Pulse News Preference Prediction

Jing Ma Chi Zhang
CS229 Machine Learning Course Project, Stanford University

***Abstract*—This paper is a summary of all the work our group has done for the final project this quarter. The topic is using algorithms taught in class to develop a way for news prediction with a relatively high accuracy. The dataset is provided by pulse application, containing 1000 user's read and click record. To make the result better, we tried a lot of algorithms, such as simple data filtering, logistic regression, feature deduction, and so on.**

I. Introduction

Natural language processing (NLP) is an important application area of machine learning and already widely used today. A large portion of text that people consume every day is news. The news we read play an important role in how we form opinions and beliefs about the world and its actors. Understanding people's news preferences is therefore an important task. If we can predict the news stories that users are most interested in we could first improve their reading experience. It could further alter how well informed we are about many issues going on in the world. In this project, we investigate the algorithms to predict users' news preferences based on their previous reading activities. We work on a large real-world dataset of news consumption which comes from a local startup that allows people to read news on their phones.

The goal is to predict the stories people are most likely to read. More specifically, we need to classify the pieces of news to two categories: one that the user is most likely to read, and one that the user most probably would not read. This problem falls into the area of document classification. Some commonly used algorithms for document classification includes expectation maximization, naïve Bayes classifier, term frequency – inverse document frequency (tf-idf) vector, latent semantic indexing, support vector machines, artificial neural network and K-nearest neighbor algorithm. In our design, we mainly used tf-idf vector and logistic regression to classify the news and predict users' reading preference.

The remaining content of the report is organized as following. First, we will briefly introduce the data set and preprocessing we have done. In methodology section, we will discuss about our feature selection and logistic regression technique. A brief description of the key concept tf-idf vector is also included.
After that, we will discuss on our simulation and results. Then, we will conclude the report with a summary and outlook.

II. Dataset

We work on a large real-world dataset of news consumption in a certain time period which comes from a local startup that allows people to read news on their phones. There are three data files for reference. Stories.log lists all the stories available for people to read. The information for each story includes the story URL, story title, feed URL, feed title and time stamp.

User_story_reads.log contains all stories read for a sample of 1000 pulse user during the time period. A read event is defined as clicking a story title in Pulse and viewing the RSS feed content in text mode. For each story, there is information on user ID, story URL, story title, feed URL, feed title and timestamp. User_story_clickthroughs.log comprises similar information as the user_story_reads.log. A click through event is defined as clicking a link to view a

story in web mode. The dataset covers reading history of about one month.

We have done a lot of data processing work to ease the simulation work. First, we removed duplicated stories from the three files. Besides, we divided the user_story_reads.log into smaller files. The original file is over 1GB, which will occupy a lot of memory space during simulation. We divided the file into 20 smaller files; each contains the reading history of 50 users. Moreover, We sorted the dataset. For stories.log, we rearranged the data such that they are ordered by date. The user_story_reads.log and user_story_clickthroughs.log are sorted by user first and they by date. After that, we extracted the features such as story title, feed title and URL element from each story.

III. Methodology

In this section, we mainly describe the features we have selected and the algorithm we are using for the news classification and prediction

A. *Feature Selection*

We consider each piece of news in three major components, story title tf-idf vector, story URL IP, and feed title. The feed URL are mostly pulse feeds, which is relatively common, so we excluded it in feature consideration.

The story tf-idf vector is the primary feature we are considering. First, we construct a key word set based on the story titles a user has read and clicked. Say we have a list of document titles; the item in the tf-idf vector corresponds to the tf-idf weight of the key word at the same index. A keyword's term frequency is the number of times the word appears in the title. Its document frequency is the number of titles the word appears over the total number of titles. The keyword's tf-idf weight is its term frequency divided by its document frequency. Then, the tf-idf vector is normalized. The tf-idf vector represents the occurrence of the important keywords a user is interested in. In this model, each document d_j is first represented as a term-frequency vector in the term-space:

$$d_{jtf} = (tf_{1j}, tf_{2j} \dots tf_{nj}) \qquad j = 1,2,3 \dots D,$$

where tf_{ij} is the frequency of the ith term in document d_j, n is the total number of the selected vocabulary, and D is the total number of documents in the collection.

