MovieOracle User Tutorial

Version 1.5
July 22, 2011

Yao_Yao@baylor.edu

TABLE OF CONTENTS

1. Overview and Introduction

Many companies from different business fields, such as entertainment, advertising, etc, are interested in this question: given a movie, will a person like it or not ? Recommender systems can help to answer this question by studying the user's social network, for example, Twitter [1], Facebook [2], or MySpace [3]. With MovieOracle, which is a recommender system for movies (based on Twitter), we can predict users' interests in movies they have not seen.

This tutorial document introduces the major functions of MovieOracle and demonstrates how it can be used to predict Twitter users' opinions and make recommendations for movies to users. It also includes important download, installation, and configuration information.

The recommender system MovieOracle can predict whether or not Twitter users like specific movies. The basic idea is to consider the social connections between people who have not seen a movie and those who have already experienced it. MovieOracle makes recommendations for people based on such connections. This project assumes that within the Twitter social network, connected people, will have similar tastes in movies. Therefore, MovieOracle uses the evaluations from friends and followers to predict a user's opinion of a movie.

The major functions of this service/software include:

1. Given a movie name, collect tweets about the movie.
2. Classify the sentimental polarities of tweets being collected (using a 3rd party classifier [4]).
3. Given a Twitter user id, predict his or her sentiment towards a specific movie.

2. Download and Installation

Exterior Software Requirements:

(1) Windows XP/Vista
(2) JDK (version 6.0 or higher)
http://www.oracle.com/technetwork/java/javase/downloads/
(3) MySQL database (version 5.1 or higher)
http://dev.mysql.com/downloads/mysql/

You can download MovieOracle from http://cs.ecs.baylor.edu/~yay/. The software documents can also be found there.

Uncompress the MovieOracle.zip file. The following folder and files should be found in

the file folder “MovieOracle”: ClassifierTrainingSet (folder), bootstrap.bat, Bootstrap.jar, config.properties, MovieOracle.jar, scripts.sql, and MovieOracle.bat.

1. The ClassifierTrainingSet file folder contains the training data set for the tweet polarity classifier[4]. There are three sub file folders in this folder: pos, neg, and unknown. They store positive, negative, and unknown tweet text files, respectively. (In 7.1 “Training data for the polarity classifier”, you can find more information about how to use your own training data.)
2. The config.properties file contains important environment variables. (You can find more details in Chapter 5 “Property File”.)
3. The scripts.sql file contains the scripts for creating database tables and importing training data of a decision tree [5]. (In 7.2 “Training data for the decision tree”, you can find more information about how to use your own training data.)
4. The bootstrap.bat creates the database for MovieOracle and loads the initial data.
5. The bootstrap.jar is the packaged jar file for the bootstrap part. It is invoked via bootstrap.bat.
6. The MovieOracle.bat is the executable script of the MovieOracle system.
7. The MovieOracle.jar is the packaged jar file of this project. It is invoked via MovieOracle.bat.

3. Quick Start

(1) Start the MySQL database service.
(2) Open the config.properties file. Modify the value of item “database_name” to the database you want to create. (Make sure the name you specify is not the same as any existing database name, since it is your first time to use this software. Otherwise, the bootstrap step will fail and you cannot start the MovieOracle service.)
(3) Run bootstrap.bat to setup the database.
(4) Run MovieOracle.bat.
(5) Within the command-line interface, enter “start [movie name]” (ex. “start star trek”).
(6) Repeat step (5) for all desired movies. The MovieOracle system starts collecting tweets referring to the movie name. It also trains a decision tree, starts the polarity classifier service and potential user search service. After a while (the time length can vary), it will print twitter users who will be interested in a movie to a “Recommendations.txt” file in the MovieOracle directory.

4. Commands

The service command line interface accepts the following commands:

collect [file name]

Collect tweets on a batch of movies. The movie names are in a file indicated by [file name].

exit
Stop the service.

oracle [user id] [movie name]
Predict a twitter users' opinion on a specified movie. The user id is indicated by [user id] and the movie name is indicated by [movie name]. The prediction result is displayed on the screen. (The result is either from the decision tree prediction or from a simplistic strategy of "best guess". See Section 1.1 and 5.6 of the Detailed Design document for more details.)

predict [file name 1] [file name 2] [file name 3]
Predict a group of twitter users' opinions on a batch of movies. The user ids are in a file with the name [file name 1] and the movie names are in a file with the name [file name 2]. The prediction result is copied to a generated file with the name [file name 3]. If there exists no file called [file name 1] or[file name 2] , the system prompts a warning message. If there already exists a file called [file name 3], the system prompts you to give a different name. (You can find examples of the file containing a group of user ids and the file containing a group of movie names in Appendix A. You can also check the explanation for an example of the prediction result file in Appendix B.)

