學號：R05921086 系級： 電機碩一 姓名：邱名彥 (Michael Chiou)

I am an international student and apologize for using English to write this report.

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

I use X_train.csv as input since all data is organized already. **All attributes** are used and are not normalized. The mean (μ_1 and μ_2) and covariance (Σ) are calculated for each attribute which allows us to calculate weights and bias. We then calculate the probability using the equation below.

$$\Sigma_1 = \Sigma_2 = \Sigma$$

$$z = \underbrace{(\mu^1 - \mu^2)^T \Sigma^{-1}}_{\boldsymbol{w^T}} x \underbrace{- \frac{1}{2}(\mu^1)^T \Sigma^{-1} \mu^1 + \frac{1}{2}(\mu^2)^T \Sigma^{-1} \mu^2 + ln\frac{N_1}{N_2}}_{\text{b}}$$

Z is a probability which is rounded to 0 or 1 where 1 is >50 and 0 <= 50k.
a public kaggle score of 0.84287 is produced using this generative model.

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

We use X_train.csv as input and use **all attributes**. Attributes are normalized and then all non-binary attributes into categories (age,fnlwgt, cap_gain, cap_loss,hours).
Example(age = 20 is placed in age group 17-25)
We use the two equations below in a logistic regression model where $f_{w,b}(x)$ is our prediction.

$$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$$

Class estimation using sigmoid function

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Sigmoid function

We use cross entropy (equation shown below) to determine and minimize our loss.

$$= \sum_n - \underbrace{\left[\hat{y}^n ln f_{w,b}(x^n) + (1-\hat{y}^n) ln\left(1 - f_{w,b}(x^n)\right)\right]}_{\text{Cross entropy between two Bernoulli distribution}}$$

We use the equations below to iteratively adjust our weights and bias

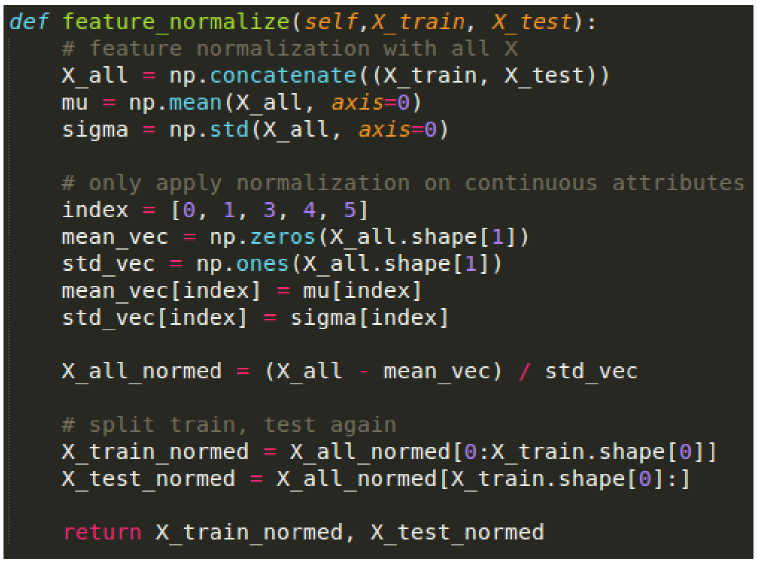
$w_i \leftarrow w_i - n\sum_n(-(y_n - f_{w,b}(x^n)))\, x_i^n$	$b_i \leftarrow b_i - n\sum_n(-(y^n - f_{w,b}(x^n)))$
Weight update equation	Bias update equation

Using 1001 iterations, a learning rate of 0.2, adagrad, and a regularization coefficient of 0.000002, we can obtain a strong kaggle score of 0.85676.

