CS498, Computational Advertising: Fall 2018

Midterm Report

Netid: qiuchen2 , songwei3, yushuof2

1. Introduction

The key goals of our project are to build a multiclass classification model to predict user reputations based on the quality of contribution created on the stack exchange. Since the given textual description are pre-processed using the bags of words representations, we will mainly focus on the side of further features extraction, the classifier model training using grid search strategy, and the specific supervised learning algorithms selections, in order to estimate the internal reputation scores from data.

To better understand the features in textual description, we'll have to explore more on the tokenization to fulfill the transition from a bags of words to feature vectors. Besides, in terms of the data pre-processing, we also plan to transform the data occurrences to the term frequencies and downscale the weights of certain words by using the sklearn built-in function named tf-idf(term frequency times inverse document frequency), to get a relative count values and avoid this potential of discrepancies raised by the large number of words that are less informative.

We settled on this topic because we believe in the significant role a reputable user played in either a successful social media applications or any UGC(user-generated content) systems. Given the training data and label in stack exchange, we would be working on dealing with a particular kind of UGC environment, namely the comment rating environment where users make comments about content items with some specific words/patterns. There's also a series of the challenges we are facing with, such as the data sparseness and several confounding factors in users' voting behavior.

2. Related work survey

We would like to try a number of machine learning models in order to get a higher accuracy, for example, Naïve Bayes classifier, Random forest. Thus, we look for papers with implementation on similar problems and reviewed the paper User Reputation in a Comment Rating[1].

In this paper, they came up with a bias-smoothed tensor model to solve the problems caused by the data sparseness and several confounding factors in users' voting behavior based on Yahoo! News, Yahoo! Buzz and Epinions datasets. Currently, they try their methods in a particular kind of UGC environment, namely the comment rating environment where users make comments about content items and rate. And these methods may further be applied to all types of UGC systems.

They've made three **technical contributions**:1) They've found the surprising lack of correlation between quality and ratings, and they've applied standard machine-learning models with standard text features and have achieved high accuracy for predicting quality-based user reputation, whereas rating-based methods (including PageRank on the rating graph) fail miserably. 2) They define a support based reputation score calculated true support score sjk of user j in context k as the average rating that user j would receive in context k from all users 3) They built a latent factor model for multi-context rating prediction and showed the strong empirical performance of the proposed model.

They've built and compared three kinds of models: Random forest, GBRank, and linear regression model and use the Monte Carlo EM algorithm to fit the model. They select the features(Length,

Lexical diversity, Textual similarity, Bad words, Spellcheck, Readability, Capital words, Rating) from the comment and define the quality score with five types of labels and assign numerical scores to these labels: Excellent=2, Good=1, Fair=0, Bad=−1, and Abuse=−2, then predict the score of that comment. And obtained the results as Table 1:

Table 1: Accuracy of Quality Score Prediction

Model	Rank correlation			
	Yahoo! Buzz		Yahoo! News	
	use vote	no vote	use vote	no vote
Random Forest	0.4164	0.4049	0.4601	0.4684
GBRank	0.4054	0.4016	0.4554	0.4515
Linear model	0.3926	0.3901	0.4706	0.4619
Two editors	0.4157		0.4280	
PageRank	0.0576		-0.0368	

The predicted reputation score of a user is the average of the predicted scores of his/her comments.

Strengths and weakness:

They used the estimate the true support scores to dealing with the lack of correlation between quality and ratings and one advantage is that the true support scores are not sensitive to gaming and attacks since it involves a summation over the entire user base. However, these scores are not computable from rating data because each user j usually only receives ratings from a small subset of users.

This model is based on a rating-generation assumption especially suited for estimating user reputation in a comment rating environment, but it has not been applied to more general situations.

Compare with the paper, they focused both on comment quality and ratings and they identify users who make high-quality comments by appropriately aggregating the ratings that they receive, and based on our data set, since we don't have the rating for the comment, we only need to focus on the quality of the comment in textual description and content type. In addition, they have considered different support scores of comments in various context, since we have different types of comments, we can also consider this element in defining our comment quality scores. So we'd like to consider the features they selected except the rating, then create the labels for the scores of comments and use the similar models in the paper that predict ratings for the comments, which means Random forest, GBRank and linear regression, then use the average or other way to calculate the reputation scores to fit into our label set[1,2,3].

3. Baseline Implementation

Since we would like to train a classifier to predict the reputation score/label for the user in the baseline implementation, we choose to start with a naïve Bayes classifier. Before we start on training and fitting the model, we also performed the feature engineering by vectorizing the textual data using the `CountVectorizer` and normalizing the words frequency using the `TfidfTransformer.` After that, we will implement some basic classifier such as *MultinomialNB* from `sklearn.naive_bayes.`The finalized version would involve building a data pipeline and trained the dataset based on the compound classifier and perform grid search for suitable hyperparameters.

4. Results Interpretation

In the initial approach, we choose to only include the “textual description” data to fit the classifier. And we split the given data to train/test by 80/20. The accuracy rate we get by using naive bayes model “MultinomialNB” is 0.384, which is almost similar to the accuracy of random guess among 3 distinct labels. This probably because the naive bayes model basically considers the selected features as independent with each other, which is not true in our case. We then implemented SGD Classifier to further optimized our model and get 0.524 instead of using Naive Bayes model. For this SGD classifier, we included hinge loss function and penalty L2, and the maximum iteration number is set as 5. In the end, we used GridsearchCV to turn the hyper-parameters. The best score we get is 0.57.

Reference:

[1] Bee-Chung Chen, Jian Guo, Belle Tseng and Jie Yang.User Reputation in a Comment Rating Environment.17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,2011.