學號：R05942114 系級： 電信一 姓名：方敬勻

1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.

```
def Normalization( my_data ) :
    my_data = np.transpose( my_data )
    for i in range(105) :
        mean = np.mean( my_data[i] )
        variance = np.std( my_data[i] ) if np.std( my_data[i] ) != 0 else 1
        my_data[i] = np.divide( np.subtract(my_data[i], mean), variance )
    my_data = np.transpose(my_data)
    return my_data
```

對 data 用 mean 以及 std 做完標準化之後發現跟沒有做標準化出來的結果差不多標準化稍微好了一點點，大概是因為標準化之後會把 rating 比較特別的點讓他影響沒有這麼大 train 出來的 model 就可以比較符合大部分的情形

本次是用的對照是使用 K_vector145 和 0.2validate data 作為觀察對象

	Kaggle:Public/Private
normalize	0.88226/0.88416
Without normalize	0.89355/0.89337

2. (1%)比較不同的 latent dimension 的結果。

一樣用 K_vector145 和 0.2validate data 作為觀察對象，可以發現當 K145 時 performance 為中間增加為 215performace 增加推測是增加 model set 可以找到更正確的 model，但在後來 K=275 performance 卻降低可能是因為 train 的 EarlyStop 的參數設定沒設好造成 model train 的不夠深。

latent dimension	Kaggle:Public/Private
145	0.88226/0.88416
215	0.87773/0.87881
245	0.91531/0.91863

3. (1%)比較有無 bias 的結果。

一樣使用 145 作為 baseline 加上 bias

使用 Bias 之後 train 的時候發現 trainset 變好但是 validate 變差因為加了 bias 可以更加貼近 trainset 讓 trainset 更加準確

Bias	Kaggle:Public/Private (30% train data)
無	0.88226/0.88416
有	0.91488/0.91565

4. (1%)請試著用 DNN 來解決這個問題，並且說明實做的方法(方法不限)。並比較 MF 和 NN 的結果，討論結果的差異。
   使用 K=275 的 deep model 發現 train 出來的結果比之前所 train 出來的結果都差很多個人推測可能是因為使用 DNN 之後 model 會更加的貼近 train set 而導致在 kaggle 上面的分數比較不理想，第二點的原因可能是因為 DNN 設的 Epoch 不夠大所以會造成 model 沒有 train 完成而導致 performance 不好。

	Kaggle
DNN	0.96321/0.96737

5. (1%)請試著將 movie 的 embedding 用 tsne 降維後，將 movie category 當作 label 來作圖。

6. (BONUS)(1%)試著使用除了 rating 以外的 feature, 並說明你的作法和結果，結果好壞不會影響評分。