Latent Aspect Rating Analysis

Dushyant Kumar 150242
Siddhant Garg 150711
Priyadarshini Chintha 150206

Abstract

In this course project, we build an information retrieval system that suggest relevant hotels to the user according to his/her needs. The information about the hotel were derived from the reviews that are given by different users. First, we analyzed the opinions given by the reviewers in an online review about the hotel. We populated the list of seed words of five pre-defined aspects using bootstrap method and tried to predict the individual aspect ratings for these pre-defined aspects using Latent Rating Regression(LRR). We also predict the individual aspect ratings using Sentiment Analysis. We have also tried to integrate the ratings predicted by sentiment analysis in LRR model and predicted aspect ratings We then processed a free-text-query given by the user to obtain what all aspects the user is concerned about and then retrieve results based on ranking according to the "hotel-score", obtained using the ratings of these aspects.

1 Introduction

With the advancement of Web, people can freely express opinions on all kinds of entities such as products and services. These reviews are useful to other users for making decisions and to merchants for improving their service. But the number of reviews are growing so rapidly that it has become increasingly difficult for users to wade through numerous reviews to find the needed information. A user might be looking for a particular aspect in the entity of interest. For example, if the entity is hotel, some users might be looking for good rooms and food, and not concerned about the location around the hotel, or wifi-services, while some other person who is on a business trip might require good wifi service.

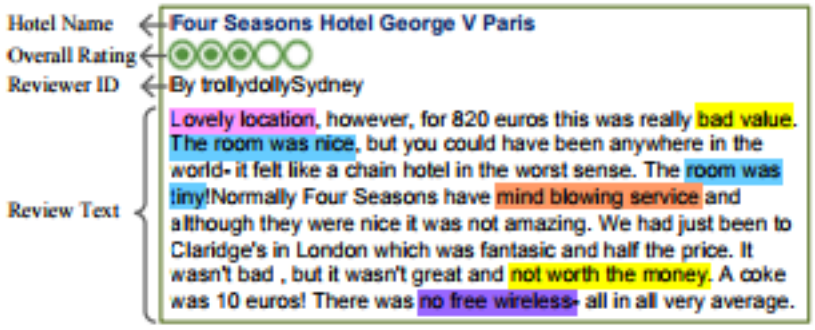

Figure 1: A Sample Hotel Review

Consider a hotel review as shown above. This review discusses multiple aspects of the hotel, such as price, room condition, and service, but the reviewer only gives an overall rating for the hotel; without

an explicit rating on each aspect, a user would not be able to easily know the reviewers opinion on each aspect. Going beyond the overall rating to know the opinions of a reviewer on different aspects is important because different reviewers may give a hotel the same overall rating for very different reasons. For example, one reviewer may have liked the location, but another may have enjoyed the room. In order to help users tell this difference, it is necessary to understand a reviewers rating on each of the major rating aspects (i.e., rating factors) of a hotel.

Individual aspect ratings are not available for every entity. Our aim is to build an information retrieval system for the user who is looking for particular aspect(s) in a hotel. We formulate a method through which we can assign rating to each pre-defined aspects of the hotel for every review and based on the user's necessity we retrieve the best options for the user. We discuss various methods in which we generate the ratings of predefined aspects by applying Sentiment Analysis and Latent Rating Regression methods [1] and retrieve the hotels as per user's interests.

We used the dataset available on TripAdvisor [3] [2]. We solve this in two stages. In the first stage, we employ a bootstrapping-based algorithm to identify the major aspects (guided by a few seed words describing the aspects) and segment the reviews into sentences describing those aspects. In the second stage, we used two methods to predict the aspect ratings. One is Sentiment Analysis and the other is generative Latent Rating Regression (LRR) model which aims at inferring aspect ratings and weights for each individual review based only on the review content and the associated overall rating. We have also predicted individual aspect ratings by integrating sentiment analysis in the LRR model.

2 Problem Definition

The project consist of two parts. In the first part we have to learn a model that predicts the ratings of predefined aspects from a given review. In the second part of the problem, we have to answer a user's query and retrieve the hotels relevant to him.

