**Recommendation Engine based on Google Cloud Platform**

Zishuo Li Yuxin Fu Deming Zheng

GSAS Mechanical Engineering Mechanical Engineering

Columbia University Columbia University Columbia University

[zl2528@columbia.edu](mailto:zl2528@columbia.edu) [yf2440@columbia.edu](mailto:yf2440@columbia.edu) [deming.zheng@columbia.edu](mailto:deming.zheng@columbia.edu)

*Abstract*— **This report presents a recommendation engine based on Google Cloud platform. Specially, the system is designed for movie recommendation. First, the motivation and current problem are described in detail, followed by a description of algorithms implementation. Finally, the testing results are demonstrated and analyzed. Consequently, the results of recommendation engine on Google Cloud platform meet the expectation. It can be concluded that recommendation engine on Google Cloud platform is very successful and promising.**

***Keywords-big-data; google cloud platform; recommendation engine; spark; tensorflow; svd; machine learning; deep learning***

# Introduction

Recommendation engines are the technology behind content discovery networks and the suggestion features of most ecommerce websites. They improve a visitor's experience by offering relevant items at the right time and on the right page. There are various components to a recommendation engine, ranging from data ingestion and analytics to machine learning algorithms. In order to provide relevant recommendations, the system must be scalable and able to handle the demands that come with processing Big Data and must provide an easy way to improve the algorithms.

Cloud is a significant technological development and is being widely adopted. Its adoption makes a lot of sense since it simplifies things and also makes them more secure at reasonable costs. Google cloud is one of the major options available and among the most well-known. Google’s cloud platform provides reliable and highly scalable infrastructure for developers to build, test and deploy apps. It covers application, storage and [computing services](https://www.netsolutions.com/cloud-application-development)for backend, mobile and web solutions. More than four million apps trust and use the platform [1].

Google Cloud platform with these essential features are well-suited to support workload of recommendation engines mentioned above. Therefore, this project presents a way to build recommendation engine using machine learning on Google Cloud platform combined four different algorithms.

More, almost everyone loves to spend their leisure time to watch movies with their family and friends. Apparently, a movie recommendation agent has already become an essential part of daily life. Therefore, this project especially focuses on movie recommendation system on Google Cloud platform.

# Related Works

As for previous work, a solution for delivering relevant product recommendations to users in an online store has been reviewed. It mentioned about how to set up an environment that supports a basic recommendation engine which can be improved, based on the needs of particular workload.

To achieve a good compromise between speed, simplicity, cost control, and accuracy, this solution uses Google App Engine, Google Cloud SQL, and Apache Spark running on Google Compute Engine using bdutil [2]. This solution utilized Spark, which offers much better performance than a typical Hadoop setup and with Spark MLib, user can analyze several hundreds of millions of ratings in minutes. However, Spark MLib only implements Alternating Least Squares (ALS) algorithm to train the models, which based on our previous homework results is not enough to predict very precise results. Therefore, this project presents a potential solution to make the recommendation results more precise by combing more four different algorithms, including Spark ALS, SVD, cosine similarity, and deep learning. Moreover, this project will run a Jupyter notebook on Google Cloud Dataproc cluster instead of using bdutil to simplify the coding process.

# System Overview

To provide recommendations, several things need to happen. At first, while knowing little about users' tastes and preferences, system might base recommendations on item attributes alone. But the system needs to be able to learn from users by collecting, analyzing, and storing data about their tastes and preferences using Cloud SQL, which is a great database option to load and get data quickly and easily on Google Cloud platform. This is the first part of the system. As for second part, over time and with enough data, the system can use machine learning algorithms, which is the combination of Spark ALS, SVD, cosine similarity, and deep learning for this project to perform useful analysis and deliver meaningful recommendations.

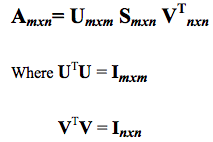
This project utilizes the a popular dataset, downloaded from MovieLens dataset by GroupLens Research group. The dataset contains 100.000 ratings (1-5 scales) from 943 users on 1682 movies. Each user has rated at least 20 movies. Also, this project gets the movie posters from The Movie Database website using its API with IMDB id.

# Algorithm

Besides using the Spark ALS algorithm for recommendation system, this project also utilizes SVD, cosine similarity, and deep learning.

Recommendation system based on the user-item matrix factorization have become more essential due to powerful algorithms such as ALS. But when the number of users and/or items is not so huge, the computation can be done using directly a SVD (Singular Value Decomposition) algorithm [3]. Singular value decomposition can be seen as a method for data reduction, by taking a high dimensional, highly variable set of data points and reducing it to a lower dimensional space that exposes the substructure of the original data more clearly and orders it from most variation to the least. Given that, besides Spark ALS, this projects also implemented SVD algorithm based on Tensorflow for recommendation engine while the number of users and/or items is small.

