Project Report

Restaurant Recommendations with Yelp ratings and reviews

CS 6220 Data Mining Techniques
Spring 2018

Submitted by
Shraddha Shah
Karan Tyagi

Under the guidance of
Prof. Dr. Sara Arunagiri

Northeastern University
College of Computer and Information Science
January – April 2018

Abstract

Yelp contains review data of various restaurants in a city and that can help the user in choosing a restaurant. But, it doesn't recommend any restaurants to the user. In this project, we have explored how to use review text for recommending restaurants to users. We have investigated features of Yelp data to build machine learning models for rating prediction and recommendation models for recommending restaurants using the predicted ratings. User's restaurant preferences can be decoded based on the users' reviews about restaurants that they have visited previously.

.

Contents

Introduction

Yelp has quickly emerged as the most popular review site which has resulted in Yelp receiving massive amount of user and business data. The data varies from people's preferences and personalities, to reviews, ratings, and general information provided by the community about any business. The consumers in the world today have a vast number of choices to make from even for the simplest of services. In spite of all this information being readily available, it is often hard for people to make these choices while relying on just the raw data provided by Yelp. The format in which this data is presented to a regular Yelp user is not optimal and doesn't help them make a quick, informed decision. Therefore, majority of the users feel overwhelmed with the raw data presented to them by Yelp and would instead prefer some means of getting organized, digestible data to make a quick, informed choice. This problem is made easier for users by recommendation systems which utilize their personal preferences along with preferences of similar users to suggest potentially best choices for them.

Problem Statement

Yelp online reviews are invaluable source of information for users to choose where to visit or what to eat among numerous available options. But due to overwhelming number of reviews, it is almost impossible for users to go through all reviews and find the information they are looking for. To solve this problem, we create a recommender system. Recommender systems typically produce a list of recommendations in one of the two ways - through collaborative or content-based filtering.
Collaborative filtering approaches build a model from a user's past behaviour (items previously purchased or selected, and/or numerical ratings given to those items) as well as similar decisions made by other users; then use that model to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item to recommend additional items with similar properties.

The text of a review is often overlooked in such predictive tasks in favour of features such as the user's and business' previous rating history. However, if the sentiment of the text of a user's review can be estimated suitably, it would be the best indicator of the rating. Ultimately, a review is what the opinion of the user is about the business in his own words. Thus, it is essential to be able to predict what the user feels about a business from the review text and this is the prediction task of predicting rating from review text was chosen.

In this project, we aim to predict rating of restaurants listed in the Yelp dataset based on the reviews given by the users. Linear regression and different classification techniques such as Naive Bayes' Classification and Support Vector Machines were used for different features. And then recommending restaurants to the users using content-based filtering and collaborative filtering.

Literature Review

With the development of e-commerce and review websites, recommendation systems have been playing a key role in helping customers search and make decisions easily. This ease is powered by the large-scale repositories that are a goldmine of information in all fields from health to sports to dining out. As a result, recommendation system design is a field of arduous research in a variety of business areas, such as movie reviewing in Netflix, Amazon datasets, beer and wine reviews, and news recommendation.

Sentiment analysis is used these days to get valuable results using the variety of available data. Sentiment analysis or opinion mining is the study of emotions toward an item or entity. Sentiment analysis can be treated as a classification problem; Sentiment analysis will determine whether the sentence expresses positive or negative opinion. The most important application of Sentiment analysis is product reviews; these reviews are important for the business holders as they can decide according to user's opinion, and for users because they can be recommended for products according to opinions of other users.

Hidden Factors as Topics (HFT) combines latent rating dimensions and latent review text dimensions, which results in more interpretable topics and more accurate rating predictions at the same time. Another related work tends to incorporate rating using a Latent Dirichlet Allocation (LDA) model which parameterizes to generate better topics with sentimental meaning and which are perceivable by humans. Another study intends to improve recommendation accuracy and rating prediction accuracy by grouping users together using clustering techniques based on review topic.

We can discover good features simply by linear regression model. Classifier ensembles are known to offer a significant improvement in prediction accuracy. Ensembles include changing the instances used for training through techniques such as Bagging and Boosting.

In adaptive boosting, several weak learners trained sequentially are combined to boost the overall algorithm performance. Recently adaptive boosting methods for classification problems have been derived as gradient descent algorithms. This formulation justifies key elements and parameters in the methods, all chosen to optimize a single common objective function.