The prediction result can also be retrieved by directly querying the database. For example, "SELECT POLARITY FROM POTENTIAL_USER WHERE MOVIE_ID = 1 AND USER_ID = 12" can get a twitter user's sentiment for a prediction that has already been made.

show commands
Display all of these commands.

show movies
Display all the movies for which the service is currently downloading tweets.

start [movie name]
Collect tweets of a movie. The movie name is indicated by [movie name]. If the movie is currently been processed, the MovieOracle ignores this command and produces a warning message.

stop [movie name]
Stop collecting tweets of a movie. The movie name is indicated by [movie name]. If the movie is not currently been processed, the MovieOracle ignores this command and prompts a warning message.

terminate [file name]
Stop collecting tweets on a batch of movies. The movie names are in a file indicated by [file name].

Note:

1. Each file name used in the commands can be either a full path plus a file name or just a file name if it is in the same directory of the MovieOracle.bat.
2. If a movie name may appear in tweets not about the movie, like the movie "Red", adding a string "movie" plus a blank space as its prefix will restrict tweets to movies. For example, "start movie Red". Although this solution can avoid gathering meaningless tweets, it also may miss some valuable tweets that do talk about the movie but have no such keyword.

5. Property File

The system property file is config.properties, which is in the MovieOracle directory. It contains environment variables for MovieOracle. (Moving or deleting this file before MovieOracle starts will cause it to fail.)

1. database_name: name of the database used by the MovieOracle system. When you first use the system or you want to use a new database for the system, you need to select a name that does not conflict with any existing database. Otherwise, the bootstrap will fail and you can find the reason in the log file "bootstrap.log". The default value is "MovieOracle".
2. related_tweets_threshold: the total number of a user's followers and friends who have evaluated a specified movie determines the technique used to make a prediction. At the related_tweets_threshold level and above, a sophisticated strategy is used, but if fewer friends and followers have made tweets on a movie, a simplistic strategy is used. Note that if you set this value too low, the sophisticated model makes less accurate predictions. The default value is 6 (more explanation for the selection of this value can be found in Section 5.4.3 of the Detailed Design document).
3. script_file: name of the file that contains the scripts for creating database tables and importing the training data for the decision tree[5]. The file must be under the MovieOracle directory. If you want to use your own training data, you may make another script file for importing different training data. In this case, you can modify this variable to your file name. The default value is "scripts.sql".
4. polarity_training_set: name of the file folder that contains the training data set of the tweet polarity classifier[4]. The folder must be under the MovieOracle directory. When a different training data set (folder) is used, you can modify this variable to the new folder name. The default value is "ClassifierTrainingSet".
5. search_limit: the number of requests allowed to invoke the search method of Twitter API within an hour. To avoid abuse, the Twitter API does not make the access

limitation public, but it is between 300 and 350. Therefore, it is better to make this value no more than 300 (you can find more explanation in Section 3.2 of the Detailed Design document). Note that if you set this value too low, the MovieOracle system will collect fewer tweets, which may significantly impact its performance. The default value is 300.

6. clear_interval: the time interval for clearing all records in the THREAD_CALL table. The table is used for each tweet-collecting thread to check Twitter API limits. Therefore, the records of an hour ago are meaningless for threads. It is not a high-priority task, but if you set this value too low, the clearance task will be executed very frequently, which is a waste of system resources (ex. CPU). The default value is 12 (hours).
7. check_tweet_interval: the time interval for classifying tweets. This value should not be set too low because system resources (ex. CPU) can be wasted and it should not be set too high because prediction methods need to know the tweets' polarities. The default value is 5 (minutes).

The following four variables were provided by Twitter when the MovieOracle system was registered. The benefit of MovieOracle's registration is to increase the access limitation of Twitter REST API from 150 to 350 (therefore, for MovieOracle, it can find 350 friends and followers of a twitter user per hour). You should not change the values of these four variables, unless you have higher access limitations. More information about these parameters are in http://dev.twitter.com/pages/auth.

8. token: token. The default value is "29361004-HnEcThXEp0GiAaqxtf5PBm4hzdmYCrxP7lvRrOVB0".
9. token_secret: token secret. The default value is "bztWPImbgml92XDFqCnjgKBYbSNv7h8C8qll9IkE".
10. consumer_key: consumer key. The default value is "pojxr7DEQLNSqyuT5x0g".
11. consumer_secret: consumer secret. The default value is "ef0p4DUx2VqaMySomczsypOGIRBoAXU2adswv3muk".

