DECEMBER 5, 2016

MOVIE RECOMMENDER ON TWITTER MOVIE DATASET

MSBA 6420: PREDICTIVE ANALYTICS

TEAM - OVERFITTERS

SUBMITTED BY:

AAYUSH AGRAWAL

WENDY LU

WENRUI LIU

WILLIAM EERDMANS

Contents

1. Introduction 2
2. Business Understanding 2
3. Data Understanding 4
4. Data Preparation 5
5. Modeling 6
6. Evaluation 10
7. Deployment 13
8. References 16

1. Introduction

In the last twenty years, major strides have occurred in personalized analytics. Examples are ubiquitous today such as targeted ads, suggested friends on a social network, and a request from a friend to “Netflix and Chill”. As mentioned, one of the most prominent and prevalent in pop culture individualized-applied, analytics applications are movie recommendation systems. These systems use a variety of algorithms to achieve a list or ranking of recommended movies for a particular user. We are a startup focused on utilizing movie ratings data created by Twitter users to recommend streaming media and movie trailers via their Twitter feeds and other media platforms. Not only will ratings data, users, and movies be used, but other movie attributes such as genre, runtime, and plot summaries are incorporated into the system. The topic of this paper is to discuss the use case of this recommender system, what understanding of the data is required, how the data is prepared and used in the modeling phase, how the recommender is evaluated, and the end deployment and lessons learned from the entire CRISP process.

2. Business Understanding

Movie streaming services generate revenues based on period subscribers. Thus, the value of the business is in the ability to drive increased media streaming via the service or retain subscribers. In order to achieve these two ends, streaming services such as Netflix use a highly-tuned recommender system to provide a personalized feed of content to subscribers. In our business case, we have found a creative wrinkle inspired

by Netflix. Our planned revenue is based on targeted advertisements of media content such as movies, trailers, and documentaries on Twitter and other media platforms. Due to Twitter's falling revenue, we found this opportunity a low buyin into Twitter's new "First View" video advertisements [1]. This is an advertisement shown to a user upon logging in and the advertisement is only shown once in the 24 hour period. Some have voiced concern over this advertisement being a "homepage takeover", however, we believe our advertisement is an elegant compromise to an intruding ad [1]. Our advertisements would be individualized to the user, but the most important aspect is the time being advertised is media. Supported by research cited in the Harvard Business Review, movie trailers and other media receive significantly more attention based on the situation [2]. We believe trailers are able to "Entertain people to grab their attention"[2]. To automate the process of suggesting movies, we required an algorithm that addressed personalized recommendations, even if an individual never had ranked a movie or a movie had never been rated. Thus, in order to retrieve optimized advertising revenues, we decided to focus on the creation of a matrix factorization, content-based recommender system. Similar to other streaming media companies, our basis for success is in the recommendation algorithm we will put into development. This data mining tool is specially tuned to create top "n" recommendations for a user based on ratings and movie attributes. The goal is to provide precise recommendations to users based on RMSE and other evaluation metrics touched on later in the paper. In order to move forward in our plan to target Twitter users with relevant media, we dug into the data with which we built our models.

3. Data Understanding

In order to create a recommender system, at least two features are required. These two features are users and movies represented by identification numbers and corresponding ratings. Ratings are at a scale from 0 to 10. With these, a simple collaborative recommender can be built. With the introduction of rankings, the recommender now has the ability to recommend movies that not only show up often between users, but can now recommend users movies they would actually like due to the ratings functioning as the target variable. The initial data sets with user id, movie id, and rating were taken from Simon Dooms' MovieTweetings dataset via Github. Dooms reports as of this writing the number of ratings to be 554,992, over 47,000 unique users, and 26,534 unique movies collected since February 2013 [3]. The average number of ratings per user is 12 and the average ratings per movie is 21. The data in the database were well-structured Tweets referencing the International Movie Database (IMDb) website. An example provided in Dooms' paper on the subject, "MovieTweetings: a Movie Rating Dataset Collected From Twitter", is shown below:

'I rated The Matrix 9/10 http://www.imdb.com/title/tt0133093/ #IMDb' [3]

