ML2017FALL-HW5-Report

學號：B03801039　系級： 電機四　姓名：楊福恩

1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.
   (collaborator:)
   無 normalize: kaggle public RMSE=0.85567
   有 normalize: kaggle public RMSE=0.91243
   本次作業實作的無 normalize 的 matrix factorization 模型，latent dimension 設為 110，epochs 取 18，optimizer 為 adamax，以上為測試過各種不同數值與不同 optimizer 後所決定的 model。
   本題 normalize 的方式如同 TA Hour 的 ppt 中所描述的方法，取 training data rating 的 mean 及 standard deviation，再把 training data 的 rating 做 normalization 後，用 normalized 過後的 rating 做為 training 的 target。Testing 的時候則是將 predict 後的結果乘上 training data rating 的 standard deviation，再加上 training data rating 的 mean。
   由 public 上的結果可看出，對 rating 做 normalize 後，結果反而變差，推測原因可能為在 testing 的部分，最後輸出的結果是 predict 後再乘上 training 的 standard deviation 再加上 mean，這個換算的過程可能產生額外的誤差，造成 RMSE 變大。

2. (1%)比較不同的 latent dimension 的結果。
   (collaborator:)
   以下列舉其中幾個實驗後的結果，下表中的 RMSE 皆為 kaggle public 的分數

Latent dimension	90	100	110	120	130
RMSE	0.85986	0.85688	0.85567	0.85887	0.85607

   由上表可知在 Latent dimension 為 90~130 這個區間，dimension=110 時結果最佳，但和其他 dimension 的結果差距並不大。

3. (1%)比較有無 bias 的結果。
   (collaborator:)
   無 bias: kaggle public RMSE=0.85567
   有 bias: kaggle public RMSE=0.87745
   此題所使用的依然是 latent dimension 設為 110，epochs 取 18，optimizer 為 adamax 的 model，加上 bias 後發現結果略差，觀察在相同 epoch 的訓練下，無 bias 的 model 在 training data 上的 loss 為 0.5731，有 bias 的 model 在 training data 上的 loss 則為 0.5308，可推測可能因為加入 bias 後的 model 參數量較多，故 overfitting 較嚴重，使得在 public 上結果變差。

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。
(collaborator:廖宜倫 B03901001)
DNN model: kaggle public RMSE=0.89983
MF model: kaggle public RMSE=0.85567

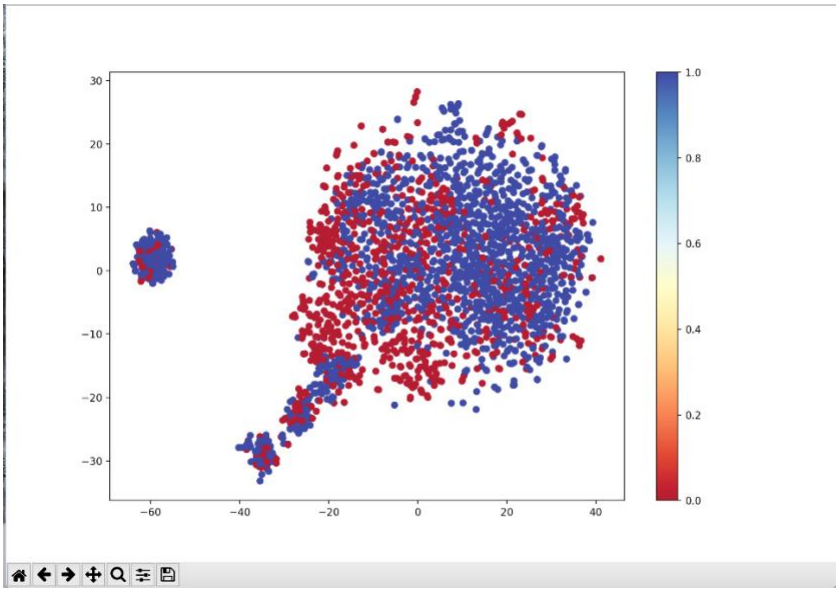
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 1)	0	
input_2 (InputLayer)	(None, 1)	0	
input_3 (InputLayer)	(None, 1)	0	
embedding_1 (Embedding)	(None, 1, 75)	453075	input_1[0][0]
embedding_2 (Embedding)	(None, 1, 75)	296475	input_2[0][0]
embedding_3 (Embedding)	(None, 1, 75)	453075	input_3[0][0]
flatten_1 (Flatten)	(None, 75)	0	embedding_1[0][0]
flatten_2 (Flatten)	(None, 75)	0	embedding_2[0][0]
flatten_3 (Flatten)	(None, 75)	0	embedding_3[0][0]
concatenate_1 (Concatenate)	(None, 225)	0	flatten_1[0][0] flatten_2[0][0] flatten_3[0][0]
dense_1 (Dense)	(None, 150)	33900	concatenate_1[0][0]
dense_2 (Dense)	(None, 50)	7550	dense_1[0][0]
dense_3 (Dense)	(None, 1)	51	dense_2[0][0]

Total params: 1,244,126
Trainable params: 1,244,126
Non-trainable params: 0

本題 DNN model 的架構如上圖，實作方式為將 user embedding 以及 movie embedding concatenate 在一起再過 DNN 得出 rating，此題經過多次嘗試後，選定了以下參數，atent dimension 為 75，epochs 取 18，optimizer 為 adamax，由實驗結果可知 MF model 在 kaggle public 的結果較佳，但觀察 training data 上的情況，兩模型訓練相同 epoch 後，DNN model 的 loss 為 0.4789，MF model 的 loss 為 0.5731，故推測應為 DNN model 的 overfitting 較嚴重，造成 kaggle public 的結果較差。

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

下圖紅色點為將'Thriller',' Horror',' Crime' 這幾個相似的類別視為相同的大類別，藍色點則為將' Drama',' Musical 這兩個視為一個大類別，觀察上圖中，中間偏右上的部分，可發現紅點和藍點分布上略有分開的趨勢，但並沒有非常顯著。



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。

(collaborator:)

本題為從 users.csv 中，取 Age 這一項 feature，加入第 4 題的 DNN model 中，加入方式為找到每一個 user 所對應的年齡，依照每一個 user 的順序形成一個 array，在將其 embedding 後，與 user embedding 以及 movie embedding concatenate 在一起再過 DNN。實驗結果為 kaggle public RMSE=0.95570，明顯結果變糟，推測原因可能為年齡和 rating 的關連並不大，加入此 feature 反而造成結果變差。