Prediction of User Appreciation of Yelp's Businesses Based on Text Reviews Hidden Features

Umberto Di Fabrizio
University of Illinois at Chicago
udifab2@uic.edu

Vittorio Selo
University of Illinois at Chicago
vselo2@uic.edu

Abstract

This work aims to predict the vote (i.e.stars) that an user gives to a business on Yelp. Each user is modeled by a set of his most important words, which are extracted from his reviews and his *tastes* are used to predict how much he will like a business. The contribution given by this work are: 1) A new framework is build to profile the users' tastes based only on the text in his reviews 2)The user profiling algorithm is used to predict the likelihood that the user likes a business. The method reaches 54% of accuracy revealing the effectiveness of the approach and encourages further investigation on this area, in particular combining this approach with classical high level features.

1 Introduction

Nowadays with the growth of websites that offer user-generated content (e.g. Yelp, IMDb, TripAdvisor) there is an increasing need, for companies, to extract the tastes of the users in order to make unique the user experience.
For this reason, recently, there have been a lot of researches in this area aimed to increase the effectiveness of the contents provided to the user (e.g. Netflix competition(Netflix, 2009), Amazon, Yelp(Nikulin, 2014)). The traditional approach is based on a system that tries to predict the rating that a user would give to an item, this type of engines are called recommendation systems and have gained popularity in the recent years.
Generally to achieve this outcome the system is trained using high-level features, for instance in a movie platform those features would be: the rating of the film, the genre, the actors and so on.
Those characteristics are easily obtained from the database and are extensively used.
Our aim is to extract *hidden-features* of an user in order to understand their personal tastes and extract this information only by the text reviews. We call *hidden-features* those personal tastes of an user which can even be unconscious for the user himself. For example suppose a user writes reviews of Mexican restaurants, why is one restaurant better than another although the food is good in both? Usually, people pay attention to details such as lights, atmosphere, the type of customers and so on, and sometimes without even being aware of their pet peeves.
The issue is how to find out, in an automatic way, those tastes that rule people feelings about a place (or an item) and which source to use.

To extract the *hidden-features* we will mine the reviews of an user in order to collect the

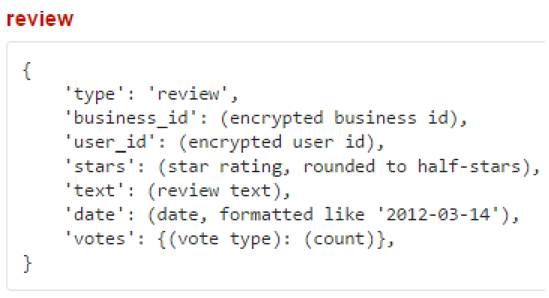
most common topics (furniture, lights, etc.) and understand what he really observes to judge a business (e.g. restaurant, pub, hotel). The hypothesis is that if an user always talks about certain topics then he cares a lot about that theme and this can guide the recommendation system.
Once the user hidden features are extracted we use them to train an ensemble of classifiers (SVM, Naive Bayes, Linear Regression) that learn how to predict the starts vote by those features.
We develop two methods: the first one is composed by one classifier that predicts one of the 9 possible classes (i.e. {1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5}) the second method is composed by three classifier which independently predict three subsets of the classes (i.e. {1, 1.5 ,2},{2.5 ,3 ,3.5},{4, 4.5, 5}) and then trains a Adaptive Boosting classifier on the predictions of those three preceding classifiers to obtain the final prediction.
The approach developed has the strength that it does not make any assumption on the type of reviews, user or platform, nor it make use of any specific domain-knowledge, this makes the framework highly flexible and ready to be used across different platform with no changes.

2 RELATED WORK

The classical approach based on user behavior and ratings is the one exploited by (Nikulin, 2014), in his work the collaborative filtering approach is combined with the content-based filtering to create an Hybrid Recommender System. A collaborative filtering approach build a model from a user's past behavior as well as similar decisions made by other users whilst the content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. The approach works pretty good by it completely ignores text reviews from users which express the reason why a user has bought a certain item. The idea to address the user profiling by exploiting the text reviews is quite knew in the literature,because it is a hard task and because the existing methods are mainly based on users ratings and ignore text reviews.The common approach is the one by (Dupuy, 2014) where meaningful words are filtered from a database of text reviews then the topics are extracted using LDA and finally a similarity measure is used to score the similarity between topics and based on the result the user receives a certain suggestion.
The main problem with user reviews on Yelp is that there is a great percentage of them which is fake, infact almost 1 out of 5 review is marked as fake by the Yelp's algorithm. As demonstrated in the work by (Luca, Zervas, 2015) economic incentives are a heavy factor into the decision to commit fraud. Moreover they show that , a restaurant is more likely to commit review fraud when its reputation is weak and this may explain the highly skewed distribution of ratings of Yelp's review that we have to face. We took inspiration from the work by (McAuley, Leskovec, 2013), the basic idea is to exploit the richness of information in the text reviews to extract topics. They develop statistical models that combine latent dimensions in rating data with topics in review text moreover they combine ratings with review text and predict ratings more accurately than approaches that consider either of the two sources in isolation. Our approach tries to dig deeper in the user profiling, with the purpose to find those unique characteristics of the user and to suggest a business that he will surely like. To follow this idea we decided not to use LDA but to extract frequent words in a user reviews and use them as indicators for user tastes.