Since "The Matrix" was rated from IMDb, the corresponding movie ID is listed within the dataset. In order to obtain the data from MovieTweetings, an understanding of the formats of the data was required. MovieTweetings is separated into three different flat files: users.dat, items.dat, and ratings.dat. Ratings.dat has all relevant information to make a recommender plus the rating timestamp and items.dat includes genre categories. Format examples of the data from users.dat, items.dat, and ratings.dat are below for reference:

Users - 1::177651718

Movies - 0110912::Pulp Fiction (1994)::Crime|Thriller

Ratings - 14927::0110912::9::1375657563 [3]

Believing it necessary to find other features beyond user id, movie id, ratings, rating timestamp, and genre, we searched for a dataset that referenced IMDb's movie identification numbers and had extra features not included in MovieTweetings. We eventually discovered the Open Movie Database and API by Brian Fritz [4]. This API contains granular content on many forms of media such as documentaries, movies, television programs, and more. This includes media title, year of release, parental advisory rating, runtime, director, writer, actors, a plot summary, number of awards, distribution language, and ratings from movie rating sites such as IMDb and Rotten Tomatoes [4]. Mentioned later in the paper, we added some of these features to the recommender to improve our performance in our recommender system.

4. Data Preparation

We downloaded the datasets from github twitter repository and converted the .dat files of movie, user and ratings file to .csv files using R [5].

RapidMiner Approach

We began our analysis using a simple model in RapidMiner. This model utilized the Twitter identification numbers, movie identification numbers, and ratings. The only preparation applied was identifying the user and item columns utilizing the

Recommender package downloaded from the market place. We then selected the user and movie columns. This ended the data preparation for RapidMiner.

GraphLab Approach

1. Created 28 binary features from Movie Genres using R
2. Mined OMDB features for more than 13,000 movies using OMDBI package in R from the OMDB api
    1. Pulled subplots from IMDB api using imdb package in Python for 13,000 plus movies
    2. Converted movie length feature from text like “1 h 35 m” to numbers like 95 minutes
    3. Created binary dummy feature for awards column as awards won 1 or 0
    4. Created a binary dummy variable of item type as is_movie 1/0
    5. Convert all the required datasets into SFrame

A dataset structure can be represented as both Python data frame and Sframe. However, SFrame are different from python in many ways -

1. SFrame is defined in Graphlab but dataframe is defined in Pandas dataframe
2. SFrame can be scaled to big data, “S” stands for scalable
3. SFrame is column immutable and are stored on disk to avoid burden on memory, hence they are not restricted by memory size

5. Modeling

We explored the following three types of recommender systems (RSs):

Collaborative Filtering RS - recommend items that the users' peers also like. This is the most common approach in generating recommendations.

Content Based RS - recommend items that are similar to items the user likes.

Hybrid - A combination of collaborative filtering RS and content based RS.

We summarized the strength and weakness of the three approaches in the table below:

	Strength	**Weakness**
Collaborative Filtering RS	● Do not require any knowledge of users or items themselves. ● Has the ability to provide serendipitous recommendations	● Cannot deal with new items or new users for which rating data are not available. ● Cannot deal with similar items with different names.
Content Based RS	● New items can be recommended once item attributes are extracted.	● Subjective characteristics of an item that are important to users' preferences can be difficult to measure in some domains. (i.e. ease of use) [6]
Hybrid	Will provide better or more	No specific weakness.

	precise recommendations than a single algorithm as the disadvantages of one algorithm can be overcome by another.	However, how the results of different techniques should be weighted is one of the specific questions that need to be considered using this approach.

RapidMiner Approach:

When developing the RapidMiner workflow, we looked at both user-based and item-based collaborative filtering recommender systems. Within these there were also two types: rating prediction and matrix factorization. The rating prediction method utilized Item-based KNN and User-based KNN to predict the numerical values of the ratings. These then could be evaluated using RMSE, Mean absolute error (MAE) and Normalized mean absolute error (NMAE). Along with the rating prediction, matrix factorization can be used. Matrix factorization finds the latent features and their interactions with ratings given users and movies. Pictures of the user-based KNN and matrix factorization models are shown below. These initial models were created with the direction of our professor, Panagiotis Adamopoulos.