The SVD theorem states:



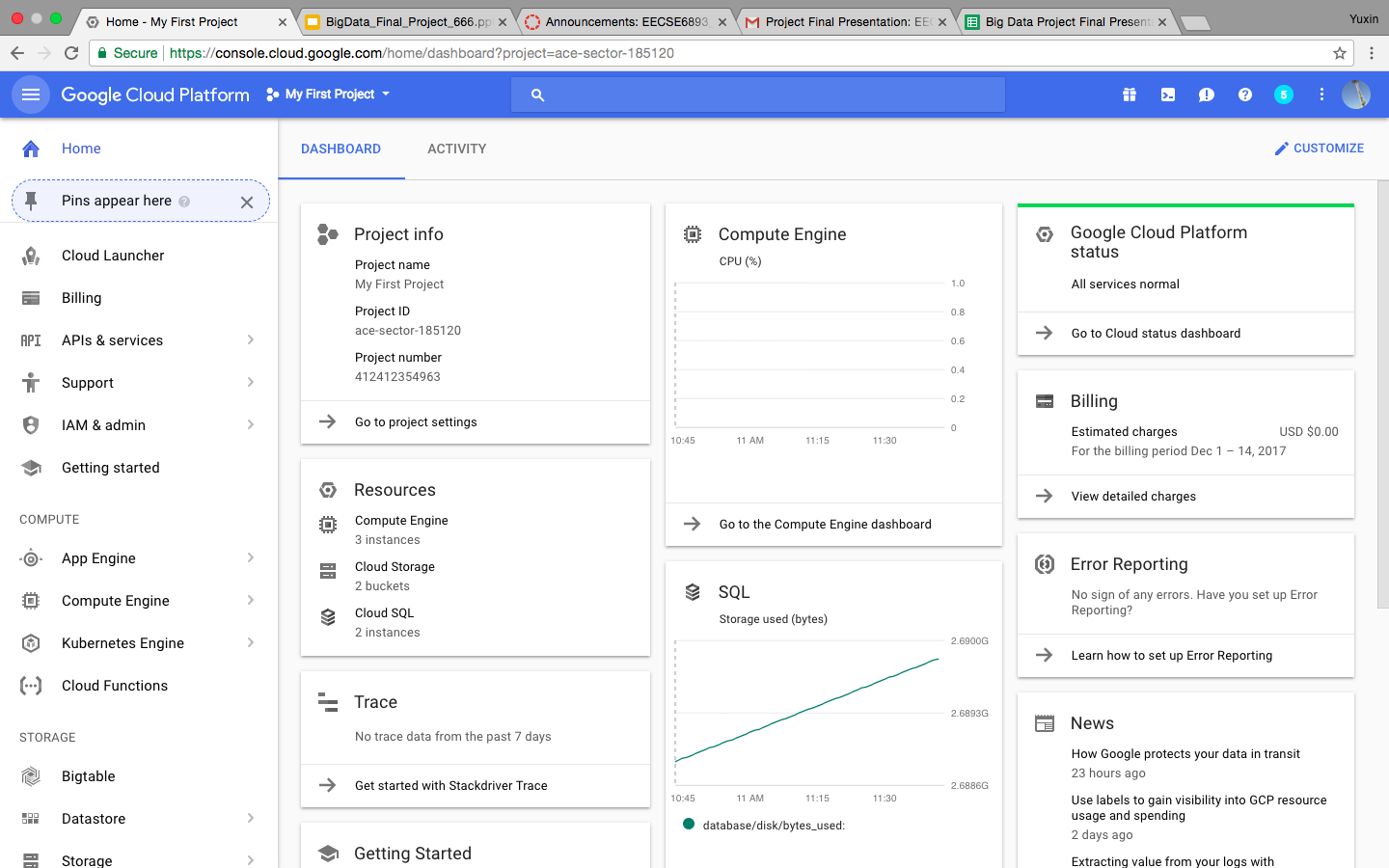
Where the columns of U are the left singular vectors; S has singular values and is diagonal; and VT has rows that are the right singular vectors.

As for cosine similarity algorithms, one of them is item-item collaborative filtering. Item-item models resolve these problems in systems that have more users than items. Item-item models use rating distributions *per item*, not *per user*. With more users than items, each item tends to have more ratings than each user, so an item's average rating usually doesn't change quickly. This leads to more stable rating distributions in the model, so the model doesn't have to be rebuilt as often. When users consume and then rate an item, that item's similar items are picked from the existing system model and added to the user's recommendations [4]. Another one is user-user collaborative filtering. This system uses the given profile of given user and provide recommendation system completely based on that’s preference and liking [5].

