Sentiment Analysis and Recommender System on Amazon Reviews

Harika Andugula, Shravanthi Bhoopalam Sudharshan, Thanmai Gajam, Thasleem Tajudeen

Computer Engineering Department
San José State University (SJSU)
San José, CA, USA
Email:{harika.andugula,shravanthi.bhoopalamsudharshan,thanmai.gajam,Thasleembanu.tajudeen}@sjsu.edu

***Abstract*—Recommendation systems are tools for interacting with large and complex information spaces. They provide a personalized view of such spaces, prioritizing items likely to be of interest to the user. In e-commerce (suggest to buyers articles that could interest them), recommendation systems are today unavoidable in our daily online journeys. Reviews on Amazon are not only related to the product but also the service given to the customers. In this project, we applied topic modeling to generate predominant topics from the reviews given by customers for all products to build a recommendation system based on topic similarity and compare the recommendations with standard collaborative filtering. We then extended our work on analyzing the sentiment of reviews in order to score items by topics mentioned in their reviews.**

I. Introduction

Amazon is one of the largest online vendor in the World. People often gaze over the products and reviews of the product before buying the product on amazon itself. With the increase in anonymous customer-generated content, efforts must be made to understand the information in the correct context, and develop methods to determine the intent of the author. With this intent, we could analyze the sentiments of the reviews to better understand the opinion and emotions of customer's purchase experience.

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body.

The purpose of this paper is to investigate the sentiments of customers purchasing items on Amazon and generate topics related to all the products using topic modeling, thus, with the help of topics generated we build a recommendation system. We also build recommendation system using only user ratings for products from the scale 1.0 to 5.0. Recommendations are provided for users and items. A comparative analysis of the recommendation system using traditional methods and using topics modeling gives us a good picture of performance. We also perform sentiment analysis on the topics generated to find the highlights and opinions of customers feedback

II. Background

Much work has been done analysing statistical data from the subset of Amazon review data set. A large number of recommendation systems are developed using different techniques. AI based procedures from collaborative filtering through matrix factorization [1] have proved to be more successful in producing desired results but, one of the drawbacks of collaborative filtering is, it is limited by cold start problem where the recommendations can't be made for new items with no rating history. To achieve our project goals we researched and read few papers related to LDA nad Recommendation systems. The Latent Dirichlet Allocation (LDA), the model used for training is proposed by this article [2] described LDA as a generative probabilistic model of a corpus whose idea is that the documents are represented as weighted relevancy vectors over latent topics, where a distribution over words characterizes a topic. LDA is being applied in various Natural Language Processing tasks such as for opinion analysis, for native language identification. This paper [3] proposed a tag recommendation system using LDA to improve search. This article [4] shows how to infer topic similarity of topics generated from LDA.

III. Methodology

Our first objective is to build a recommendation system using LDA topic similarity and compare the recommendations with collaborative filtering using ALS. Our next goal is to do sentiment analysis on customer reviews to score topics generated from LDA and to pinpoint the highlights.

A. Recommendations using LDA Topic modeling

Latent Dirichlet Allocation is a type of unsupervised learning algorithm in which topics are inferred from a dictionary of text corpora whose structures are not known (are latent). Gensim package has been used for LDA implementation as it offers excellent APIs for training LDA with many parameters that can be tuned for specific case. The module [5] also allows to get inference of topic distribution on new, unseen documents. Inference is run on all the reviews for each product to find the most representing topic. Gensim offers an API called similarity [6] which computes similarities across a

collection of documents in the Vector Space Model. After the products were represented by probabilities of topics, Cosine similarity was calculated to find similar items based on topics. Cosine similarity was measured between bag of words which gave similarities in the range of -1 to 1 (the greater, the more similar).

B. Sentiment Analysis on LDA Topics

A product's rating is often considered by the customer's review rather than the rating given by user explicitly in the scale of 1 to 5. In this project we are experimenting to analyse the reviews for a given product and present how much content of the review is talking positive or negative or neutral with respect to the topic for that product. These topics are generated from the LDA model which was explained above. In addition to this, LDAMallet model (another version of LDA model) is applied to generate more accurate topics. Once the topics are generated, which will be detailed in later sessions, sentiment analysis is performed on the reviews that has these topics. Sentiment analysis is generally a process of determining whether the sentence or collection of words is positive , negative or neutral. We used VADER(Valence Aware Dictionary And sEntiment Reasoner) which is a rule-based library for sentiment analyzer [11]. This library can capture the essence of the word in the way it is used (annotated, bold or in quotations etc.,) by human. It gives not only the positive , negative and neutral scores. It also gives compound score which is the normalized form of these scores in the range of -1 and 1. From this attribute we can depict how positive a given sentence actually is. We also tried TextBlob's polarity score which is a score between -1 to 1 to assign positive and negative sentiments to sentences.