LATENT ASPECT RATING ANALYSIS

In this section, we formally define the problem of Latent Aspect Rating Analysis (LARA). As a computational problem, LARA assumes that the input is a set of reviews of some interesting entity (hotels, in our case), where each review has an overall rating.

Formally, let $D = \{d_1, \ldots, d_D\}$ be a set of review text documents for an interesting entity or topic, and each review document $d \in D$ is associated with an overall rating r_d. We also assume that there are n unique words in the vocabulary $V = \{w_1, \ldots, w_n\}$.

Overall Rating: It is the score, ranging between 0-5 that a user give to the hotel.

Aspect: It is an attribute/topic on which text is concentrated. An aspect is characterized by the set of keywords that descibes that aspect. For example, words like "price", "value", "expensive" describe the "Value" aspect of the hotel. Formally, aspect $A_i = \{w | w \in V, A(w) = i\}$, where A(.) is a mapping function from a word to an aspect label.

Aspect Ratings: Aspect rating $\boldsymbol{s_d}$ is a k dimensional vector, where the i-th dimension is a numerical measure, indicating the degree of satisfaction demonstrated in the review d toward the aspect A_i.

Aspect Weights: Aspect weight α_d is a k dimensional vector, where the i-th dimension is a numerical measure, indicating the degree of emphasis placed by the reviewer of review d on aspect A_i , where we require $\alpha_{di} \in [0, 1]$ and $\sum_{i=1}^{k} \alpha_{di} = 1$

Latent Aspect Rating Analysis (LARA Given a review collection D about a topic T where each review document d is associated with an overall rating r_d , and k aspects $\{A_1, A_2, \ldots, A_k\}$ to be analyzed, the problem of Latent Aspect Rating Analysis (LARA) is to discover each individual reviews rating $\boldsymbol{s_{di}}$ on each of the k aspects as well as the relative emphasis α_{di} the reviewer has placed on each aspect.

QUERY PROCESSING

Given a user's query $\boldsymbol{Q} = \{q_1, \ldots, q_m\}$, m is the number of query words, we have to separate the query into different groups such that each group describes a particular aspect. We define a single group as a "chunk" and the process as "chunking". Formally a chunk is defined as a set $< f; m_1, \ldots, m_k >$, where f is the aspect and m_i's are the modifiers that describes those aspects. Most of the times these modifiers are adjectives that are use to put weights on each aspect of the query by the user.

3 Approaches and Developments

Given a review and its overall rating, we want to generate the ratings for each pre-defined aspects and for each rating we also want to find the amount of weight that the user has given on individual aspects. We used two approaches to generate the aspect ratings for each review. In the first approach we assume that the ratings of each aspect is based on the sentiment of the user towards that aspect. We analyse the review and group the sentences into different categories. Each category of sentences describes a particular aspect of the hotel. We segment the review into sentences using the output **aspect keyword list** of the **Bootstrap algorithm**, which is described in the next section. The bootstrap algorithm requires a list of **Seed Words**, that describes a particular aspect and a set of reviews. **Seed Words list** is a list of words that describes a particular aspect. For example, words like "cheap", "price" and "expensive" describe the Value aspect of the hotel. By the end of Bootstrapping algorithm, each review is segmented into different categories of sentences and also the seed words is populated by appending more and more words in the list describing a particular aspect. Words are selected on the basis of $\boldsymbol{\chi^2}$-values. Detailed algorithm is explained in the next section. We trained a sentiment analysis classifier on a labeled dataset of hotel reviews where each review was marked as positive or negative. We used the trained model to predict the sentiment of the sentences of every aspect in the review. The predicted probability is scaled-up to five and assigned as rating for that aspect.

Although we get each aspect rating for a review, we do not get the weights assigned to each aspect by the user. Therefore, employ another generative method to compute the rating of the individual aspect namely, **Latent Regression Model**. Specifically, we assume that the reviewer generates the overall rating of a review based on a weighted combination of his/her ratings on all aspects, and the rating on each aspect is generated based on another weighted combination of the words in the review that discusses the corresponding aspect. After fitting such a two-fold regression model to all the review data, we would be able to obtain the latent aspect ratings and weights, thus solving the problem of LARA. The list of words that describe each aspect is obtained from the bootstrap algorithm.

