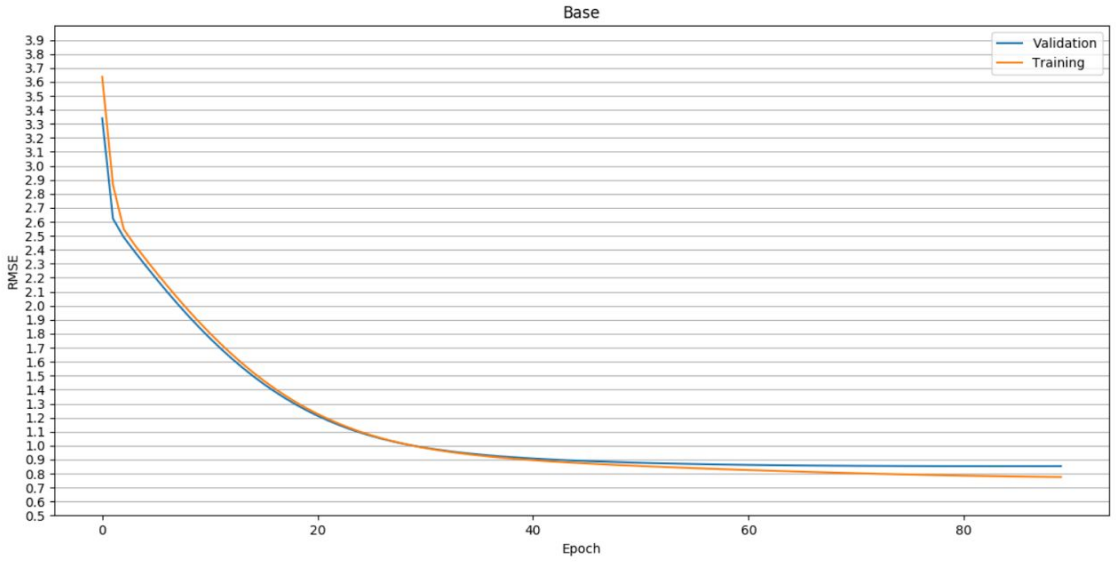
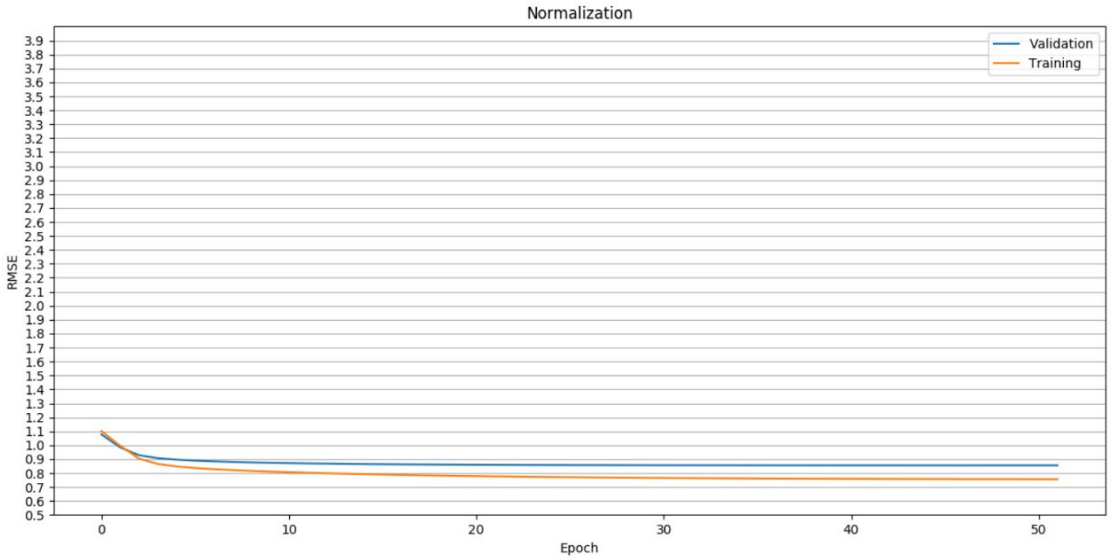
學號：B04901102 系級： 電機三 姓名：簡仲明