C. Recommendations using ALS

A recommendation system is built for Amazon ratings for products by customers using collaborative filtering approach. Two methods are implemented using collaborative filtering, user based and item based. The alternating least squares algorithm is used through out the industry of recommender systems. The ALS algorithm lies within a method for recommending called Collaborative Based Filtering(CBF). Formally, CBF models from a users past activity and behavior, then tries to recommend an item based on the similarity to other users. Within the CBF, the approach taken by the ALS is one known as Matrix Factorization models.

For understanding matrix factorization approach, we represent our data as a (very) large matrix A, say of dimensions m,n. We impose no size on these, but the usual case is that the number of items (artist) outnumbers the number of users, and generally m¡¡n. In our case rows of A represent users, and the columns of A correspond to the artist. So we have that Aij the (i,j)th entry of the matrix depicts the the number of times the ith user played the jth artist. The goal of Matrix Factorization Models is to approximate A with two smaller matrices, U and V, each of these matrices representing the rows and columns of A respectively (users and artist). Formally U will be a matrix of users u each described in k-dimensions. Similarly V will be the matrix of artist described in k-dimensions.

$$U_{k\times m} = \begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ u_1 & u_2 & \cdots & u_m \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix} \quad V_{k\times n} = \begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ v_1 & v_2 & \cdots & v_n \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix}$$

The vectors ui and vi are called factors (sometimes latent factors), moreover k is usually picked to be something small, and certainly k¡¡m,n. Here k represents features we believe associate each user to the artist, and so entries in the row vectors of U and V express how much association each has with the k features. We then predict the rating of user ui of artist vj to be $r_{ij} = u_i^T v_j$ And so our aim to is to approximate (and complete) $A \approx U^T V$. Hence, the best parameters are found.

IV. Data Analysis

A. Data Collection and Preparation

The amazon review data is freely available online [8]. For this analysis we used a dataset specific to the Home and Kitchen category. Initially we explored the complete review data which had 21million entries, later we decided to work on a smaller subset which had 7million entries, due to technical difficulties faced during processing the huge amount of data. The dataset has following fields reviewerID - the id of the user, asin - Amazon product id, reviewerName - name of the user, vote - the number of times the review was voted helpful, reviewText - the actual content of the review, overall - the rating of the product ranging from 1 to 5, summary - the title of the review, unixReviewTime - the time of the review in Unix format, reviewTime - the time of the review, style -a dictionary of the product metadata, e.g., "Format" is "Hardcover" , image - images that users post after they have received the product. We also used metadata to infer titles for the product ids to make better sense of the recommendations.

B. Data Cleaning

The freely available Amazon User Review dataset was originally in JSON format. The data was parsed and loaded to pandas dataframe and the following steps were performed.

1) The column name 'vote','style','image' does not provide much information for this project and 'reviewerName' is not required since the dataset contains 'reviewer id'.
2) Drop all the unverified reviews.
3) Convert reviewtime to date format.
4) Find all duplicate entries (I.e If the reviewer has reviewed the same product more than once, Keep the latest review).
5) Check for null values remove entries where both summary and reviewtext are nan.
6) Find all the url in summary text and replace them with ".
7) Combine Review text and summary column and drop summary column and reset index.
8) Find all the url in review text and remove.
9) Check for null values.

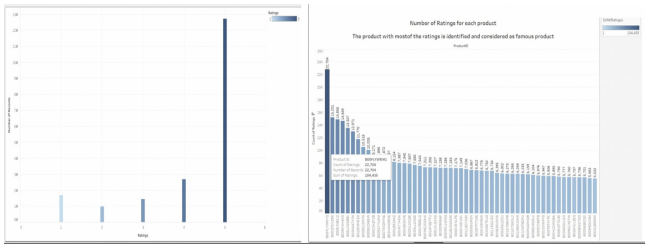

Fig. 1. Rating Distribution and Ratings of products

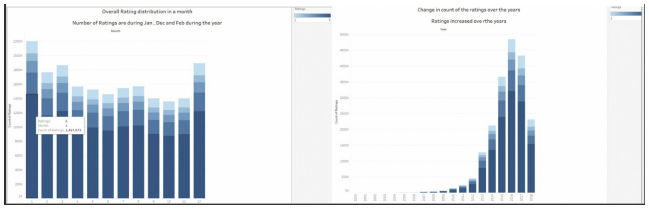

Fig. 2. Ratings over Month and years

After performing all the preprocessing steps we ended up having a dataset with 6053296 rows 5 columns which we will be using for Exploratory data analysis and topic modeling.

C. Data Visualizations

The initial data exploration is visualized using tableau as it can accommodate huge datasets. Some of the visualizations are explained below.

- The overall rating distribution is shown in figure **??**. From this graph we can infer that most of the ratings are biased towards rating 5 making them positive.
- All the products are sorted in decreasing order with respect to their number ratings. By this we can get top products with highest number of reviews, and planning to apply topic modeling and sentiment analysis on these products.
- The number of ratings for all the products has changed over the years.
- The number of ratings are plotted against month of the year and it can be inferred that most of the products are viewed and rated in the months of January , February and December. In all the plots representing ratings, color coding is applied based on the actual rating.