Further we extended this model by integrating Sentiment Analysis of the reviews in predicting the aspect ratings for a particular hotel.

4 Latent Dirichlet Allocation

We have run the LDA model on the some of the hotel reviews for the aspect segmentation task. But the results were not that good. Following are the list of words that are obtained for various topics.

Topic 1	Hotel, Room, Breakfast, Seattle, Staff, Great, Clean, Good, Night
Topic 2	Room, Hotel, Stay, Nice, Airport, Area, one, Good, Pool, Night
Topic 3	Room, Hotel, Breakfast, Suite, Embassy, Area, Stay, Night, Staff, Nice

Table 1: Output list of topics words of LDA

We can see that the results obtained were not so good, because a particular of words cannot be put under a single topic heading. Therefore, we opt for more efficient method for aspect segmentation, called Bootstrap Algorithm in which the aspects were pre-defined.

5 Aspect Segmentation using Bootstrap Method

The goal of the aspect segmentation is to map the sentences in a review corresponding to each aspect. A few **seedwords** are given describing each aspect and then they are fed into the bootstrap algorithm to analyse the review and obtain the more related words used for those aspects.

Algorithm 1: Bootstrap Algorithm

input : A collection of reviews $\{d_1, \ldots, d_{|D|}\}$, a set of aspect keywords $\{T_1, \ldots, T_k\}$,vocabulary V, selection threshold p and iteration step limit I.
output: Reviews split into sentences with aspect assignments.
Step 0: Split all reviews into sentences, $X = \{x_1, x_2, \ldots, x_M\}$;
Step 1: Match the aspect keywords in each sentence of X and record the matching hits for each aspect i in Count(i);
Step 2: Assign the sentence an aspect label by a i = argmax i Count(i). If there is a tie, assign the sentence with multiple aspects.
Step 3: Calculate χ^2 measure of each word (in V);
Step 4: Rank the words under each aspect with respect to their χ^2 value and join the top p words for each aspect into their corresponding aspect keyword list T_i;
Step 5: If the aspect keyword list is unchanged or iteration exceeds I, go to **Step 6**, else go to **Step 1**;
Step 6: Output the annotated sentences with aspect assignments.

The χ^2 statistic of a term w and aspect A_i is defined as follows:

$$\chi^2(w, A_i) = \frac{C \times (C_1 C_2 - C_2 C_3)^2}{(C_1 + C_3) \times (C_2 + C_4) \times (C_1 + C_2) \times (C_3 + C_4)}$$

C_1 : The number of times w occurs in sentences belonging to aspect A_i.

C_2 : The number of times w occurs in sentences not belonging to A_i.

C_3 : The number of sentences of aspect A_i that do not contain w,

C_4 : The number of sentences that neither belong to aspect A_i , nor contain word w

C : The total number of word occurrences.

After aspect segmentation, we would get k partitions of each review d, and represent them as a$k \times n$ feature matrix $\boldsymbol{W_d}$, where $\boldsymbol{W_{dij}}$ is the frequency of word w_j in the text assigned to aspect A_i of d normalized by the total counts of words in the text of that aspect.

6 Sentiment Analysis

Recent years have seen rapid growth in online discussion groups and review sites (e.g.www.tripadvisor.com) where a crucial characteristic of a customers review is their sentiment or overall opinion for example if the review contains words like great, best, nice, good, awesome is probably a positive comment. Whereas if reviews contains words like bad, poor, awful, worse is probably a negative review. However, Trip Advisors star rating does not express the exact experience of the customer. Most of the ratings are meaningless, large chunk of reviews fall in the range of 3.5 to 4.5 and very few reviews below or above.

