From Reviews to Ratings: A model to predict ratings based on review text

Sanjeev Shenoy
sshenoy@eng.ucsd.edu

Anwaya Aras
aaras@eng.ucsd.edu

User generated ratings have a profound effect on the success of a business on popular websites like Yelp. While a particular business may have an averaged rating associated to it, it doesn't necessarily abide by the tastes of all users. Although some users may find the business particularly outstanding, it might not cater to another user's tastes. Besides being related to user preferences, such varied inclinations might be related to a businesss ratings on hidden topics.
In this project, we predict the rating for a given business customized to a given user. We present two methods for predicting ratings. The first one involves clustering the users probabilistically, based on hybrid features that seem the most important in determining a users interest. The second method makes use of LDA in extracting latent subtopics from review texts to determine the star rating. Finally, we present our findings, compare our model to the existing state of art techniques and explain the results thus obtained.

DATASET

Description:

With over nine years of operation, Yelp has amassed large amount of raw business data that was used for our experiment. The dataset consists of json files where each file is composed of a single object type, one json-object per-line. Data was available for business, reviews, checkin, user and tips. Our project involved working with the business, user and reviews file. Each of the files has been briefly described as follows:

Business data:
Data points: 61,184
Data Shape:

```
{
    'type': 'business',
    'business_id': (encrypted business
       id),
    'name': (business name),
    'neighborhoods': [(hood names)],
    'full_address': (localized address),
    'city': (city),
    'state': (state),
    'latitude': latitude,
    'longitude': longitude,
    'stars': (star rating, rounded to
       half-stars),
    'review\_count': review count,
    'categories': [(localized category
       names)]
    'open': True / False (corresponds to
        closed, not business hours),
    'hours': {
        (day_of_week): {
            'open': (HH:MM),
            'close': (HH:MM)
        },
        ...
    },
    'attributes': {
        (attribute_name): (
           attribute_value),
        ...
    },
}
```

Properties: Business data spanned across multiple neighborhoods,cities and countries. The following table would give a better understanding of the data.

Feature	Number	Example
Neighborhoods	175	Eastland, West View, Carrick, East Carnegie, Schenk - Atwood
Cities	378	Seattles,LAS VEGAS,Saint-Laurent,Florence
States	26	WA, NC, PA, QC
Categories	783	Advertising,Gay Bars,Cafes,Mexican,Spanish,Bike Repair/Maintenance, Beer Wine & Spirits
Attributes	38	Alcohol,Dietary Restrictions, Happy Hour,By Appointment Only, Good For Kids

User Data: Data points: 366,715
Data Shape:

```
{
    'type': 'user',
    'user_id': (encrypted user id),
    'name': (first name),
    'review_count': (review count),
    'average_stars': (floating point
       average, like 4.31),
    'votes': {(vote type): (count)},
    'friends': [(friend user_ids)],
    'elite': [(years_elite)],
    'yelping_since': (date, formatted
       like '2012-03'),
    'compliments': {
        (compliment_type): (
           num_compliments_of_this_type)
           ,
        ...
    },
    'fans': (num_fans),
}
```

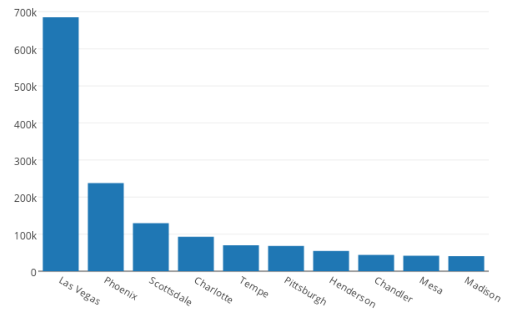
Reviews Data: Data points: 1569264
Data Shape:

```
{
    '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)},
}
```

EXPLORATORY ANALYSIS

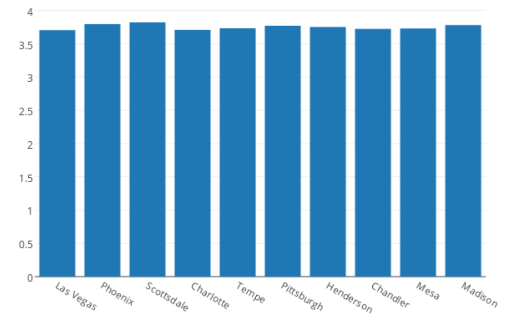
The yelp data set incorporated reviews from over 378 cities spanning over 26 states. To have an initial understanding of our data, we wanted to decipher any obvious biases towards a business rating. We try to understand the cultural influences in ratings by considering data across different cities and considering businesses with different attributes.