Methodology

We aim to build a recommendation system that will enable us to make sophisticated restaurant recommendations for Yelp users by applying learning algorithms to develop a predictive model of customers' restaurant ratings. We begin by providing a brief explanation of the dataset we used while creating our recommendation system. We follow this with a relevant exploratory analysis of data. We discuss the performance metrics we used to evaluate our results. We provide an explanation of our baseline algorithm and its performance, the algorithms we implemented, and the processes we used during our development. Finally, we conclude with comparing results obtained from the different approaches and a discussion of future work that could be explored.

Exploratory Analysis

The primary features of a business being used in our data analysis are business category and location (state and city). The preliminary exploratory analysis of the dataset includes study of distribution of reviews with respect to category of the business and its location.

Dataset Characteristics

Yelp.com is a website where users can review local business on a 5-star scale where 5 is for the best and 1 is for the worst. A user can also write a review text which gives an account of his/her experience. The website contains reviews about several different types of businesses including restaurants, shops, nightlife, and beauty spas. Our primary dataset is the Yelp Dataset Challenge data http://www.yelp.com/datasetchallenge that contains actual business, user, and users' review data along with check-in information of users and the tips users suggest for different businesses. As we intend to work with restaurant reviews, we have constrained this project to Yelp businesses with Restaurant as a category.

The dataset used for this task was obtained from the Yelp dataset challenge, which consists of 1.6M reviews and 500K tips by 366K users for 61K businesses. The dataset consists of five files – business, review, user, check-in and tip. Business and review files are primarily used for this predictive task. More specifically, it includes the data of 61184 businesses, 1569264 reviews and 366715 users.
All data was in JSON format.

We converted the data in JSON format to CSV files and the two main CSV files are business.csv and reviews.csv

Our project involves working with the business and reviews files. Each of the files has been briefly described here. The business file contains data related to all the businesses listed on Yelp. Its attributes are listed in Table 1. The review file contains the reviews posted by each users on different businesses. The main attributes are the rating stars, review text and the usefulness measure of the review. All of its attributes are listed in table 2.

Table 1: Attributes of restaurant data

Field	*Description*
business id	encrypted business id
name	business name
city	name of the city
state	name of the state
latitude	latitude
longitude	longitude
stars	star rating
review count	number of reviews
categories	category of the business
type	business

Table 2: Attributes of review data

Field	*Description*
review id	encrypted review id
user id	encrypted user id
business id	encrypted business id
stars	star rating
text	review text
useful	number of useful votes received

business.csv looks like:

