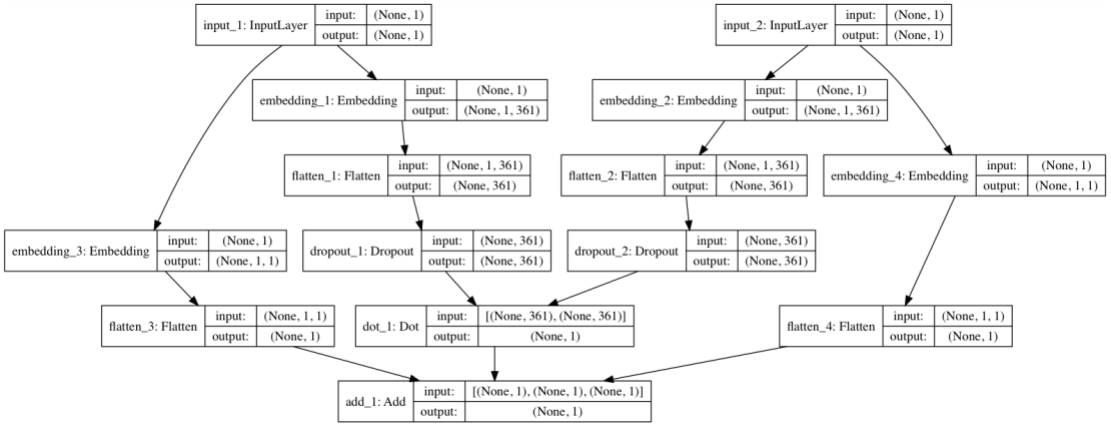
學號：B03902096 系級： 資工四 姓名：陳柏屹

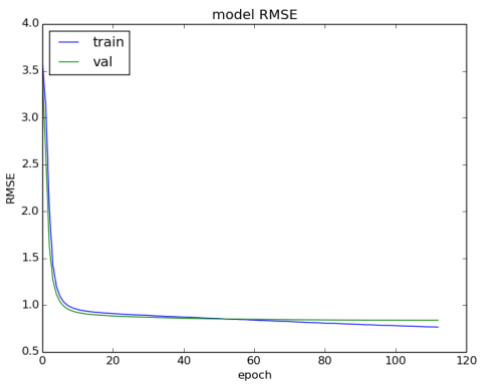
第 1.到第 3.的基本模型均使用以 Adam(lr=0.00005), batch_size=32, validation=10%, Dropout rate=0.5 , Embedding initializer = glorot_normal, bias embedding initializer = zero, EarlyStopping (monitor=val_RMSE, patience=5)



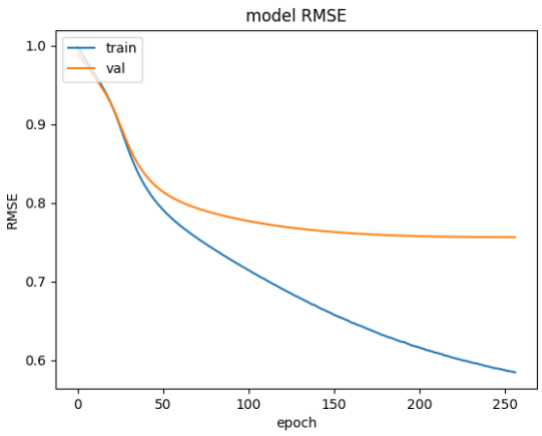
1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize. (collaborator:)

   Normalize 的方法為：Training 時先用 np.mean()找出 rating 的平均值 M, 並將所有 rating 減去 M, 再用 np.std()計算標準差 S, 並將 rating 除以 S。Testing 時再將算出的 rating * S + M。實作時我使用 bias, latent dimension=361, 其餘設定在最上面附註。看起來是否有使用 Normalize 的結果並沒有顯著差距，但

	Kaggle public accuracy	Kaggle private accuracy
Not Normalized	0.84796	0.84729
Normalized	0.84781	0.84822