An interesting trend was observed when we tried to plot the number of reviews against cities. While the top cities had around 50-100k reviews, Las Vegas was an outlier with around 680k reviews. Looks like what happens in Vegas, doesnt stay in Vegas anymore!



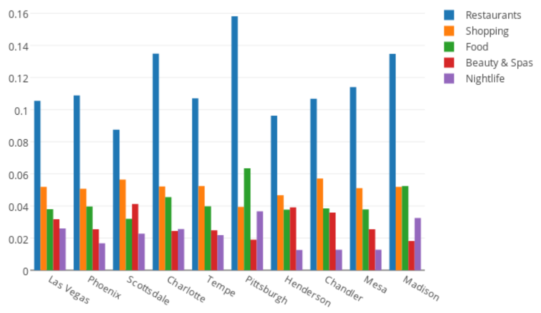
Average ratings per city

We didn't observe much variation across average ratings for every city.



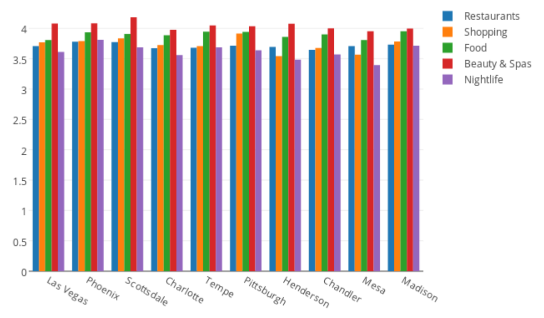
Top categories per city

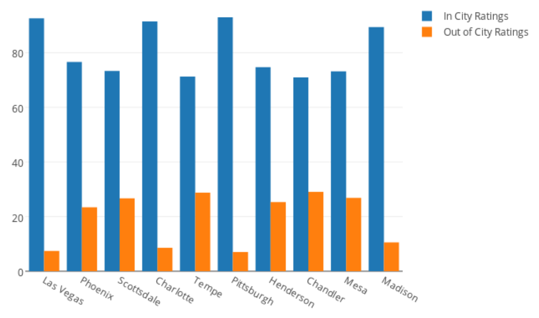
After going through all the businesses data we identified 5 major categories of businesses: Restaurants, Shopping, Food, Beauty & Spas and Nightlife.
We now compare the percentage of these businesses in each of the top ten cities:



This data gives us insights into what categories a city's residents prefer or what categories is a city good for.

While there are variations in how businesses of various categories are spread in different cities, a natural question to ask is how these categories are rated in such cities.



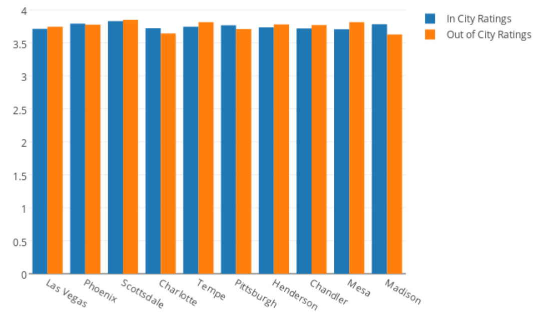


We observe small variations in the ratings given which indicate that ratings for businesses of different categories depend on the location.

But why does such a variation occur? Is it because of the quality of the businesses or because of the tastes of the residents of the cities. We compare how users rate businesses which are located outside their home city.

To determine the home city, we check all the businesses that a user has reviewed. If more than half of them are from the same city, then we conclude that this city is the user's city. If there is no such city, then we ignore that user.

The plot for the average ratings a user gives to businesses in his home city vs some other city is given below:



This does not seem to suggest a lot about how users rate based on location. We now compare how many reviews a user gives to businesses in his home city vs other cities.

PREDICTIVE TASK

Our goal is to predict what rating a user will give to a particular business. We have used a content-based predictor which uses the user's and the business's past reviews.

