausssian Distribution
x=test_data[i]
index_B=(-0.5)*(np.dot(np.dot(x-B_mu,np.linalg.inv(cov)),(x-B_mu)))
f_B_x=np.exp(index_B)
index_S=(-0.5)*(np.dot(np.dot(x-S_mu,np.linalg.inv(cov)),(x-S_mu)))
f_S_x=np.exp(index_S)
#Result
P_B_x=(P_B*f_B_x)/(P_B*f_B_x+P_S*f_S_x)
res =1 if P_B_x>0.5 else 0
result.append(res)
```

左圖則是將上面所算出來的 MEAN、COV 帶入高斯並求出答案。

下圖則是我利用 generative model 在 kaggle 上的結果，可以預先知道的這不會是一個最好的方法。



2.請說明你實作的 discriminative model，其訓練方式和準確率為何？

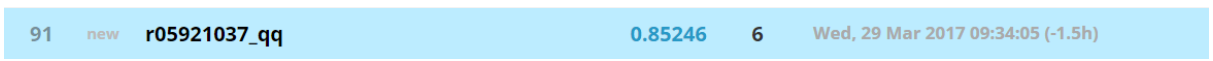
答：

```
#data extraction
train_data,target_data,test_data,predict=data_extraction()

#data normalization
train_data,test_data=normalization(train_data,test_data)

#weight generation
W,b=weight(len(train_data[0]))

#training
W,b=training(X=train_data,y=target_data,W=W,b=b)
```

```
zz=np.dot(X,W)+b
#reshape a into column
aa=np.reshape(sigmoid(zz),(len(zz),-1))
```

如同 linear model 我也是先取出 data，然後標準化，產生初始化參數，最後進入迴圈訓練，不一樣的是如下圖在求出 zz 後我需要經過一個 sigmoid function。

此次的準確率就比 generative model 高出了 1.2%，可以看出確實 discriminative model 效果是比較好的。



3.請實作輸入特徵標準化(feature normalization)，並討論其對於你的模型準確率的影響。

答：

下面兩張圖都是 logistic regression 在同樣是 SGD 以及 eta 一樣的情況下做的比較

左圖是沒有標準化的數據可以看出不只反映精確度的數據亂跳而且有向下跳動的趨勢，而右圖則是有標準化的數據可以看出精確度穩定上升中。

```
acc: 0.798194158656
acc: 0.803967937103
acc: 0.708700592734
acc: 0.824821105003
acc: 0.702496852062
acc: 0.816774669083
acc: 0.706243665735
acc: 0.72387211695
acc: 0.754737262369
acc: 0.720217438039
acc: 0.73130432112
acc: 0.801940972329
acc: 0.800651085655
acc: 0.791867571635
acc: 0.758637633979
acc: 0.783176192377
acc: 0.789502779399
```

```
acc: 0.835048063634
acc: 0.842848806855
acc: 0.844476520991
acc: 0.84619636989
acc: 0.84754767974
acc: 0.848407604189
acc: 0.848438315777
acc: 0.848684008476
acc: 0.848929701176
acc: 0.849359663401
acc: 0.849574644513
acc: 0.849912471976
acc: 0.850188876263
acc: 0.850311722613
acc: 0.850526703725
acc: 0.850741684838
acc: 0.850895242775
```

而在 generative model 的部分則是差異不大，我想可能是因為 generative model 本身就是利用其平均以及標準差來衡量新的數據，所以如果作標準化後感覺只是把所有的資料平移縮放，對於高斯函數所界定的邊緣界線不會差太多。

4. 請實作 logistic regression 的正規化(regularization)，並討論其對於你的模型準確率的影響。

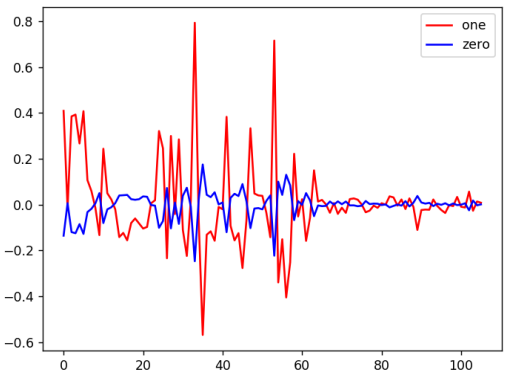
答：

```
acc: 0.805564939652
acc: 0.812075796198
acc: 0.788458585424
acc: 0.795030865145
acc: 0.791621878935
acc: 0.788888547649
acc: 0.781640613003
acc: 0.802708762016
acc: 0.785540984613
acc: 0.809219618562
acc: 0.807254076963
acc: 0.798716255643
acc: 0.811338718098
acc: 0.791038358773
```

```
acc: 0.835048063634
acc: 0.842848806855
acc: 0.844476520991
acc: 0.84619636989
acc: 0.84754767974
acc: 0.848407604189
acc: 0.848438315777
acc: 0.848684008476
acc: 0.848929701176
acc: 0.849359663401
acc: 0.849574644513
acc: 0.849912471976
acc: 0.850188876263
acc: 0.850311722613
acc: 0.850526703725
acc: 0.850741684838
acc: 0.850895242775
```

左圖為沒有正規化的 logistic regression，其精確度一直卡在 80%左右不上不下，我想主要原因是因為每一步走的步伐太大所以無法往 min 的地方走下去一直卡在那個凹口，而右圖則是有正規化的結果，會進入那個凹口不會卡在那邊。

5.請討論你認為哪個 attribute 對結果影響最大？



左圖是將 106 個 feature 對各自的 1 或 0 作平均後的圖，可以看出紅色於(33,53)也就是 Married-civ-spouse 以及 Husband 有最大值而藍色於這兩個位置剛好是反指標，所以可以確定這兩個指標較為明顯。