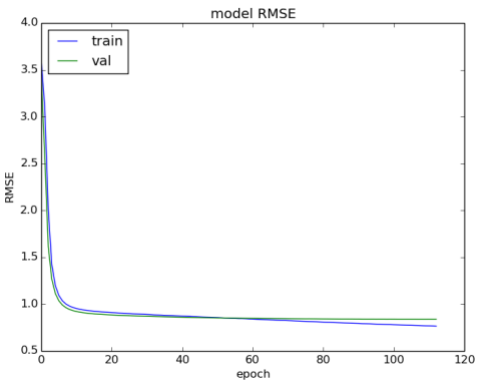
Not Normalized



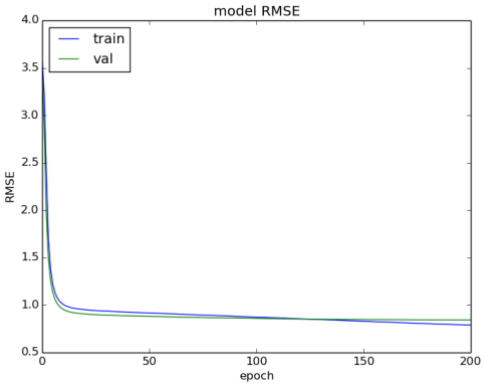
Normalized

2. (1%)比較不同的 latent dimension 的結果。
(collaborator:)
實作時我使用 bias 並不使用 normalize，其餘設定在最上面附註。可以發現 latent dimension 變大還是對於準確率有一定正面幫助。

	Kaggle public accuracy	Kaggle private accuracy
Dimension=361	0.84796	0.84729
Dimension=180	0.84954	0.84952



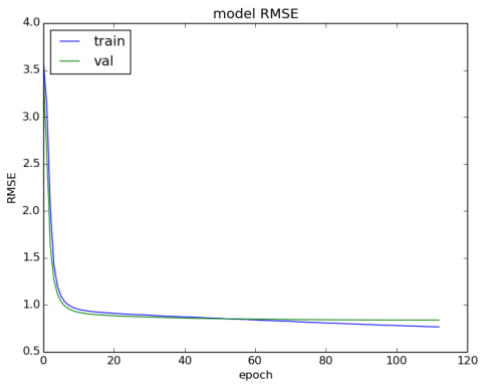
latent dimension = 361



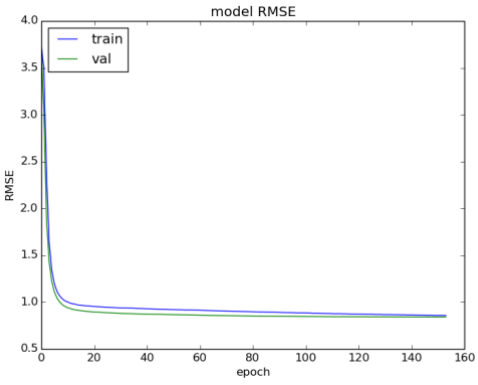
latent dimension=180

3. (1%)比較有無 bias 的結果。
(collaborator:)
實作時我使用 latent dimension=361 並不使用 normalize，其餘設定在最上面附註。model 去除 bias 部分，也就是變成 dot 完後直接 output。觀察結果後發現加入 bias 後準確率較佳。

	Kaggle public accuracy	Kaggle private accuracy
Biased	0.84796	0.84729
Not biased	0.84942	0.85131

