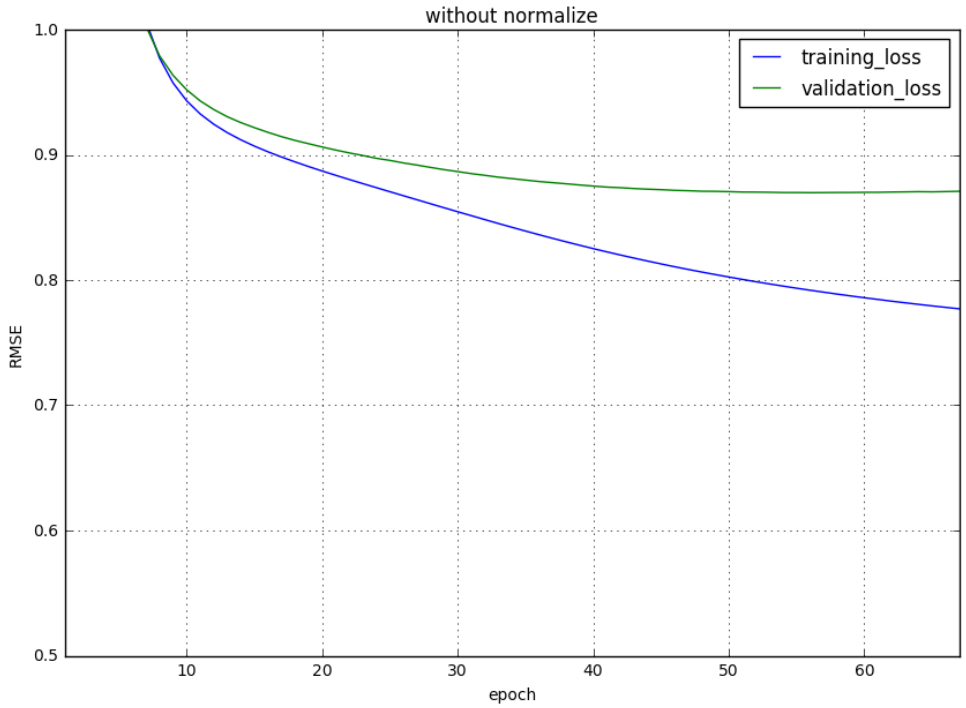
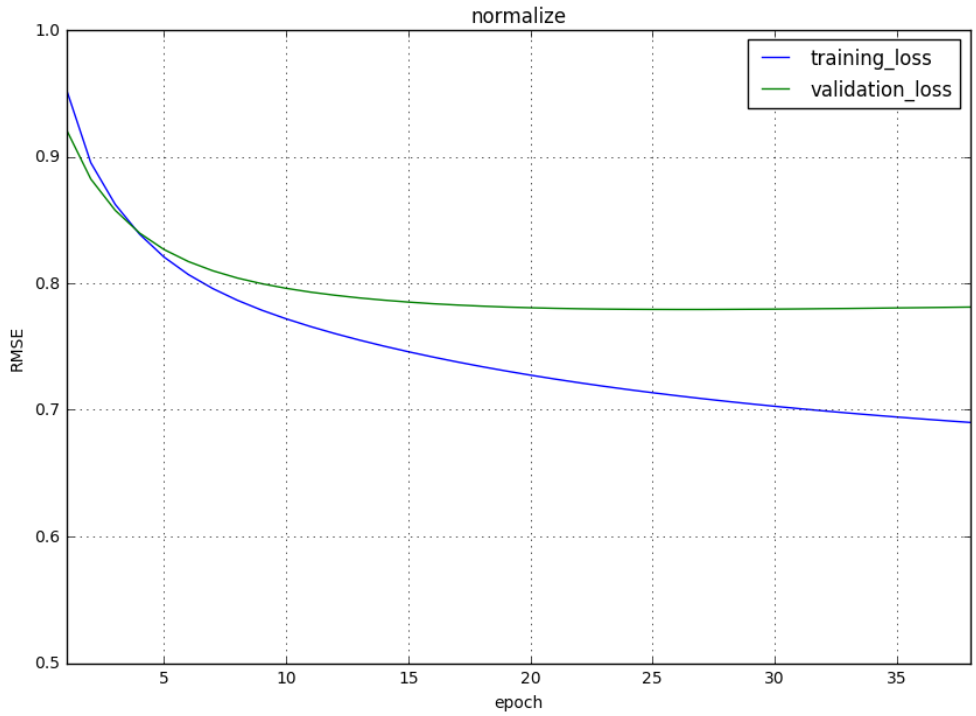
學號：B03901076　系級：電機三　姓名：林尚謙