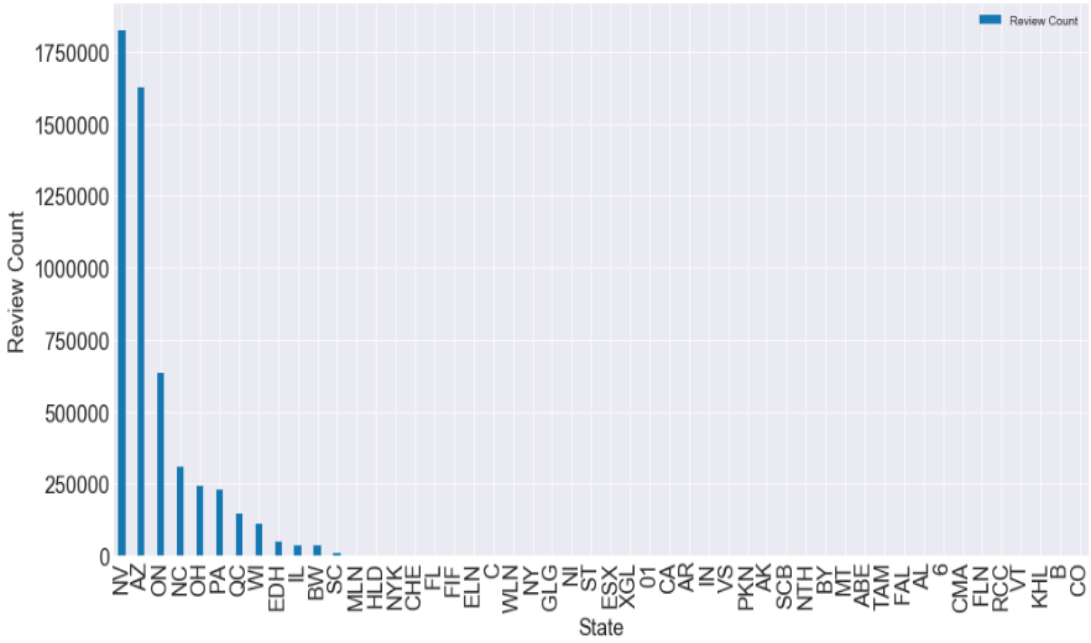
	business_id	categories	name	city	state	postal_code	latitude	longitude	stars	review_count
0	b'FYWN1wneV18bWNgQjJ2GNg'	['Dentists', 'General Dentistry', 'Health & Me...	b'Dental by Design'	b'Ahwatukee'	b'AZ'	b'85044'	33.330690	-111.978599	4.0	22
1	b'He-G7vWjzVUysIKrfNbPUQ'	['Hair Stylists', 'Hair Salons', "Men's Hair S...	b'Stephen Szabo Salon'	b'McMurray'	b'PA'	b'15317'	40.291685	-80.104900	3.0	11
2	b'KQPW8lFf1y5BT2MxiSZ3QA'	['Departments of Motor Vehicles', 'Public Serv...	b'Western Motor Vehicle'	b'Phoenix'	b'AZ'	b'85017'	33.524903	-112.115310	1.5	18
3	b'8DShNS-LuFqpEWIp0HxijA'	['Sporting Goods', 'Shopping']	b'Sports Authority'	b'Tempe'	b'AZ'	b'85282'	33.383147	-111.964725	3.0	9
4	b'PfOCPjBrlQAnz__NXj9h_w'	['American (New)', 'Nightlife', 'Bars', 'Sandw...	b'Brick House Tavern + Tap'	b'Cuyahoga Falls'	b'OH'	b'44221'	41.119535	-81.475690	3.5	116

reviews.csv looks like:

	review_id	business_id	user_id	text	stars
0	8oFUMhJ7fR2-X3We9TiK7g	kfo1hXvNtGGThfrZGaWtVw	tL2pS5UOmN6aAOi3Z-qFGg	b"I've been here a few times over the years. I...	4
1	xryg94pDLOO71veGcQINuQ	dfRAK2mgdHbL2_YsFqtCdQ	tL2pS5UOmN6aAOi3Z-qFGg	b"I used to come here back in the 90s when thi...	1
2	adNS6X4TnaxuFFxzoezGzg	So132GP_uy3XbGs0KNyzyw	tL2pS5UOmN6aAOi3Z-qFGg	b"I've eaten here a couple of times. Great foo...	5
3	JyOWXyxpN0PmPAF3OXkfCQ	cZBCVzd4lg_jx8liFz-lag	tL2pS5UOmN6aAOi3Z-qFGg	b"Have grabbed a quick bite to eat hear many t...	4
4	ClB81WvkDJwDnHN6evg84w	WzkbnhI-fxdH_tMzT3evtA	tL2pS5UOmN6aAOi3Z-qFGg	b'This is one of the best bars in town. Not ma...	5