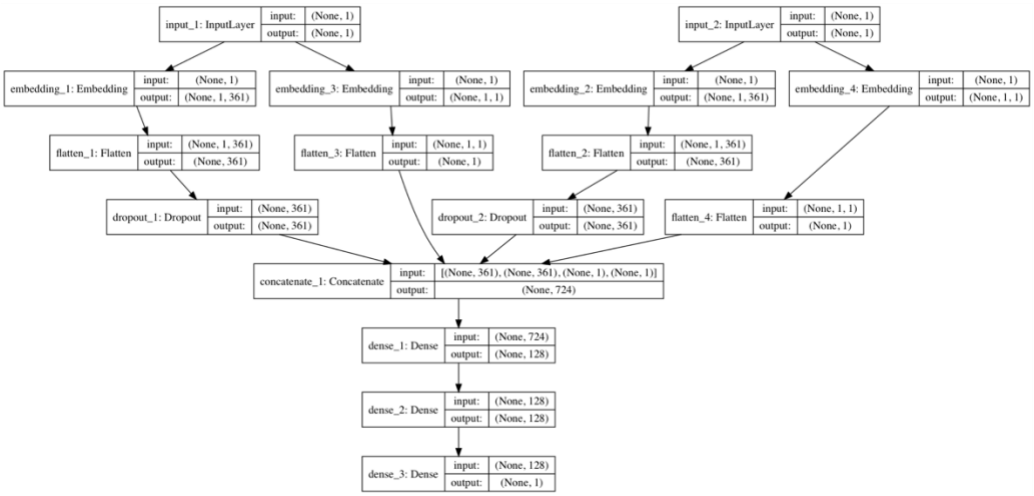


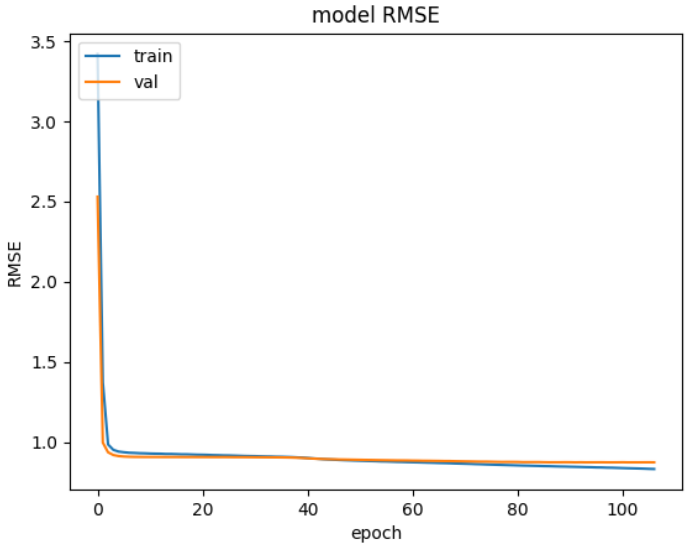
biased



unbiased

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:)





除了改使用 Dense 其餘參數皆和原本相同。可以發現 Dense model 的 RMSE 收斂的速度比原 dot model 還要快，但是 Dense 所能夠達到的最小 RMSE 卻輸給 dot model。或許是我的 dense model 不夠 Deep，或者需要更多 dropout。

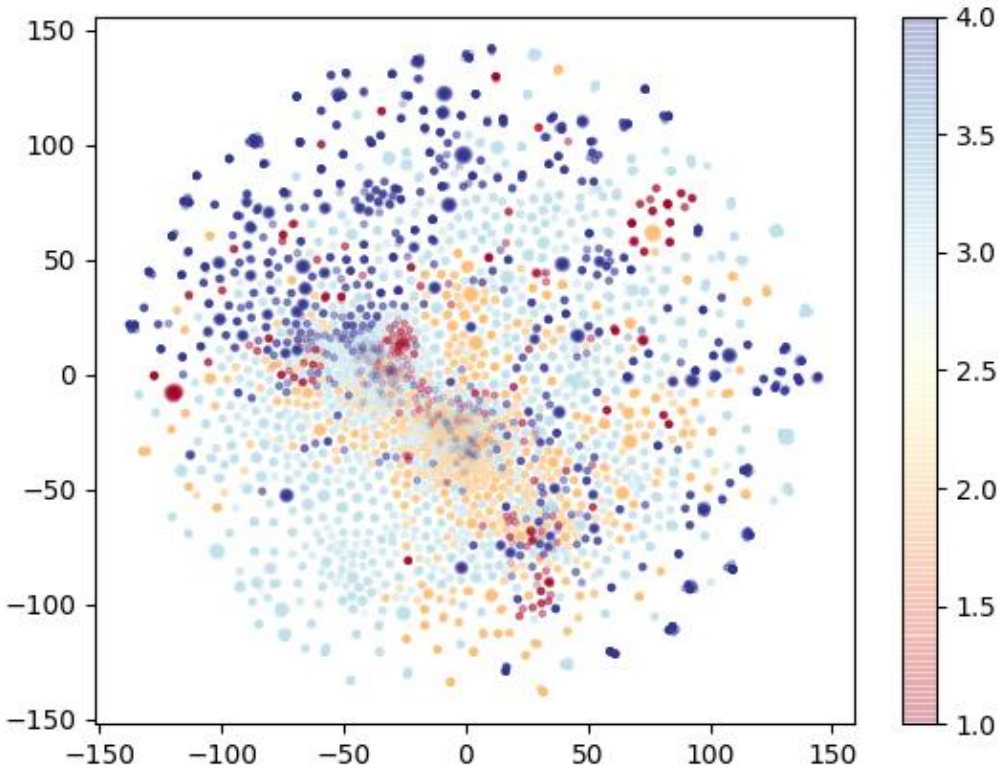
	Kaggle public accuracy	Kaggle private accuracy
Dense model	0.87573	0.87826
Dot model	0.84796	0.84729

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

(collaborator:b03902016 周聖荃)

我的分類:

	包含的 category
分類 1	"Thriller", "Horror", "Mystery"
分類 2	"Drama" , "Musical" , "Romance"
分類 3	"Children's" , "Animation", "Adventure", "Sci-Fi", "Fantasy", "Comedy"
分類 4	"War" , "Action" , "Documentary" , "Western", "Film-Noir", "Crime"