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

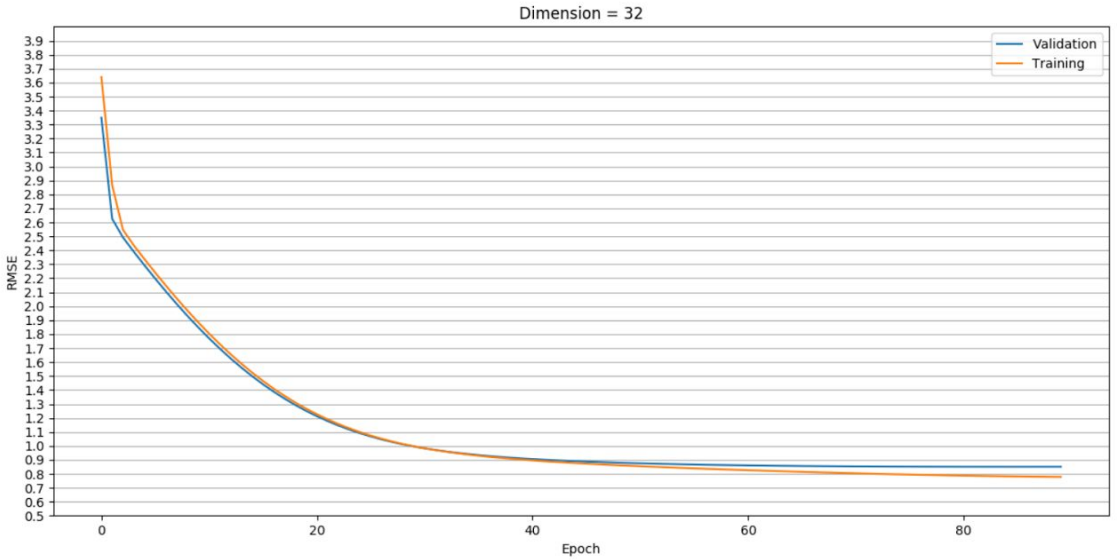
沒有normalization的情況下，最終的test rmse為0.85224，有normalization時則為0.85594，雖然前者較佳，但差距不大。接著比較兩者的training process，可以發現經過normalization後training的速度加速極多，甚至在第一個epoch時就可以得到沒有normalization第25個epoch的表現。而若比較rmse降到0.9的時間(兩者一個epoch耗時幾乎一樣)，可以發現沒有normalization時需要花40個epoch，但有normalization時只要不到5個epoch。在一樣的early stop條件下，總共少了30個epoch。因此normalization在各方面來說都是較佳的選擇。