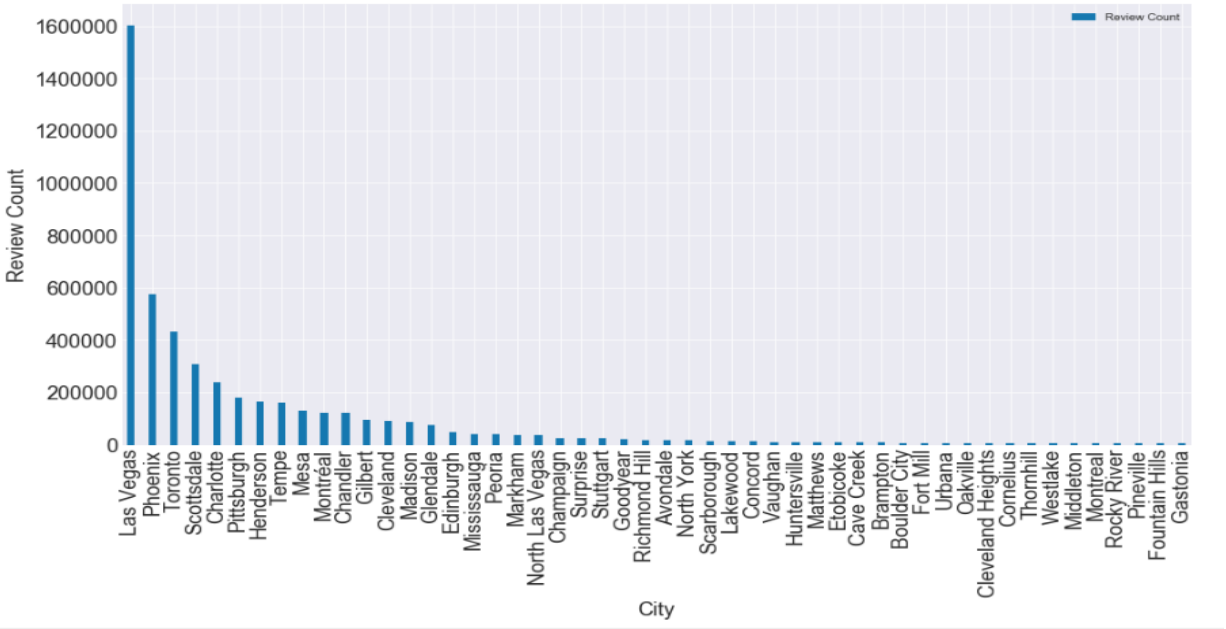
Frequency distribution of State v/s Number of reviews

From the plot of states vs the number of reviews for each state, it was observed that there are 1604173 reviews for the state of Nevada which was the highest. And the total number of states are 68.



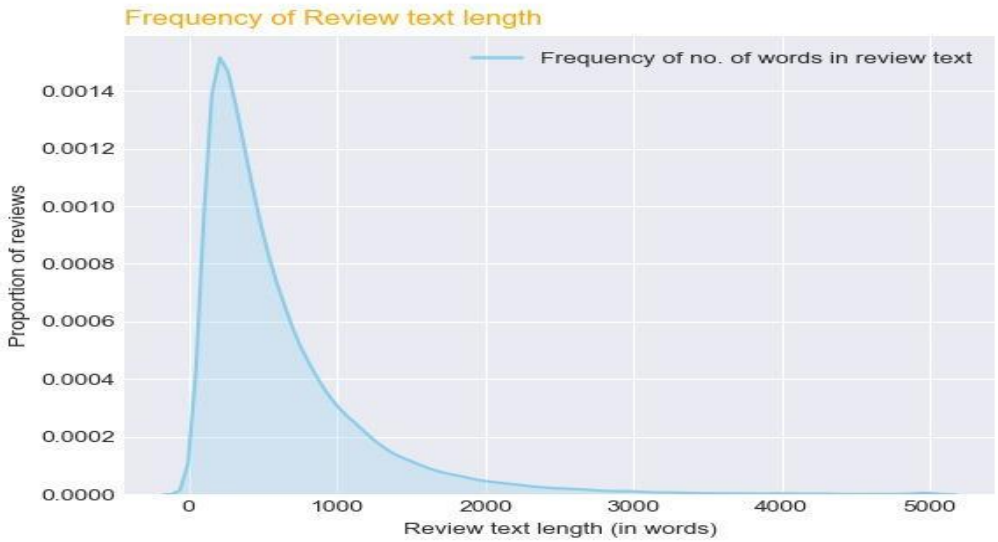
Frequency distribution of Cities v/s Number of reviews

From the plot of cities vs the number of reviews for each city, it was observed that there are 1824387 reviews for the city Las Vegas which was the highest. And the total number of cities are 1094.