We want to use the Sentiment Analysis [4] to generate the sentiment oriented aspect ratings of a the hotel from a given review. We first divide the review into 5 (# of aspects) parts. Each part contains the sentences describing that aspect. We did this using the populated aspect words list after running the Bootstrap Algorithm. Review segmentation is similar to the one in Bootstrap, i.e.,we look for maximum number of aspect keywords in a sentence and tag the sentence with that aspect that contains maximum words of that aspect from the keywords list. Then each group of sentences is send to the Sentiment Analysis Classifier. The probabilities obtained were scaled upto five and are assigned as the sentiment-oriented aspect ratings for that review.

7 Latent Rating Regression Model(LRR)

So far we have obtained the seedwords corresponding to each aspect and also aspect segmentation results for each review. Using these segmentation results for each review we apply LRR model to analyze both aspect ratings $\boldsymbol{s_d}$ and aspect weights α_d for each review.

7.1 The General Assumption

Our assumption of reviewers rating behavior is as follows: to generate an opinionated review, the reviewer first decides the aspects he wants to comment on; and then for each aspect, the reviewer carefully chooses the words to express his opinions. The reviewer then forms a rating on each aspect based on the sentiments of words he used to discuss that aspect. Finally the reviewer assigns an overall rating depending on a weighted sum of all the aspect ratings, where the weights reflect the relative emphasis he has placed on each aspect.

7.2 The LRR Model

The reviewer for d would be assumed to first generate an aspect rating for each A_i as a linear combination of W_{di} and β_i, i.e.

$$s_i = \sum_{i=1}^{n} \beta_{ij} W_{dij} \tag{1}$$

where $\beta_i \in \mathbf{R}$ represents the word sentiment polarities on aspect A_i. The overall rating would then be $r_d = \alpha^T s_d = \sum_{i=1}^{n} \alpha_{di} s_{di}$ The overall rating is assumed to be a sample drawn from a Gaussian distribution with mean $\alpha^T s_d$ and variance δ^2, which indicates the uncertainty of the overall rating predictions.

$$r_d \sim N(\sum_{i=1}^{k} \alpha_{di} \sum_{j=1}^{n} \beta_{ij} W_{dij}, \delta^2) \tag{2}$$

Intuitively, the key idea here is to bridge the gap between the observed overall rating and the detailed text descriptions through introducing the latent aspect weight α_d and term sentiment weight β, which enable us to moodel the overall rating based on ratings of specific aspects. To capture the dependencies among different aspects, we employ a multivariate Gaussian distribution as the prior for aspect weights, i.e.

$$\alpha_d = N(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{3}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and variance parameters respectively. Combining Eq (2) and (3), we get a Bayesian regression problem. The probability of observed overall rating in a given review in our LRR model is given by:

$$\begin{aligned} P(r|d) &= P(r_d|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta, W_d) \\ &= p(\alpha_d|\boldsymbol{\mu}, \boldsymbol{\Sigma}) p(r_d| \sum_{i=1}^{k} \alpha_{di} \sum_{j=1}^{n} \beta_{ij} W_{dij}, \delta^2) \end{aligned} \tag{4}$$

where r_d and W_d are the observed data in review d, $\Theta = \boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta$ are the set of corpus-level model parameters, and α_d is the latent aspect weight for review d. Suppose we are given the LRR model parameters $\Theta = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta)$ we can apply the model to get the aspect ratings and weights in each review as follows: (1) the latent aspect rating s_d in a particular review d could be calculated by Eq (1); (2) we appeal to the maximum a posteriori (MAP) estimation method to retrieve the most probable value of α_d in the given review. The object function of MAP estimation for review d is defined as:

$$L(d) = \log p(\alpha_d|\boldsymbol{\mu}, \boldsymbol{\Sigma}) p(r_d| \sum_{i=1}^{k} \alpha_{di} \sum_{j=1}^{n} \beta_{ij} W_{dij}, \delta^2) \tag{5}$$

We expand this object function and associate all the terms with respect to α_d in each review.