In order to further improve precision, the project tries to use deep learning to recommend movies to users. VGG16 in Keras is used to train this neural network. There is no target in the data set, therefore only the fourth-to-last layer as a feature vector has been considered. The project uses this feature vector to characterize each movie in data set. In the codes, similar as before first step is to get the movie posters from TMDB website using its API with IMDB id, then these posters are fed to VGG16 and are used to train the neural networks, finally, cosine similarity can be obtained using the features learned by VGG16. After getting the movie similarity, movies with the highest similarity will be recommended to similar users.

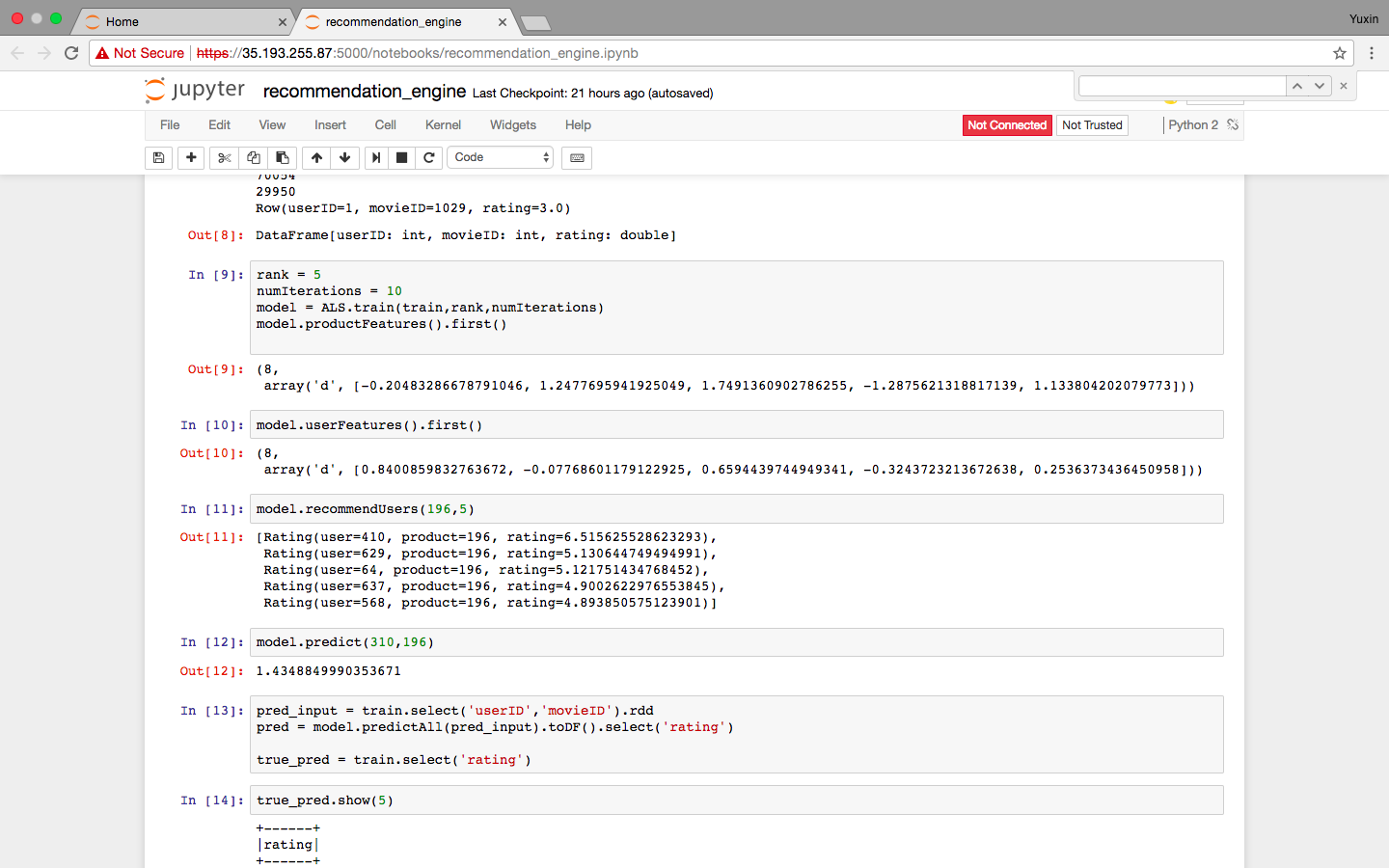
# Software Package Description

In this project, we mainly utilize the Big data services on Google Cloud platform. It enables us to process and query big data in cloud to get fast answers to complicated questions.