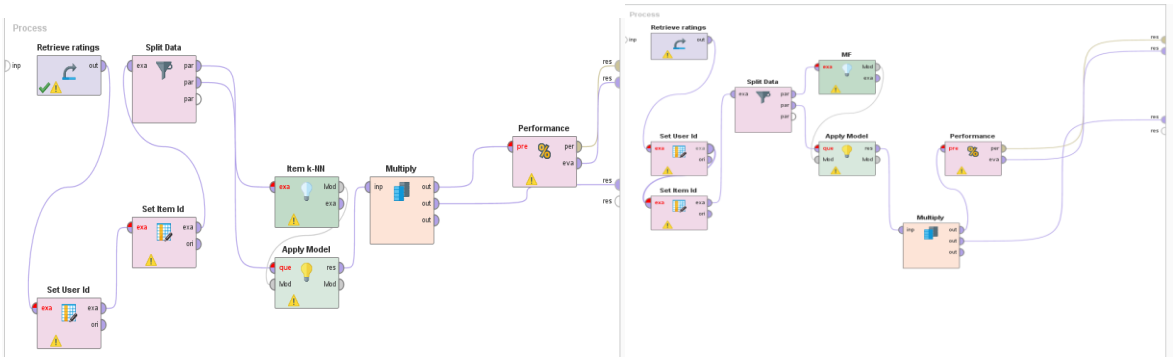

Figure 1 : Left: User based KNN movie recommender , Right : User based KNN movie recommender in Rapid Miner

The second method used was a weighted regularized matrix factorization. This is a collaborative, implicit feedback approach and only the user identification number and movie identification number were used. Since the ratings are not being predicted, the evaluation of this model is by area under the curve (AUC), Precision, Normalized discounted cumulative gain (NDCG), and Mean average precision (MAP).

Python Approach (from scratch):

We built a basic user-based collaborative filtering (CF) recommender system in Python in order to understand the building blocks of the system. The only input for a CF recommender is the ratings file. The main idea of a user-based CF recommender is to identify users who are similar to the target user and make recommendations to the target user based on these similar users' ratings on the same movie. We used the K Nearest Neighbor (kNN) algorithm to find similar users and used Pearson correlations to measure similarities between users, as it is the most commonly used measure for user-based CF recommender systems. This method required up to 80 lines of code using only the pandas library. Using the GraphLab package in Python (which will be discussed in the following section), we were able to accomplish the same thing in under 10 lines of code. As such, we continued to explore different recommender system models using the GraphLab approach.

GraphLab Approach:

We started initially with same ideas as we used in our RapidMiner approach utilizing Matrix factorization with ratings. Matrix factorization is an approach of factorization or decomposing a matrix into many matrix decompositions. Then multiplying these matrices together again provides us a dataset which is complete and has no missing values. This approach helps us in dealing with the sparsity of the dataset and was first used in the Netflix competition for recommenders [6]. We also collected features about the movies because we wanted to include them in our recommender systems. For this task we used a generalized version of recommender systems: factorization machines. Factorization machines basically combine the advantages of SVM and factorization models. Factorization machines also can incorporate side features of users and movies and learn latent variables. We added side features such as movie genre and other movie's attributes taken from the OMDB website and optimized the recommender using a five-fold cross validation framework with RMSE as our evaluating metric.

6. Evaluation

The value of a recommender system is determined by how much it will help to achieve the business goal. Will it help to improve customer satisfaction, improve revenue, or attract new customers? As an online advertising agency, our main goal is to improve the click through rates that contribute directly to our revenue.

The best way to evaluate the effectiveness of our model is through controlled experiments and A/B testing on real users. However, due to the limitation of resources

in the beginning phase, the scope of this project is to use off-line experiments to simulate user behaviors from historical data.

Although we realize accuracy is not necessarily the most important measurement, we assume that a recommender system that provides more accurate predictions will be preferred by the users. Our system will provide each user with ten recommended movies and predicted ratings. We used five-fold cross validation that separates the data to training and testing set. We used RMSE that compared the difference between the actual user rating and our prediction. Adding up the squared errors provides a measurement of how the prediction deviated from the true rating.