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。

We do normalization by calculating the mean and standard deviation of each attribute.
We calculate the normalized value with the following equation and code shown below.

$$X_{normalized} = (X_{val} - \mu)/\sigma$$



```python
def feature_normalize(self,X_train, X_test):
    # feature normalization with all X
    X_all = np.concatenate((X_train, X_test))
    mu = np.mean(X_all, axis=0)
    sigma = np.std(X_all, axis=0)

    # only apply normalization on continuous attributes
    index = [0, 1, 3, 4, 5]
    mean_vec = np.zeros(X_all.shape[1])
    std_vec = np.ones(X_all.shape[1])
    mean_vec[index] = mu[index]
    std_vec[index] = sigma[index]

    X_all_normed = (X_all - mean_vec) / std_vec

    # split train, test again
    X_train_normed = X_all_normed[0:X_train.shape[0]]
    X_test_normed = X_all_normed[X_train.shape[0]:]

    return X_train_normed, X_test_normed
```

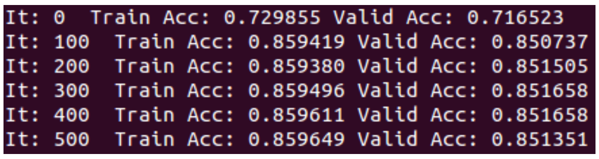
If we do not normalize values, we get a kaggle score of 0.76327 using all sorted attributes.
If we normalize values, we get a kaggle score of 0.85676 using all sorted attributes (from Q2).
A large change in accuracy may be due to weights being biased towards the continuous variables which have non-unity elements. This would skew values as the sigmoid would output values not representative of the actual model.

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。

When performing logistic regression with regularization, we get a strong kaggle score of 0.85676 using all sorted and normalized attributes.
Without regularization, we get a kaggle score of 0.85627 using a lambda coefficient of 0.00000000005. We use the following regularization equation

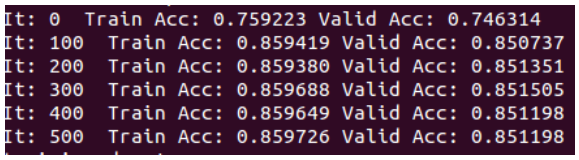
$w_i \leftarrow w_i - n\sum_n((-(y_n - f_{w,b}(x^n)))\, x_i^n) + \lambda w^2$	$b_i \leftarrow b_i - n\sum_n(-(y^n - f_{w,b}(x^n))) + \lambda w^2$
Regularizedd Weight update equation	Regularized Bias update equation

Validation data is shown below.



```
It: 0   Train Acc: 0.729855 Valid Acc: 0.716523
It: 100  Train Acc: 0.859419 Valid Acc: 0.850737
It: 200  Train Acc: 0.859380 Valid Acc: 0.851505
It: 300  Train Acc: 0.859496 Valid Acc: 0.851658
It: 400  Train Acc: 0.859611 Valid Acc: 0.851658
It: 500  Train Acc: 0.859649 Valid Acc: 0.851351
```

No regularization

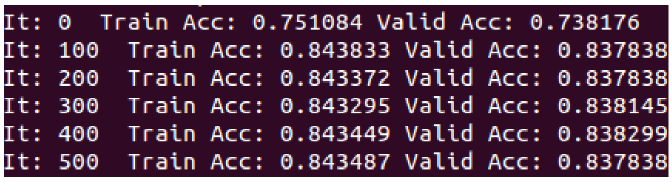


```
It: 0  Train Acc: 0.759223 Valid Acc: 0.746314
It: 100  Train Acc: 0.859419 Valid Acc: 0.850737
It: 200  Train Acc: 0.859380 Valid Acc: 0.851351
It: 300  Train Acc: 0.859688 Valid Acc: 0.851505
It: 400  Train Acc: 0.859649 Valid Acc: 0.851198
It: 500  Train Acc: 0.859726 Valid Acc: 0.851198
```

With regularization

5.請討論你認為哪個 attribute 對結果影響最大?

By removing attributes one by one, we can see that Capital Gain has a significant correlation to >50K or not. Removing Capital gain reduces accuracy by a whole 1% shown below when compared to the results shown in Q4. Other attributes with strong correlation to >50K include age, relationships, and education.



```
It: 0  Train Acc: 0.751084 Valid Acc: 0.738176
It: 100  Train Acc: 0.843833 Valid Acc: 0.837838
It: 200  Train Acc: 0.843372 Valid Acc: 0.837838
It: 300  Train Acc: 0.843295 Valid Acc: 0.838145
It: 400  Train Acc: 0.843449 Valid Acc: 0.838299
It: 500  Train Acc: 0.843487 Valid Acc: 0.837838
```