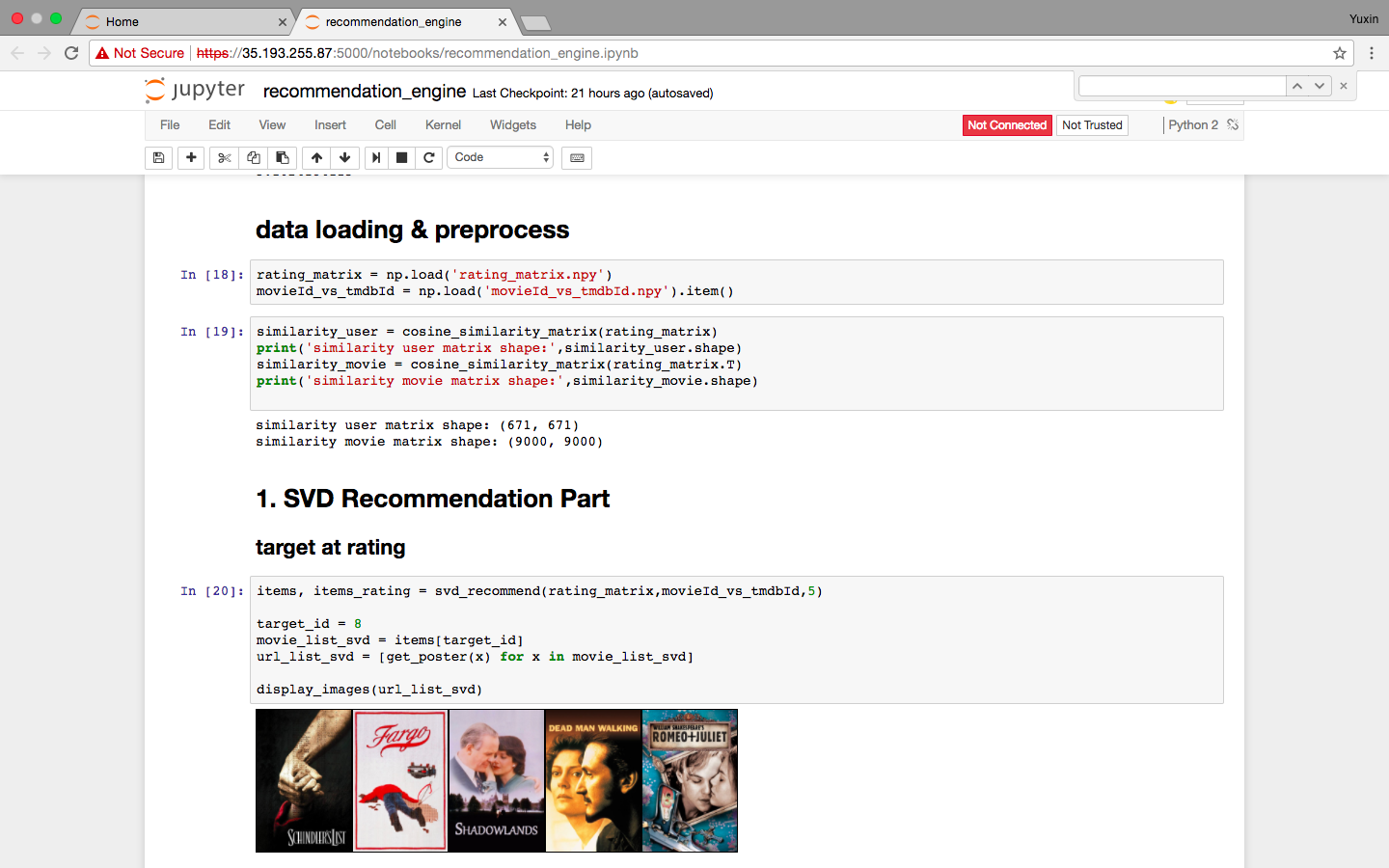


Picture above shows the final setup of Google Cloud platform, which can be considered as a simple web UI for users. Different modules are organizedly demonstrated. Users can choose any module they want by a simple click. Cloud SQL database module is used for data storage. Computer Engine module is used for delivering high-performance scalable virtual machines. Moreover, Jupyter notebook is used in the Cloud Dataproc module to input the commands and output the results to simplify the operation and visualize the data set. Cloud Dataproc is a fast, easy-to-use, fully-managed cloud service for running Apache Spark and Apache Hadoop clusters in a simpler and more cost-efficient way. After installing and configuring Jupyter notebook and PySpark kernel, users are able to freely run this project and check results.