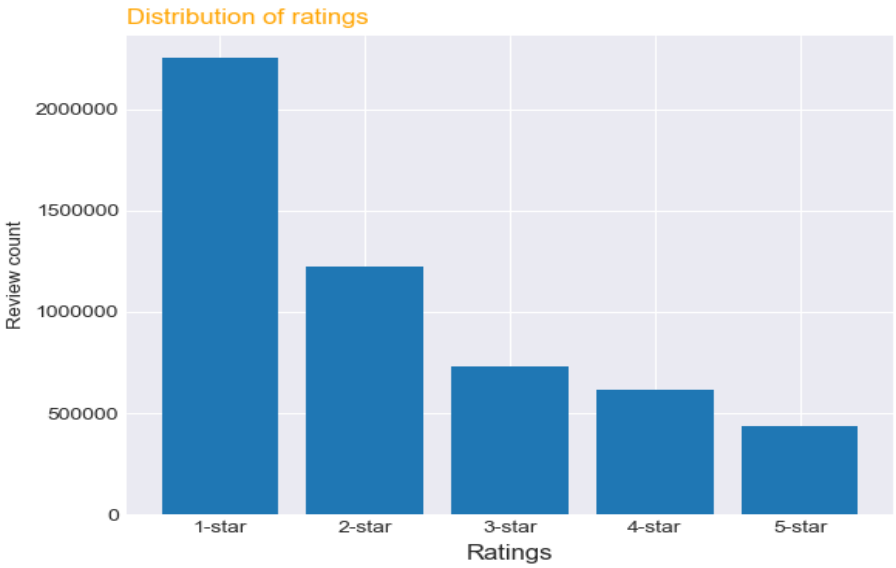
Frequency distribution of length of review text v/s Number of reviews

From the plot of length of review text v/s proportion of reviews, we observe that this word frequency is increasing till around review length 100 after which there is a steady decrease in the number of reviews till 200. Following review length of 200 words, there is a sharp decline in the number of reviews. We have considered number of words instead of number of characters as the parameter as the number of words is a better indicator of length as written by humans. That is to say that the length of the words is not of the same importance.



Frequency distribution of ratings v/s Number of reviews

From the plot of ratings v/s review count, we observe that most of the reviews have 1-star and 2-star rating.



Dataset Reduction

After exploratory analysis, we trimmed our dataset to overcome sparsity problem of the Yelp Dataset.

We selected instances with:

- Business category as 'Restaurants'
- State as 'Nevada'
- City as 'Las Vegas'
- Length of reviews between 100 and 200 words

Predictive Tasks

There are two major tasks in our project:

- Predicting rating from the review text alone
- Clustering users based on similarity and recommending restaurants to them

Predicting rating from the review text

We implemented the following two models:

- Naive Bayes Classifier
- Linear Support Vector Machine Classifier

Naive Bayes Classifier

The Naive Bayes algorithm was our first choice for rating prediction. Naive Bayes is a generative machine learning algorithm that uses learned conditional probabilities via Bayes Rule to generate the probability that a particular feature set belongs to a class. A generative model is one that is capable of creating new pieces of data from an underlying probability distribution. Naive Bayes works by storing the probability of the classifications we observe in our training data, as well as the conditional probabilities of all features we observe given that classification. Therefore, when attempting to predict the probability of a piece of test data given its features, we can use Bayes Rule to determine the probability of that piece of data belonging to a specific classification. This process is demonstrated in the figure below.

$$P(Class) = c$$
$$P(Feature\ 1 \mid Class) = x$$
$$P(Feature\ 2 \mid Class) = y$$
$$P(Feature\ 3 \mid Class) = z$$

$$P(Class \mid Features\ 1, 2, 3) = P(Feature\ 1 \mid Class) \cdot P(Feature\ 2 \mid Class) \cdot P(Feature\ 3 \mid Class) \cdot P(Class)$$
$$P(Class \mid Features\ 1, 2, 3) = x \cdot y \cdot z \cdot c$$

Once we've calculated the probability of a piece of data belonging to different classifications, the classification with the highest probability given a document's features becomes the class that the document becomes classified as. For this reason, it is easy to think about modeling the tokens of a document using a Naive Bayes classifier. In addition to this, the algorithm is quite efficient, making it worth taking note of if your system has performance restrictions.

To build a Naive Bayes Classifier using the reviews text, we carried out the following preprocessing steps:

- Removed the punctuations
- Removed the stop words

The classifier needs some sort of **feature vector** in order to perform the classification task.
We used the bag-of-words feature to convert the review text into vector format. So, each unique word in the review text will be represented by one number.

Initially, we tested our classifier for 5 classes (1,2,3,4 and 5-star rating). And we observed that 1 and 5-star reviews are easiest to predict, and we get good F1 scores for them. Neutral reviews are more difficult to predict and that was evident by their F1 scores.

So, instead of trying to predict the exact star rating, we tried to classify the review as positive (4 and 5-star reviews) or negative (1 and 2 star reviews). And the model performed better for the 2 classes.