We have divided the review dataset into a 90:10 (training,test) dataset. Thus, once the features are entered, we use a supervised model to train our model on the training data. To evaluate the performance of our model, we compare the predicted ratings with the actual ratings.

We have used MSE to calculate the error.
If $\hat{Y}$ is a vector of n predictions, and Y is the vector of the true values, then the (estimated) MSE of the predictor is: $\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$.

Baselines

We use a naive model which simply predicts the average rating of the training data to compare our proposed models with a baseline.

PRIOR WORK

The dataset we have used is a part of the yelp dataset challenge. The dataset has been used to perform several predictive tasks such as ratings prediction using hidden factors[5], Improving Restaurants by Extracting Subtopics[1], and Inferring Future Business Attention[2].

The paper Improving Restaurants by Extracting Subtopics uses LDA to do subtopic discovery and therby generate subtopic ratings to help restaurants make inferences based on the idea that subtopic ratings indicate in what area the restaurant is doing good or bad. Our project includes LDA for all types of businesses. We use the topic distribution extracted as features for our prediction model.

The Amazon.com example

Amazon.com is a e-commerce website in which users can buy books, music and others goods. It has a databases containing more than 29 million customers and several million catalog items.

Amazon.com use a algorithm based on item-based collaborative filtering to make their recommendations. Their algorithm, called item-to-item collaborative filtering, works by first matching each of the users purchased and rated items to similar items (as with the item based CF, this is use to create an item-to-item matrix where elements are the similarities between items). Afterward, it combines those similar items into a recommendation list.

To improve the scalability and the performance, Amazon.com has built its recommender as two components. An offline component that creates the expensive and costly item- to-item matrix offline. The other component is the online component that look at the item-to-item matrix to produce the recommendations. The online component is dependent only on how many titles the user has purchased or rated [4].

Another paper uses a regression based technique on the review text to make predictions.[3]

TOPIC MODELING

While statistics give a certain approximation of a user's preferences, review texts provide exact information about the user's opinions. We hope to identify what the users care about when they write their reviews. This gives us insights into what a user looks for when he visits a business and what users notice the most about a particular business. However the review text is high dimensional data where the useful information density is very low. Thus, we reduce the dimensionality by adopting topic modeling. We have implemented this using LDA.

Latent Dirichlet allocation (LDA) is an unsupervised, generative model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. This model enables us to break the review text into latent subtopics.

In LDA, Each document di from a corpus samples (latent) topics from a multinomial distribution Mult(), and each topic z_i samples words in the vocabulary from a multino- mial distribution $p(w|z_i)$. Gibbs sampling, can be used to estimate the posterior probability on configurations of the model. A given configuration is an assignment $z < z1, z2, ..., zn >$, where each entry corresponds to the topic of a given word in the corpus. The full generative model for a simplified version of LDA is as follows:

$w_i|z_i, \phi^{(z_i)} \sim Discrete(\phi^{(z_i)})$
$\phi \sim Dirichlet(\beta)$
$z_i|\theta^{d_i} \sim Discrete(\theta^{d_i})$
$\theta \sim Dirichlet(\alpha)$

Implementation:

Initially the reviews are split into sentences. Stopwords are removed using the standard stopwords in the Natural Language Processing Toolkit. The parts-of-speech tags are extracted for all the remaining tokens. We have used lemmatization to look up the lemma of each noun and then store the lemmas. Lemmatization is used for the following reason: For grammatical reasons, documents are going to use different forms of a word, such as organize, organizes, and organizing. Additionally, there are families of derivationally related words with similar meanings, such as democracy, democratic, and democratization. In many situations, it seems as if it would be useful for a search for one of these words to return documents that contain another word in the set. The goal of lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

Finally, we add the 10,000 most frequently occuring words into our vocabulary. The number of topics chosen is 50.

Results: We get 50 topics with a word distribution for each of them. It is possible to assign some of the topics a meaningful representational name after considering the top words in that topic.

For example:
Topic 3: 0.106*egg + 0.093*breakfast + 0.068*juice + 0.045*fruit + 0.038*waffle + 0.030*morning + 0.028*hash + 0.022*sausage + 0.021*biscuit + 0.020*orange
is related to breakfast items.
Topic 7: 0.061*flavor + 0.054*cake + 0.049*chocolate + 0.039*dessert + 0.034*birthday + 0.031*sweet + 0.022*milk + 0.021*butter + 0.020*apple + 0.020*sample
is related to desserts.