The RMSE of our initial baseline model was 1.6944. Including latent genre features did not improve the model performance and increased RMSE to 1.8760, but including implicit features from the OMDB site along decreased RMSE to 1.6567. Finally, when we combined those two attribute sets, our final model RMSE was 1.6498. This means on average our prediction can be positive or negative 1.6537 from the true rating on a 10-score scale.

As a sanity check, we introduced a new user and asked him to rate 5 movies we provided (*Spring Breakers, The Martian, Star Wars IV, The Lion King,* and *Titanic*) and provide some best and worst movie he had watched (See the entire movie rating in Appendix). Based on his rating of 9 movies, we recommended 10 movies to him (see recommendation in Appendix). From the user's feedback, our recommender system did a good job.

Snapshot of Recommender system below -

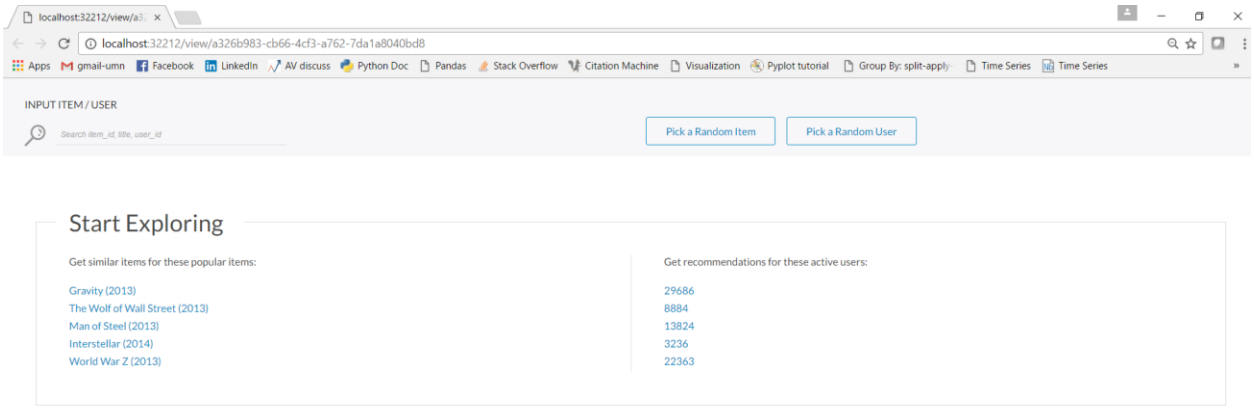

Figure 2: Home page of Graphlab recommendation dashboard

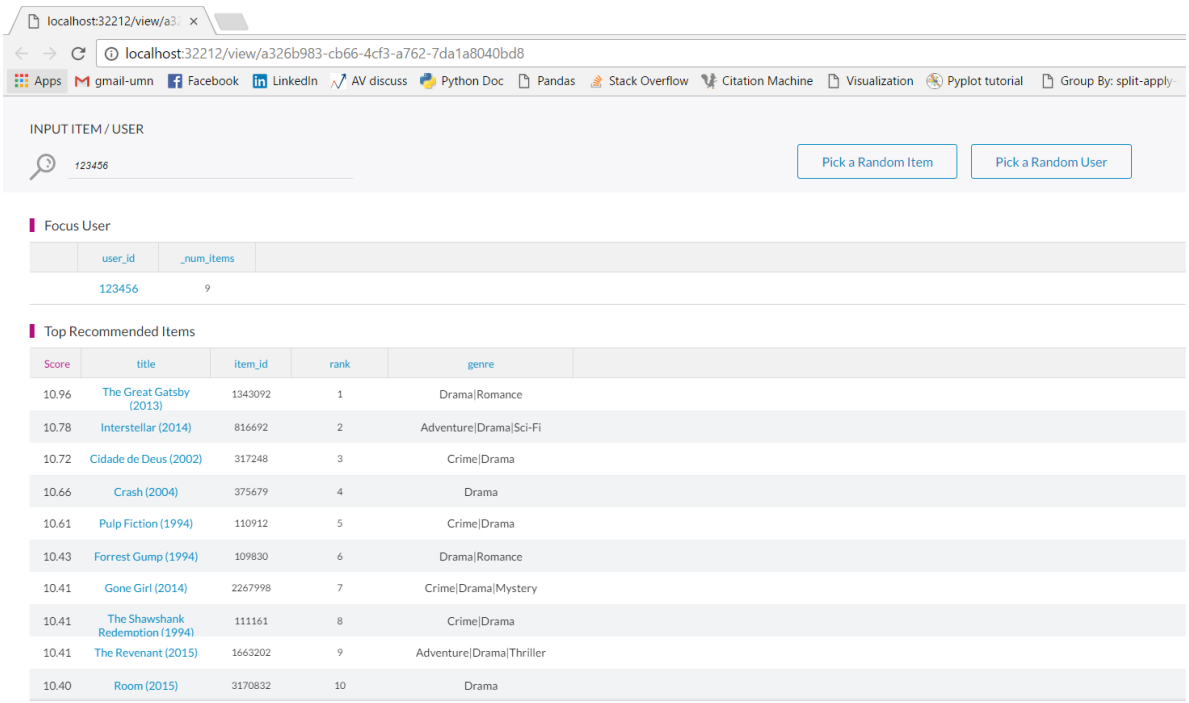

Figure 3: User specific recommendation of Graphlab recommendation dashboard

We do realize that there are many more measurements beyond accuracy to consider about, including:

- Coverage: if the recommender system only recommends the most frequently rated popular movies or covers a large proportion of movies in our database

- Serendipity: how surprising and relevant are our recommendations to the users. It can be measured by the difference of how likely a movie will be recommended to user A and how likely it will be recommended to any users [7]
- Utility: what are other factors that increase the click through rates other than accuracy? For example, quality of the trailer, length of the trailer, year of the movie, etc.
- Novelty: does the system recommend movies that are new to the user and they cannot find otherwise
- Robustness: how well can the system handle outliers and anomalies

We also did not consider the cost of inaccurate recommendations. In a real business environment, overrating a movie and recommending it to a user would have a worse impact than underrating a movie and failing to recommend. However, adding the cost factor to an offline evaluation may not be able to generalize online users.

7. Deployment

RapidMiner for Quick Deployment