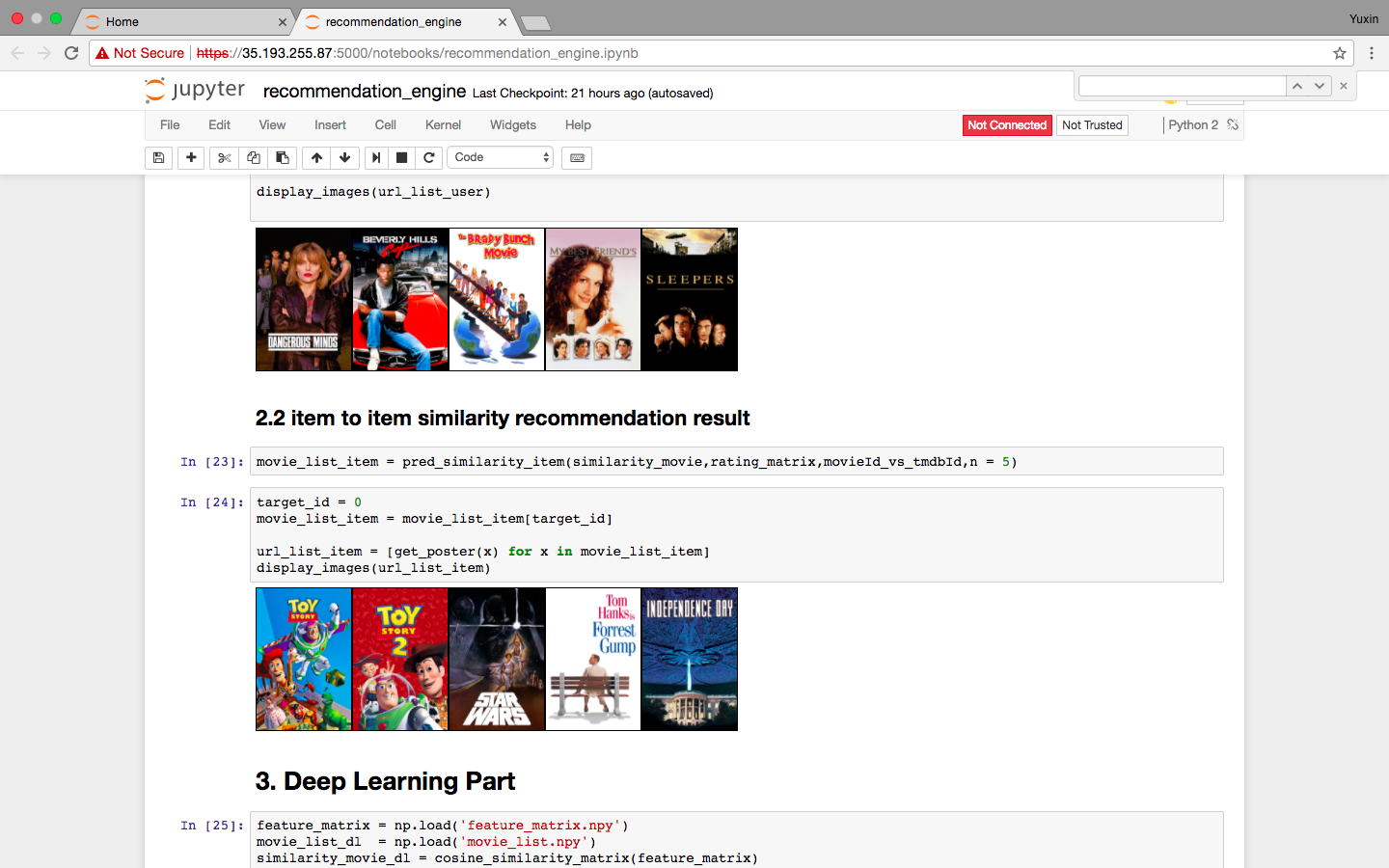
# Experiment Results

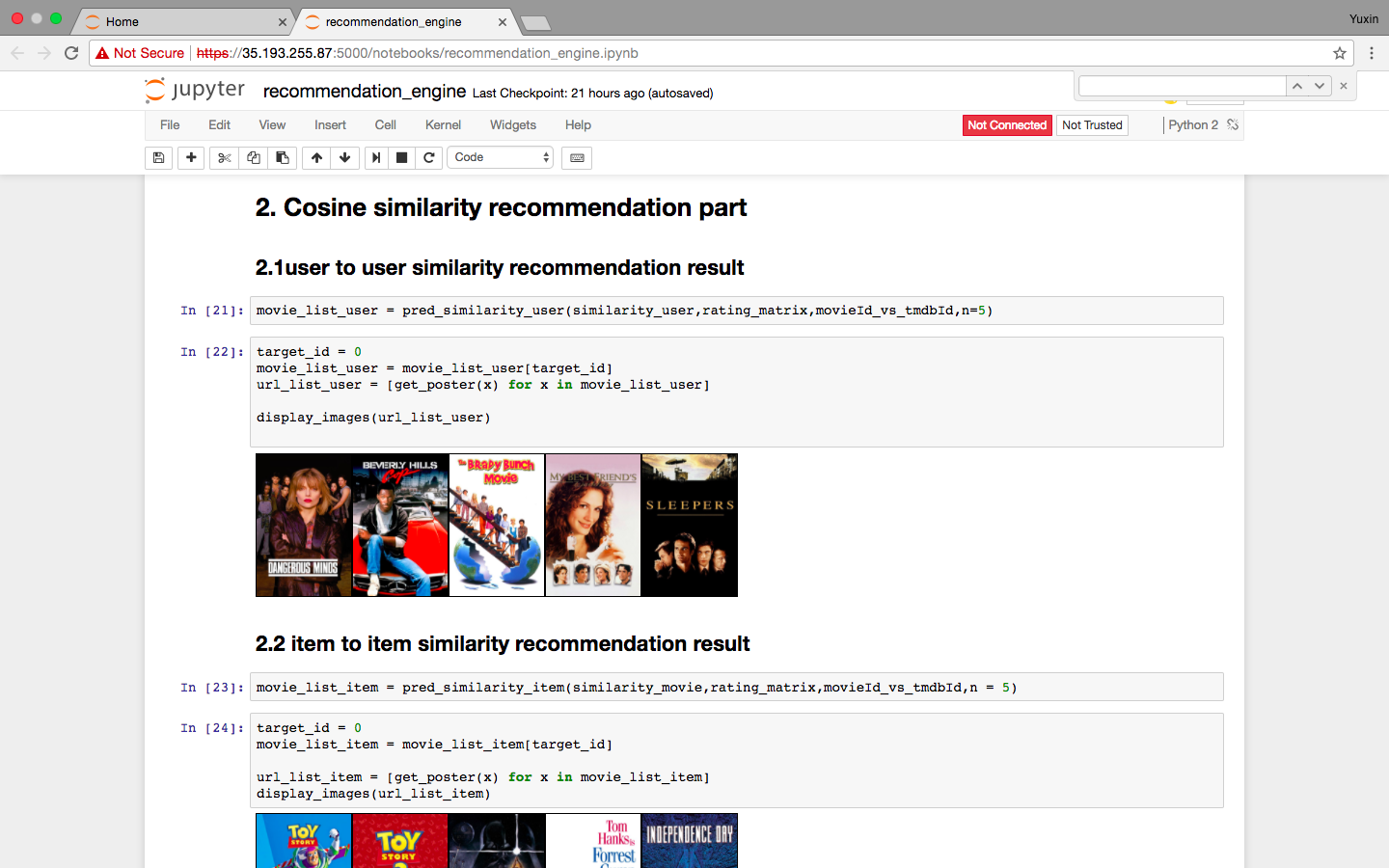


This is the result for Spark ALS algorithm. Compared to previous homework, accuracy is quite close. But it is not good enough. Movies recommended are not very similar based on users’ preference. Therefore, methods have to be improved and more algorithms are needed to get better predictions.