V. Experiments and Results

A. Recommendation system using LDA

The idea here is to generate meaningful topics from the review comments and recommend items based on similarity or by querying the model. Because of the high computation cost, HPC cluster has been used for this part. The intention of using LDA for recommendation is to solve the cold-start problem present in the standard recommendation system, whereby analytics can be done on texts to derive similarities in the dictionary's corpus.

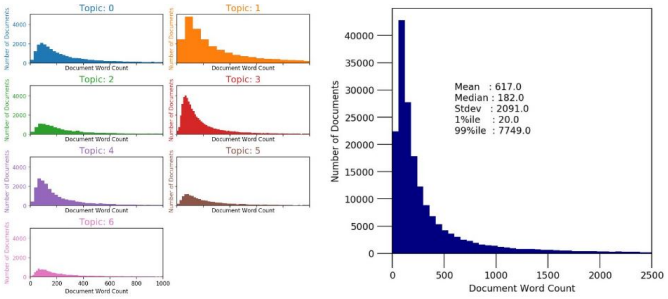

Fig. 3. Distribution of Document word counts

1) *Preprocessing the data for LDA input*
   - The dataset was grouped by product ids such that all the reviews for each product ids are in one cell
   - Review comments of each item id was considered as a document for LDA
   - The review texts were converted to lowercase and tokenized using nltk library.
   - Numeric tokens without characters were removed.
   - Tokens which were less than 4 characters were removed.
   - Stopwords such as 'during', 'doing', 'you', 'or', "they'll", 'after', 'the', "weren't" were removed using nltk stopwords library.
   - Tokens were lemmatized to reduce inflectional forms, Lemmatization was preferred over stemming because of its ability to provide meaningful words.
   - Applying the above steps for all the reviewtexts generates a list of lists.(i.e Document term matrix).
   - Removed words that appear in less than 20 documents or in more than 10 percent of the documents.
   - Created a dictionary using corpora(id- term dictionary).
   - Gensims Dictionary method encapsulates the mapping between normalized words and their integer ids. This can be viewed using token id method.
   - doc2bow method was used to convert a document (a list of words) into the bag-of-words format (list of (token id, token count) tuples).

   Figure 3, shows the distribution of Document and word token counts and also it is understood that most of the documents contain less than 500 unique tokens.
2) *Training LDA*
   LDA transforms this bag of words counts into a topic space of lower dimensions. The important hyper parameters for LDA are number of topics and dirichlet parameters alpha and eta.Tuning them is bit tricky and each run takes a lot of time due to huge aount of data. The gensim LDAmodel offers an option called 'auto' for these two hyper parameters which automatically adjusts based on number of topics and documents. For Recommendations based on topics the baseline model with a number of topics 7 was used. Figure 4, shows the top 10 words that contribute to each of the topics generated. These

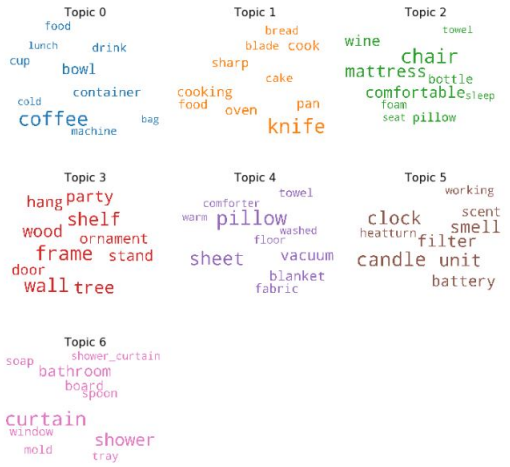

Fig. 4. Wordcloud of Top 10 words

```
#Sample recommendation from Topic 2
Product_recommender('B000HK03HI')
```

```
Your item's most prominant topic is:
0.026*"chair" + 0.019*"mattress" + 0.015*"comfortable" + 0.014*"wine" + 0.010*"bottle" + 0.009*"pill...
-----
Recommendations for : ZYLISS Lock N' Lift Can O...
**********************************************
["Chef'n EzSqueeze One-Handed Can Opener (Black and Meringue)", 0.99800944]
['EZ-DUZ-IT Deluxe Can Opener with Red Grips', 0.99777937]
['Swing-A-Way 6090 Easy Crank Can Opener, Black', 0.99773985]
['Resource for Cooking Stainless Steel Manual Can Opener', 0.997704]
['Commercial Swing-a-way Easy Crank Can Opener Heavy Duty - Ergonomic Design', 0.9974011]
```

Fig. 5. Recommendation using LDA - item id as input

topics were assigned a meaningful labels by looking at the words within each topic.

- Topic 0 - Food storage supplies
- Topic 1 - Baking and kitchen appliances
- Topic 2 - Furniture and Party supplies
- Topic 3 - Home Decor
- Topic 4 - Bed room essentials
- Topic 5 - Home improvements
- Topic 6 - Bathroom supplies

Though some of the words in topics overlap, it is pretty good classification to start with.

