CMPS 242 : Project Fall 2016

Greeshma Swaminathan GSWAMINA@UCSC.EDU
Neha Ojha NOJHA@UCSC.EDU
Jianshen Liu JLIU120@UCSC.EDU
https://github.com/ljishen/yelp-dataset-challenge
University of California, Santa Cruz, 1156 High Street, Santa Cruz, CA 95064

Project Description

The aim of our project is to predict restaurant ratings in the Yelp dataset using classification. The degree of success of restaurants or the label that we are predicting is the attribute: ”stars”, which is one of the attributes provided with the business dataset. As a part of our initial analysis, we have only considered useful features from the business dataset(like city, attributes, open hours) in order to make predictions. Our next level analysis, will involve creating additional features using the checkin, tip and user datasets and trying to improve our classifier performance by increasing the number of dimensions.

The problem that we are trying to solve is a multiclass classification problem, with stars 1-5, as five labels. The dataset has floating point values for stars, but, we have rounded the star value to create buckets for the labels values. We have used the **Naive Bayes classifier**.

Preparing Data

0.1. Preprocessing

We are using Python (specifically iPython notebooks) for the implementation. The JSON files were loaded into pandas dataframes. This helped us to get a quick overview of the data. Since ’Restaurants’ form the majority of the data, we are focusing currently on restaurants by filtering the other businesses out. Next to facilitate classification we are rounding the column ’stars’ in the business dataset so that we can form five classes (1 star, 2 star etc.). We also filtered the data to get only the ’Open’ businesses. Out of all the attributes in the business dataset, we found ’city’, ’attributes’ and ’open hours’ as the most useful attributes for analysis. We dropped other irrelevant and redundant attributes like ’latitude’, ’longitude’ etc.

New attribute	Values
Credit Cards Accepted	True,False, UnKnown
Alcohol	Full_bar, beer_and_wine, None, Unknown
Take out	True,False, UnKnown
Noise level	Very_Loud, Loud, Average, Quiet, UnKnown
Price Range	1,2,3,4,Unknown
Caters	True,False, UnKnown

Table 1. Features generated from business ’attributes’ column

0.2. Feature generation

Most of the datasets have nested relevant attributes, therefore we transformed data by decomposing in to new features.

Similarly for the attribute ’hours’ we have generated features like breakfast, lunch, night, weekend etc. For the user dataset, features like compliments and votes which have multiple sub entries we have broken them into separate features.

Naive Bayes Algorithm

0.3. Theory

We are using Naive Bayes multiclass classifier for the classification. Naive Bayes classifier uses Bayes theorem for prediction between classes. ’Naive’ indicates the assumption that the features are independent of each other.

It considers each attribute and class label as random variables. Given a record with attributes $A_1, A_2 ... A_n$, the goal is to predict class C, which for our case is ’stars’. This can be estimated directly from the data. As per Baye’s rule, we can write the posterior probability of each class as below. $P(C|A_1, A_2, , A_n) \propto P(A_1, A_2, , A_n|C).P(C)$
And with the Naive Baye’s assumption this becomes $P(C|A_1, A_2, , A_n) \propto \prod_{i=1}^{N} P(A_i|C).P(C)$ We calculate the posterior probability estimate for each class and then pick the class that has the maximum probability.

$argmax_{k=1}^{K} P(C_k|A_1, A_2, , A_n)$ where k is the number of classes

We have divided the data into training and test sets. We use the training set to train the classifier and evaluate the performance of the classifier by comparing the results produced by the classifier with true labels of the test set.

0.4. Math

To calculate the feature vector we follow the following approach. First, a group by is done on the data with the attribute 'stars'. For each star group, we find the number of restaurants matching each dimension (frequency) and calculate the probability as frequency/total number of restaurants in this class. Additionally to include Laplace smoothing we add +1 to the numerator and number of unique dimension values to the denominator.

0.5. Evaluation criteria

The performance of our classifier can be determined by measuring the number of correct predictions that it makes. For example, M is the number of correct prediction in terms of the attribute of star, N is the total number of restaurants in the test set, then the success ratio is $\frac{M}{N}$.

Evaluation

yelp-dataset-challenge/demo/Analysis.ipynb is the source code file in the repository which has the merged final analysis.

We separate the business data set into two parts, 80% as the training data and rest as the test data. It is better to use cross-validation to derive a more accurate estimate of model prediction performance, but right now we think the basic separation is good enough to show if we have chosen the right probability model and made use of enough information of the data set. By only using 8 attributes of the business data set and for the best run, we achieved **27.2% of accuracy**, which is better than the random result but not good enough. However, we also calculated the **average of absolute distance** between our predictive result and the true star, and it gave **0.693**, which meant for those missed predictions they are actually very close to the true values. Therefore, it is reasonable to assume that our model has scope for vast improvement by investigating more of other features/attributes, e.g. user reviews and tips.

Observations

Based on the prediction results and the average of absolute distances we have gotten so far, we are not sure if the rounding the value of star is a good decision for restaurants. The default behaviors of function $round()$ in Python 3 is not as widely known as it ought to be, and it is called "round to nearest, ties to even". As you can see from our notebook, both 4.5 and 3.5 are rounded to 4, which look helpful of getting rid of the bias toward the higher number, but may not be fair this specific case. On the other hand, we believe only including the most useful features is better than abusing all of features because the non-relevant attributes that could pollute our final classifier. Thus, our future improvement would stick to choose those contributing features to probability model.

Future work

In our initial analysis, we have only used the attributes of the business dataset in order to get a baseline performance for our classifier. We understand that ways like 1. adding other attributes from reviews, tip, checkin and user datasets 2. scaling and normalizing the data 3. dropping sparse attributes 4. tuning parameters 5. using cross validation may improve the accuracy of the classifier. At this point, we have preprocessed the other datsets but haven't used them in our initial analysis. We hope to see interesting results by incorporating useful attributes from the reviews, tip, checkin and user datasets as features for the Naive Bayes classifier.

We also intend to perform text mining on the 'reviews' and 'tips' of users, in order to categorize the restaurants as 'good' or 'bad'. Hopefully, this information will improve the classifier accuracy.

Bibliography

- https://en.wikipedia.org/wiki/Naive_Bayes_classifier