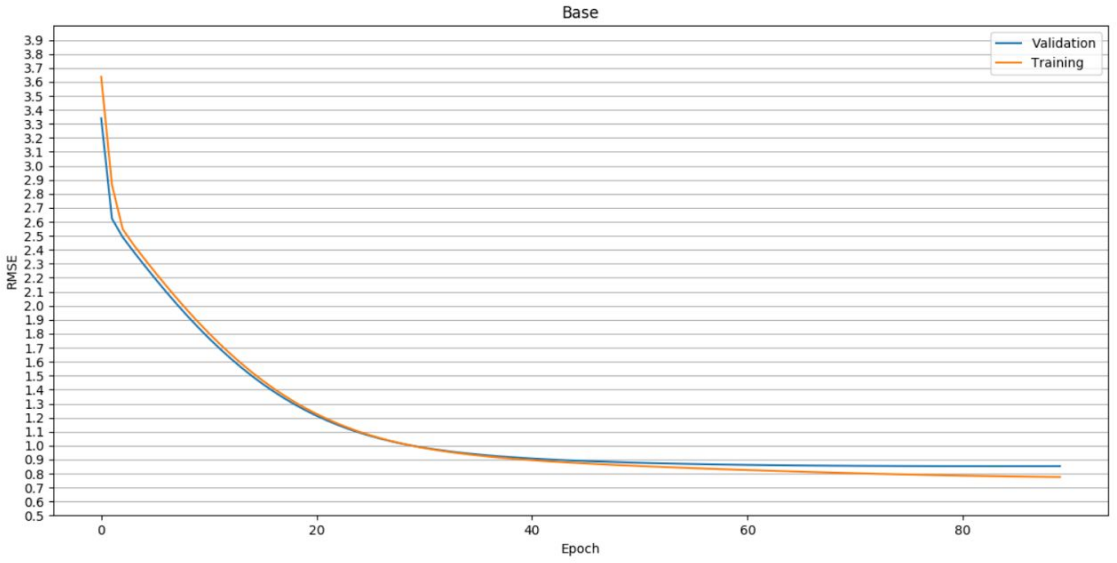


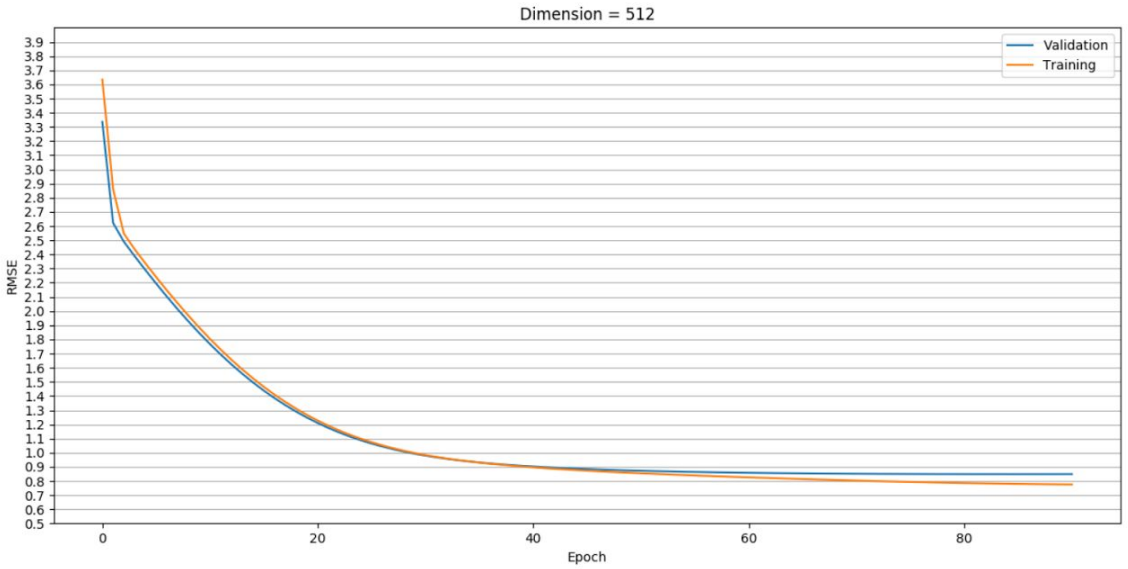


2. (1%)比較不同的latent dimension的結果。
(collaborator:無)

latent dimension為16、32、128、512時，testing rmse分別為0.85277、0.852235、0.85224、0.852725，幾乎完全在誤差範圍內(尤其後三者)，可視為相等。接著觀察training process，可以發現dimension越大時，收斂速度越快(但在一樣的early stop條件下，結束的epoch數量相同)，但因為一個epoch所花的時間大致與latent dimension(即為參數數量)成正比，因此不需要選擇那麼大的dimension。