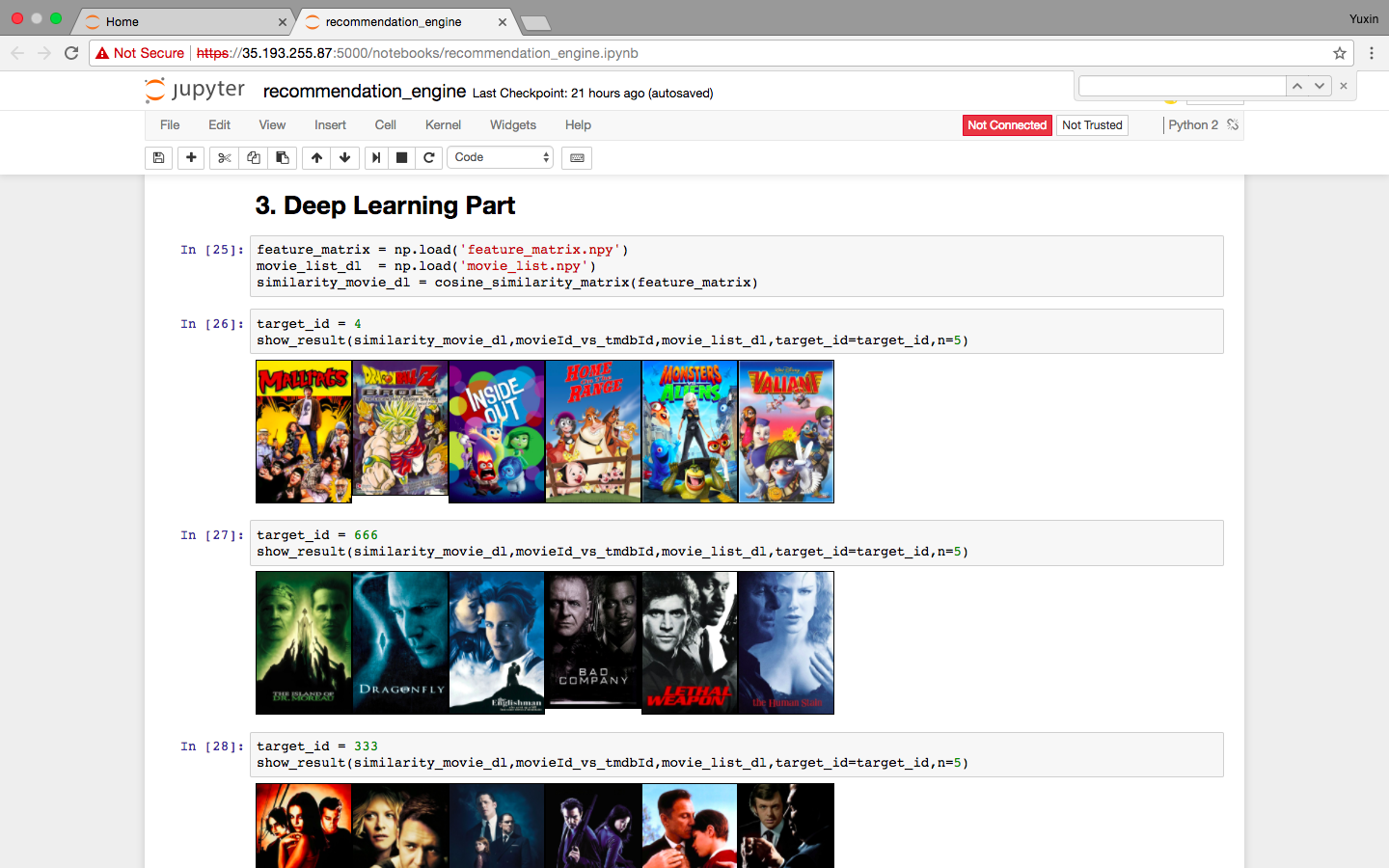


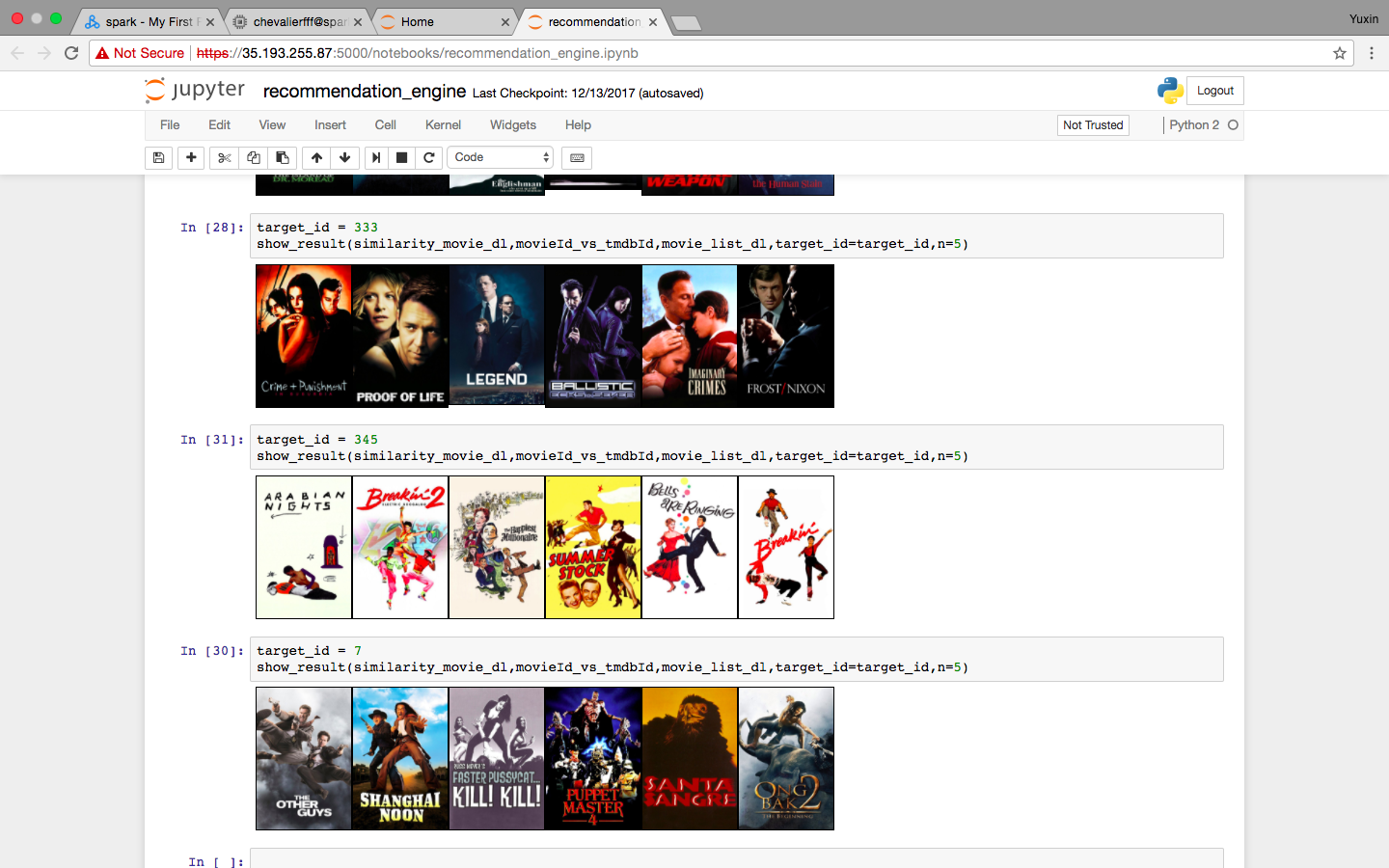
This is the result for SVD. Based on the pictures, it is obvious that they are all about romance. Resulting movies have similar genre.