$$\begin{aligned} \hat{\alpha_d} &= \underset{\alpha_d}{argmax} L(d) \\ &= \underset{\alpha_d}{argmax} [\frac{(r - \alpha^T s_d)}{2\delta^2} - \frac{1}{2}(\alpha_d - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\alpha_d - \boldsymbol{\mu})] \end{aligned} \tag{6}$$

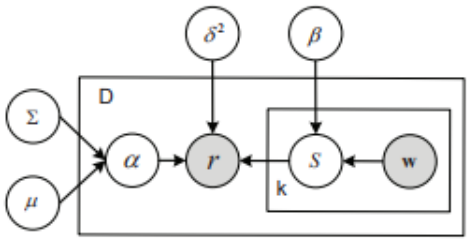

Figure 1: Graphical Representation of LRR. The outer box represents reviews, while the inner box represents the composition of latent aspect ratings and word descriptions within a review.

subject to

$$\sum_{i=1}^{k} \alpha_{di} = 1$$

$$0 \leq \alpha_{di} \leq 1 \text{for} i = 1, 2..k$$

To address above constraint non-linear optimization problem, we apply the conjugate-gradient-interior-point method with the following formula for the derivatives with respect to α_d

$$\frac{\partial L}{\partial \beta_i} = \sum_{d \in D} \Big(\sum_{i=1}^{k} \alpha_{di} \beta_i^T \boldsymbol{W}_{di} - r_d \Big) \alpha_{di} \boldsymbol{W}_{di}$$

7.3 LRR Model Estimation

We estimate the model parameters using the Maximum Likelihood (ML) estimator. The log-likelihood function on the whole set of reviews is:

$$L(D) = \sum_{d \in D} \log P(r_d | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta, W_d) \tag{7}$$

Thus the ML estimate is

$$\hat{\Theta} = \underset{Theta}{\operatorname{argmax}} \sum_{d \in D} \log P(r_d | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta, W_d)$$

To compute this ML estimation, we would first randomly initialize all the parameter values to obtain $\Theta^{(0)}$ and then use the following EM-style algorithm to iteratively update and improve the parameters by alternatively executing the E-step and then M-step in each iteration: **E-Step**: For each review d in the corpus, infer aspect rating s_d and aspect weight α_d based on the current parameter $\Theta^{(t)}$ (the subscript t indicates the iteration) by using Eq (1) and (6). **M-Step**:Given the inferred aspect rating s_d and aspect weight α_d based on the current parameters $\Theta^{(t)}$, update the model parameters and obtain $\Theta^{(t+1)}$ by maximizing the complete likelihood, i.e., the probability of observing all the variables including the overall ratings r_d and the inferred aspect ratings s_d and aspect weights α_d for all the reviews. First, we look at the case of updating the two parameters of the Gaussian prior distribution of the aspect weight α_d. Here our goal is to maximize the probability of observing all the α_d computed in the **M-Step**: for all the reviews, thus we have the following updating formulae based on the ML estimation for a Gaussian distribution.

$$\mu^{(t+1)} = \underset{\Theta}{\operatorname{argmax}} \sum_{d \in D} (\alpha_d - \mu)^T \Sigma^{-1} (\alpha_d - \mu)$$

Therefore $\mu^{(t+1)} = \frac{1}{|D|} \sum_{d \in D} \alpha_d$.

$$\Sigma^{(t+1)} = \underset{\Sigma}{\operatorname{argmax}} [-|D| \log \Sigma - \sum_{d \in D} (\alpha_d - \mu^{(t+1)})^T \Sigma^{-1} (\alpha_d - \mu^{(t+1)})]$$

Since α_d is assumed to be known, we can update δ^2 and β to maximize $P(r_d|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \beta, W_d)$ (defined in Equation 2). Solving this optimization problem, we have the following updating formulae:

$$\begin{aligned}\delta^{2(t+1)} &= \underset{\delta^2}{argmax}[-|D|\log\delta^2 - \frac{\sum_{d\in D}(r_d - {\alpha_d}^T s_d)^2}{\delta^2}] \\ &= \frac{1}{|D|}\sum_{d\in D}(r_d - {\alpha_d}^T s_d)^2\end{aligned} \tag{8}$$

$$\beta^{(t+1)} = \underset{\beta}{argmax}\sum_{d\in D} -\frac{(r_d - \sum_{i=1}^{k}\alpha_{di}{\beta_i}^T W_{di})^2}{2\delta^{2(t+1)}} \tag{9}$$