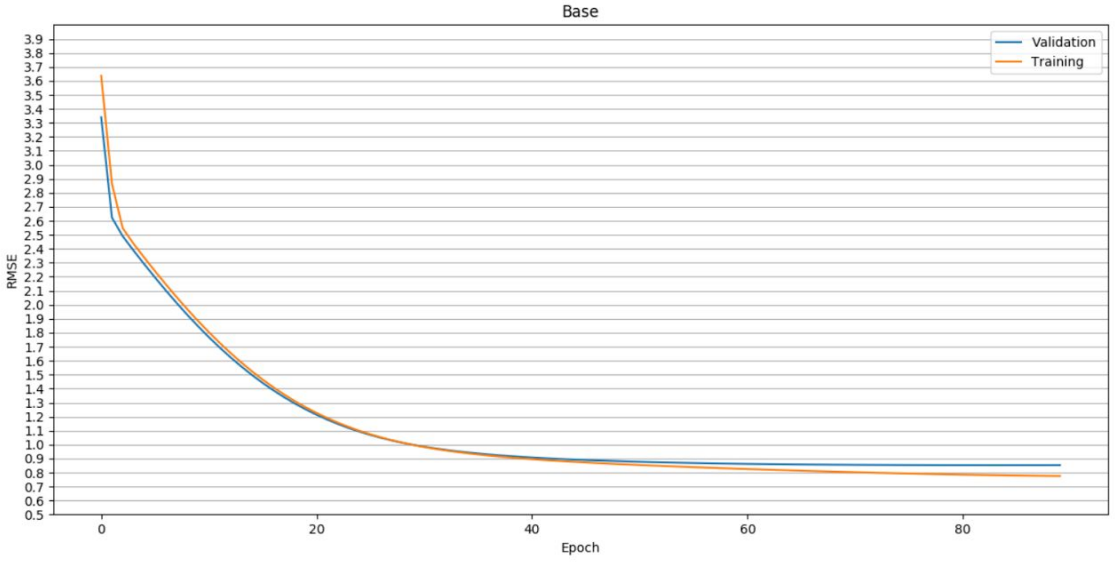


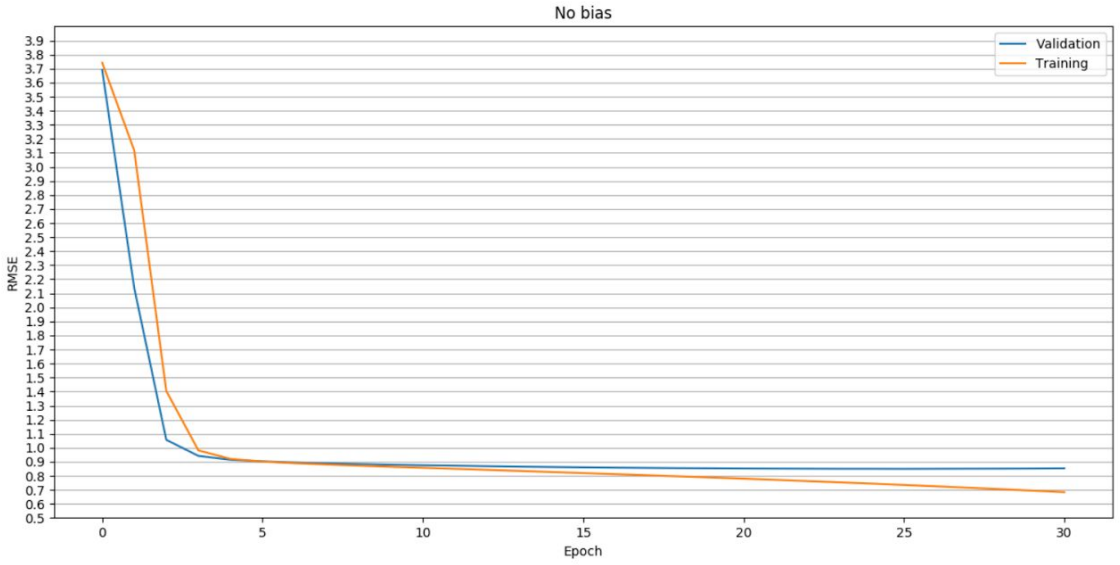




3. (1%)比較有無bias的結果。
(collaborator:無)

有bias時，testing rmse為0.85224，沒有時則為0.85611，雖然差距不大但看出bias可以些微的增進表現，接著觀察兩者的training process，可以發現不加上bias的情況下training速度快上許多，甚至高過normalization可以達到的效果，且early stop的條件也在30個epoch左右就能達到。最後，沒有加上bias的情況下，每個epoch只需要花不到二分之一的時間，因此不加上bias不一定是比較差的選擇。