Our deployment of our recommender system first began with creating a small-scale example that was accessible for business users to understand. We determined RapidMiner and its user-interface to be business-user friendly and allowed for quick runs of the model. Using RapidMiner, we developed a basic recommender system that uses collaborative filtering approach with matrix factorization. This method requires data

on ratings only and enables faster predictions especially on high-dimensional data that have large numbers of users and movies. A main advantage of this method is that the recommender model application can be explained without the prerequisite for any particular programming language. As no code is involved in building the process in RapidMiner, it is easier for the audience to focus on the application of the model itself as well as be involved in the training and evaluation process. The RapidMiner process could also be a good candidate for testing proof of concept for businesses, as the process in RapidMiner can be set up within minutes and can be refined and repeated based on different business requirements. Users can apply different types of collaborative filtering recommender systems (user based vs. item based) and compare the different models based on cross validation performance within minutes in RapidMiner. However, when it comes to implementing a large scale solution, our GraphLab recommender is ideally suited to fulfill a client's recommendation needs.

Support for Big Data Analytics

The main advantage of using GraphLab is its parallel programming design and its capability of integrating with big data platforms. GraphLab is designed to build efficient, parallel machine learning algorithms that work for large scale datasets. Given the nature of our business problem, it is crucial to be able to train the model on as much data as we could collect. Implementing the recommender system using GraphLab coupled with the Hadoop environment would allow us to run algorithms on large user ratings datasets from Twitter and still provide fast predictions. Furthermore, GraphLab also provides a simplified deployment process to production. Typical challenges with deploying recommender system code into production include rework on code to support dynamic

configurations and integration of workflow management tools as well as monitoring tools. GraphLab Create provides an easy way to manage the deployment process by enabling users to create and monitor jobs on Hadoop and EC2 clusters without having to change code [8].