Next, we weight each term based on its inverse document frequency (IDF) and obtain a tf -idf vector for each document:

$$d_j = (tf_{1j} * idf_1, tf_{2j} * idf_2 \dots tf_{nj} * idf_n) \quad j = 1,2,3 \dots D$$

Story URL IP is another important feature to consider. By observing the story URL of specific users' reads and clicks, we found that the websites a user is interested in are a limited number of webs and they do not vary a lot within a certain period of time. This is also true in real time experience. For example, user A may usually check cnn.com and newyorktimes.com for news while user B usually prefers msn.com and googlenews as his or her news source. By story URL IP we mean the main website IP. For example, the story URL IP is www.youtube.com for the story with story URL of *http://www. youtube.com/watch?v=2ee_VQJ_d4cw*. Based on the user's previous reads and clicks, we established a URL IP pool that the user is generally interested. For each available story, if its URL IP is in the user's URL IP set, then this feature is 1, otherwise, this feature is marked as 0.

Besides, we also consider the feed titles. The feed title can be regarded as a general category label of the detailed stories. The feed titles of all the stories the user reads can be consolidated as certain areas that the user is interested in. Each user usually has a limited number of areas that they are interested in and would like to read the news on them. Thus, similar to the story URL IP, we constructed a feed title set including the feed titles that the user has previously read from. In prediction, if an available

story's feed title is in the user's feed title pool, then this feature is labeled as 1, otherwise, it is marked as 0.

B. News Prediction Algorithm

There is 1000 sample users information. For each user, there is a two-step classifier and evaluate it and update it over the time series: (say there are N total number of days we have for the user)

Train on day 1 and test on day 2, 3, …, T
Train on day 1, 2 and test on day 3, …, T
Train on day 1, 2, 3 and test on day 4, …T

The news classification and prediction algorithm comprises two steps: feed title filtering and logistic regression based on story title tf-idf vectors and story URL IP.

First, a story will pass through a feed title filter. For each user, a feed title set is constructed and updated during training. The feed title of the stories that the user has read will be added to the set everyday during training. In classification, if a story's feed title can be found in the feed title set, this story will be passed to the next step; otherwise, this story will be classified as negative, meaning that the user would probably not read it. This step is to see whether a story falls into a user's interested categories of reading.

Next, if a story is within a user's interested area, it will be classified based on the story title and story URL IP using logistic regression. For each story, we obtain a story URL IP value 1 or 0 as described in previous session. This value together with the story title's tf-idf vector forms the features for logistic regression. The logistic regression classifier is updated everyday based on all reading activities before that day. This step further checks whether the story is from the user's preferred web list for news and whether the user is interested in the keywords of the story titles.

C. Feature Deduction

We noticed that the dimension of feature vector is growing as we process more days' data. So we believe it is a good idea to add feature deduction into this algorithm. However, due to the size of the tf-idf vector, the PCA didn't work for the whole dataset on our computers.

IV. Simulation and Discussio

We define positive set as the set of stories the user read and negative set as the set of stories the user did not read. The flow of our simulation is as following. On day i, we predict the user's reading preference using the classifier obtained from previous day and evaluate this classifier in terms of positive set error rate and negative set error rate. Next, we update the training set. We add the stories the user read on day i to positive training set and add a portion of the stories the user did not read to negative training set such as the positive and negative training set have the same size. Then we update the feed title set and train the new logistic regression classifier. After the training, we update the negative training set. We use the classifier to test the current negative training set, if a story is predicted correctly as negative, remove this story from the negative training set. This step allows new negative stories be added to the negative training set.

One key point of our implementation is that the positive and negative training set are balanced to train the logistic regression classifier fairer. Also, we used gradient ascent for the logistic regression training. We attempted to use Newton's method at the beginning but found that a lot of matrices are singular and not able to get an inverse. The simulation is implemented in Python. We used sklearn and Numpy modules in the implementation of tf-idf vector.

We obtained the learning curve as shown in Figure 1 and 2. For positive test set, the average error rate is very high in the first few days. This is because the training size is very small for positive training set. As the training size increases day by day, the error rate drops very quickly in first seven days.