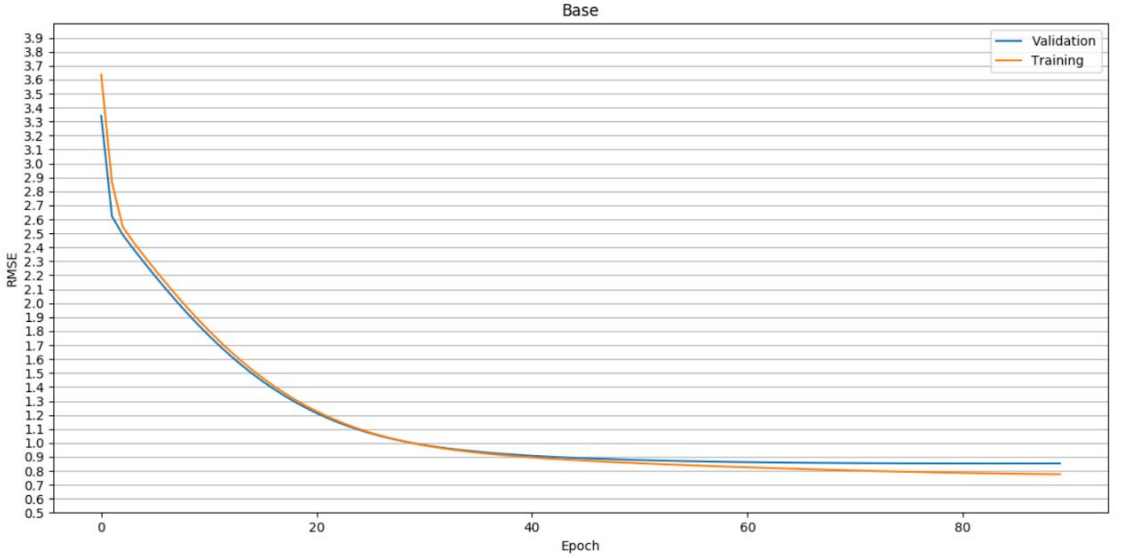


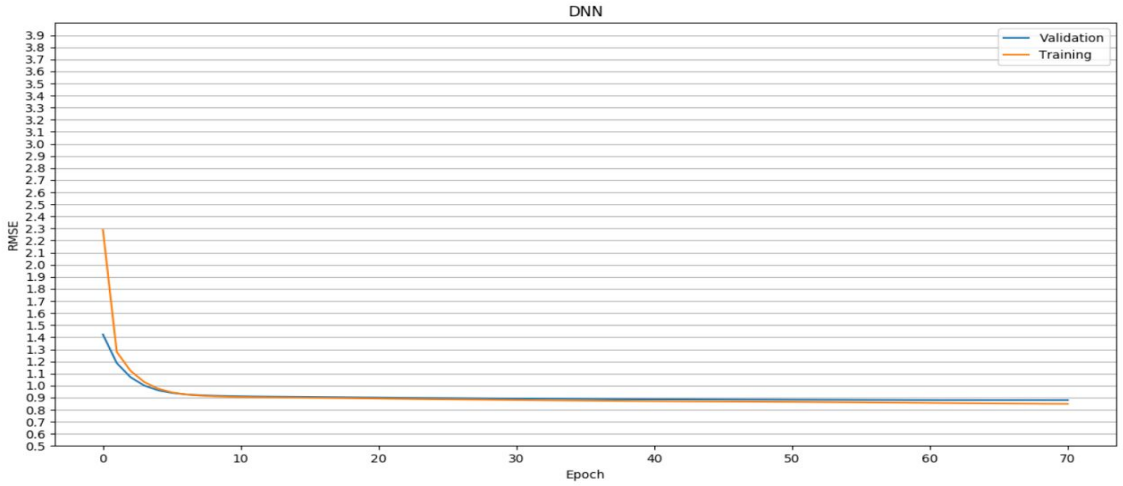


4. (1%)請試著用DNN來解決這個問題，並且說明實做的方法(方法不限)。並比較MF和NN的結果，討論結果的差異。
(collaborator:無)

```
====================Model Building====================
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1)                 0
_________________________________________________________________
input_2 (InputLayer)         (None, 1)                 0
_________________________________________________________________
embedding_1 (Embedding)      (None, 1, 128)            773248
_________________________________________________________________
embedding_2 (Embedding)      (None, 1, 128)            505984
_________________________________________________________________
flatten_1 (Flatten)          (None, 128)               0
_________________________________________________________________
flatten_2 (Flatten)          (None, 128)               0
_________________________________________________________________
concatenate_1 (Concatenate)  (None, 256)               0
                                                       flatten_1[0][0]
                                                       flatten_2[0][0]
_________________________________________________________________
dense_1 (Dense)              (None, 32)                8224
_________________________________________________________________
dense_2 (Dense)              (None, 8)                 264
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 9
=================================================================
Total params: 1,287,729
Trainable params: 1,287,729
Non-trainable params: 0
```

