學號：B04901138 系級：電機三 姓名：張景程

✧ 一到五題當中所有 model 皆只有使用 user_ID 和 movie_ID 兩項 feature

✧ optimizer 皆為 Adam

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

(Collaborators：無)

latent dimension : 256

normalize 方式：

取出 training data rating 的 mean 和 std，以(training data rating – mean) / std 來進行訓練，而 model predict 出的結果則是先乘 std 後再加上 mean 來還原。

| | **epochs** | **public** | **private** |
|---|---|---|---|
| 有 normalized | 40 | 0.85412 | 0.85367 |
| 無 normalized | 109 | 0.84804 | 0.84856 |

雖然做 normalized 的結果沒有比較好，但是可以大幅減低訓練次數。

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

(Collaborators：無)

無 normalized

| **latent dim** | **epochs** | **public** | **private** |
|---|---|---|---|
| 4 | 385 | 0.88522 | 0.88219 |
| 8 | 330 | 0.87484 | 0.87266 |
| 16 | 290 | 0.86311 | 0.86202 |
| 32 | 275 | 0.85459 | 0.85497 |
| 64 | 211 | 0.85065 | 0.84956 |
| 128 | 134 | 0.85003 | 0.84949 |
| 256 | 109 | **0.84804** | **0.84856** |
| 512 | 47 | 0.84897 | 0.85406 |
| 1024 | 38 | 0.85202 | 0.85460 |

由上表可以觀察出 latent dimension=256 的時候結果最好，且當 dim 越大，training 時所需的 epochs 數量也隨之減少。

3. **(1%)比較有無 bias 的結果。**

(Collaborators：無)

latent dimension : 256

| | **public** | **private** |
|---|---|---|
| 有 bias | 0.84804 | 0.84856 |
| 無 bias | 0.85075 | 0.85145 |

有加了 bias 的成果比較好，這其實也很 make sense，舉例來說，每位 user 的喜好都不同、評分標準也不一樣，可能有些 user 習慣會把電影的評分給高一點，有些可能習慣會給低一點等等，同時每部電影背後的屬性也不相同，所以加了 bias 之後表現比較好是可以預期的。

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

(Collaborators：無)

latent dimension : 256、無 bias

DNN 架構：

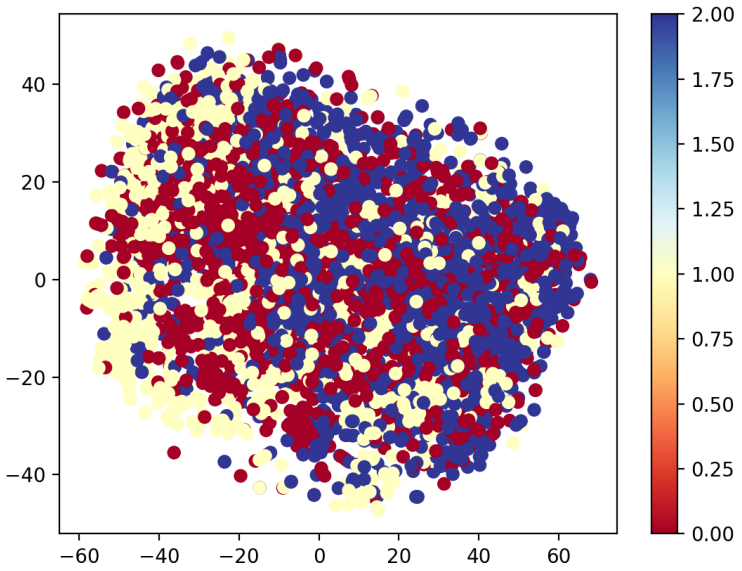
按照助教的方式，將 user_ID 和 movie_ID 分別做 dimension 為 256 的 embedding 後 Concat 起來，再通過三層 units = 128, 64, 32 的 Dense，最後 output layer 為 Dense(1)，得到預測的評分結果。

| | **public** | **private** |
|---|---|---|
| DNN | 0.88735 | 0.88975 |
| MF | 0.85075 | 0.85145 |

DNN 的結果比 MF 差 0.03~0.04 左右，且疊的層數越多越容易 overfit。

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

   (Collaborators：無)




   Legends:

   紅色：Adventure, Animation, Children's, Comedy, Fantasy

   米色：Action, Crime, Film-Noir, Horror, Thriller, War, Western

   藍色：Documentary, Drama, Musical, Mystery, Romance, Sci-Fi

   這是由 latent dim=32 的 model 的 movie_emb 得到的圖，由圖中可以觀察到藍色大多位於右邊，米色大多位於左側，而紅色雖然跟藍/米色都有混雜，但主要都分佈在中間偏左的區域，算是分得蠻開的。

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

   (Collaborators：無)

   - 額外使用的 feature : user 的性別、年齡、職業，以及 movie 的類別
   - 讀檔時將 user 職業和 movie 類別這兩項分別做成 dimension 為 21 和 18 的 one-hot 形式。其中，movie genres 每一個 entry 分別代表 ['Action' , 'Adventure' , 'Animation' , 'Children\'s' , 'Comedy' , 'Crime' ,'Documentary' , 'Drama' , 'Fantasy' , 'Film-Noir' , 'Horror' , 'Musical' ,'Mystery' , 'Romance' , 'Sci-Fi' , 'Thriller' , 'War' , 'Western']

- 將 user 性別、年齡如同 ID 也各自做一個 dim=256 的 Embedding，職業和電影類別則分別通過 units=latent dim 的 Dense
- 將 user_ID, movie_ID, movie_genres 分別對其他五項做內積，共有 3*5 – 3(重複) = 12 項。
- 將這 12 項內積的結果 Concat 起來，形成一個 12 維的向量。
- 最後通過一層 unit=1 的 Dense layer，再加上 user、movie 的 bias，即得到預測的評分。

- 實驗結果：

  kaggle public : 0.84686

  kaggle private : 0.84829

  使用額外的資料得到的結果有好一些些，但相差不大。