學號：B04901061 系級： 電機三 姓名：蔡忠紘

1. (1%)請比較有無normalize(rating)的差別。並說明如何normalize.
   (collaborator : None)
   < latent dimension = 12 >

	Public Score	Private Score	Average score
有Normalize	0.86517	0.86610	0.8656
無Normalize	0.86733	0.86767	0.8675

   將training data的rating扣掉平均，再除以標準差後，當作training的target；在testing時再將結果乘以rating的標準差再加上平均，即是最後結果。
   實驗結果顯示，有做Normalization比沒做準確。

2. (1%)比較不同的latent dimension的結果。
   (collaborator: None)

Latent dimension	Public Score	Private Score	Average score
8	0.86736	0.86624	0.8668
10	0.86446	0.86454	0.8645
12	0.86517	0.86610	0.8656
16	0.86682	0.86768	0.8673
20	0.87281	0.87064	0.8717

   實驗結果顯示，Latent dimension在10附近時，結果最為準確。

3. (1%)比較有無bias的結果。
   (collaborator : None)
   < latent dimension = 10 >

	Public Score	Private Score	Average score
有bias	0.86446	0.86454	0.8645
無bias	0.86985	0.86845	0.8692

   實驗結果顯示，有加bias的結果較準確。

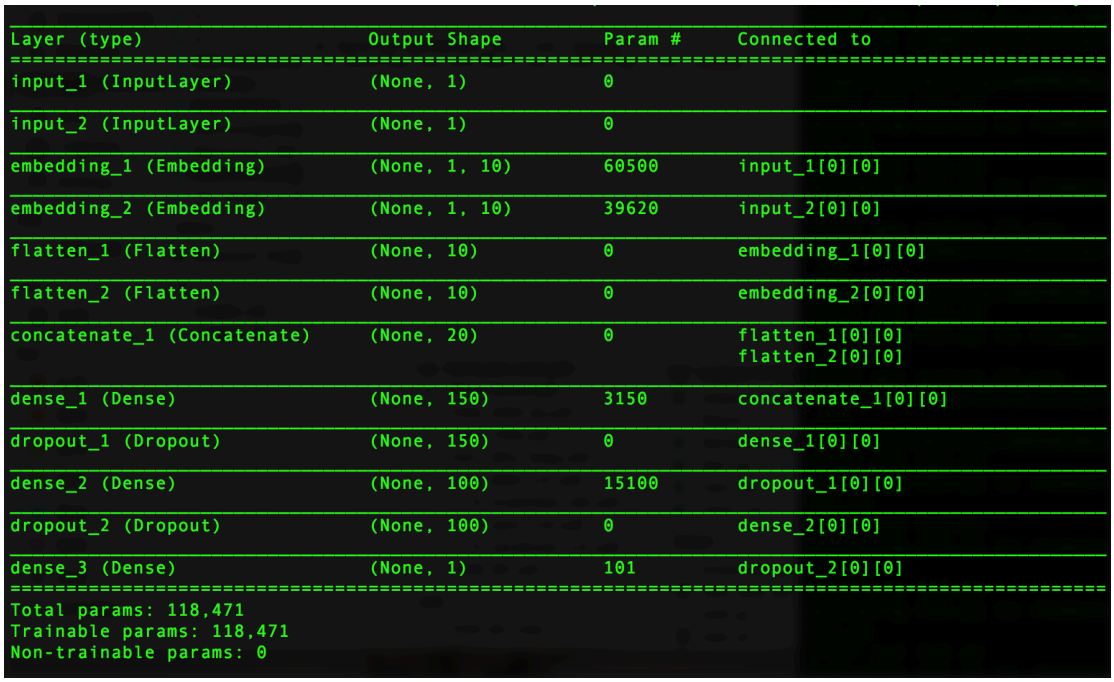
4. (1%)請試著用DNN來解決這個問題，並且說明實做的方法(方法不限)。並比較MF和NN的結果，討論結果的差異。
(collaborator : None)

MF settings:

< latent dimension = 10、normalization = True、bias = True >

DNN settings:

< latent dimension = 10、normalization = True >



Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 1)	0	
input_2 (InputLayer)	(None, 1)	0	
embedding_1 (Embedding)	(None, 1, 10)	60500	input_1[0][0]
embedding_2 (Embedding)	(None, 1, 10)	39620	input_2[0][0]
flatten_1 (Flatten)	(None, 10)	0	embedding_1[0][0]
flatten_2 (Flatten)	(None, 10)	0	embedding_2[0][0]
concatenate_1 (Concatenate)	(None, 20)	0	flatten_1[0][0] flatten_2[0][0]
dense_1 (Dense)	(None, 150)	3150	concatenate_1[0][0]
dropout_1 (Dropout)	(None, 150)	0	dense_1[0][0]
dense_2 (Dense)	(None, 100)	15100	dropout_1[0][0]
dropout_2 (Dropout)	(None, 100)	0	dense_2[0][0]
dense_3 (Dense)	(None, 1)	101	dropout_2[0][0]

