學號：R05921086 系級： 電機碩一 姓名：邱名彥

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

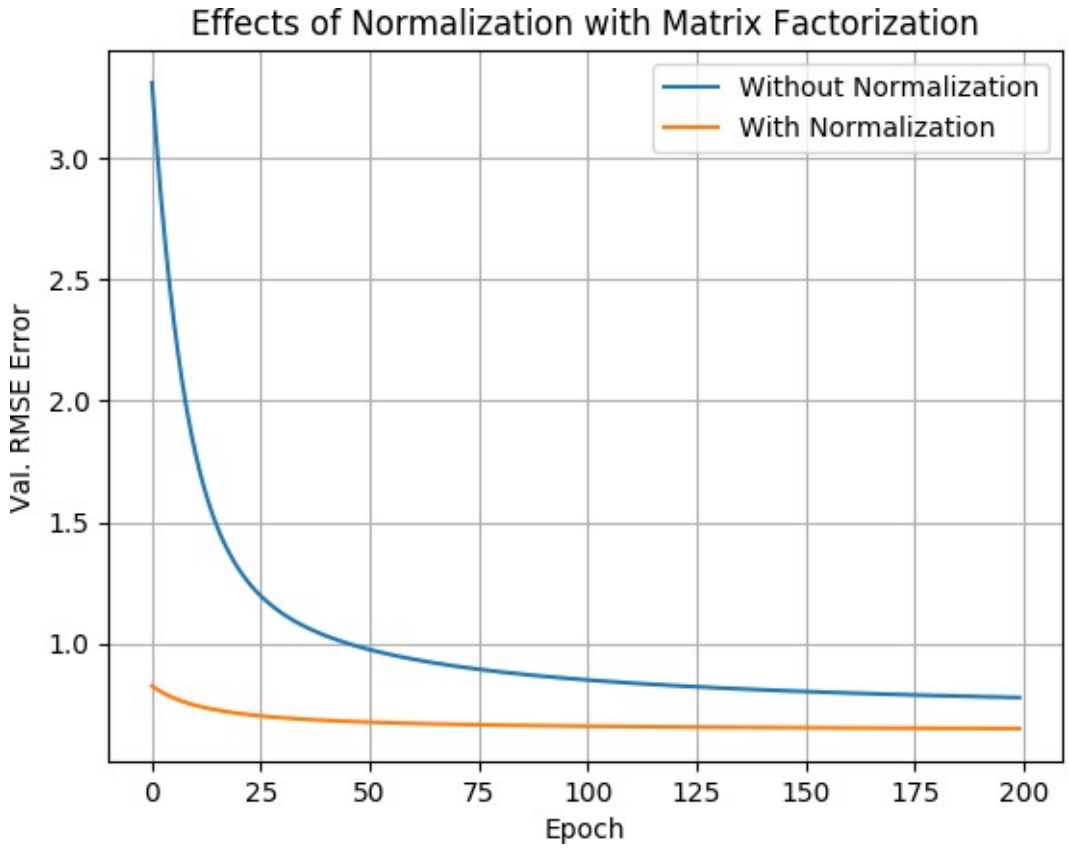
To normalize data, we take the Ratings data and center it such that the data has Zero Mean and a Unity Standard Deviation using the code shown below on the left. After we finish training our model, our predictions must be adjusted to account for its original mean and standard deviation. The code is shown on the bottom right.

```
yMean = np.mean(yData)
yStd = np.std(yData)
yData = (yData-yMean)/yStd
```

```
yTest = model.predict([testUsers, testMovies])
yTest = yTest * yStd + yMean
yTest = np.around(yTest)
```

Type	Private Kaggle Score
With Normalization	1.19940
Without Normalization	0.86614

