1. **Data Set (1M MovieLens DataSet)**

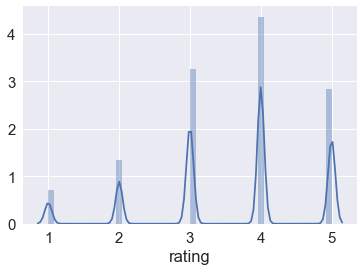
数据集地址： <https://grouplens.org/datasets/movielens/>

Rating record: 1,000,209

Rating score:1-5

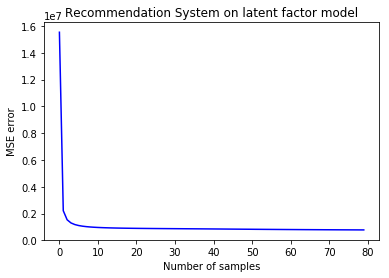
Total Number of users: 6040

Total Number of Movies: 3883

****

主要任务就是使用三种模型在MovieLens数据集上进行rating预测

1. **K-Latent-Factor-Model**

****

**RMSE: 0.8375718995424994**

**K = 3**

**Lambda = 5.0**

**Learning rate = 0.0001**

**Optimize Method: Stochastic Gradient Descent**

**Evaluation Method: RMSE**

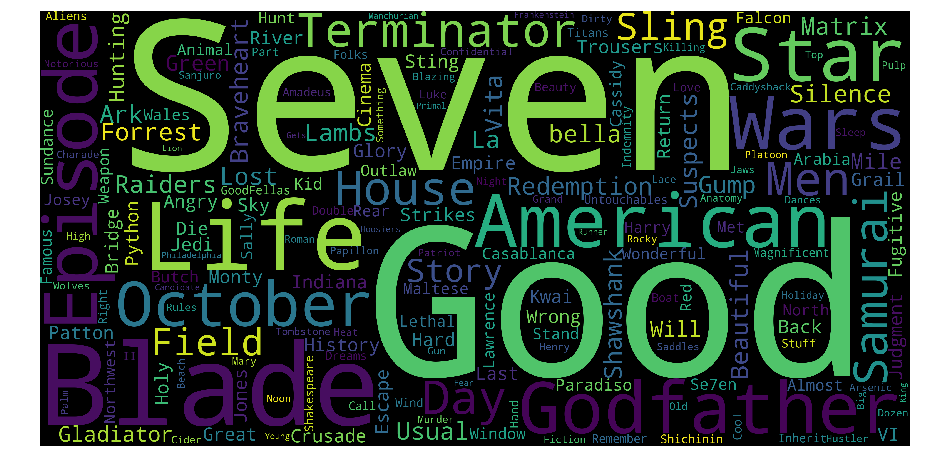
K-latent-factor model takes the relation between users and items into considerations. The equation is:

Where lam\_u and lam\_i are m X k and k X n matrix. k is the hyper parameters, m is the number of users and n is the number of items.

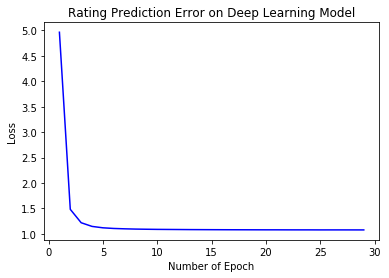
Our objective function now becomes:

Here, we use **Stochastic Gradient Descent** **(SGD)** to optimize this objective function. The derivative function for lam\_u[k] is as follows:

**使用 K-Latent-Factor-Model，对用户可能感兴趣电影关键字的挖掘。具体的做法是筛选出用户没看过的电影，然后对这些进行评分预测，再把电影按照预测分数进行排名。我们选取top100的电影名称，挖掘出了以下高频的关键词：**

****

1. **Deep Learning Method[1][2]**

****

**Number of Epoch: 30**

**RMSE: 1.0437**

**Evaluation Method: RMSE**

**具体方法论可以参考作者文章，模型实际跑出来没有K-latent-factor Model 好，RMSE也没有达到作者在网上说得那么低。**

**Reference:**

1. **Collaborative Filtering Method[1]**

**Evaluation Method: RMSE**

**基本的思想是使用similarity对rating就行加权求和，得出预测的rating数值**

**数据量很大的时候训练非常消耗内存, 以下为作者自己实验结果， RMSE非常大**

User-based CF RMSE: 1447.6814769930884

Item-based CF RMSE: 1678.7256599700413

**这一部分参考可见：[3]**

1. **Github**

源代码在这里：

<https://github.com/LinbinYoung/CSE258-UCSD/tree/master/code/CSE258-FinalProject>

1. **Reference**

[1] <https://medium.com/@james_aka_yale/the-4-recommendation-engines-that-can-predict-your-movie-tastes-bbec857b8223>

[2] <http://www.fenris.org/2016/03/07/index-html>

[3] <https://blog.dominodatalab.com/recommender-systems-collaborative-filtering/>