These are the prediction results for item-item and user-user similarity. Based on the results, the algorithm seems to work very well. Resulting movies for item-item similarity are obviously belong to romance theme, whereas the prediction results for user-user similarity are not as good as those of item-item similarity, but still good enough. We can easily tell the genre. The reason for this is because the number of items is greater than the number of users in the dataset, item-item similarity will have better results.





This is the result of Deep learning. We have done five groups of testing. From the screenshots above, it can be seen that this algorithm has highest accuracy. The genre of movies belong to each of these five groups are quite similar.

# Conclusion

Movie posters have elements which create the hype and interest in the viewers. This project uses Spark ALS, SVD, cosine similarity, and deep learning as unsupervised learning approaches and learns the similarity of movies by processing different movie posters. Precise results of this project have proved its validity and functionality. As for contributions of each team member, Zishuo is mainly in charge of Google Cloud platform environment setup, and Yuxin and Deming are mainly in charge of presentation preparation and final report writing. We three together worked on algorithms implementation and test. Apparently, this is just the first step of using machine learning in recommendation systems. There are many new things to try. Another possible direction for the future work is songs recommendation. Instead of image processing, machine learning can be used to predict latent features derived from collaborative filtering by processing sound of a song. In a conclusion, this is a promising, successful and interesting project.

##### Acknowledgment

We would like to thank Professor Chingyung Li for his informative and interesting lecture, as well as Gongqian Li for his expert instruction and advice throughout all homework and this project. Finally, we would like to express our special thanks of gratitude to our TA Mohneesh Patel for his helpful and valuable feedback, which guides us through the fog to the light.

##### Appendix

See Github documentation

##### References

1. "Cite a Website - Cite This For Me", *Netsolutions.com*, 2017. [Online]. Available: https://www.netsolutions.com/insights/what-is-google-cloud-its-advantages-and-why-you-should-adopt-it/. [Accessed: 14- Dec- 2017].
2. "Using Machine Learning on Compute Engine to Make Product Recommendations  |  Solutions  |  Google Cloud Platform", *Google Cloud Platform*, 2017. [Online]. Available: https://cloud.google.com/solutions/recommendations-using-machine-learning-on-compute-engine. [Accessed: 14- Dec- 2017].
3. G. Bonaccorso, "SVD Recommendations using Tensorflow - Giuseppe Bonaccorso", *Giuseppe Bonaccorso*, 2017. [Online]. Available: https://www.bonaccorso.eu/2017/08/02/svd-recommendations-using-tensorflow/. [Accessed: 14- Dec- 2017].
4. "Item-item collaborative filtering", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Item-item\_collaborative\_filtering. [Accessed: 14- Dec- 2017].
5. "USER-USER Collaborative filtering Recommender System in Python", *Medium*, 2017. [Online]. Available: https://medium.com/@tomar.ankur287/user-user-collaborative-filtering-recommender-system-51f568489727. [Accessed: 14- Dec- 2017]