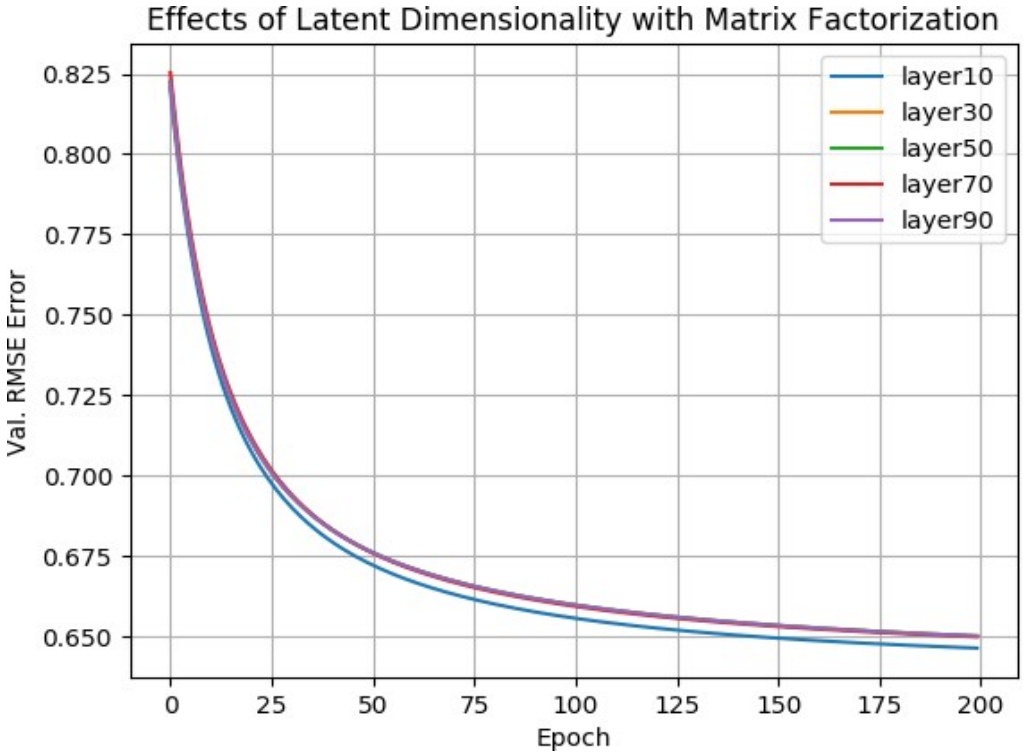
With normalization, our prediction is significantly off when compared to training without normalization. This could be due to how data is a rating on a scale which may have fluctuations since the data is on a subjective scale. Training validation shows that normalization should have positive effects on accuracy.



2. (1%)比較不同的 latent dimension 的結果。

We compare the effects of using different numbers of latent dimensions. The more latent dimensions used, the more graphic memory was required to run each model since greater latent dimensions mean more latent features to use. A greater number of latent features could lead to overfitting which can leads to bad accuracy results. We show the kaggle scores for 3 different layers in the table below and the validation scores for 5 different models with various layers.