6. Build Project

1. Download the source code from http://cs.ecs.baylor.edu/~yay/.
2. Uncompress the src.zip. Two file folders can be found: one is "src" which stores all of the source code and the property file. The other is "system lib" which stores all of the system dependency jar files.
3. Copy all of the jar files in the "system lib" folder to the "src" folder.
4. With the DOS command-line (located in the "src" directory), type "javac -cp ci-bayes-1.0.4.jar;javolution-5.5.1.jar;lingpipe-4.0.1.jar;mysql-connector-java-5.1.7-bin.jar;twitter4j-core-2.1.12.jar;. SystemController.java" to compile the whole project.
5. After successfully compiling, type "java -cp ci-bayes-1.0.4.jar;javolution-5.5.1.jar;lingpipe-4.0.1.jar;mysql-connector-java-5.1.7-b

in.jar;twitter4j-core-2.1.12.jar;. SystemController" to start the service.

7. Advanced Usage

There are two training data sets for this project. One is used by the polarity classifier and the other is used by the decision tree. Both of these two data sets can be replaced or modified for more accurate predictions.

7.1 Training data for the polarity classifier

The training data for the polarity classifier[4] is stored in a "ClassifierTrainingSet" folder (which is indicated in the property file "config.properties"). This folder has three subfolders; they are "pos", "neg", and "unknown". Each file in the "pos" folder represents a positive tweet. Each file in the "neg" folder represents a negative tweet. Each file in the "unknown" folder represents a unknown tweet. In order to modify the original training data or add new training data for the polarity classifier, the files in each subfolder must be modified.

In the training data set, a file contains exactly one tweet. This makes it easy to use n-fold cross-validation to validate the accuracy of the training data.

7.2 Training data for the decision tree

The training data for the decision tree[5] is stored in the DECISION_TREE_TRAINING table. In order to modify the original training data or add new training data for the decision tree, use regular SQL to manipulate the records of the table. The data model for the table is:

Field	Type	Null	Key	Default	Extra
TWEET_ID	BIGINT(10)	NO	YES		
RELATIVE_POS_NUMBER	NUMERIC(6,5)	YES	NO		
RELATIVE_NEG_NUMBER	NUMERIC(6,5)	YES	NO		
RELATIVE_UNKNOWN_NUMBER	NUMERIC(6,5)	YES	NO		
POLARITY	VARCHAR(10)	YES	NO		

TWEET_ID: tweet status id
RELATIVE_POS_NUMBER: relative number of positive related tweets
RELATIVE_NEG_NUMBER: relative number of negative related tweets
RELATIVE_UNKNOWN_NUMBER: relative number of unknown related tweets
POLARITY: sentimental polarity of a tweet (pos: positive, neg: negative, unknown: unknown)

Line 52 to 227 of the scripts.sql file are 176 commands used for importing the training data for the decision tree. These scripts can be replaced or new commands can be added to them. For example, a positive tweet is found with 1 negative related tweets, 3 unknown related tweets, and 6 positive related tweets. Its tweet status id is 123456. In this case, the following script can be made for adding to the original scripts: INSERT INTO DECISION_TREE_TRAINING (TWEET_ID, POLARITY, RELATIVE_POS_NUMBER, RELATIVE_NEG_NUMBER, RELATIVE_UNKNOWN_NUMBER)
VALUES (123456, 'pos', 0.6, 0.1, 0.3);

APPENDIX A

Contents of user id file

243022170
94109281
43071113
5871
…

Contents of movie name file

Avatar
Iron Man 2
X-Men
…

APPENDIX B

File Content

Avatar
243022170: positive [user prediction]
94109281: positive [simplistic prediction; 6 scores]
43071113: negative
…
Iron Man 2
243022170: unknown
94109281: negative [simplistic prediction; 5 scores]
43071113: negative
…

Explanation

This result is similar to the result of “oracle” command.

1. The decision tree (which means higher accuracy) predicts that 243022170 is a twitter user who holds a positive attitude to the movie “Avatar”.
2. The user 94109281 possibly has a positive attitude to the movie “Avatar” because major users hold positive sentiments to this movie by now but the system does not have sufficient information to use decision tree to predict this user. The “6 scores” indicates the confidence degree of this prediction. Higher score means higher confidence (The score value is between 3 and 10).
3. If the system has no information for a movie, no prediction result will be displayed.

References

[1] Twitter Inc. Engineers. Twitter / Home http://twitter.com/ Twitter Inc.

[2] Facebook Inc. Engineers. Welcome to Facebook http://www.facebook.com/ Facebook Inc.

[3] MySpace Inc. Engineers. Myspace | Social Entertainment http://www.myspace.com/ MySpace Inc.

[4] Joseph Ottinger. CI-Bayes https://ci-bayes.dev.java.net/ Open Source

[5] Jiawei Han and Micheline Kamber. Data Mining: Concepts and Techniques, Second Edition (Chapter 6.3.2)