Linear Support Vector Machine Classifier

Support Vector Machine(SVM) is primarily a classifier method that performs classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. SVM is effective in high dimensional spaces. It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient. And different kernel functions can be specified for the decision function. In this project, we use the open python library scikit-learn to implement the classifier.

To build a Linear SVM Classifier using the reviews text, we carried out the following preprocessing steps:

- Removed the punctuations
- Removed the stop words

The classifier needs some sort of **feature vector** in order to perform the classification task.
We used the TF.IDF feature to convert the review text into vector format. So each review is now represented as a set of coordinates in a high-dimensional space. During training, the SVM will try to find some hyperplanes that separate our training examples. When we feed it the test data, it will use the boundaries it learned during training to predict the rating of each test review.

Initially, we tested our classifier for 5 classes (1,2,3,4 and 5 star rating). And we observed that 1 and 5-star reviews are easiest to predict and we get good F1 scores for them. Neutral reviews are more difficult to predict and that was evident by their F1 scores.

So, instead of trying to predict the exact star rating, we tried to classify the review as positive (4 and 5-star reviews) or negative (1 and 2 star reviews). And the model performed better for the 2 classes.

Clustering users based on similarity and recommending restaurants to them

We implemented two types of recommender systems:

- Content-based filtering
- Collaborative filtering

Content-based filtering

Content-based recommendation engine works with existing profiles of users. A profile has information about a user and their taste. Taste is based on user rating for different items. Generally, whenever a user creates his profile, Recommendation engine does a user survey to get initial information about the user in order to avoid new user problem. In the recommendation process, the engine compares the items that are already positively rated by the user with the items he didn't rate and looks for similarities. Items similar to the positively rated ones will be recommended to the user. Here, based on user's taste and behavior a content-based model can be built by recommending articles relevant to user's taste.

In our system, content-based filtering recommends new restaurants based on the similarity of a restaurant's characteristics to a user's profile. Each restaurant is characterized by its

- Average review score
- Category (e.g. chinese, mexican, italian)
- Attributes (e.g. non-smoking, accepts credit card)

The category and attribute features were represented by one-hot encoding.

A user's profile is a weighted average of the features of the restaurants she reviewed, weighted by his/her rating of the restaurant. Finally, the algorithm recommends the restaurants that are closest in cosine distance to the user's profile.

Collaborative filtering

The idea of collaborative filtering is finding users in a community that share appreciations. If two users have same or almost same rated items in common, then they have similar taste. Such users build a group or a so-called neighborhood. A user gets recommendations for those items that user hasn't rated before but was positively rated by users in his/her neighborhood.

Collaborative filtering has basically two approaches:

User Based Approach

In this approach, Items that are recommended to a user are based on an evaluation of items by users of the same neighborhood, with whom he/she shares common preferences. If the article was positively rated by the community, it will be recommended to the user. In the user-based approach, articles which are already rated by a user, play an important role in searching for a group that shares appreciations with him/her.

Item Based Approach

Referring to the fact that the taste of users remains constant or change very slightly, similar articles build neighborhoods based on appreciations of users. Afterwards, the system generates recommendations with articles in the neighborhood that a user might prefer.

We implemented both the user-based and item-based approach.

We implemented the following three variations for the collaborative filtering model each for user-based and item-based:

- Using the biased rating from the dataset
- Using the unbiased rating from the rating prediction task (using linear SVM)
- Using the unbiased rating from the rating prediction task (using Naive Bayes)

Code

We used the Yelp Challenge Dataset for this project. We used the 'business.json' , 'reviews.json' and 'user.json' datasets from https://www.yelp.com/dataset. We converted these datasets into .csv files and then trimmed them further.

Our code is divided into three parts:

1) Exploratory analysis of datasets
2) Predicting ratings from review text
3) Recommendation models

We started with '*business.json'* datasets and converted them into csv files after trimming unnecessary attributes. '*analyze_business.py'* was used for creating the csv files.

Dataset created	**Description**
business-categories-reviews.csv	Contains 2 columns, categories and reviews in each category
business_full_all_attr.csv	Contains complete business.json dataset with all columns
valid_restaurants.csv	Contains business_ids of all Restaurants in Las Vegas
cities-reviews.csv	Contains 2 columns, city and reviews in each city
states-reviews.csv	Contains 2 columns, state and reviews in each state

The same python file also generated and plotted figures which are saved in the plots folder.