3) *Recommendation based on Topics*

The next thing to do after getting meaningful topics is to find the most representing topic for each product. Fitting the corpus to the trained LDA model will generate vectors with topic probability of every document in the corpus. Then the most dominant topic and keywords were found and added to the dataframe. Then using the Similarity Api from Gensim index was built for all the given documents.(index the documents in their semantic representation (topics)).

At this point the data frame is merged with meta data to infer the title for each product id. Recommendations were made in two ways. In the first method product was given as input. The LDA vectors of the given product id was found and the similarity score was retrieved from the built index and top 5 similar items were recommended. Figure 5 shows the top 5 recommendations for item with

```
Query_Rec('Dimmable Light Bulbs for Indoor Outdoor Patio Decor')
```

```
Query comprised of the following topics [(0, 0.0109096), (3, 0.3274316), (5, 0.6267742)]
Your item's most prominant topic is:
0.015*"candle" + 0.013*"clock" + 0.012*"unit" + 0.011*"smell" + 0.011*"filter" + 0.009*"battery" + 0...
-----
Recommendations for : Dimmable Light Bulbs for ...
**********************************************
['LightFairy Wall Art Framed Canvas Decoration, Design for Home Decor, Glow in The Dark Painting for Liv
tem (32 x 16 inch)', 0.999881]
['Candle Choice Set of 6 Indoor and Outdoor Remote Controlled Multi-color / Color Changing Flameless Vot
tive Candles /Battery-operated Candles', 0.999881]
['Sunnydaze Tiered Pitchers on Brick Steps Tabletop Water Fountain with LED Light, 10 Inch', 0.999881]
['12 Inch Natural Beeswax Glitter Candles, Gold Color, Boxed Set of 2 candles', 0.999881]
['12 Flameless Battery Operated Tealight Candles. These Are Great for Wedding Decorations, Events, Banqu
Party. Can Be Used in Centerpieces, Floral Displays, or Votive Holders. ON SALE!!!!', 0.99988097]
```

Fig. 6. Recommendation by LDA - Query as input

id "B00COK3FD8". As we can see from the title the product belongs to Topic 0 (Food storage supplies) and the items recommendations make much sense.

In the second method the query was given as input, The query is preprocessed and represented as bag of words (word,frequency pair) and then queried to trained LDA model to get topic probabilities that represents the given query. Those lda vectors are queried to index built earlier which will return the similar documents(Each item's review is considered a document). The top 5 similar items were recommended as shown in 6. As we can the that the query belongs to topic 5 (Home Improvements) and the recommendations are also related to similar topic. The generated topics can still be fine tuned by number of topics and hyperparameter alpha and beta to get more accurate model.

B. Recommendation system using ALS:

The spark ml library uses a model-based collaborative filtering approach, in which users and products are described by a small set of latent factors matrix as in case of singular value decomposition (SVD), which is used to predict missing entries in the user-item matrix. This library uses the ALS algorithm to derive these latent factors. The implementation in spark.mllib has the following parameters:

- numBlocks is the number of blocks used to parallelize computation (set to -1 to auto-configure).
- rank is the number of features to use (also referred to as the number of latent factors).
- iterations is the number of iterations of ALS to run. ALS typically converges to a reasonable solution in 20 iterations or less.
- lambda specifies the regularization parameter in ALS.
- implicitPrefs specifies whether to use the explicit feedback ALS variant or one adapted for implicit feedback data.
- alpha is a parameter applicable to the implicit feedback variant of ALS that governs the baseline confidence in preference observations.
- nonnegative specifies whether or not to use nonnegative constraints for least squares (defaults to false)

The ALS algorithm takes the users and products columns as input and and rating as the rating output column. All the columns are of integer type. Thus, few preprocessing steps are needed in order to apply the algorithm to our data.

1) *Preprocessing the data for ALS input*
   - The preprocessed ratings data (usersID,productsID and ratings) is loaded into sparks RDD dataframe
   - The titles data (productsId and title) is loaded into spark's dataframe.
   - Drop review text and review date columns as they are not needed for the ALS.
   - Merge the titles column into ratings data using left outer join.
   - Use pysparks ml feature StringIndexer library to convert the ratings into integer type.
   - Apply this library on userID and productID columns, generating userId and itemId columns which are of integer type.
   - Drop the old userID and productID columns and rename the new columns
   - Create two new spark dataframes, first with userId,itemId and ratings columns. Second, itemId and titles columns
   - Save the two dataframes as csv files which will be used as a input to ALS algorithm
2) *Exploratory data analysis*
   In this, we are analyzing the Amazon's product ratings data. First we zoom in to visualize the ratings for a subset of users (user id ¡ 100 and product id ¡ 100). We see a sparse matrix. Here, ratings are on the scale of 1.0 to 5.0, where we each value of the rating is color-coded to understand which rating appears to be maximum. As observed in Figure[7], the ratings matrix is very sparse. In this plot, we can see different ratings with different color code. For example User 20 has rated movie 40 with the highest rating of 5.