3 DATA

We have used the Yelp dataset(Yelp, 2015) which is right now freely available because of the Yelp challenge that is currently ongoing.
The dataset is composed by 1.6M reviews and 500K tips by 366K users for 61K businesses and spans across several cities(Edinburgh, Karlsruhe , Montreal, Waterloo, Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas, Madison).
Because of the dimensions of the dataset and the computational power we have access to, we decided to limit our investigation to businesses in the area of Edinburgh(23780 reviews, 3150 users,4576 businesses). Anyway this work can be easily generalized to the all dataset or virtually to any dataset with the same info.
The format of the dataset is shown in fig 1.



review

```
{
    'type': 'review',
    'business_id': (encrypted business id),
    'user_id': (encrypted user id),
    'stars': (star rating, rounded to half-stars),
    'text': (review text),
    'date': (date, formatted like '2012-03-14'),
    'votes': {(vote type): (count)},
}
```

Figure 1: Yelp's Dataset review format.

4 METHODS

4.1 PREPROCESSING

The reviews of Edinburgh businesses are tokenized using the tweeter tokenizer (keeps smilies ':)'), then we use the nltk package for python trained on the Penn Tree Bank dataset for the POS-tagging.
For our first approach we scan each user's reviews, we remove stop-words and detect the nouns that he uses finally for each noun we collect three features. Let M be the set of all users, R the set of all reviews. Given user m $\in$ M let r_m be the set of all reviews of user m then define X_m as the set of all nouns of the user m. Let n_{xmz} be the number of times that the word x belonging to X_m appear in review z $\in r_m$. Let s_{mz} be the rank that user m gave to review z.
Now we can define:

- **Frequency**: frequency of that noun compared to the other nouns used
i.e. how much the user talks about X?

$$f(x,m) = \frac{\sum_{i \in r_m} n_{xmi}}{\sum_{i \in r_m} \sum_{k \in X_m} n_{kmi}}$$

- **Regularity**: how constantly is that noun used in the reviews
i.e. does the user talks about X in most of the reviews or not?

$$r(x,m) = \frac{\sum_{i \in r_m} n_{xmi} \frac{1}{\sum_{k \in X_m} n_{kmi}}}{|r_m|}$$

- **Relevance**: how influent is the noun to predict the rank?
i.e. is X important to the final business score?

$$i(x,m) = \frac{\sum_{i \in r_m} n_{xmi} \frac{s_{mi}}{\sum_{k \in X_m} n_{kmi}}}{|r_m|}$$

As shown in the formulas above all the scores are normalized, in this way we avoid any biasing problem for the classifier.

4.2 FIRST METHOD

The pipeline for our first method is shown in figure 2, for each user we extract nouns, adverbs and adjectives and for each of them we have the 3 features. We select the top n (30) based on frequency and we intersect

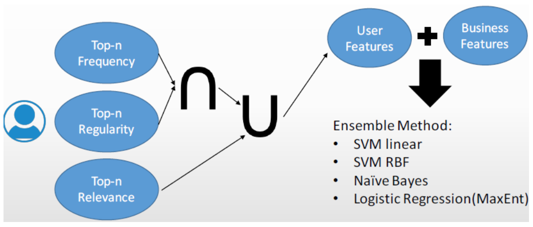


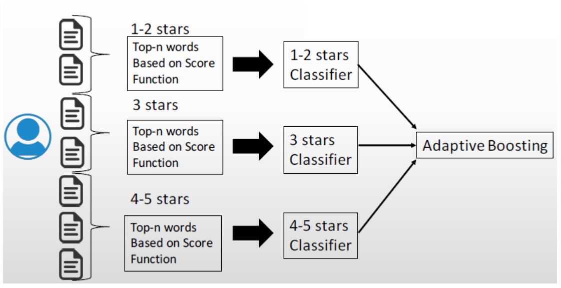
Figure 2: First Method pipeline.