We then ran '*analyze_reviews.py'* with *'reviews.json'* and created csv files after trimming unnecessary attributes.

Data exploratory analysis.ipynb

The csv files were used in the above Jupyter notebook
We explored the dataset by plotting various graphs like:

- Frequency distribution of category v/s number of reviews
- Frequency distribution of state v/s number of reviews
- Frequency distribution of cities v/s number of reviews
- Frequency distribution of length of review text v/s number of reviews
- Frequency distribution of length of review text v/s ratings
- Frequency distribution of rating v/s number of reviews

And then we trimmed our dataset using the insights we got from the exploratory analysis.

To predict the rating of a restaurant based on the review text alone, we implemented the following two models:

Naive Bayes Classifier.ipynb

- Pre-processed the review text by removing the stop words using NLTK and also removed the punctuations.
- Converted the review text into vector format using bag-of-words approach using the countVectorizer of sklearn.
- Split the dataset into train and test set (80:20) using train-test split of sklearn.
- Built a multinomial Naive Bayes model and fitted it to our training set.

- Evaluated the model for 5 classes (1,2,3,4,5 star rating) and 2 class (1 and 5 star rating)

Dataset created	Description
reviews_restaurants_text_unbiased_nb.csv	Contains review data subset for Restaurants in Las Vegas with review length 100-200 words, with biased (predicted from review text) ratings, using Naive Bayes model

Linear Support Vector Machine Classifier.ipynb

- Pre-processed the review text by removing the stop words using NLTK and also removed the punctuations.
- Converted the review text into vector format using TF-IDF approach using the TfidfVectorizer of sklearn.
- Split the dataset into train and test set (80:20) using train-test split of sklearn.
- Built a linear SVM model and fitted it to our training set.
- Evaluated the model for 5 classes (1,2,3,4,5 star rating) and 2 class (1 and 5 star rating)

Dataset created	Description
reviews_restaurants_text_unbiased_svm.csv	Contains review data subset for Restaurants in Las Vegas with review length 100-200 words, with biased (predicted from review text) ratings, using SVM model

Content-Based Filtering.ipynb

- To characterize a restaurant, we represented the city, category and attribute features by one-hot encoding
- Generated a feature union using FeatureUnion of sklearn
- Created a profile for a user based on the restaurants she/he had reviewed.
- Given unreviewed restaurants, calculated the similarity between the unreviewed restaurant and the user's profile using cosine similarity
- Recommended the top 10 restaurants to the user

Collaborative Filtering .ipynb

- Got the unique users and restaurants from the dataset
- Splitted the dataset into train and test set (80:20) using train_test_split of sklearn.
- Created a user-item matrix (for train and test data), which consists rating of each user-item pair
- Created a user-user matrix(user_similarity) which consists of a similarity score for each pair of user using pairwise_distances of sklearn
- Created an item-item matrix(item_similarity) which consists of a similarity score for each pair of restaurant using pairwise_distances of sklearn
- We calculated the predicted rating for each user-item pair for both the user-based and item-based approach.
- Evaluated the models using RMSE and MAE

Results

For Rating Prediction using review text:

Model	Feature	Precision	Recall	Accuracy	Number of Classes
Naive Bayes	Unigram + TF	62%	66%	66.42%	5
Naive Bayes	Unigram +TF	88%	88%	88.29%	2
Linear SVM	Bigram + TF-IDF	65%	69%	69.12%	5
Linear SVM	Bigram + TF-IDF	92%	93%	92.62%	2

For Recommendation models:

Model	RMSE - Testing Data	RMSE - Training data	MAE - Testing Data	MAE - Training Data	Rating used
User based collaborative filtering	3.422425	3.405979	3.101183	3.086098	Biased rating (rating from the dataset)
Item based collaborative filtering	3.424497	3.406809	3.103325	3.086804	Biased rating (rating from the dataset)
User based collaborative filtering	3.393075	3.413332	3.073444	3.093057	Unbiased rating (rating from the rating prediction task using Linear SVM)
Item based collaborative filtering	3.395080	3.414164	3.075493	3.093765	Unbiased rating (rating from the rating prediction task using Linear SVM)
User based collaborative filtering	3.572904	3.567457	3.226867	3.219094	Unbiased rating(rating from the rating prediction task using Naive Bayes)
Item based collaborative filtering	3.575060	3.568335	3.229085	3.219826	Unbiased rating(rating from the rating prediction task using Naive Bayes)

Discussion

Predicting Ratings:

Evaluation Metrics

We use Precision and Recall as the evaluation metric to measure our rating prediction performance. SVM has better performance than Naïve Bayes, as a naive Bayes classifier simply assumes that the value of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. SVM on the other hand is primarily a classier method that performs classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels.