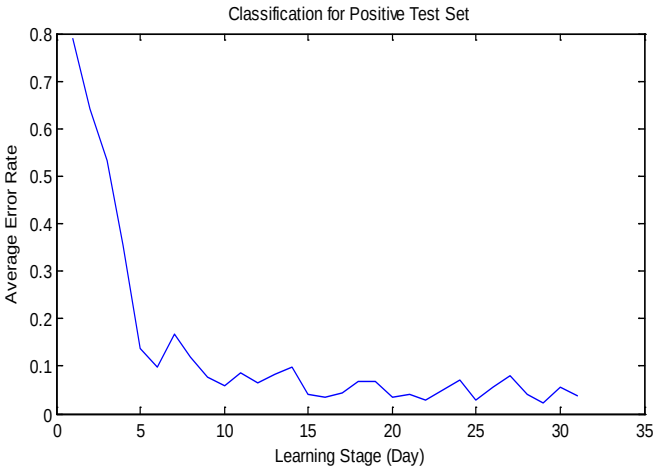

Figure 1. Two-Step Classifier Learning Curve For Positive Test Set

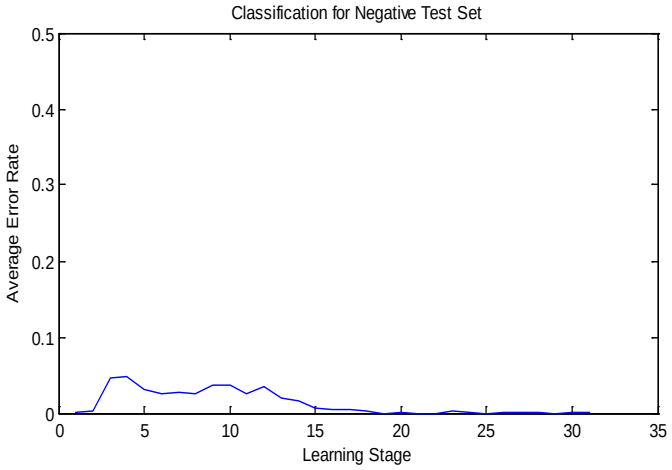

Figure 2. Two-Step Classifier Learning Curve For Negative Test Set

After about ten days, the training size is relatively adequate and the average error rate for positive set fluctuates within 10%. In contract, the average error rate for the negative set is much smaller; it is around 5% in the first 15 days and drops to within 1% in later days. Note that in the starting 2 days, the error rate is very small. This is because the logistic regression classifier has not been properly trained yet, it tends to predict a story as negative at the beginning due to the very small size of positive training example. Also, the feed title filter plays a very important role in the classification of negative examples. It rules out around 90% of the negative stories and speed up the algorithm a lot.

Our simulation focused on the news prediction of user_story_reads. We also ran the algorithm over user_story_clicks. However, the positive set average error rate is relatively high, around 30%. This is because the dataset for each user is very small compared to the huge data set of user_story_reads. On average, a user only reads a few stories by click through everyday, and less days are recorded. We also tried to correlate the clickthroughs into prediction of reads, but it did not help improve the prediction accuracy.

V. Conclusion and Outlook

The tf-idf vector provides a very good indication of keyword weight, and this model is very useful to classifier the users' reading preference in term of keywords. The two-step classification algorithm is relatively effective. The first step feed title filtering greatly reduces the complexity of the classification problem. It does not affect the positive examples much, but it filters out the majority of the vast stories available everyday. The second step of logistic regression focus on the story title keyword tf-idf vector and also takes story URL into consideration. The algorithm is able to predict the users' reading preference with more than 90% accuracy with the increase of learning stages and growing of training examples. One drawback of the algorithm is the long running time. The primary work of future work of the project is to reduce the vocabulary set and keyword features to improve the running time without scarification of much accuracy. Moreover, we shall give more thoughts on the user_story_clickthoughs data file to make use of this relatively small dataset. We can also collect more data on the clickthroughs to improve the prediction accuracy.

References

[1] 1. Manning, Christopher D., "Foundations of statistical natural language processing", 1999

[2] 2. Christian S. Perone, "Machine Learning: Text feature extraction (tf-idf)", http://pyevolve.sourceforge.net/wordpress/?p=1589