The closed-from solution for β requires an inversion on a $|V||V|$ matrix, which is expensive to directly compute. To avoid this, we apply the gradient-based method to find the optimal solution of β with the following gradients:

$$\frac{\partial L(\beta)}{\partial \beta_i} = \sum_{d\in D}(\sum_{i=1}^{k}\alpha_{di}\beta_i^T W_{di} - r_d)\alpha_{di}W_{di}$$

The E-step and M-step are repeated until the likelihood value of Eq (7) converges.

7.4 LRR Model with Sentiment Analysis

In earlier section we saw that the $\boldsymbol{\beta_i}$ matrix was responsible for word sentiment polarities for aspect A_i and the rating vector $\boldsymbol{s}$ was obtained using

$$\boldsymbol{s_i} = \sum_{i=1}^{n}\beta_{ij}\boldsymbol{W_{ij}}$$

In this model we will add the sentiment oriented rating $\boldsymbol{r_i}$ for aspect A_i, obtained from the sentiment analysis above, to the $\boldsymbol{s_i}$. The new equation obtained:

$$\boldsymbol{s_i} = \frac{\sum_{i=1}^{n}\beta_{ij}\boldsymbol{W_{ij}} + \boldsymbol{r_i}}{2}$$

The overall rating was then generated from a Gaussian Distribution, using the weighted sum of α and $\boldsymbol{s}$, ie $\alpha^T\boldsymbol{s}$ as the mean and some variance δ^2.

$$r_d \sim N(\alpha^T\boldsymbol{s}, \delta^2)$$

We also initialised this δ^2 with the sample variance of the overall ratings of the hotels.

$$\delta^2 = \frac{\sum_{i=1}^{N}(r_{di} - \bar{r_d})^2}{N-1}$$

where r_{di} is the hotel rating, $\bar{r_d}$ is the sample mean of the overall ratings of all the hotels and N is the total number of hotels.

8 Query Processing

User's are allowed to give the free-text-query. The basic assumption taken is that the user will ask for only those aspects that are pre-defined here. We will parse the query and extract the aspects which the user is concern about.

The following steps involved in query processing :

- Tokenize the query by using nltk word_tokenizer
- POS tagging of tokens of the query.
- Noun phrase chunking, where we search for chunks corresponding to noun phrases.

In order to create noun phrase chunker, we defined the chunk grammar, consisting of rules that indicate how sentences should be chunked. We defined a simple grammar with a single regular expression rule. This rule says that noun phrase chunk should be formed whenever chunker finds any number of adjectives(JJ), adverbs(RB) followed by a noun(NN). Using this grammar, we create a chunk parser.

Consider the query
"Hotels with good room and very clean bathrooms and exception wifi."

The chunked output will be

Hotels
, [good room], [very clean bathrooms], [exceptional wifi]

The need to consider the adjectives and adverbs in noun phrase chunker is to know the emphasis that the user has put on different aspects.

Within each phrase we identify aspects and the adjective and adverbs associated to them. We then assign rating to each aspects based on adjectives and adverbs.

Aspects in example query :
good room - Room aspect, very clean bathrooms - cleanliness aspect, exceptional wifi - service aspect.

Rating assigned : Room - 3, cleanliness - 4, service - 5

The aspects that have not been talked about in the query assigned a default value of 2. And the overall rating has been assign default weight as 5.