FEATURE SELECTION

In this section, we describe the various features at hand, their relevance and our methodology for generating additional features. We identified three types of features for user clustering :

1. Simple features: Purely based on meta data
2. Enhanced features: Incorporating business preferences
3. LDA: Incorporating reviews data

1. **Simple features Model**:
   The first approach toward rating prediction was based on clustering the users based on the dominant features and then predicting rating based on a cluster preferences. The following features were chosen for clustering users:

   (a) Number of reviews (u['review_count'])
   (b) Average stars (u['average_stars'])
   (c) Number of 'funny' votes (u[val]['funny'])
   (d) Number of 'useful' votes (u[val]['useful'])
   (e) Number of 'cool' votes (u[val]['cool'])
   (f) User experience (u['yelping_since'])
   (g) Number of fans on yelp (u['fans'])

   Thus a user vector could be generated using the above features. However, it is important that the features selected improve the prediction quality of our data and reflect the properties of our sample space. A naive approach would be to sample over all possible subsets of existing features and consider the ones that minimize the test error. However, such an approach would require excessive computational prowess to run over 1.5 million reviews and would generate redundant and superfluous features.

2. **Enhanced features Model**:
   With the current features incorporated into user clusters, they do not give relevant information about a particular users preferences. To accurately predict the affinity of a user towards a business, we need to include features that reflect his tastes and interests. The enhanced features model, divides businesses into various clusters and includes the weighted preference of a user towards a business into consideration.

To segregate businesses, we performed clustering on the business based on the features that made them as separable as possible.

(a) City where the business is located: u['city']
With 378 cities in the existing data set, each business feature incorporated a 378 dimensional vector that contributed to its city. Although it was possible to reduce the dimensions of the vector, we though giving unnecessary weights to to one city over other would skew our results. Also, as indicated in the exploratory analysis section of this paper, the location of a business heavily contributes towards the ratings it receives. Thus, to make sure that the city features are completely independent of each other, we converted used a 378 dimensional vector to model the city for each business.

(b) State of the business: u['state']
The principle behind choosing a vector for a businesss state is similar to the city vector. With 28 states for our businesses, we have a 28 feature vector for every state that has been incorporated into the business vector

(c) Review count: u['review_count']
The number of reviews for a particular business indicates the level of interest circling it and hence provided meaningful insights to its significance on the yelp community. Thus, this particular feature was weighed twice as compared to the other features to incorporate its significance in our model.

(d) Number of stars: u['stars']
The number of stars for a business indicate it's general perception among users. While incorporating this aspect of our data, we modified the existing value of average stars provided to encompass the strength of user affinity. Strongly positive or negative ratings are enhanced while average ratings are mitigated. This gives us sharp distinction between the various excellent, good, okay, bad, horrible businesses. Alas! It's like how grading works at UCSD!

The following table indicates our relative weights that aggravate the star ratings for restaurants.

Average rating for the business	Indicated sentiment	Weight for our model
(0-1)	Strongly negative	10
[1-2)	Negative	6
[2-3)	Average	3
[3-4)	Positive	6
[4-5)	Strongly Positive	10

(e) Categories of businesses: u['categories']
Although we had 783 categories in the existing yelp data set, incorporating all of them would heavily fine grain into the data. Thus we identified following categories and reduced transformed the existing 783 dimensional vector into a 10 dimensional form.
C = {Food, Entertainment,Service, Fashion, Medical, Exclusive,Stores, Professionals,Bars and Breweries, Others}

The features are independent but an item could belong into multiple categories. The following table indicates an example of each.