Use Case for Twitter Ads

A use case of our GraphLab recommender would be to support Twitter's new in-feed advertising strategy. During March of 2016, Twitter launched its new algorithm that reorders users' Twitter feeds so that tweets that are more relevant to the users are displayed on top. This change in algorithm suggested a transition in Twitter's strategic focus from chronologically ordered content to content relevancy. Additionally, Twitter has been experimenting with different ways of displaying advertisements to its users. Their goal is to target more relevant advertisements to users and bring brands and people who are interested in their products together without encroaching on users' feeds [9]. Twitter has seen positive results in its movie advertising. A recent study shows that the combination of Twitter Ads and TV ads boosted ROI and was more effective than either one alone [10]. Different types of Twitter advertisements include promoted Tweets, promoted accounts and promoted trends. In order to make the promoted content more relevant, Twitter utilizes users' information (user profiles, apps installed on users' mobile devices, user location, etc.) as well as users' interaction with other Twitter accounts to determine what content to promote. In alignment with Twitter's focus on content relevancy, our recommender system solution could add value to Twitter's advertisement program as it allows them to promote trailers for movies that users are likely to be interested in. Our hybrid solution enables movie recommendations

based on the user's past ratings, movie features, as well as users who are similar to the targeted user. This is particularly beneficial when we are making recommendations to new users or recommending new movies for which movie ratings data are not available. One important application-specific consideration is the diversification of movie recommendations. As Twitter users are able to opt out of promoted tweets, it is important to consider the diversity and novelty aspects in the movie recommendations. As discussed in the *Evaluation* section, we should be cautious with relying on accuracy as the only criteria and consider using additional metrics to improve the novelty and diversity of the recommender system results [11].

Please see codes and data inputs and outputs at the associated github link https://github.com/aayushmnit/predictive_analytics_project.

8. References

[1] Peterson., T. (2016). Twitter's New Video Ads Will Greet Visitors Right Away Each Day. Retrieved December 04, 2016, from http://adage.com/article/digital/twitter-s-newest-ad-unit-pair-algorithmic-feed/302604/

[2] Teixeira, T. S. (2015). When People Pay Attention to Video Ads and Why. Retrieved December 04, 2016, from https://hbr.org/2015/10/when-people-pay-attention-to-video-ads-and-why

[3] Dooms, S., Pessemier, T. D., & Martens, L. (n.d.). MovieTweetings: A Movie Rating Dataset Collected From Twitter. Retrieved from http://crowdrec2013.noahlab.com.hk/papers/crowdrec2013_Dooms.pdf

[4] OMDb API. (n.d.). Retrieved December 04, 2016, from https://www.omdbapi.com/

[5] Source_GitHubData: A simple function for downloading data from GitHub into R. (2013). Retrieved December 04, 2016, from https://www.r-bloggers.com/source_githubdata-a-simple-function-for-downloading-data-from-github-into-r/

[6] Koren, Yahedu, Robert Bell, and Chris Volinsky. "MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS." IEEE. Columbia University, 2009. Web.

[7] B, A. (2016). Recommender Systems - It's Not All About the Accuracy. Retrieved December 04, 2016, from https://gab41.lab41.org/recommender-systems-its-not-all-about-the-accuracy-562c7dceeaff

[8] Arya, R. (2014, July 21). GraphLab in Production Data Pipelines. Retrieved from http://www.slideshare.net/turi-inc/gl-conference2014-deploymentrajat-47590398

[9] TOLENTINO, M. (2013). Twitter's New Tailored Ads: How They Work and How to Block 'Em - SiliconANGLE. Retrieved December 04, 2016, from http://siliconangle.com/blog/2013/07/15/twitters-new-tailored-ads-how-they-work-and-how-to-block-them/

[10] Ciszek, Thomas. "New Movie Marketing Research Reveals Twitter Ads Deliver Increased Ticket Sales." Twitter. Twitter, Research Department, 10 June 2016. Web. 4 Dec. 2016.

[11] Sandoval, Saul Vargas. "Novelty and Diversity Enhancement and Evaluation in Recommender Systems." Ed. Pablo Castells Azpilicueta. Programa Oficial De Posgrado En Ingeniería Informática Y De Telecomunicación (n.d.): n. pag. Http://www.mavir.net/. Universidad Autonoma De Madrid, Apr. 2012. Web.