9 Retrieval

Now that we have the aspect ratings of each hotel from the reviews and also from the user's query, we can retrieve relevant hotels to the user. For ranking the hotels we define a **Hotel Score** that is the weighted average of the aspect ratings and the overall rating of the hotel. The weights used, are the query aspect weights that were calculated in the previous section. They have been normalized before calculating the hotel score. The assignment of default weights is described in the retrieval section.

$$\boldsymbol{H} = \frac{\sum_{i=1}^{6} w_i A_i}{\sum_{i=1}^{6} w_i}$$

,

where w_i are the normalized query weights and A_i is the i^{th} aspect rating. Note that we assigned default weights to the aspects that were not mentioned in the query. The main goal of this assignment was to consider the ratings of those aspects too in the ranking. The overall rating was given the highest default weight to rank the best hotel higher.

10 Experiment and Results

10.1 Dataset and Preprocessing

We evaluated our models on a hotel data set crawled from **TripAdviser** (www.tripadvisor.com). The data consist of the large number of reviews and for each review the individual aspect ratings are

available that serves as the ground truth. TripAdvisor provides the ratings of 7 pre-defined aspects namely Overall rating, Value aspect rating, Rooms aspect rating, Location aspect rating, Cleanliness aspect rating, Check in/front desk aspect rating, Service aspect rating and Business Service aspect rating. We only used five of them in our project - **Overall**, **location**, **cleanliness**, **room**, **service** and **value** aspects. We merged the Service and Business Service and Check in/front desk into into one aspect namely Service.

We did a simple pre-processing on the data. We chose total of 58,288 reviews from 500 hotels. The pre-processing have the following steps:

1. Converting words into lower cases
2. Remove the stop words except 'not', or 'no' and other words that negate the sentence, because they we changing the polarity of the sentence and it was essential that our learned their weights.
3. Remove all the punctuation available in python "nltk" library.

10.2 Bootsrapping

We fed the following list of aspect seed words to the Bootstrap algorithm. The Bootstrap algorithm runs for the time till the aspect key word list stop changing or maximum of 10 iterations are completed. We took top 5 words for every aspect which have highest χ^2 values.

Aspects	Seed Words
Value	value, price, cheap, overpriced, expensive, inexpensive, costly, worth, quality
Room	room, bed, space, small, bathroom, bedroom, big, spacious, comfortable, luxurious, suite, deluxe
Location	location, locate, scenery, warm, beach, traffic, area, quite, outskirts, restaurant
Cleanliness	clean, smell, bad, dirty, old, dusty, hygiene, maintain
Service	manager, staff, helpful, knowledge, food, breakfast, desk, clerk, lunch, dinner, friendly, buffet

Table 2: Aspect Seed Words

After Bootstrapping Algorithm we get the a newly populated aspect keyword list. We just gave the new words that were added to the list.

Aspects	New Words Added
Value	"reason", "well", "money", "wizz", "wew", "hotel", "room", "stay", "not", "n't"
Room	"luxurious", "suite", "clean", "dirty", "size", "great", "deluxe", "quite", "attractive", "fabulous", "locat", "time", "'s", "us", "hotel", "stay", "staff", "good", "night"
Location	"traffic", "hotel", "night", "staff", "good", "nice", "breakfast", "bathroom", "room", "not", "n't", "time"
Cleanliness	"dusty", "hygiene", "room", "great", "year", "musti"
Service	"people", "pleasant", "hotel", "stay", "room", "not", "night", "pool", "bathroom", "view", "beach", "bedroom"

Table 3: Aspect words by Bootstrap Algorithm

10.3 Hotel Rating Prediction Using Sentiment Analysis

The Classifier that we used for sentiment analysis is the **Probabilistic Naive Bayes Classifier**. The dataset that was used to train the classifier was scrapped from Booking.com. It contains 515k hotel reviews that are labeled as positive or negative. The classifier was first trained on this data. Then, for predicting the sentiments of each aspect on our hotel reviews, we first divided the review into 5 groups. Each group contains the sentences that describes a particular aspect. This segmentation was done using the aspect keyword list outputted from the bootstrap algorithm. We treated each group as a sub-review and fed it into the trained classifier. The probabilities obtained were scaled-up to five. Below is an example of a segmented review along with the actual and predicted ratings.