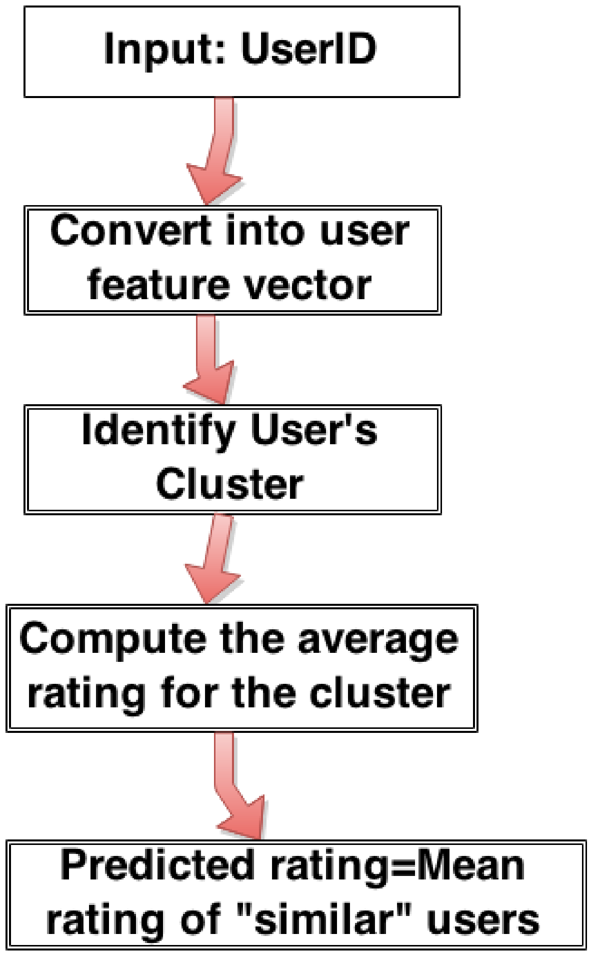
Subcategories of Categories	Example
Food	Spanish
Entertainment	Opera & Ballet
Service	Event Planning & Services
Fashion	Salons
Medical	Obstetricians & Gynecologists
Exclusive	Tattoo Removal
Stores	Beverage Store
Professionals	Psychics & Astrologers
Bars and Breweries	Brewing Supplies
Others	Aquarium

(f) Business Attributes: The 38 attributes of every business were modeled using a 38 dimensional vector. However, since the data contained large amounts of noise and features that weren't extremely, we ran a PCA on the data to include the features with maximum variance. Thus, the features were compressed to 10 principle dimensions that described most of the information included in the attributes section of our data.

3. **LDA: Incorporating reviews data**
After predicting topics for reviews with LDA, we get a probability distribution over the 50 topics. We use these as our features in predicting ratings.

PREDICTING RATINGS USING USER CLUSTERING:

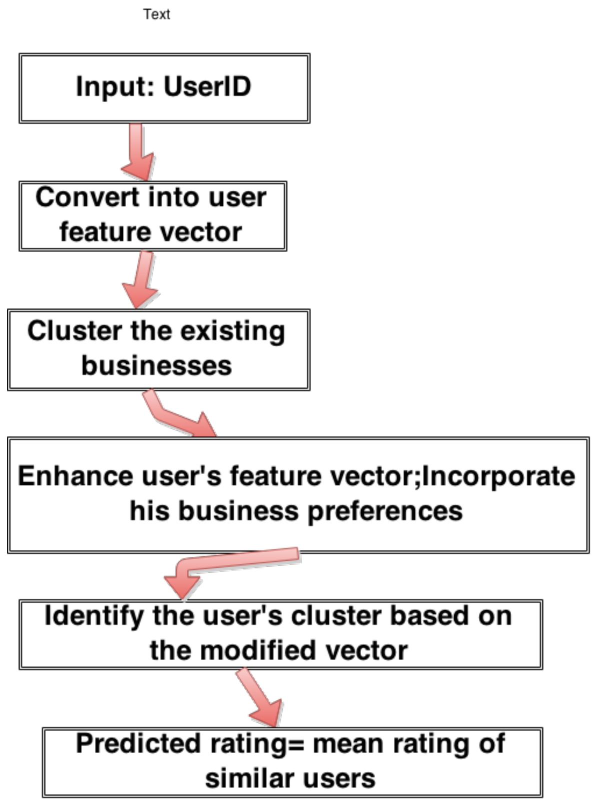
1. Simple Baseline Model:
Our initial naive model was based on the simple assumption that to predict the rating a user might give a restaurant, we just need to average the ratings given by similar users. This notion of similarity is captured by the probabilistic clusters assigned to each user. The working of our model is shown below