Tf.idf with bigrams is performing better. These results are intuitively aligned to the observation that that we need to factor in phrases like ‘not great’, ‘not bad’ to understand the sentiment of the review. SVM has a better performance than Naïve Bayes for predicting ratings from reviews.

Recommender System:

Recommender systems typically produce a list of recommendations in one of the two ways - through collaborative or content-based filtering. Collaborative filtering approaches build a model from a user’s past behaviour (items previously purchased or selected, and/or numerical ratings given to those items) as well as similar decisions made by other users; then use that model to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties.

Evaluation Metrics

To evaluate a recommender system, we need metrics that can measure how close the estimates are to the actual preferences. For testing a particular prediction value, we need to compare them with the actual preference value of that user. But actual preference value doesn’t exist, because nobody knows for certain how they might like some new items in the future. So we can simulate this by setting aside a small part of the real data set as test data. These test preferences are absent in the training data. Then the recommender is asked to estimate preferences for the missing test data, and estimates are compared to the actual values.

We chose the root mean square error (RMSE) and the mean absolute error (MAE). Both these metrics are highly accurate for measuring the effectiveness of a recommender system. MAE is the absolute value of the difference between the predicted value and the actual value. It tells us how big of an error we can expect from the prediction on average. In MAE, we may understate the impact of big, but infrequent, errors. To adjust for such large, rare errors, we calculate the Root Mean Square Error. By squaring the errors before we calculate their mean, and then taking the square root of the mean, we arrive at a measure of the size of the error that gives more weight to the large but infrequent error than the mean.

Future Work

- To use parts-of-speech in feature selection process to differentiate between the same word features that are used as different parts-of-speech while building the rating prediction model. We can keep reducing the feature dimension, using top n keywords instead of all the keywords. In Naive Bayes, we can also try different smoothing parameters.

- To build a novel approach that combines content-based filtering with collaborative filtering to arrive at more relevant recommendations.

- To build a group recommendation system. This system can consider the information of all individuals in a group and recommend restaurants that satisfies the group of users.

Conclusion

In conclusion, we have experimented with various feature selection techniques (like tf, tf.idf) and various supervised learning algorithms to predict star ratings of the Yelp dataset using review text alone. We evaluated the effectiveness of different algorithms based on precision and recall measures. We concluded that Linear Support Vector Machine Classifier with TF. IDF and bigram feature is found to give the best results.

We extended our model to incorporate rating of restaurants predicted as features to perform user restaurant recommendations by clustering. We implemented content-based and collaborative filtering. Both the models work well. But, we realized that content-based leads to over-specialization i.e. recommended restaurant is like already visited/reviewed restaurant and may not be useful for the user. Whereas collaborative filtering relies on past preferences or rating correlation between users. However, this technique can lead to bad prediction if the restaurant is unpopular and very few users have reviewed that restaurant.So, a hybrid model that considers the aspects of both content-based and collaborative filtering should give the best results.

References

[1] *Yelp Challenge Presentation: http://www.ics.uci.edu/~vpsaini/*

[2] *http://www.ics.uci.edu/~vpsaini/files/technical_report.pdf*

[3] *Scaria, Aju Thalappillil, Rose Marie Philip, and Sagar V. Mehta. "Predicting Star Ratings of Movie Review Comments."*

[4] *https://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/017.pdf*

[5] *Chada, Rakesh, and Chetan Naik. "Data Mining Yelp Data Predicting rating stars from review text."*

[6] *Li, Chen, and Jin Zhang. "Prediction of Yelp Review Star Rating using Sentiment Analysis."*

[7] *https://nycdatascience.com/blog/student-works/yelp-recommender-part-1/*

[8] *https://cambridgespark.com/content/tutorials/implementing-your-own-recommender-systems-in-Python/index.html*

[9] *Arun Babu, Rahool Arun Paliwal and Syamsankar Kottukkal. "Content-Aware Collaborative Filtering for Yelp Restaurant Recommendation"*