

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

訓練方式: `loss='mse' optimizer='adam' batch_size=512 epochs=30`

`validation_split=0.2 EarlyStopping latent dimension=256`

需要注意的是 `loss` 是 `mse` 而 `Kaggle` 的評分是 `rmse`，所以最好的 `validation loss` 要開根號才會接近 `Kaggle` 上的分數。

`normalize` 的方式是把所有的 `rating` 扣掉平均再除以標準差。

無 `normalize`:

epoch	1	2	3	4	5	6	7
train loss	5.0368	0.8377	0.7214	0.5994	0.4715	0.3506	0.2536
val loss	0.9038	0.8232	0.7820	0.7692	0.7759	0.7998	0.8315

有 `normalize`:

epoch	1	2	3	4	5
train loss	0.8368	0.5298	0.3443	0.2028	0.1233
val loss	0.6568	0.6043	0.6136	0.6482	0.6839

平均: 3.5817 標準差: 1.1169

比較有無 `normalize` 的 `val loss` 時必須先把有 `normalize` 的結果乘以標準差才正確。

$0.6043 \times 1.1169 = 0.6749$ 比沒有 `normalize` 的情形好上許多，且也比較快收斂。

2. (1%)比較不同的 `latent dimension` 的結果。

這裡都使用有 `normalize` 的情況，比較不同 `latent dimension` 最低的 `val loss`。Best epoch 是指訓練花了幾個 epoch 才達到這個最好的 `loss`。

latent dimension	8	16	32	64	128	256
lowest val loss	0.6133	0.6099	0.6048	0.6034	0.5997	0.6001
Best epoch	12	7	5	4	3	2

可以觀察到其實在 `latent dimension=16` 以後最佳的 `val loss` 都幾乎已經不再下降，但是收斂速度越來越快。

3. (1%)比較有無 `bias` 的結果。

有 `normalize`, `latent dimension=256`

epoch	1	2	3	4	5
with bias val loss	0.6454	0.5981	0.6170	0.6549	0.6897
without bias val loss	0.6545	0.6001	0.6087	0.6443	0.6792

差異並不顯著，這裡的 `bias` 實做方式是把 `user id` 和 `movie id` 都通過一層 `embedding layer` 在接到 `merge layer` 加起來。

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。

分別將 user id 和 movie id 通過一層 embedding layer 後 flatten，利用 Concatenate Merge 把兩個向量接起來後經過一層 Dense(256, 'relu')，最後給 Dense(1, 'linear')得到最後的 output。

MF

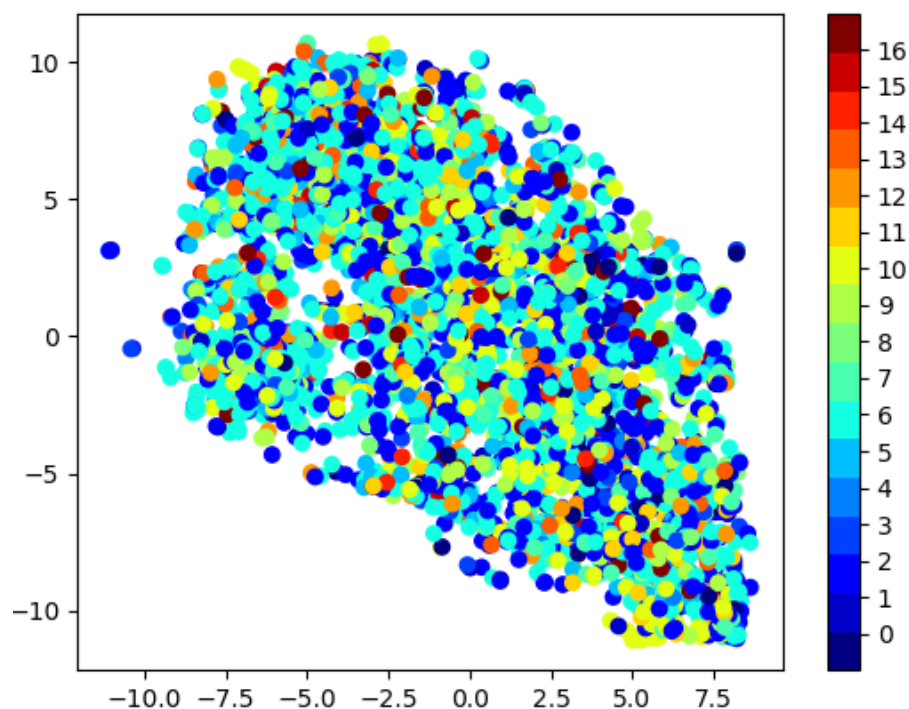
epoch	1	2	3	4	5
train loss	0.8368	0.5298	0.3443	0.2028	0.1233
val loss	0.6568	0.6043	0.6136	0.6482	0.6839

DNN

epoch	1	2	3	4	5	6	7	8
train loss	0.6885	0.6326	0.6099	0.5910	0.5672	0.5401	0.5116	0.4849
val loss	0.6518	0.6310	0.6221	0.6188	0.6129	0.6133	0.6156	0.6207

DNN 的結果比 MF 還要差，也收斂的比較慢。

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



共有 17 類的電影。

因為 tsne 適用於維度比較低的降維，所以我重新訓練了一個 50 維的 embedding，用 tsne 降維後做出的圖如上，數據點完全沒有分開。

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。