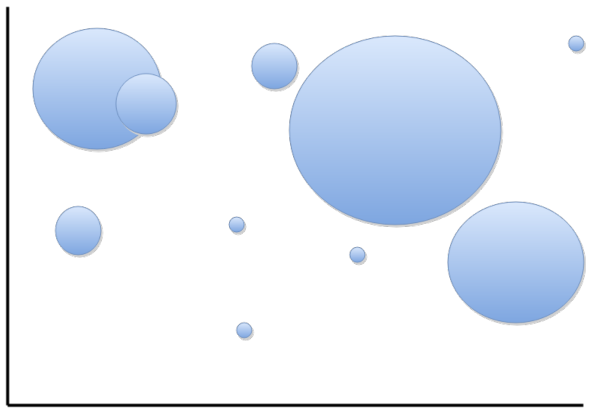

Input: UserID
Convert into user feature vector
Identify User's Cluster
Compute the average rating for the cluster
Predicted rating=Mean rating of "similar" users

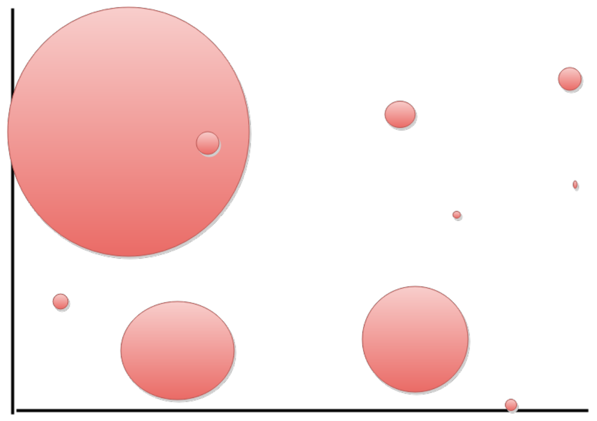
2. Enhanced features incorporated :

   This model clustered the businesses using K means clustering into 20 different types. To predict the rating for a given user say Ui and a given business say Bj, the model computed the cluster for both the user and the business. Then to predict the rating, the ratings for all the businesses in Bjs cluster as given by all the users in Uis cluster were averages and normalized against the number of businesses. Figure below describes the working of this model

Clustering Techniques used
We used the K means clustering technique to cluster both users and data. Both business and users were classified into 20 clusters each. The distribution of data for user clusters seemed to be somewhat uniform however the distribution of businesses was extremely skewed. With some business such as restaurants/bars/breweries had many more entries, a few others didnt. The data distribution across both clusters is shown in the two following figures. Since both user and business vectors are multidimensional, the two dimensions with the most variance have been included in the bubble plots.


Text
Input: UserID
Convert into user feature vector
Cluster the existing businesses
Enhance user's feature vector;Incorporate his business preferences
Identify the user's cluster based on the modified vector
Predicted rating= mean rating of similar users





PREDICTING RATINGS USING TOPIC MODELING:

Training and Testing data: We split the review data into reviews for each user. For each user we again split the reviews in a 90:10 (train,test) ratio.

We first try the naive approach where rating $= \alpha + \beta_u + \beta b$. This model assumes that the rating is only dependent on the user and business independently. The user has a constant weight β_u and the business β_b

Our second approach is an extension of the first one: rating $= \alpha + \beta_u + \beta_{ub}$. Here, a business has different weights associated with each user. For implementing this model, we only consider the reviews for one user at a time. We use the following linear model for prediction: rating $= alpha + w.B$ where B is a sparse vector indicating which business is being considered.

For our third approach, we try to create the model purely using topic data. We try to predict what rating a user will give to a business using feature vectors for the user and the business. These feature vectors are populated using the topic data. User features: We consider all the reviews posted by that user and predict the topic model using LDA for those reviews. We sum up the probability distribution over topics for all these reviews. We normalise these sums to get our 50-dimensional user feature vector. Business features are calculated using in a similar way using all reviews that are posted for that business. Instead of summing the probabilities, we sum the product of probability of a topic with the (rating-average rating) for each of the reviews.

The user features try to represent the topics the user generally is interested in/ notices first. Since we are taking sum of probabilities, features with higher weights will represent the topics which influence the ratings the most. When we pair a user with a business, intuition suggests that the probability of the user liking the business will be higher if the business is rated well for the topics the user cares about.

Now that we have our features, we try to fit a linear model to it. For each of the training data reviews, we have a (user,business,rating) tuple. Thus, our linear model looks like: rating $= \alpha + \beta_u + \beta_b + \sum_{i=1}^{50} (uf_i bf_i) w_i$
where uf_i is the i^{th} feature of the user vector and bf_i is the i^{th} feature of the business vector.

