學號：B06901066　系級：電機三 姓名：孟妍

1. (2%) 請比較實作的 generative model 及 logistic regression 的準確率，何者較佳？請解釋為何有這種情況？

logistic model

```
Training loss: 0.26436490334768664
Development loss: 0.2842827489675203
Training accuracy: 0.8858488634036453
Development accuracy: 0.8763361592333211
```

generative model

```
Training accuracy: 0.8693232084930699
```

實作 logistic regression 跟 generative model，logistic model 的效果好很多。Logistic regression model 的是 posterior probability，在有足夠 data 的情況下，logistic regression 的正確率就會比較高。Generative model 使用的是 Naive Bayes，有點類似腦補的概念，會需要先假設機率分佈，在資料少或資料有很多雜訊時才會比較有幫助。

2. (2%) 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。接著嘗試對正規項使用不同的權重 (lambda)，並討論其影響。(有官 regularization 請參考 https://goo.gl/SSWGhf p.35)

with regularization

```
Lambda:  1
Training loss: 0.49961044666957083
Development loss: 0.4971205770555701
Training accuracy: 0.7942658201925046
Development accuracy: 0.7963509030593439


Lambda:  0.1
Training loss: 0.47085049414408664
Development loss: 0.46895058058412203
Training accuracy: 0.7942658201925046
Development accuracy: 0.7963509030593439


Lambda:  0.01
Training loss: 0.37450921135739
Development loss: 0.37584650426719896
Training accuracy: 0.8178373950440303
Development accuracy: 0.8168079616660524


Lambda:  0.001
Training loss: 0.2987419982894614
Development loss: 0.3085380951159162
Training accuracy: 0.871738685234487
Development accuracy: 0.8661997788426097
```

without regularization

```
Training loss: 0.26436490334768664
Development loss: 0.2842827489675203
Training accuracy: 0.8858488634036453
Development accuracy: 0.8763361592333211
```

這裡用了四個不同的 lambda 值，[1,0.1,0.01,0.001]，得到的結果是 lambda 值越小的時候，accuracy 會較高，但在這裡加了 regularization 項對 accuracy 並沒有太好的影響。此題是只用 google colab 的 sample code 跑的，右圖是未加 regularization term 的結果，看起來沒有加的 accuracy 甚至還稍好一些。另外，當 lambda 值較大的時候，training 和 development set 上的 loss 和 accuracy 會變得很接近，甚至 development loss 比 training loss 稍低，development accuracy 稍微高於 training accuracy。加入 regularization 的目的是要避免 overfitting，但可能在這題跑的原始 model 還沒有那麼好，所以加入 regularization 的幫助不大，還沒有必要加入。

3. (1%) 請說明你實作的 best model，其訓練方式和準確率為何？

我實作的 best model 做了 feature engineering 跟套了一個小的 keras nn model。feature engineering 的部分，因為是 one-hot encoding，所以我先把一些幾乎都是 0 的 column 刪掉，然後把 continuous 項挑出來，乘 0.8 加了三次（不知道為時麼直接加會比*0.8 準確率差很多），對於一些奇怪的 column (ex. 有’ ?’ 的)也*0.8 降低權重，另外多加了一項 'wage per hour'*'weeks worked in year'。nn model 的部分只有三層因為試了滿多次發現比較深的 model 結果都會爛掉。Training 的時候最好的一次 model 在 development set 上 accuracy 有到 0.88 左右，kaggle public 成績為 0.89240。

4. (1%) 請實作輸入特徵標準化 (feature normalization)，並比較是否應用此技巧，會對於你的模型有何影響。

with feature normalization

```
Training loss: 0.26300343773818885
Development loss: 0.2832427707978645
Training accuracy: 0.8861970100348147
Development accuracy: 0.8767047548838923
```

without feature normalization

```
Training loss: 4.085520639092836
Development loss: 4.114608385619859
Training accuracy: 0.7782101167315175
Development accuracy: 0.7766310357537781
```

沒有做 feature normalization 的 accuracy 會下降，loss 也大很多，因為做 feature normalization 的一個好處就是可以加速收斂，所以在原本 sample code 裡 iter 次數很少的時候，做 feature normalization 會得到較好的結果。把 iter 次數提高後， 沒有做 feature normalization 的 accuracy 有上升，但還是比有做 feature normalization 差。可以透過做 feature normalization 來降低達到一個比較好的結果所需的 iteration。