Number of Layers	Private Kaggle Score
10	0.86614
30	0.87356
50	0.87701

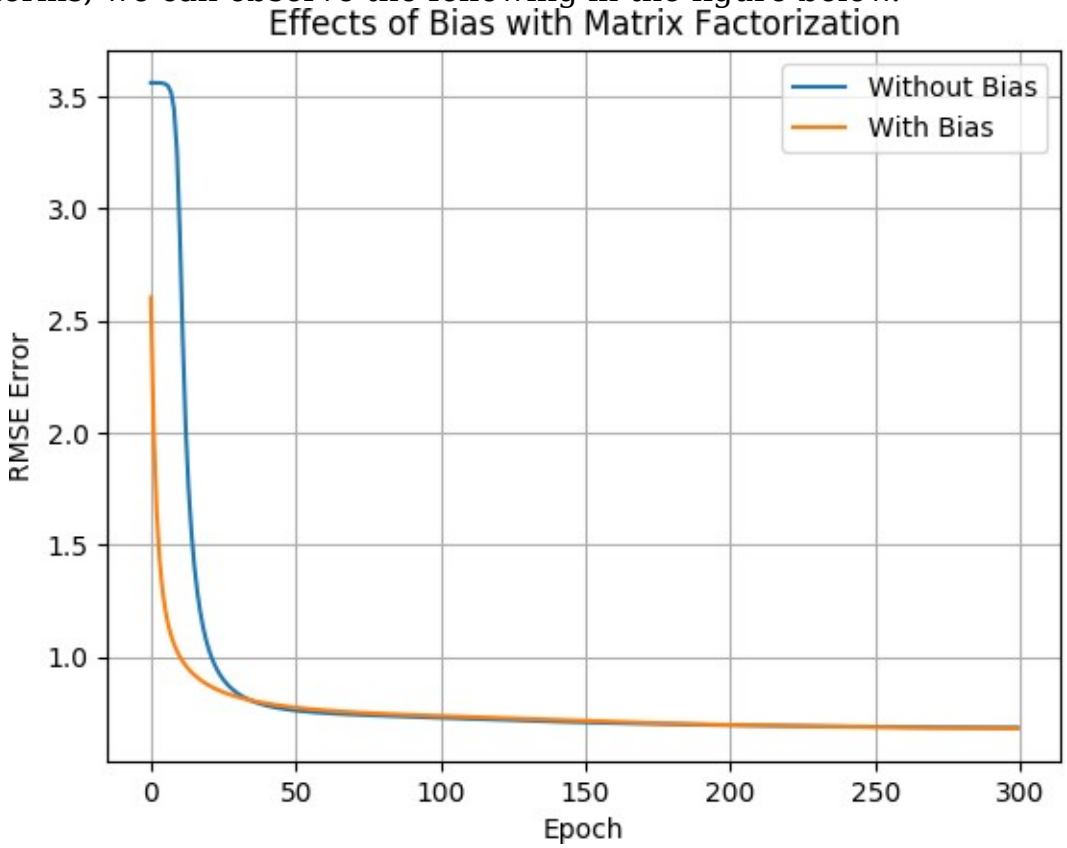


3. (1%)比較有無 bias 的結果。

If we compare the effects of the bias term during training, we can get the following in the table shown below.

Type	Private Kaggle Score