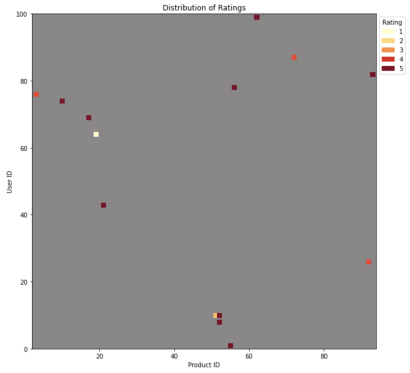

Fig. 7. Sparsity of Ratings Matrix (data subset)

We can analyze the same for entire data to look at the ratings for all of the users across all of the products. We can see some clear patterns from Figure[8]. The vertical lines indicate that the products are rated similarly by all users. The horizontal lines could also indicate that a person ranks all products fairly similarly - if a pale line, they tend to rate negatively and dark red positively. There are some interesting grey patterns too, where users have not rated products at all. We can also notice that some products have dense ratings, indicating some of the popularly purchased products. The left part of the graph indicated highly purchased/ordered products. Grey parts indicating no ratings for those products.

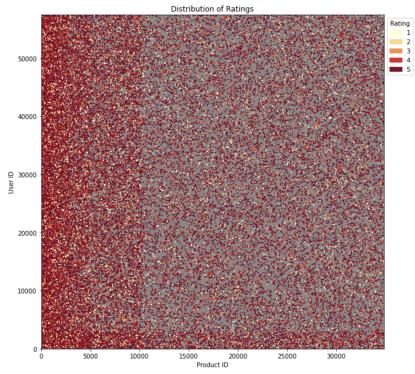

Fig. 8. Sparsity of Ratings Matrix (complete data)

3) *Training for ALS algorithm*
   First, the data is split into train, validation and test data in the ratio 0.6,0.2,0.2 using randomSplit. Defining the model, building the recommendation system using ALS on the training dataset. The cold-start strategy is set to drop and is used for new users/items that have no history and model has to be trained. For this purpose, ALSModel.transform is used when an item/user is unseen. Spark has a coldstartstrategy parameter to be set to drop to resolve this issue. The hyper-parameter tuning is done as shown below:
   - Rank: [10, 12, 14, 16, 18, 20]
   - maxIter: [5,10]
   - regParam: [0.001, 0.01, 0.05, 0.1, 0.2]

   The trainValidationSplit/Cross validation is done and the model is fit on the training data. Thus the model defined is used to predict ratings on the test dataset and evaluate root mean square error [RMSE] using RegressionEvaluator to find the rmse value on the test dataset. This gives us predictions for all the userID and itemID in the test dataset.
   We can plot and visualize the learning curves for this model based on the number of iterations. We can see a plot of the learning curves in ALS as the number of iterations increases in Figure[9](a). Here, the RMSE value decrease as the number of iterations increase and stabalizes after a threshold. We can also plot a learning curve for the changing values of K(rank) or the latent factors. As shown in Figure[9](b), as the number of latent factors increase the RMSE value decreases. Hence, appropriate value of latent factors needs to be chosen.
4) *User-based Recommendations*
   Recommendations for a particular user is evaluated to test the recommendation system. In this, pyspark's ml library is used for implementing ALS. A user ID is selected at

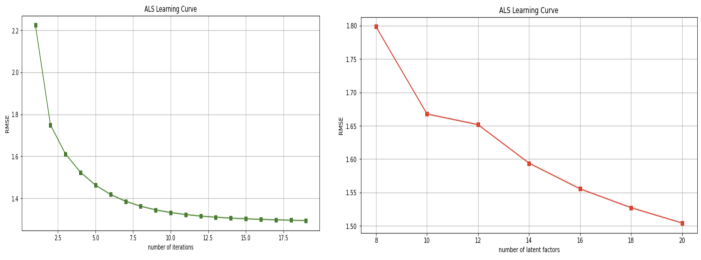

Fig. 9. ALS Learning Curve for Latent Factors vs RMSE

random. The items rated by that user is filtered out and ratings are predicted for all the items. Now, the title data is imported from the metadata to get the names of the productID in our predictions. Now the data from both tables is merged on the basis of asin i.e., productID and the final predictions are displayed for that specific user. A link to the recommended product on Amazon's website is added for a quick reference. The top recommendations for the user "32921" is shown in Figure[10]

