ML2017 HW6 Report

學號：B02901124 系級： 電機四 姓名：黃柏翔

1. **(1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.**
   以下是我實作 MF 及 DNN 架構，並分別比較有無 normalization 的差別。並有我在 validation data 及丟上 Kaggle 的 public 的 RMSE 的表格。MF 的 latent dimension 為 16，DNN 的為 256。
   Normalize 的方式跟助教投影片上的一樣：先減掉 rating 的 mean 並除以 standard deviation。

Normalization	MF(16)-validation	MF-Kaggle	DNN(256)-validation	DNN-Kaggle
有	0.876615	0.87384	0.862619	0.8603
無	0.869411	0.87136	0.8608	0.8606

   由此表可以發現，普遍來說 normalization 對於預測結果的效果不佳，我想是因為在更新 matrix 時原本 rating 的範圍在 0~5 之間，matrix 學到的值可以比較精確；而經過 normalize 後 matrix 學到的值在 predict 的時候有點像是做一些 upsample 的感覺，所以 predict 出的值會比較不精確。

2. **(1%)比較不同的 latent dimension 的結果。**
   下表是我在 MF 及 DNN 上利用不同 latent dimension 在 validation data 上所得到 RMSE 的值：

Latent Dimension	8	16	32	64	128	256
DNN	0.872673	0.861879	0.86063857	0.861445	0.860908	**0.86051**
MF	0.866519	0.862662	0.86294607	**0.861026**	0.863263	0.864743

   單就 MF 來看的話，latent dimension 設為 64 時有最好的效果，參數再更多的話明顯看出有變差的趨勢
   而 DNN 在 latent dimension 設為 256 時在 validation data 上有最好的效果。

3. **(1%)比較有無 bias 的結果。**

下表為 MF 在某些 latent dimension 下有無加 bias 的比較：

Bias	16	32	64	128	256
有	0.862662	0.862946	0.86102596	0.863263	0.864743
無	0.864623	0.865363	0.86293479	0.86348	0.86578

可以發現在有加入 bias 後在各個 latent dimension 下都能有效降低 RMSE，代表每 user 可能有自己的一套評分標準，也可為每部電影定一個水平。

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。**

關於 MF 及 DNN 的比較如下表(與第 2 題同)：

Latent Dimension	8	16	32	64	128	256
MF	0.866519	0.862662	0.86294607	0.861026	0.863263	0.864743
DNN	0.872673	0.861879	0.86063857	0.861445	0.860908	0.86051

從結果看來在大部分的狀況下 DNN 的效果都比 MF 來的好(在 Kaggle 上更明顯)
我所用的 DNN 架構只是單純將 user embedding 以及 movie embedding 的 output 做 batch normalization 並 concatenate 在一起後通過一層與 latent dimension 相同的 neuron 數(128 維)的 dense layer，接上一層 0.5 的 Dropout，最後再接一層一個 output 的 dense layer 做 regression。
這樣的架構在 Kaggle 上的 public 卻可以達到 0.858 左右的 rmse(MF 最高只能達到 0.871 左右)。

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