The third model requires us to do topic modeling over all the reviews. However, this process was too time consuming. The estimated time for this was 52hrs. Thus, we selected a subset of 100 random users and selected the union of all the businesses they visited. The reviews now required for the training and data were all the reviews posted by these 100 users and all the reviews posted for the businesses. After picking a good random set of users, we ended up with 1800 businesses and around 60,000 reviews.

The strengths for the first two models lies in the fact that they are computationally very easy to compute. However, the number of dimensions being so low, these models are very restrictive and can't represent the system adequately. The third model uses topic model data to capture the unique relation between a user and a business. The problem in this model lies in how the feature vector for the businesses are being calculated. The magnitude of a feature represents how much a particular topic is being talked about at a business and the sign represents the sentiment. Because of the product, this does not work particularly well when the number of ratings is small.

RESULTS

While our the predicted rating for our model was a real number, the actual rating given by a user is an integer. Hence, the results obtained on traditional classification ,metrics such as precision/recall were considerably. This was because a rating wasnt considered to be accurately predicted unless and until it gave an exact value as the given rating. For example, if the predicted rating was 4.1 and the given rating was 4, then it is considered to be wrongly predicted. Thus,we used a new other metrics for evaluating the model.

1. Correct,AlmostCorrect and Wrong.

To get a better idea of the model performance, we calculated 3 values:

Correct: Abs (Given rating - predicted rating) ¡ C1

Almost correct: Abs (Given rating - predicted rating) ¡ C2

Wrong: Everything else

Constant	Value	Interpretation	Example
C1	0.6	Correctly Classified	Actual=3, Predicted=3.5
C2	2	Almost Correctly classified	Actual= 3, Predicted =4
C3	N/A	Wrong	Actual= 4, Predicted= 2

This metric is derives inspiration from the L1 norm and MSE techniques but tries to evaluate the problem at hand in a fairer sense. Also,it takes into account that a predicted rating that is close enough to the actual rating can be labelled as correct. Additionally, the metric can be fine tuned as per the expected classification performance and thus generalized to regression tasks.

In our project, weve used the values of the constants as indicated in the above table.The performance of each of the model is judged based on the computed values of the three metric measurements. The results obtained were have been shown in table below

Model	Correct Ratio	Almost Correct Ratio	Wrong ratio
Simple Features	0.67281	0.24968	0.07749
Enhanced features	0.79724	0.23316	0.05958
LDA features	0.68238	0.25673	0.00602

Significance of the results:

The results gave some insightful results onto our model. While Enhanced feature model guessed 79% of the ratings, correctly (by correct, we mean correct as defined earlier), it did not perform as well as LDA feature model on the almost correct ratio. Although we had expected the LDA model to perform better than the feature clustering models, it signifies the role of raw statistical features which were completely absent from the LDA model.

While trying to gauge the performance of the simple versus enhanced features model, we observe that incorporating the business related features indeed improved the performance by 10% over the simple feature model.

CONCLUSION

In this paper, we described three models to predict ratings for a business on the yelp data set. Each of these models had a combination of features generated from business statistics and raw review text. All three models were composed of mutually exclusive features :

1. The simple model included basic user specific features
2. The enhanced model included the inclination of users towards certain types of businesses
3. LDA incorporated the latent hidden topic features.

In our experiments, we analyzed all of these features to see their effects,influence on ratings and then attempt to infer the rating for a business customized to a user. We found that the model with business specific data performed considerably better as compared to the other two techniques.

In future work, we aim on generating more culture-specific, temporal and seasonal features to improve the accuracy of the model. We also plan on using a combination of the three models to generate ratings specific to the user data available. Additionally we plan to undertake a more sophisticated approach towards LDA by including features which were included in the first two models. Such a model however would require a large amount of training data to estimate the parameters.

REFERENCES

1. Yelp dataset challenge: Improving restaurants by extracting subtopics from yelp reviews.
2. Yelp dataset challenge: Inferring future business attention.
3. Ganu, G., Elhadad, N., and Marian, A.
4. Linden, G., Smith, B., and York, J. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE 7*, 1 (2003), 76–80.
5. McAuley, J., and Leskovec, J. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, ACM (2013), 165–172.