Top 20 recommendations for the userID => 32921 are:

	productID	prediction	title	Link
0	B00005IX9T	0.120711	Braun KF400-WH Aromaster 10-Cup Coffeemaker, White	link
1	B0000665TD	1.620185	J-2000 Jiffy Garment Steamer with Plastic Steam Head, 120 Volt	link
2	B00061J2B6	1.489314	Steamfast SF-680 Digital Steam Press with Multiple Fabric Settings and Steam Burst Function,Stainless/Black	link
3	B000A2M2EK	0.993996	Universal Slicer/Shredder with Three Discs	link
4	B000I0WQWC	2.542618	Holmes Arm and Hammer Air Purifier Booster Filter, 4 Pack	link
5	B000LNS2HE	1.646729	Anchor Hocking Glass Prep Bowls Custard Cups, 6 Ounce, Set of 4	link

Fig. 10. User-based top 5 recommendations

5) *Item-based Recommendations*
Recommendations of items which are similar to a particular item is been evaluated using pyspark's mllib library. First, the model is trained with train data and evaluated on test set. The RMSE value is found to be 1.4883. Then a new favourite item-list is created, then the corresponding itemmId's are fetched. The new data is added to train data list with the rating 5.0, and then finally an inference data is fed to the trained model excluding the items from favourite list. The top n recommendations are displayed which are similar to the favourite item-list. The top recommendations for the list of item 'Proctor-Silex 26500Y Durable Belgian Waffle Baker' and 'OXO Good Grips Garlic Press' is shown in Figure[11]

```
Recommendations for ['Proctor-Silex 26500Y Durable Belgian Waffle Baker', 'OXO Good Grips Garlic Press']:
1: Amco Stainless-Steel Cocktail Picks, Set of 4
2: Pyrex 6022369 Storage 14-Piece Round Set, Clear with Blue Lids
3: FoodSaver 1 Quart Size Bag, Package of 48
4: simplehuman KT1025 System Stainless-Steel Dish Rack
5: KitchenAid KHM900ER 9-Speed Hand-Mixer, Empire Red
6: Classic 7" Hollow Edge Santoku Knife with Board
7: MASTERCOOL (52224-A-SP Gray Infrared Thermometer with Laser
8: Polder 77-90 Diet Utility Food Scale, 18 oz./500 g Capacity, White
9: OXO Good Grips Vegetable Brush (color may vary)
10: William Bounds Sili Gourmet V-Shaped Roasting Rack
```

Fig. 11. Item-based top 10 recommendations

C. Topic Modeling and Sentiment Analysis:

This section explains about our experiments on topic modeling and sentiment analysis on the review texts from amazon dataset. Topic modeling is the process of generating topics from the underlying corpus of entire review texts. Sentiment analysis is the process of analysing the customer's feeling from each sentence written by the customer. The goal of this module to give a summary of the goodness of the product in terms of various aspects.This could serve as a structured review by giving aspect wise rating and the top most positive and negative sentences talked by users. Here, topic modelling has been used to derive topics from review text which serve as aspects and the rating is given based on percentage of statements with positive sentiments for a product. We have experimented with different kind of grouping over the product reviews for this purpose.

1) *Data Preparation and Preprocessing*
We have taken three different groupings of data-sets to understand what gives most reasonable results. First is to group random products together. Our data-set for amazon is huge and as we have GPU and memory limitations we used the subset of this data-set, which is top25 products having highest number of review texts as part of this mixed grouping. Second is to group similar products and for the purpose of this experiment, we grouped 3 product Ids of sheets and one product Id belonging to a comforter which had 37618 unique reviews and 134078 unique review sentences. Third is to take a single product which in our case was a cooker which had 22704 unique reviews making up to 116456 review sentences. As a pre-processing step for LDA tokenization, preparing and removing stopwords, lemmatization are applied on these reviews. After this the dictionary is created by assigning unique id for each token. From this bag of words is generated and then final corpus that is needed as input is formed. The libraries we used and their step-by-step implementation is explained in earlier section.
2) *Topic Modeling LDAMallet* LDA model is applied using gensim library on the corpus and dictionary to generate the number of topics. The accuracy of the topics generated is evaluated by the coherence coefficient. The more the coherence coefficient, the higher the accuracy is with the topics. This coefficient varies as change the number of topics to be generated that we pass as a parameter to the LDA model. For this LDA model, the coefficient value we got is -7.205 which is not satisfying. So another version of LDA model , LDAMallet is tried on the same corpus and coherence coefficient is calculated against the number of topics parameter with different values. The number which gave the highest coefficient is chosen. This hyper parameter tuning is shown in figure [12].
Since the data is less and is for only 25 products next best parameter is chosen which is 10 and the coherence score is 0.5853.
3) *Topic Distribution Visualization* The topics are nothing

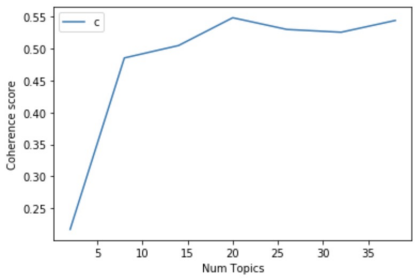

Fig. 12. Coherence Coefficient scores

Fig. 13. Topic Modeling for Sentiment Analysis

but the collection of keywords and their weights. The top 10 keywords are presented in the figure [13]. The intensity of the word in the figure depicts how frequent the word is repeated in overall reviews. The general insights of the topics can be presented using various libraries. Another interesting visualization is presenting keyword weight and its count in overall reviews for each topic. This was produced for all the topics but for space constraints showing just for topic4.