1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.
將 training data 的 rating 減去平均後除以標準差來做 normalize，以 normalize 過後的 rating 作為 training 的 target，並在 testing 的時候根據 training 的 mean 和 std 做還原。使用相同模型架構(MFmodel，有加入 bias，latent_dim 為 15)，對有無 normalize 來做比較：

無 normalize：



有 normalize：



可以發現在有 normalize 的情況下，因為 training 的 target 是經過 normalize 的較小範圍，loss 相較之下會比較小，但最後還要做還原，所以不代表結果是比較好的。在訓練的過程中，有經過 normalize 的 valid_loss 會比較快進入收斂(overfitting)，而最終預測的準確率，沒有 normalize 略高於有 normalize 過的，這次的 case 中 normalize 並沒有帶來明顯的好處。

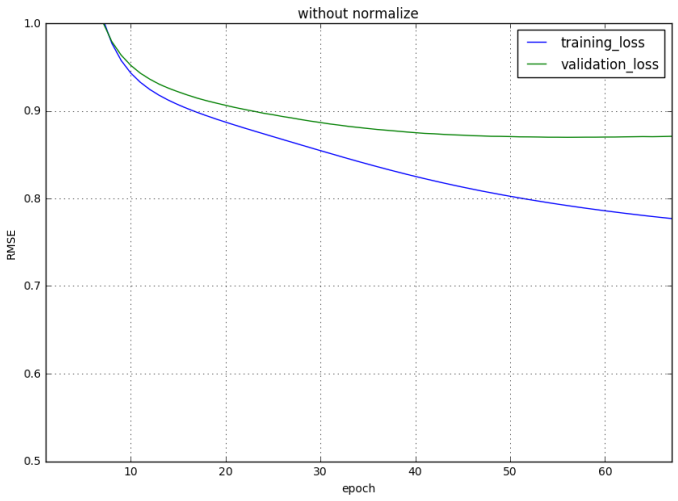
有無 normalize	無 normalize	有 normalize
RMSE(public score)	0.86799	0.87047

2. (1%)比較不同的 latent dimension 的結果。
分別使用相同的模型架構(MFmodel，有加入 bias，沒有做 normalize)，改變兩個 Embedding layer 的 latent dimension 來做比較：

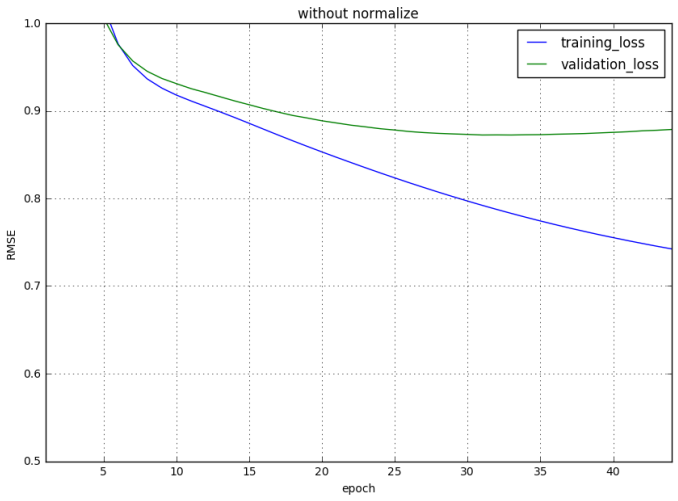
laten dimension	10	15	20
RMSE	0.86806	0.87050	0.87006
laten dimension	25	30	35
RMSE	0.87002	0.87244	0.86669

latent dimension 在 10~35 的範圍內，訓練出來的模型預測準確率沒有明顯相關性，而訓練時 batch_size 的影響還比較大。而觀察訓練過程的 traing_loss 和 valid_loss，發現如果 latent dimension 越大，讓 training_loss 下降的程度會越快，也會越早進入 overfitting，應該是 Embedding layer 有越大的 latent dimension，會有更複雜的結構和參數可以去趨向 training 的 target，但並不會使最後的預測準確率跟著增加。

latent dimension = 15



latent dimension = 30



3. (1%)比較有無 bias 的結果。

使用相同的模型架構(MFmodel，latent_dim 為 15，沒有做 normalize)，對有無加入 bias 來做比較：

有無 bias	有 bias	有 user bias	有 movie bias	無 bias
RMSE	0.86799	0.87000	0.87111	0.87229

$$r_{i,j} = U_i \cdot V_j + b_i^{user} + b_j^{movie}$$

有加入 bias 可以提升模型預測的準確率，bias 等於每個 user 和每部 movie 自己的趨勢特性，例如可能有一個 user 評分都傾向偏低，加上 bias 後就會考慮進去，使最後預測出的 rating 也會降低。