Total params: 118,471
Trainable params: 118,471
Non-trainable params: 0

	Public Score	Private Score	Average score
MF	0.86446	0.86454	0.8645
DNN	0.88346	0.88325	0.8834

實驗結果顯示，MF的效果較好。
DNN model調過很多次參數，疊深、增加參數量結果都沒有變好，可能這個task還是適合用MF做。

5. (1%)請試著將movie的embedding用tsne降維後，將movie category當作label來作圖。
(collaborator : None)

Category 1（紅色）:

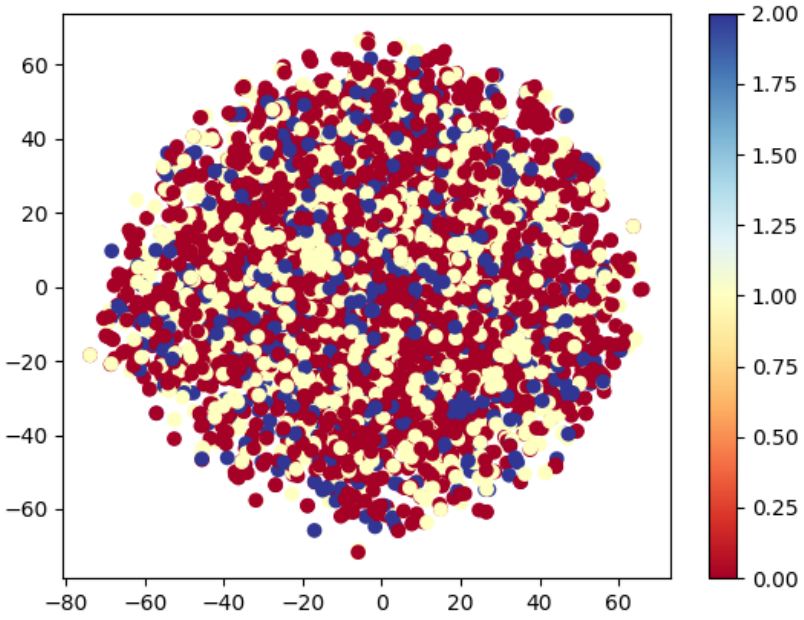
['Action','Adventure','Sci-Fi','Western','Documentary','Drama']

Category 2（鵝黃色）:

['Animation','Children\'s','Comedy','Musical','Romance','Fantasy']

Category 3（藍色）:

['Crime','Film-Noir','Horror','Mystery','Thriller','War']



因為是用Ratings去train embedding，三個大類資料沒有分很開其實合理。

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: None)

< latent dimension = 10、normalization = True、bias = True >

將Movie genres做one-hot encoding，每部電影只取其中一個類別，進去MF model中當第三層input，並將其與user做dot，再加上原先movie id和user dot的結果和兩層bias後作為output。

結果顯示，只用rating的效果較好，可能是因為取電影類別只取一項，若全取說不定會有更好的結果。

	Public Score	Private Score	Average score
只用rating	0.86446	0.86454	0.8645
加上genres	0.87591	0.87649	0.8762

```
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, 25)             151250      input_1[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)      (None, 1, 25)             99050       input_2[0][0]
__________________________________________________________________________________________________
embedding_3 (Embedding)      (None, 1, 25)             99050       input_3[0][0]
__________________________________________________________________________________________________
flatten_1 (Flatten)          (None, 25)                0           embedding_1[0][0]
__________________________________________________________________________________________________
flatten_2 (Flatten)          (None, 25)                0           embedding_2[0][0]
__________________________________________________________________________________________________
flatten_3 (Flatten)          (None, 25)                0           embedding_3[0][0]
__________________________________________________________________________________________________
embedding_4 (Embedding)      (None, 1, 1)              6050        input_1[0][0]
__________________________________________________________________________________________________
embedding_5 (Embedding)      (None, 1, 1)              3962        input_2[0][0]
__________________________________________________________________________________________________
dot_1 (Dot)                  (None, 1)                 0           flatten_1[0][0]
                                                                   flatten_2[0][0]
__________________________________________________________________________________________________
dot_2 (Dot)                  (None, 1)                 0           flatten_1[0][0]
                                                                   flatten_3[0][0]
__________________________________________________________________________________________________
flatten_4 (Flatten)          (None, 1)                 0           embedding_4[0][0]
__________________________________________________________________________________________________
flatten_5 (Flatten)          (None, 1)                 0           embedding_5[0][0]
__________________________________________________________________________________________________
add_1 (Add)                  (None, 1)                 0           dot_1[0][0]
                                                                   dot_2[0][0]
                                                                   flatten_4[0][0]
                                                                   flatten_5[0][0]
==================================================================================================
Total params: 359,362
Trainable params: 359,362
```