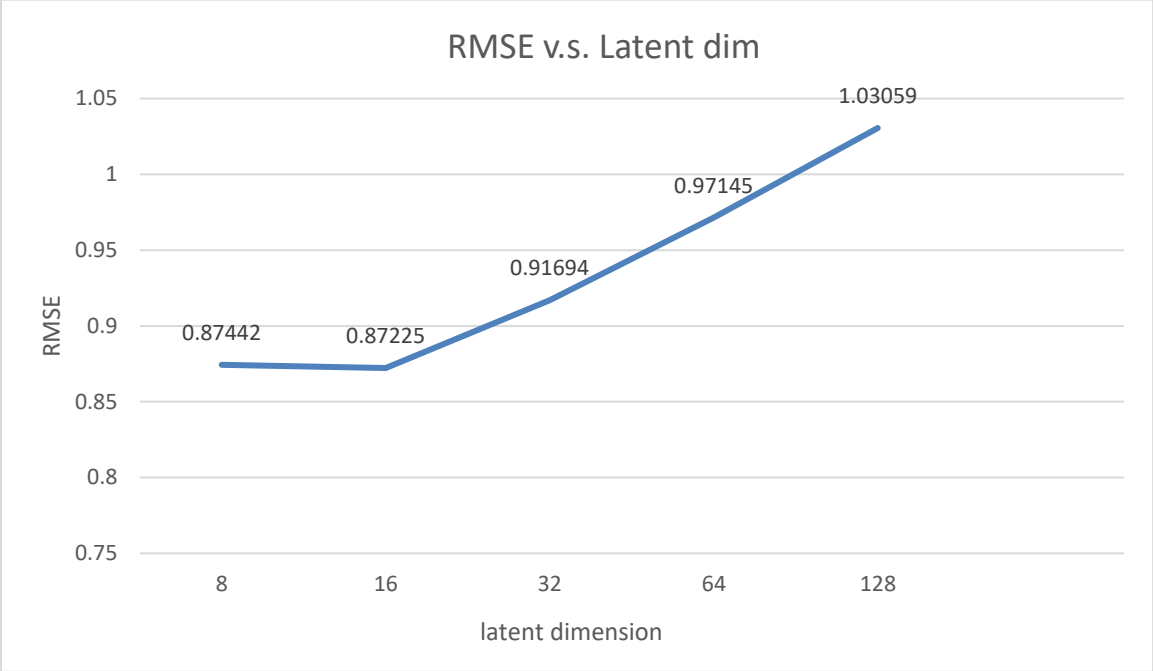
學號：B03901030 系級： 電機三 姓名：蕭晨豪

1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.
   我 normalize 的方式是將 rating 減去其平均值，再除以標準差。
   在 test 時必須將 model 的 output 乘上 training 時計算的標準差再加回平均值。
   在沒有 normalize 時 Kaggle 上 MF 最好的結果為 0.89061
   在 normalize 後最好的結果變為 0.87225(兩者均有加 bias)
   可看出有顯著進步

2. (1%)比較不同的 latent dimension 的結果。



   在有 normalize 的情況下，幾乎都在前十個 epoch 就會到達最低的 val_loss，在 latent dimension 很高的時候，幾乎在前一兩個 epoch 就會有最低的 val_loss 了，後面的 epoch 只會 overfit，但前一兩個 epoch 由於才剛開始 train 因此 loss 還是很高，造成必須取低一點的 latent dimension 才會有較好的結果，我最好的結果是取 latent dimension = 16

3. (1%)比較有無 bias 的結果。
   由於電影的 rating 可能受個別使用者的影響(例如 A 給大部分的電影較高分，B 卻均給較低分)，因此加上 user 和 movie bias 可以有效的消除此影響，較準確地得到針對個別使用者以及個別電影的 rating。
   在沒有加 bias 時 Kaggle 上 MF 最好的結果為 0.89697
   在加上 bias 後變為 0.89061(兩者均沒有 normalize)
   可看出有些許進步

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。

   我 MF 最好的結果為加上 bias 和 normalization，並且 latent dimension 為 16，其 RMSE 為 0.87225

   DNN 的 model 前面和 MF 一樣先將 user 和 movie 的 ID 通過 embedding layer 變成一個 64 維的 vector，之後 concatenate 兩個 embedding layer 後通過三次有 256 unit 的 dense，中間均使用 relu 做 activation 以及加上 0.5 的 dropout，最後用 dense(1)去 output 一個單一的數字做為 rating。

   此方法做出來的 DNN 參數量為 MF 的一半，但訓練過程反而較慢，並且收斂的時間較久，在 kaggle 上最好的結果為 0.87135，較 MF 稍好一些

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

   我取電影檔案中的第一個 label 作為該電影的 genre(原本是 multilabel)

   由於電影種類太多，我手動將其分成五類:

   (1)Animation, Children, Comedy

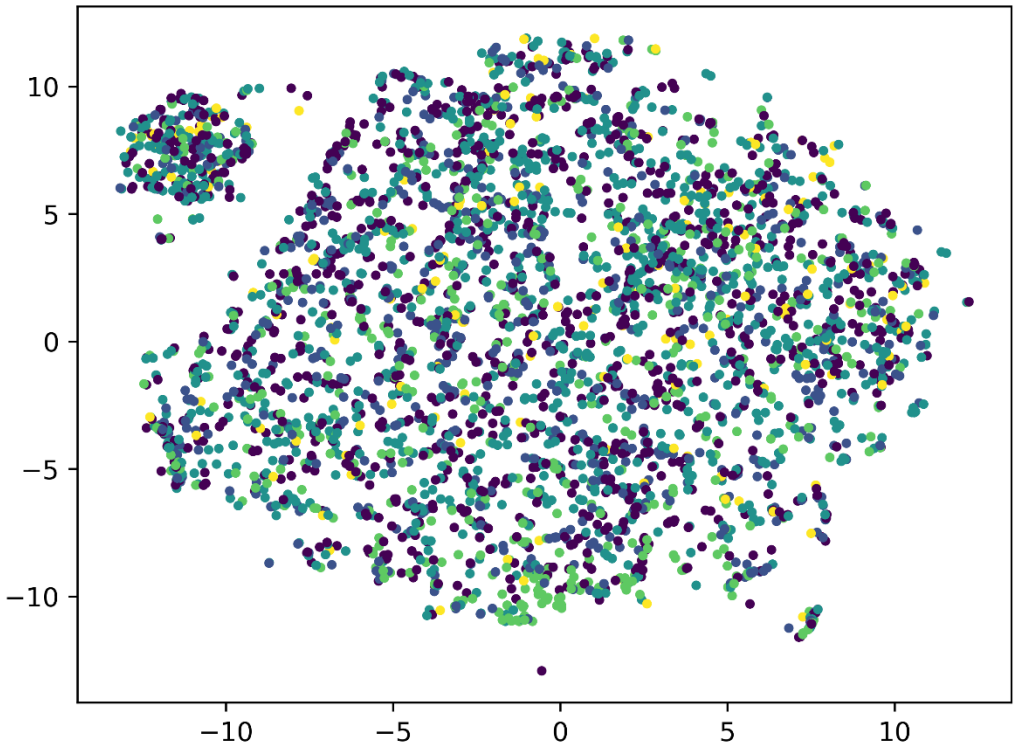
   (2)Fantasy, Adventure, Action, Sci-Fi

   (3)Drama, Musical, Romance

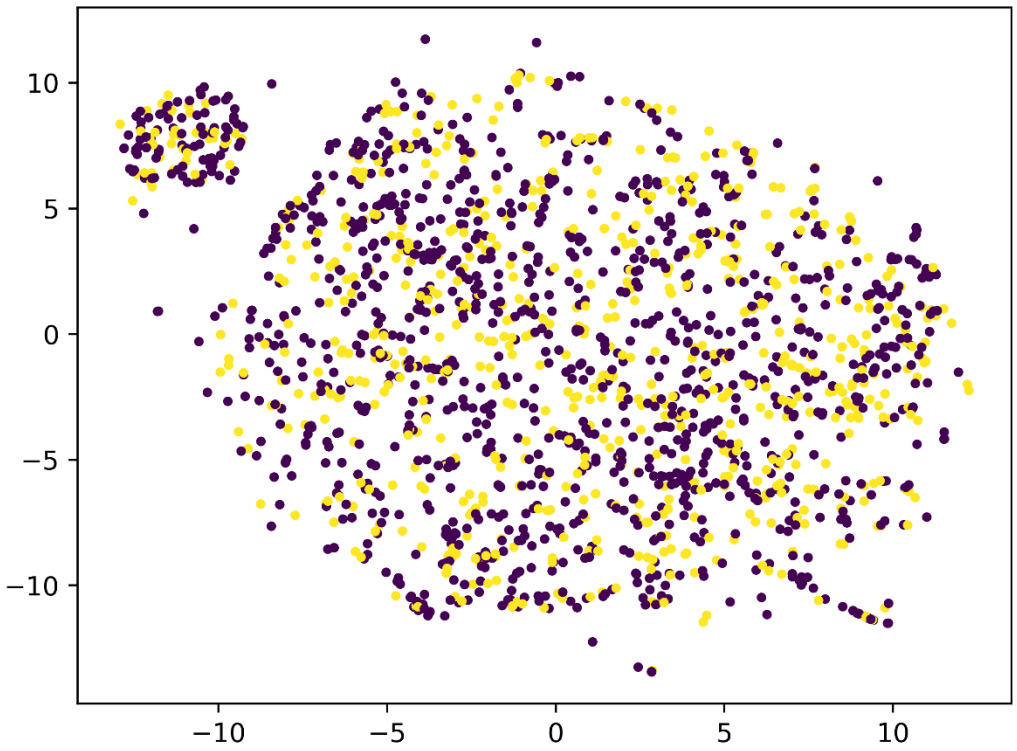
   (4)Crime, Thriller, Horror, Film-Noir, Mystery

   (5)Western, War, Documentary

   將 movie embedding layer 的 weight 用 tsne 降維後的結果表示如下:



可從上圖看出分布非常混亂，完全沒有規則可言，事實上我就算只拿只屬於(1)和(2)的電影來作圖依舊如此，結果如下:



即使只看兩群依然沒有任何分開的傾向。
可以看到不論是哪張圖，均有一小群的電影群聚在左上角，可以把他們當作在高維空間(embedding space)中比較相近的電影，可惜這些電影並不能用顏色也就是 genre 來區分。
我猜測會造成此結果除了 train 的沒有很好以外，也有可能是由於電影的 genre 是 multilabel 的，我只取第一個可能不是一個太好的做法，另外即使是被分到不同種類的電影之間也會有一些劇情或是類型的相似性，因此不太容易分開。

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。
   我將 user information 和 movie information 拿來使用:
   將 user 的 age 和 gender 還有 occupation 作為 feature(age 為數字，gender 為 0/1，occupation 為 one-hot encoding)
   以及 movie 的 genre 作為 feature(one-hot encoding)，將這兩個 information 與 DNN 中的 user 和 movie 的 embedding layer concatenate 起來，之後與第 4 題所述架構一樣，通過三層 dense(256)，加上 relu 和 dropout。
   有顯著進步，為我目前在 kaggle 上最好的結果，RMSE 為 0.8484