With Bias Term	0.87356
Without Bias Term	0.86614

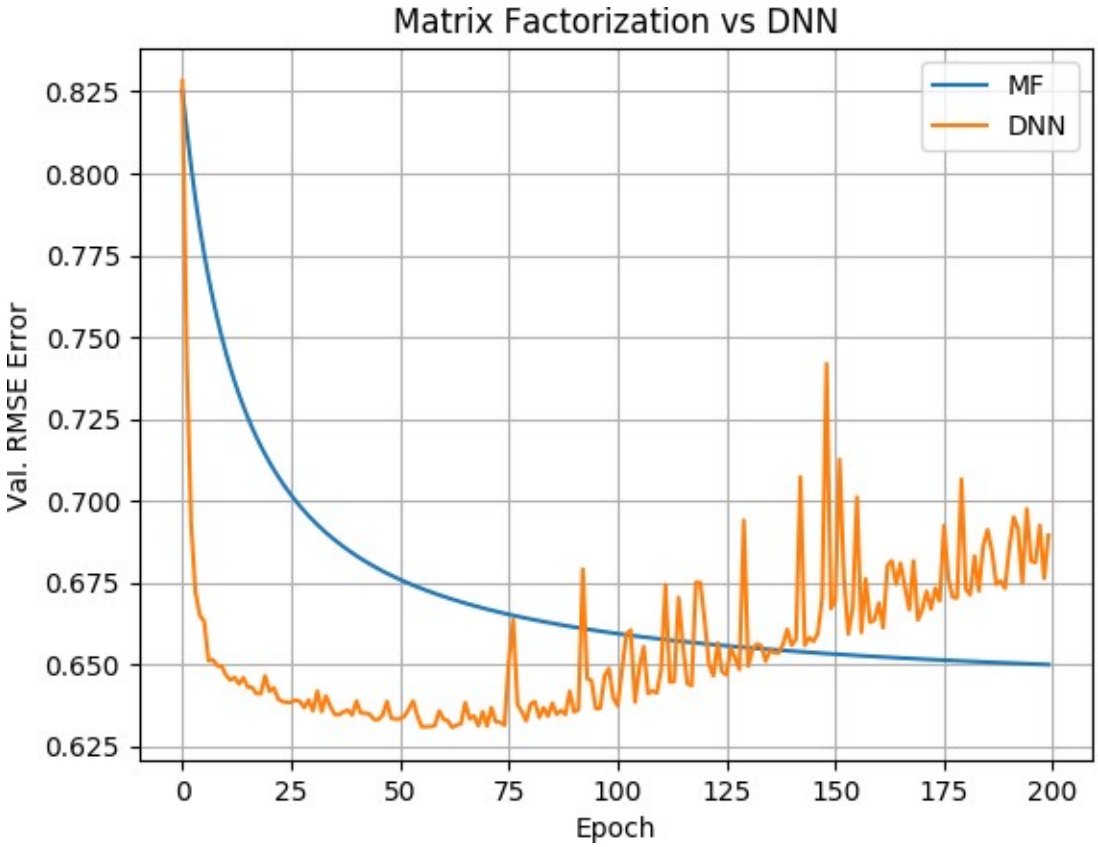
This shows that having a bias term has a negative impact on accuracy. However, if we plot compare the validation accuracy with and without bias terms, we can observe the following in the figure below.