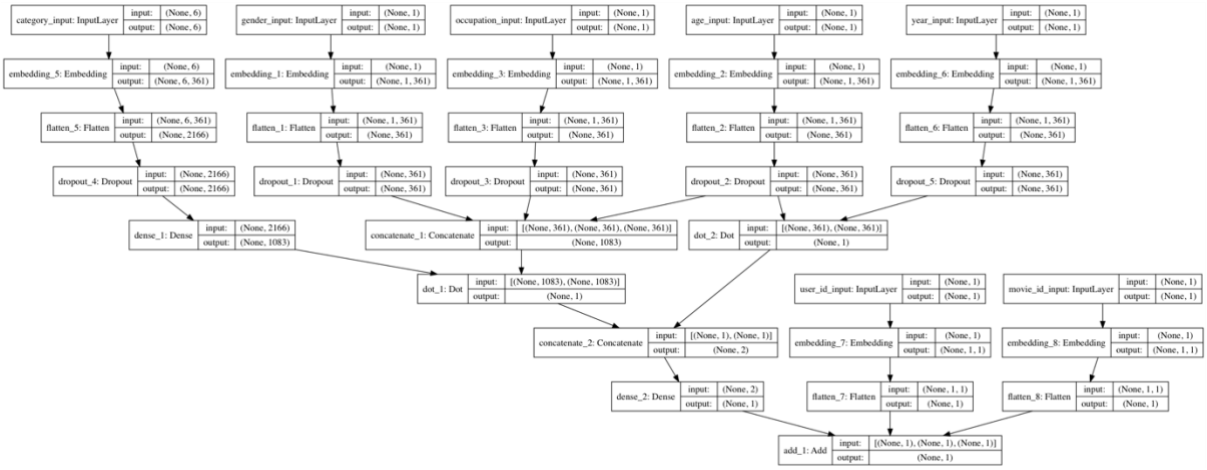


我使用 train 裡面前 20 萬個資料，可以發現分類 4(深藍色)與分類 2(黃色)有明顯的分佈區域，而分類 2（紅色）可以發現大致可以分為三團，或許分別代表它包含的 3 種 category。而分類 3(淺藍)的分佈則最平均，並沒有明顯區域。

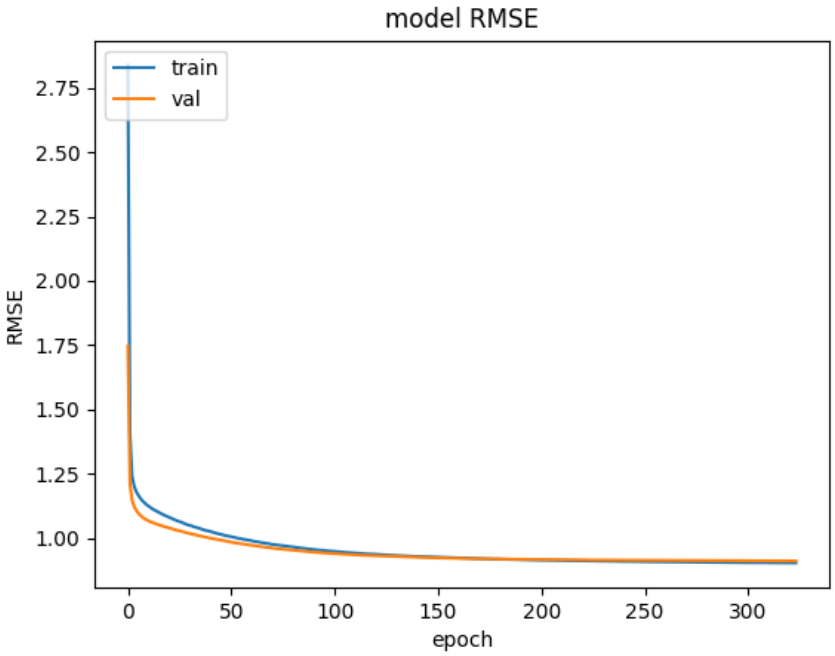
6. (BONUS)(1%)試著使用除了 **rating** 以外的 **feature**, 並說明你的作法和結果, 結果好壞不會影響評分。

(collaborator:)

我除了原本的 id 加入 user 的 gender, age, occupations, movie 的 category 與 year。其中 category 由於有可能有多種標籤，因此我先求出最多標籤為資料 6，並將所有資料都 pad zero 到長度 6。由於直接下去 embedding 的話中間有許多 sparse space，因此我先將他們 tokenize 才丟入模型，這部分我使用 dictionary 來實作。將 gender, age, occupation 三者 concatenate 後與 dense 後濃縮的 category 做 dot 運算。再將 age 和 year 做 dot 運算，然後兩者的 dot 值 concatenate 後經過一些 dense 再加入 user 與 movie 的 bias。最後的 Add 結果就是輸出。其他參數都和附註相同。



	Kaggle public accuracy	Kaggle private accuracy
New model	1.07703	1.0777


model RMSE
train
val
RMSE
epoch
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
0
50
100
150
200
250
300