

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

(collaborator: 無)

Normalize 的部分使用  $x' = \frac{x - \mu}{\sigma}$ ，其中  $\mu$  為 training data 的 rating 平均、 $\sigma$  為 training data 的 rating 標準差，predict 的時候再將 model 的輸出乘上  $\sigma$ 、加上  $\mu$  即為 rating 的預測值。

以下比較有無 **normalize** 的 validation loss (RMSE)：

(有 **normalize** 的 RMSE 在此數據中已經乘回  $\sigma$ )

Latent dimension 皆為 64 的情況下：

有 **Normalize**      0.8690    (15 個 epoch)

沒有 **Normalize**    0.9243    (31 個 epoch)

由此結果可以看出：

1. 有 **normalize** 會更快收斂
2. 有 **normalize** 的 RMSE loss 更小

故整體來說 **normalize** 是比較好的。

2. (1%)比較不同的 **latent dimension** 的結果。

(collaborator: 無)

以下結果皆有對 rating 進行 **normalize**，比較 8~256 的 latent dimension：

Latent dimension	loss	val_loss	epochs
16	0.8019	0.8732	32
32	0.7833	0.8734	22
64	0.7742	0.8659	14
128	0.7264	0.8672	11
256	0.6859	0.8681	8

可以注意到 latent dimension 越高 training loss 越低，但 validation loss 則沒有明顯下降，也就是 latent dimension 太高的情況下更容易 overfit。整體而言 latent dimension 大約選取 64 即可有不錯的效果。

此外也可以注意到 latent dimension 越高的情況下，只需要更少個 epoch 即可達到最低的 val\_loss，也就是 training 可能會稍微更快一些。

3. (1%)比較有無 bias 的結果。

(collaborator: 無)

加上 bias 的方法為使用輸出為一維的 Embedding layer。

以下比較有無 bias 的 validation loss (RMSE)：

取 latent dimension 為 64 進行本題實驗，rating 有先進行 normalize。

有 bias                      0.8616    (15 個 epoch)

沒有 bias                  0.8668    (14 個 epoch)

由上列兩個結果來看，加上 bias 後的 loss 會稍微小一些，但不是特別明顯。

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)

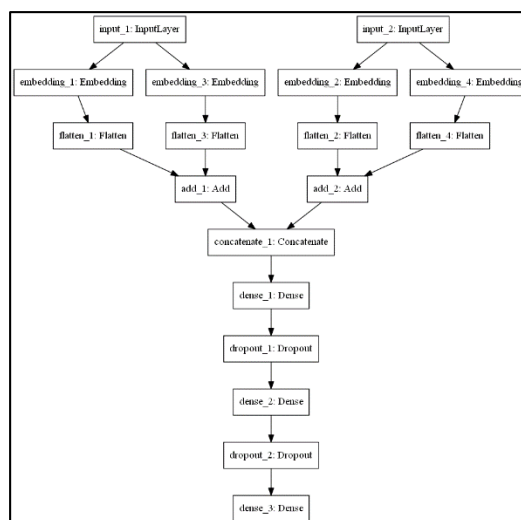
本次利用 DNN 的方法為：將 MF 中的 Dot layer 部分改為 Concatenate 後，再經過數層 Dense layer，最後輸出一個實數代表 rating。

架構如右圖，其中前兩層 Dense 皆為 512 個 units，最後一層為 1 個。Latent dimension 和 ML 相同皆為 64 兩種 model 的 loss 和 val\_loss 比較如下：

	loss	val_loss
MF	0.7742	0.8659 (14 epochs)
NN	0.8052	0.8663 (7 epochs)

經過 NN 作法後得到的結果和 MF 差不多，但

可以觀察出 NN 的 model 更快得到最低的 loss，這應該是由於 NN 參數較多。

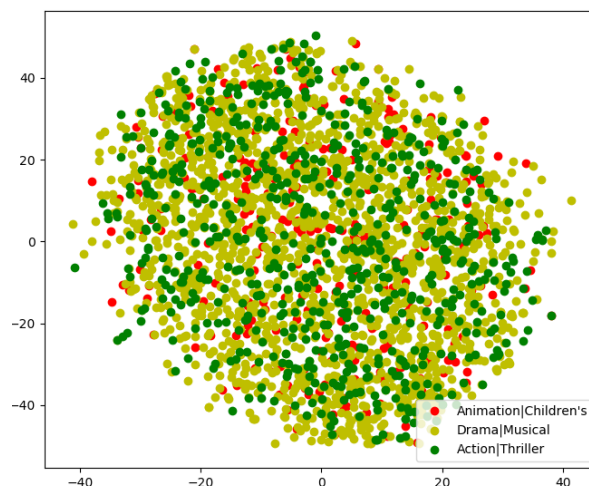


5. (1%)請試著將 movie 的 embedding 用 tsne 降維後，將 movie category 當作 label 來作圖。

(collaborator: 無)

取 latent dimension 為 64 的模型之 movie embedding layer 來看，t-sne 降維後如右圖

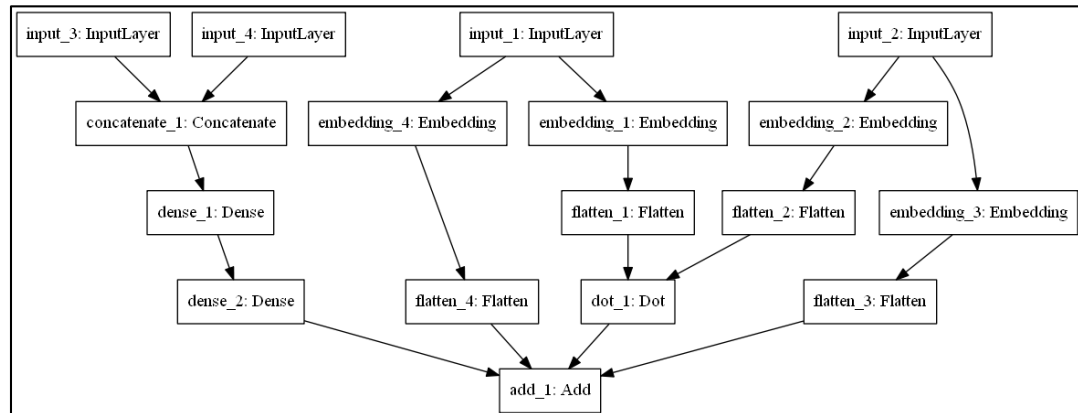
雖然已經將電影大致分為三類，但仍無法看出 embedding layer 有什麼特別的分類效果，這大概是由於 training 的時候使用 rating 當作 target，而電影類別和 rating 並沒有什麼直接關係，導致 embedding 的結果也和類別沒什麼關係。



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)

本次額外使用了 user 的性別、年齡資料。首先將年齡用 one-hot encoding 處理成 7 維的向量，再和性別合併成 8 維向量，輸入到兩層 FC 後，FC 的輸出當作第二個 user bias。最後一樣把 MF 中內積的結果和各 bias 相加當作輸出。

架構如下圖。



和不使用額外資料的 MF 相比較 Kaggle 的 RMSE：

	Public	Private
<b>MF</b>	0.86738	0.86639
<b>MF+額外 user 資料</b>	0.86648	0.86962

兩者比較起來差異並不是很大，故性別年齡資訊在本模型中沒有太大的效果。