Bias does help with initial training but all values converge to relatively similar results in the end. An interesting note as that the training result with bias has a slightly higher validation RMSE error when compared with training without bias. This may explain the slight variation in terms of Kaggle scores.

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。

In training our model, we use ModelCheckpoint to save the best model with the best validation score. We can see from the figure below that using a DNN network gives better results than training with Matrix Factorization. The DNN validation slowly increases in error most likely due to a learning rate that is too large. It may be possible for DNN results to increase with we used CheckLRonPlateau which adjusts the learning rate as needed when training.



The private Kaggle scores are shown below which compare DNN and MF outputs.

Model Type	Private Kaggle Score
DNN	0.86603
MF	0.86614

We show the architecture for our DNN and MF architecture in the following page.

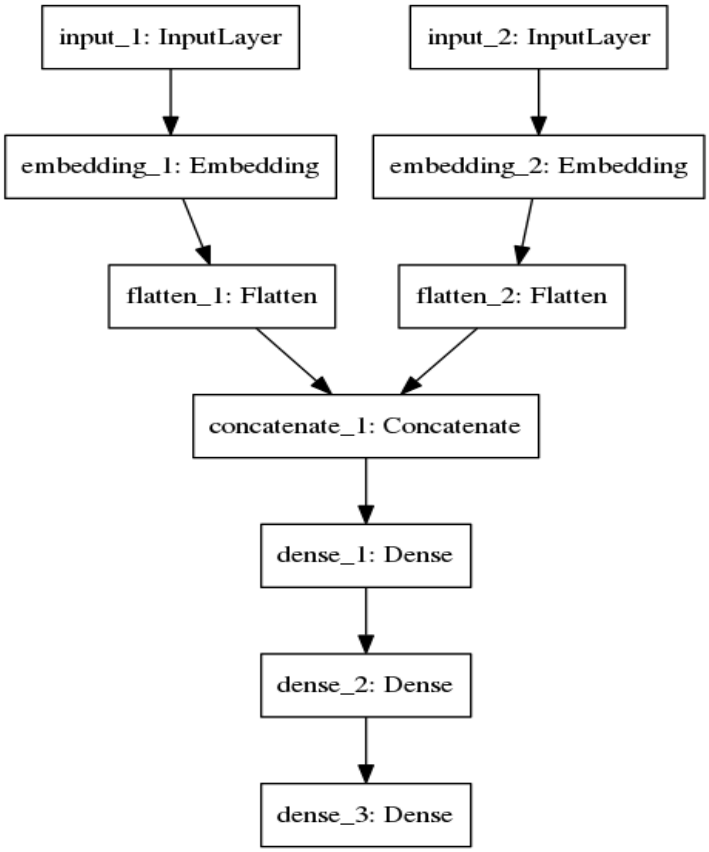


Illustration 1: DNN Architecture

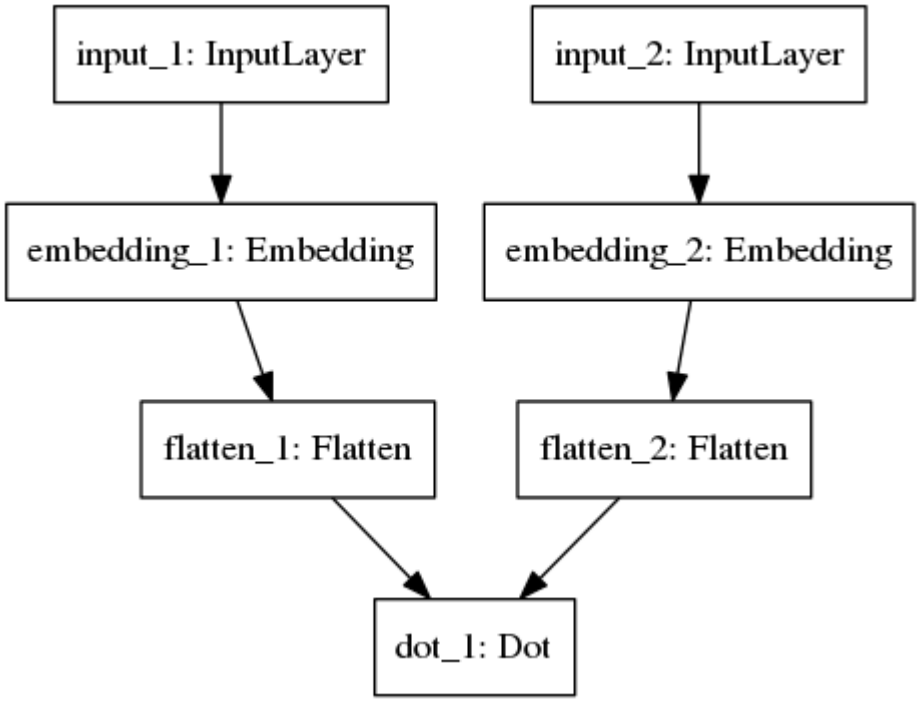


Illustration 2: MF Architecture

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

*Due to a lack of time, unable to produce photos.

To achieve this, it would look similar to the architectures shown in Q4 with the exception of an additional branch shown in the table below

Name	Layer Type	Description
Input 3: InputLayer	InputLayer	Accepts embedding of movie categories
Embedding_3	Embedding	Embeds/vectorizes movie categories
Flatten_3	Flatten	Flattens into 1D vector

The total unique types of movie categories include:

Mystery	Drama	Sci-Fi	Fantasy	Horror	Film-Noir
Crime	Romance	Children's	Musical	Animation	Adventure
Action	Comedy	Documentary	War	Thriller	Western

TSNE function would vectorize these and reduce the dimensionality of the movie categories into inputs that we can use for our model. Some factors that may influence training would be the distribution of movie categories and if every category was completely independent of each other.

Under ideal conditions, it should be possible to train using the additional features that can be generated using this feature to improve accuracy.

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。

We include the data from users.csv into our training data to provide our model with more features. User.csv contains data such as UserID, gender, age, Occupation, and zip code, which may allow extra relations to be mapped and remove that one particular's user's bias such as preferences for 'animation' or 'drama' films which may result in higher ratings compared to a non-biased user.

A private kaggle score submission to test the effects of including user.csv data is shown below.

Type	Private Kaggle Score
Without User.csv Data	0.86603
With User.csv Data	0.86792

From the results, we can see that including user.csv data shows a marginal boost of 0.001 in terms of RMSE data.