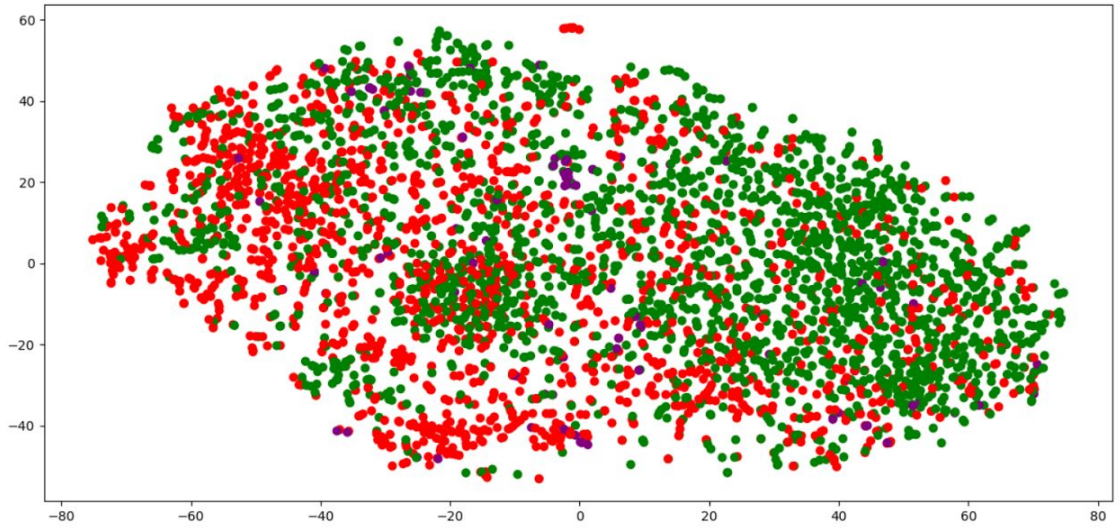
以上是我採用的DNN model，先將user以及movie丟進128維的embedding，接著concatenate，最後放進兩層hidden layer的DNN，最後得到的testing rmse為0.88161，比起MF得到的0.85224要差不少。接著觀察兩個model的training process，發現雖然DNN的收斂速度快許多，但early stop的epoch數量差不多，雖然每個epoch的耗時大概只需要二分之一，但使用3.中的不加bias方法或是1.中的normalization也可以在幾乎不影響rmse的情況下達到同樣效果，因此MF仍然是較佳的選擇。





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

(collaborator:無)



圖中綠色點代表Horror、Crime、Action以及War等等較負面、刺激或是暴力的題材，紅色代表Romance、Drama、Musical、Documentation或是Film-Noir等較溫和、人文小品類的題材，而紫色則為Animation、Fairy以及Mystery等等兒童類的題材(數量比想像中少很多...)，大致可以看出區分。值得注意的是，latent dimension太高的情況下，降維後的分區會十分不明顯。

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:無)

我使用的方法為先將user及movie丟進128維的embedding，接著互相dot，另一方面將user的gender、age以及occupation分別embedding成4維的vector再concatenate，和12維embedding的movie genre互相dot來取代bias的效果。最後將128+12=140維的vector丟進一層hidden layer的DNN，完成我的model。可以得到0.84813的testing rmse，比前面任何一個model都有一定程度的進步。觀察training process可以發現，收斂的速度以及early stop的epoch數皆與前幾個model中最佳的表現相當，只需要不到30個epoch，而且一個epoch所需的時間也與前面model幾乎相同(因為DNN的參數比起embedding來說極少)，可以說是在各方面都有最佳表現的model。