4) *Sentiment Analysis - VADER*

The LDAMallet model gives the topic distribution for each sentence. For making things interesting , topics have been labeled with generic words with most suitable words manually.Sentiment analysis is applied on each sentence to produce sentiment intensity scores using Vader Sentiment Intensity Analyser for randomly grouped products and TextBlob for single product and similar product groupings which was discussed in the earlier sections. In Vader, each sentence is assigned with the sentiment positive, negative or neutral based on the compound score which is the normalized form of sentiment polarity scores. If the compound score is less than 0 or equal to 0 or greater than 0, they are considered are negative , neutral and positive respectively. The dominant topics for each sentence represents which topic is most discusses in the piece of writing.These sentiments are now used to evaluate the following.

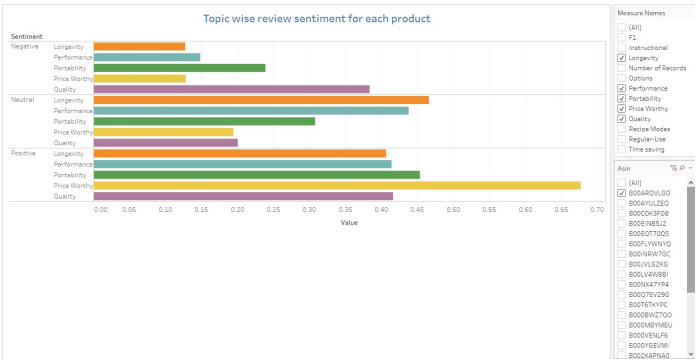

Fig. 14. Sentiment analysis Vader Result

Product summary for 'InstaPot'

Topic wise rating is...

Topic	
Time to cook	1
How likely are you to recommend this product	5
Product Design	5
Product Features	2
Product Quality	5
Number of food options this product is great for	2

Name: Rating, dtype: int64

Most positive sentences are:

0)

I'm really impressed with the quality of this cooker! It feels and looks like it was very well thought out and constructed
1) Both dishes I tested which were brown rice and Ossobuco turned out delicious
2) The brown rice was perfect
3) Wonderful
4) Thank you for a wonderful product

Most negative sentences are:

0) Tried to make yogurt 3 times and it turned out awful, grainey and watery
1) Anybody who has worked with plastic before knows that plastic has a nasty habit of letting things penetrate inside
2) I ended up discarding the rice pudding, it was that awful
3) Only reason not a 5 star
The manual's cooking times and water mix are terrible
4)
Why? The terrible silicone ring that absorbed food smell, and even the detergent's smell when I tried to wash it, and the mixture made me ditch the pot

Fig. 15. Sentiment analysis - TextBlob Result

- The topic's sentiment on each sentence (with respect to dominant topic).
- The number of topics discussed positively for each product
- The ratings of each product with respect rating of the topic.

The final result of Vader Sentiment Analyser is shown in the figure [14].This figure shows for the product B00ARQVLGO and the topics Longevity, Performance, Portability , Price Worthy, Quality.

5) *Sentiment Analysis - TextBlob*

The sentences for textblob are lemmatised sentences passed to get a single score for each sentence. Grouping mixed products of home and kitchen such as cooker, coffee mug and sheets gave topics which were difficult to tag to specific products. Where as grouping over similar products such as sheets and comforters could share certain common topics such as 'fabric'. On, the other hand taking one single product with large review set had clear topic separation.

VI. Comparisons

A. Recommendations by LDA vs ALS

Here we compare recommendations offered by two methods implemented in this project. Figure[17] and Figure[16] shows the recommendations made for "B00004S576" which is "zojirushi nhs-10 6-cup (uncooked) rice cooker" using LDA

and ALS respectively. The two methods cant be compared apples to apples because the ALS recommendation was made only on the subset of data. It is clear that Topic modeling provides better recommendations because it is built on words that describes the product rather than the rating history.

```
Recommendations for Zojirushi NHS-10 6-Cup (Uncooked) Rice Cooker:
1: Fiesta 10-1/4-Inch Deep Dish Pie Baker, Sunflower
2: Norpro Silcone Basting/Pastry Brush
3: Mastrad Silicone Spoon, Blue
4: Mastrad 12-Inch Stainless and Silicone Tongs
5: Le Creuset Enameled Cast-Iron 15-3/4-by-10-3/4-Inch Rectangular Roaster, Cherry
```

Fig. 16. Recommendations by ALS

B. LDA and Sentiment Analysis for different product groups

Ratings are generated based on the sentiments of each topic for every product. These ratings are produced with respect to the sentiments produced by texblob and VADER sentiment analyzer. The results for both approaches are shown in figure [18] Apart from ratings, sentiments for sentences are compared between textblob and VADER sentiment analyzer shown in figure[19].

