學號：B03901026 系級： 電機三 姓名：許凱傑

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

   kaggle 分數有 normalization 是: 0.85115，沒有 normalization 是: 0.87875

   我的作法是把所有 train data 的 rating 算出 mean 和 std 後做 normalization，因為因為 train data set 可能會有一個固定的 bias，把這個 bias 的因素去掉(i.e. normalization)，這樣可以使得 train 較快，可能也比較不會卡在 local minimum。

```
if args.normal:
    r_mean = np.mean(rating)
    r_std = np.std(rating)
    rating = (rating - r_mean) / r_std
    print('rating mean(%f) and rating std(%f)'%(r_mean,r_std))
```

2. *(1%)比較不同的 latent dimension 的結果。*

Latent Dimension	5	10	20	40	80
Kaggle 分數(RMSE)	0.86104	0.85115	0.85981	0.85863	0.86359

   都有 normalization。

   在 dimension 較小的情況下可能不足以把資料好好描述以及分開，但 dimension 太大又可能加入很多不需要的維度的資料(可看作 noise 的感覺)，所以反而結果變差。

3. *(1%)比較有無 bias 的結果。*

Bias	皆無	只有 movie bias	只有 user bias	皆有
Kaggle 分數(RMSE)	0.86801	0.85770	0.85938	0.85115

   Under dimension 10 with normalization

   因為每個人對於評分可能都有一個 bias(i.e. 普遍給高分，普遍給低分)。

   對於某個電影的種類，也可能會有較高/低的標準。

   從 KAGGLE 分數可以看出 movie bias 好像比 user bias 更有助於這次的分析，此此外兩個都有 bias 的進步幅度相對從沒有到有一個來的小。

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。*

   前面對於 movie/user id 的做法和 MF 一樣

```
if args.dnn:
    print('Use dnn model without extra feature')
    concat = Concatenate()([User_reshape, Movie_reshape])
    dnn = Dense(256, activation='relu')(concat)
    dnn = Dropout(DROPOUT_RATE)(dnn)
    dnn = BatchNormalization()(dnn)
    dnn = Dense(256, activation='relu')(dnn)
    dnn = Dropout(DROPOUT_RATE)(dnn)
    dnn = BatchNormalization()(dnn)
    dnn = Dense(256, activation='relu')(dnn)
    dnn = Dropout(DROPOUT_RATE)(dnn)
    dnn = BatchNormalization()(dnn)
    output = Dense(1, activation='relu')(dnn)
    model = Model(inputs=[User_input, Movie_input], outputs = output)
```

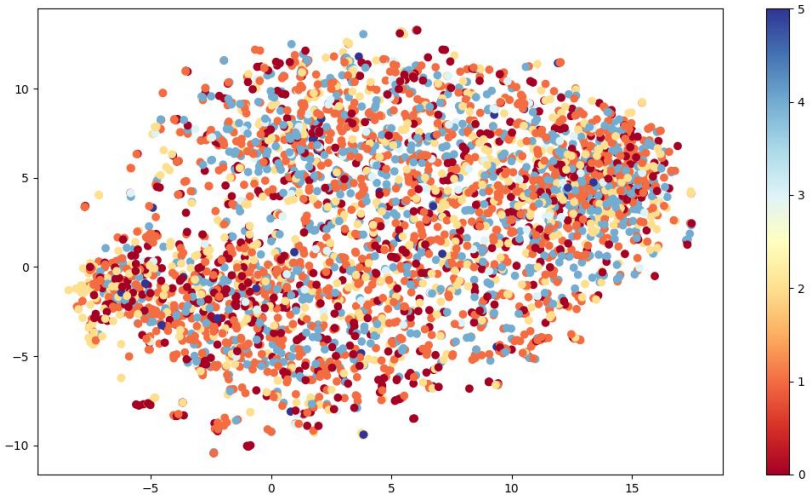
	DNN	MF(dimension = 10 with normalization)
Kaggle 分數(RMSE)	0.85876	0.85115

從 KAGGLE 分數看出，MF 的效果比較好，尤其如果把參數量調得差不多時，效果更明顯。

在我的測試中 DNN 跟我最好的 MF 分數其實差不多，可能再微調參數，就能往上衝。和同學討論他們加入其他 feature 後是可以到 0.84 的，不過我在 BONUS 中做的方法並沒有使分數提高太多。我覺得這次的問題比較單純，所以用 MF 也可以有很好的效果，而不需要用到 DNN。

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

```
cat1 = ['Action', 'Adventure', 'Western']
cat2 = ['Animation', "Children's", 'Comedy', 'Romance']
cat3 = ['Crime', 'Thriller', 'Film-Noir', 'Horror', 'Mystery']
cat4 = ['Documentary', 'War']
cat5 = ['Drama', 'Musical']
cat6 = ['Fantasy', 'Sci-Fi']
```



我覺得其實這個 label 沒有辦法把 data 分得很開，一來可能因為 movie 的分類本來就很可能一部包含很多種類，再來是從高維直接壓成 2 維，並不一定能很好的表示分布情形。

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。*

   我把 user 的 age 做 normalization，occupation 跟 gender 做 one-hot encoding，movie 的 genre 也做 one-hot encoding，再把它們跟 userid 和 movieid embedding 後的 vector concatenate 起來，後面再接三層的 DNN

   在 KAGGLE 上的分數是 0.85600，略遜於 MF，不過比單純 DNN 好一點。