Aspect	Sentences	Actual	Predicted
Value	1. I did not find any other hotel cheaper with that kind of standard. 2. I think it worth the price I pay.	3.91	4.05
Room	1. The room was a little small, probably because we had a weirdly shaped room near the elevators. 2. Room was Immaculate, bed comfortable, heating / hot water worked well.	3.72	3.05
Cleanliness	1. Room and Bathroom were not as clean as I'd like to see. 2. The suites were clean	4.15	2.33
Service	1. The majority of the staff are very friendly, however, we did encounter several that couldn't even say hello. 2. oh and bring your own wine bottle opener, one of the many things they have overlooked and are unprepared to help you with.	3.65	2.72
Location	1. It is conveniently located right near the corner of William and Wall Streets. 2. The hotel is close to the subways.	3.98	1.58

Table 4: Caption

10.4 Hotel Rating Prediction Using LRR

The LRR model just takes the whole review as the input and generates a rating vector along with the weight vector. The following results were obtained for the same review discussed above.

Aspects	**Actual Ratings**	**Predicted Ratings**
Overall	3.4	1.38
Room	4.33	2.32
Value	4.25	1.42
Location	4.66	4.99
Service	4.25	2.78
Cleanliness	4.75	1.30

10.5 Hotel Rating Prediction Using LRR with the Sentiment oriented aspect

Aspects	**Actual Ratings**	**Predicted Ratings**
Overall	3.4	2.89
Room	4.33	3.43
Value	4.25	3.26
Location	4.66	3.45
Service	4.25	4.99
Cleanliness	4.75	3.01

Retrieval Results

The results obtained for a query are the following:

```
Query given by user...........
find hotels with good rooms and very clean bathrooms


Chunked Query........
[['find', 'hotels'], ['good', 'rooms'], ['very', 'clean', 'bathrooms']]


Query Aspect.........
[u'room', u'cleanliness']


Query Ratings.........
{u'cleanliness': '5', u'room': '4'}


Retireval Results........
```

Hotel_ID	room_rating	cleanliness_rating	service_rating	value_rating	location_rating
578849	4.99	5	4.01	5	2.61
595204	4.99	5	2.43	5	2.25
232822	5	5	1.39	4.08	1
183911	4.99	5	3.41	4.96	4.94
606503	4.99	5	4.83	4.48	3.49
1186536	5	5	4.68	1	1.68
281778	5	5	2.12	1	1.25
628099	5	5	1	4.99	2.95
142098	4.99	5	5	1	4.09
1222646	4.97	4.99	2.58	5	2.79
536111	4.97	5	4.63	4.83	4.53
86954	5	4.97	5	3.68	3.29
81150	4.97	4.99	4.23	4.81	4.72

Figure 2: Retrieved Hotel id's for the given query

11 Conclusion

We saw that the LDA did not give the appropriate topics for segmenting the reviews for finding the aspect ratings of the hotel. We then used the Bootstrap Algorithm by giving the pre-defined aspects and the seed words that describe those aspects. We first used the Sentiment Analysis model to predict the aspect ratings of the reviews. Then we used the Latent Regression Model for the hotel aspect ratings. We saw that results of the above model were not that good. We then integrated both the LRR model and sentiment analysis model. We saw the results improved slightly over both the models.

References

[1] Hongning Wang, Yue Lu, and Chengxiang Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792. ACm, 2010.

[2] Hongning Wang, Yue Lu, and Chengxiang Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792. ACm, 2010.

[3] Hongning Wang, Yue Lu, and ChengXiang Zhai. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–626. ACM, 2011.

[4] Yinglin Wang, Yi Huang, and Ming Wang. Aspect-based rating prediction on reviews using sentiment strength analysis. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 439–447. Springer, 2017.

12 Appendix

We have taken the dataset from Trip Advisor on Hotel Reviews. http://times.cs.uiuc.edu/~wang296/Data/ We used the .json format of hotel reviews.

Data set for training sentiment analysis model is obtained from https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe/data. Download the Hotel_Reviews.csv file.

Download the hotelDict.txt and hoteltag_sent.pickle from the following url: http://home.iitk.ac.in/~dushyant/