C. Sentiment Analysis using Vader vs TextBlob

Since Vader sentiment Analyzer is a ruled based approach and it contains word intensity measures, its ratings and sentiments for sentences are more accurate. Vader performed better for sentiment analysis as it considered both love and loves in the same manner. On the other hand, TextBlob required stemming to be performed before getting a sentiment score as it gave lesser score for a sentence have the word 'loves' than love. This difference in result is shown in figure[20]

VII. Conclusion

We have learned to do analyze large scale data and successfully apply sentiment analysis and topic modeling on review comments and also build recommendation system using topic modeling and collaborative filtering. The future enhancement would be combine these approaches and build an app that will recommend items based on users interest and also pinpoint the highlights from the past review comments.

```
Product_recommender('B000045576')

Your item's most prominant topic is:
0.021*"knife" + 0.008*"cook" + 0.008*"oven" + 0.008*"cooking" + 0.007*"pan" + 0.007*"sharp" + 0.007*...
-----
Recommendations for : Zojirushi NHS-10 6-Cup (Uncooked) Rice Cooker...
**************************************************
['LiveFresh Stainless Steel Heavy Duty Fine Mesh 13" Splatter Screen with Resting Feet and Handle Grip', 0.99999696]
['Zojirushi 604976-NS-LGC05XB NS-LGC05XB Micom Rice Cooker & Warmer, 11.9 x 9.1 x 7.5, Stainless Black', 0.9999936]
['Chef'sChoice 840 WafflePro Taste/Texture Select Waffle Maker Traditional Five of Hearts Easy to Clean Nonstick Pla
erature Recovery with 6-Setting Color Control Dial, 5-Slice, Silver', 0.9999926]
['Black and Decker HS900 Flavor Steamer Rice Cooker Plus', 0.9999871]
['Zojirushi NS-LHC05XT Micom Rice Cooker & Warmer, Stainless Dark Brown', 0.9999801]
```

Fig. 17. Recommendations by LDA

	asin	Options	Recipie_modes	Time_saving	Portability	Logivity	Price_worthy	Instructional	Quality	Regular-Use	Performance
0	B00T6TKYPC	5.0	5.0	4.0	4.0	4.0	3.0	5.0	4.0	3.0	4.0
1	B0015TMHSI	3.0	3.0	2.0	3.0	4.0	4.0	4.0	2.0	2.0	3.0
2	B00Q7EV29G	3.0	2.0	4.0	2.0	3.0	2.0	3.0	5.0	4.0	2.0
3	B00FLYWNYQ	5.0	3.0	5.0	3.0	1.0	5.0	3.0	3.0	4.0	2.0
4	B00COK3FD8	4.0	2.0	4.0	5.0	4.0	3.0	5.0	2.0	5.0	4.0

Fig. 18. Ratings of products from Vader

	productID	Softnest and comfort	Fit	Quality	Recommendation	After wash quality
0	B00635VODS	5	5	3	4	4
1	B00902X68W	5	5	2	4	4
2	B00LV4W8BI	5	5	3	4	5
3	B00NLLUNSE	5	5	3	5	5

Fig. 19. Ratings of products from TextBlob

	Sentiment Analyser	Loves	Love	//Love	Love!!	'Love'
0	VADER	0.6467	0.64670	0.6467	0.7482	0.69880
1	TextBlob	0.0000	0.78125	0.0000	1.0000	0.78125

Fig. 20. Textblob Vs Vader

VIII. Acknowledgement

We thank Professor Carlos Rojas for his guidance (San Jose State University) in our project work. We also acknowledge the contributions of our team members.

References

[1] Taneja and A. Arora, " Cross domain recommendation using multidimensional tensor factorization," Expert Systems With Applications, vol. 92, pp. 304-316, 2018.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3, null (3/1/2003), 9931022.

[3] R. Krestel, P. Fankhauser, and W. Nejdl, Latent Dirichlet allocation for tag recommendation, in Proceedings of the 3rd ACM Conference on Recommender Systems, pp. 6168, October 2009.

[4] Yang, A: Inferring Business Similarity from Topic Modeling [Latent Dirichlet Allocation and Jaccard Similarity applied to Yelp reviews]. (2015).

[5] https://radimrehurek.com/gensim/models/ldamodel.html

[6] https://radimrehurek.com/gensim/similarities/docsim.html #gensim.similarities.docsim.MatrixSimilarity

[7] https://blog.insightdatascience.com/topic-modeling-and-sentiment-analysis-to-pinpoint-the-perfect-doctor-6a8fdd4a3904

[8] https://nijianmo.github.io/amazon/index.html

[9] https://monkeylearn.com/sentiment-analysis/

[10] https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f

[11] https://towardsdatascience.com/fine-grained-sentiment-analysis-in-python-part-1-2697bb111ed4