them with the top 30 based on regularity, this is because we want words that are frequent AND regular, finally we compute the union between those words and the top 30 based on relevance, this is because the influence of a word to the rank is fundamental by itself. We run the same algorithm to collect words and their features for the businesses, scanning all the reviews of a business. In the end for each user we have his *best words* with their relative 3 hidden-features and for each business we have its words with their 3 hidden-features. For each best word of a user we look in the business best words to check if there is a match, in this case the business values for that word are collected otherwise a default $[0, 0, 0]$ is assigned. In the end we have a vector:
[User best nouns][Business word in common with user]
for each user-business couple. The length of the features vector is $2 * |\text{best user noun}|$, thus the input to the classifier is between $(30 * 3) * 2 = 180$ and $(60 * 3) * 2 = 360$.

The pairs for user features and business features are then used to train an ensemble classifier which predicts the final score of a business. For each user we trained an ensemble of classifiers, the validation set has been used to choose the best classifier for each user among Naive Bayes, SVM (radial basis function kernel) and Logistic Regression (MaxEnt).

4.3 SECOND METHOD

The workflow for our second method in shown in figure 3. For our second approach



1-2 stars
Top-n words Based on Score Function
1-2 stars Classifier
3 stars
Top-n words Based on Score Function
3 stars Classifier
4-5 stars
Top-n words Based on Score Function
4-5 stars Classifier
Adaptive Boosting

Figure 3: Second Method pipeline.

we identified three subset based on the users' review stars [1-2], [3], [4,5]. The reason is that we need an algorithm that can distinguish classes '1' and '2' from the rest of the world as well as discriminate between class '4' and '5'. For each subset we used two features: frequency and regularity (we omit relevance because in this context is not meaningful) and we ranked the words accordingly to the score:

$$S = log(frequency) + log(regularity)$$

This measure takes into account both features and avoids underflow. We extracted the 100 most important nouns (or less if there were not enough) for each of those subsets.
As in the first approach we applied the same algorithm to the businesses and then we extracted the business nouns in common with the user.
For each subset of user's review we train a classifier which is particularly good at discriminate between only a subset of classes (e.g. '3' or 'Other').//

We selected users with more than 50 reviews and trained three classifiers as follows (on the right the majority class baseline):

1. 1, 2, Other - 91.7%

2. 3, Other - 76.8%

3. 4, 5, Other - 46.3%

This is motivated by the fact that we need to discriminate very well between 4 and 5,and to avoid to bias the classifier towards the majority class.
Again it appears that the first classifier has an high accuracy, anyway it mostly predict the 'Other' class because of the unbalanced dataset.
Each of those classifier is actually an ensemble made by SVM(rbf kernel) and MaxEnt, the best classifier is chosen with the results on the validation test by extracting which classifier performs better with respect to the different number of minimum reviews (e.g. the users with reviews between 20 and 40 are best classified by the SVM whilst the MaxEnt is better to predict users with more than 40 reviews).
Finally we combine the predictions of each of this three classifiers to train an Adaptive Boosting algorithm with a Decision Tree base estimator that predicts the final rank of a review ([1,..,5]).

4.4 EXPERIMENTS SETUP

For our first method we selected all the user with more than 20,25,50,75 reviews and compared results. The random baseline is theoretically 1/9 = 11.1% but in our dataset there are not half stars reviews so the classes to predict are only 5 (i.e.{1,2,3,4,5}) which raises the random baseline to 20%. A more meaningful baseline takes into account the distribution of stars in the dataset and always predicts the majority class ('4'), in this case the accuracy is 42.3%. The highest baseline is actually the user majority class which predicts the user average stars vote for any business in the user test (e.g. if a business rank given by an user

Table 1: Accuracy based on the minimum number of reviews per user

Min Rev	20	25	50	75
Baseline	0.441	0.442	0.449	0.449
Method 1	0.445	0.446	0.455	0.451
Method 2	-	-	0.539*	-

is on average 3 we always predict 3 for that user), this raises the baseline to 44.9%.
Fig 4 reveals how skewed is the curve of rankings on Yelp, in fact the number of 4 and 5 stars by them selfs are 69.2% of the all dataset.
We divided the test between train(80%), validation set (10%) and test (10%).