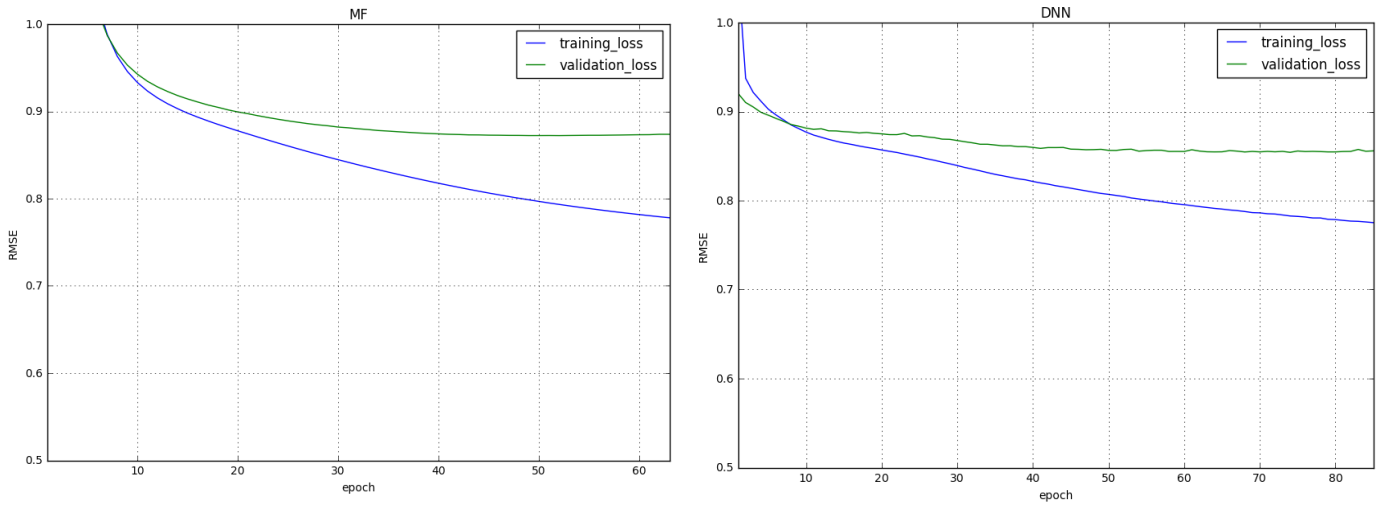
4. (1%)請試著用 DNN 來解決這個問題，並且說明實做的方法(方法不限)。並比較 MF 和 NN 的結果，討論結果的差異。

DNN 模型架構：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_57 (InputLayer)        (None, 1)                 0
_________________________________________________________________
input_58 (InputLayer)        (None, 1)                 0
_________________________________________________________________
embedding_103 (Embedding)    (None, 1, 150)            906000
_________________________________________________________________
embedding_104 (Embedding)    (None, 1, 150)            592800
_________________________________________________________________
flatten_103 (Flatten)        (None, 150)               0
_________________________________________________________________
flatten_104 (Flatten)        (None, 150)               0
_________________________________________________________________
embedding_105 (Embedding)    (None, 1, 1)              3952
_________________________________________________________________
dot_29 (Dot)                 (None, 1)                 0
_________________________________________________________________
flatten_105 (Flatten)        (None, 1)                 0
_________________________________________________________________
add_26 (Add)                 (None, 1)                 0
=================================================================
Total params: 1,502,752
Trainable params: 1,502,752
Non-trainable params: 0
```

我將 user embedding 和 movie embedding(latent_dim = 150)concatenate 在一起後，經過一個僅有一層 hidden layer 的 DNN 後，在 output layer 用 regression 問題去處理，輸出最終的 rating 分數。雖然只有一層的 hidden layer，但已經可以達到比 MF 還要好的準確率(public score 可達 0.85)。觀察 MF 與 NN 的訓練過程，訓練一開始 NN 較快把 loss 下降到一定範圍，但 NN 模型

在後半段訓練過程中 training_loss 和 valid_loss 分開的比較慢，可能是在 NN 模型中，在 concatenate 完兩個 embedding layer 進入 DNN 模型時，還有 hidden layer 到 output layer 前，都有加上 dropout layer，防止訓練太早進入 overfitting，最後也得到了較好的預測準確率。



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。