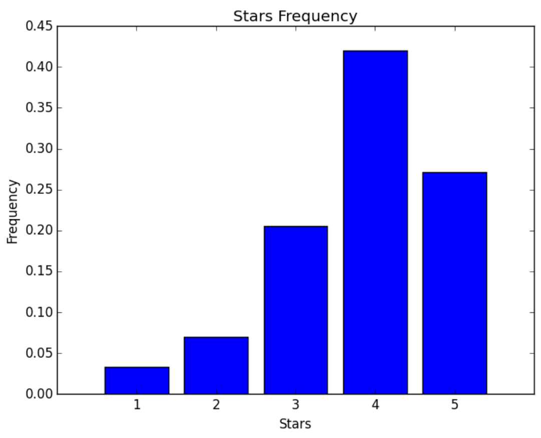


Figure 4: Distribution of stars of Yelp Edinburgh dataset.

5 RESULTS

The results for the first approach shows that 50 reviews are enough to profile an user, as presented in Table 1, this reduces the number of user selected to 72.
For the second method we achieved a statistical significant improvement reaching 54% of accuracy[1]. The analysis of the second method

[1]*p-value≈ 0

Table 2: Accuracy per class

Class	[1-2]	[3]	[4-5]
Baseline	91.7%	76.8%	46.3%
Our method	91.7%	79.5%*	58.7%*

has been further divided to understand the accuracy of each of the three classifiers: fig. 2 shows the comparison of each class with respect to its baseline; the approach cannot improve the accuracy of the class [1-2] but it outperform the baseline for the classes [3] and [4-5], especially the accuracy improves more than 12% for class [4-5].
Combining the three prediction with the Adaptive Boosting model we achieve an overall accuracy of 54% outperforming by 9% the baseline.

5.1 Error Analysis

When dealing with recommendation system it is important to understand how much does the system fails when it does not predict the right class and how does it fail.
In the experiments we found out that the baseline majority class has an average error of -0.15, which means that it usually overestimate its prediction, moreover this is by itself a very low error.
Our first model has an average error of -0.10 which means that it still overestimates the stars but with less bias.
Finally our second model has the lowest average error of -0.08. Table 3 shows the confusion matrix for method 2: it is really meaningful that our classifier is usually wrong on the adjacent classes which means that, for example, when the true prediction is '4' it predicts '3' or '5' and less frequently '2'. This is very important for a recommendation system because we can trust the fact the the system will always predict something near the true value and not just a random error. From the confusion matrix we can also see the struggle of the algorithm to predict the classes '1' and '2', this is mainly due to the low number of training samples for those classes and the highly skewed distribution of votes.

Table 3: Confusion Matrix, method 2

True/Predicted	1	2	3	4	5
1	0	5	8	5	4
2	1	17	20	43	11
3	0	17	144	113	27
4	0	22	67	422	66
5	0	14	18	153	140

6 CONCLUSION

The work has presented two methods to profile the users' tastes from the text reviews. The comparison with the baseline shows that the second one increases the accuracy up to 54%.
This demonstrates that it is possible to extract user related information only from the text. The approach is very flexible and can be adopted to any web platform because it does not depend on any domain-specific knowledge.
The limitation of our approach is mainly in the fact that the dataset is very biased towards higher scores. For this reason we would like to extend our approach and try to consider a more balanced dataset both by applying stratified sampling or by adopting a different dataset which may be very useful to compare the performances.
We would also like to extend the noun extraction by extending each noun with its synonyms and/or hypernym so that the match between user nouns and business nouns can be increased.
Finally,the framework that we developed opens the way to several extensions, for instance the sentiment linked to a certain

noun can be extracted so that the feeling towards the topic can be added to the features.

APPENDIX

The work has been evenly distributed between the students who have developed the two methods during several brain storming sessions. In particular Vittorio has worked on the POS-tagging of the dataset and on the implementation of Method 1 while Umberto has implemented the Method 2 and worked on the error analysis.

References

[Yelp 2015] Yelp Dataset. 2015 www.yelp.com/dataset_challenge

[Netflix 2009] Netflix Challenge. 2009. www.netflixprize.com/index

[Aletras 2014] Aletras et al. 2014. *Measuring the Similarity between Automatically Generated Topics*.

[Nikulin 2014] Vladimir Nikulin. 2014. *Hybrid Recommender System for Prediction of the Yelp Users*.

[Dupuy 2014] Christophe Dupuy et al. 2014. *Content Profiling from Text Reviews*.

[Luca, Zervas 2015] Michael Luca, Georgios Zervas 2015. *Fake It Till You Make It: Reputation, Competition, and Yelp Review Fraud.*

[McAuley, Leskovec 2013] Julian McAuley, Jure Leskovec. 2015. *Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text.*