Amazon Review Data Analysis

Harika Andugula *, Shravanthi Bhoopalam Sudharshan†, Thanmai Gajam‡, Thasleem Banu Tajudeen§
* Email:harika.andugula@sjsu.edu
† Email:shravanthi.bhoopalamsudharshan@sjsu.edu
‡ Email:thanmai.gajam@sjsu.edu
§ Email:Thasleembanu.tajudeen@sjsu.edu

***Abstract*—The number of people who prefer online shopping is rapidly increasing due to the technological advancement and convenience of shopping anywhere at any time. The recent studies show that the most of all online shoppers use reviews to decide on what products to buy. Reviews not only help customers, but it also helps to strengthen sellers' trustworthiness. Amazon is the worlds largest retailer with millions of products. This project deals with the study of statistical analysis of reviews and ratings by Amazon users.**

I. Data Collection and Preparation

The amazon review datasets are freely available online. 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.

II. Background

Much work has been done analysing statistical data from the subset of Amazon review dataset. A large number of recommendation systems are developed using different techniques. AI based procedures from collaborative filtering through matrix factorization have proved to be more successful in producing desired results but, one of the drawbacks of collaborative filtering is, it is limited by data sparsity problems since it recommends users/products based on rating history. Regarding The Latent Dirichlet Allocation (LDA), the model used for training is proposed by Blei, Ng and Jordan in 2003 (Blei et al., 2003a). Blei et al. (2003b) 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.In this project we try to apply LDA topic modeling on review comments in amazon dataset and recommend products based on topic similarity and compare recommendation with standard item based collaborative filtering. we also analyze sentiment of reviews in order to score items by topics mentioned in their reviews.

III. Project goals

1) Do topic modeling(LDA) on review comments
2) Build a recommendation system using collaborative filtering (ALS) method and topic modeling (LDA) method and compare their respective results
3) Use topics generated from LDA and analyze sentiment of the reviews in order to score products by the topics mentioned in their reviews.

IV. Data Preprocessing

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

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.

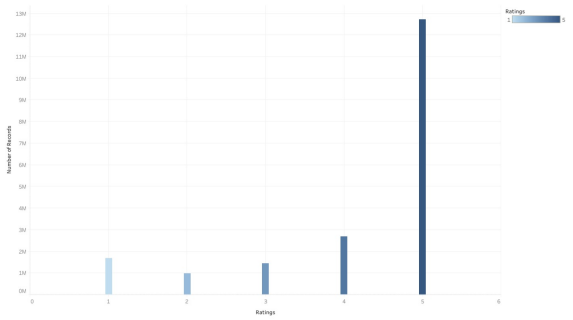

Fig. 1. Rating Distribution

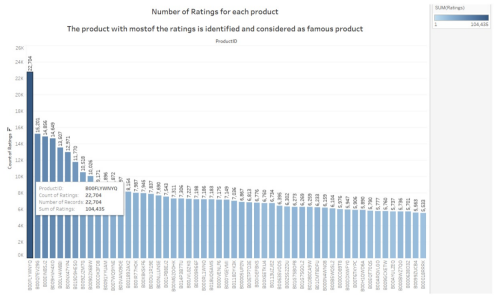

Fig. 2. Highly rated products

V. 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 1. 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. This visualization is shown in figure 2
- The number of ratings for all the products has changed over the years. This was depicted in the figure 3
- In figure 4, 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.

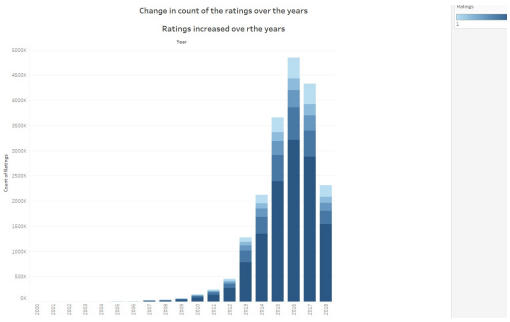

Fig. 3. Ratings in years

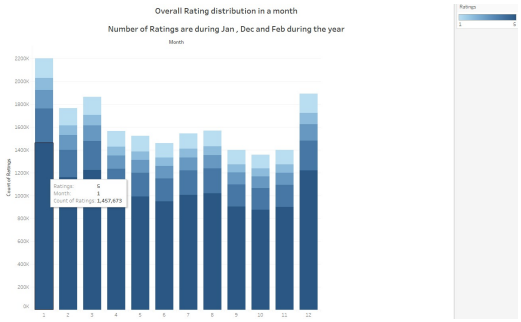

Fig. 4. Ratings per Month

VI. Algorithm Implementations

A. Topic modeling with LDA

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). The practical use of such an algorithm 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 corpuses. Gensim package has been used for LDA implementation.

1) Preprocessing the data for LDA input

- The dataset is grouped by product ids such that all the reviews for each product ids are in one cell
- Review comments of each item id is considered as a document for LDA
- The review texts are converted to lowercase and tokenized using nltk library.
- Numeric tokens which doesnt have any character are removed.
- Tokens which are less than 4 characters are removed.
- Stopwords such as 'during', 'doing', 'you', 'or', "they'll", 'after', 'the', "weren't" are removed using nltk stopwords library.
- Tokens are lemmatized to reduce inflectional forms, Lemmatization is 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).
- Remove words that appear in less than 20 documents or in more than 10 percent of the documents.
- Create 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 is used to convert a document (a list of words) into the bag-of-words format (list of (token id, token count) tuples).

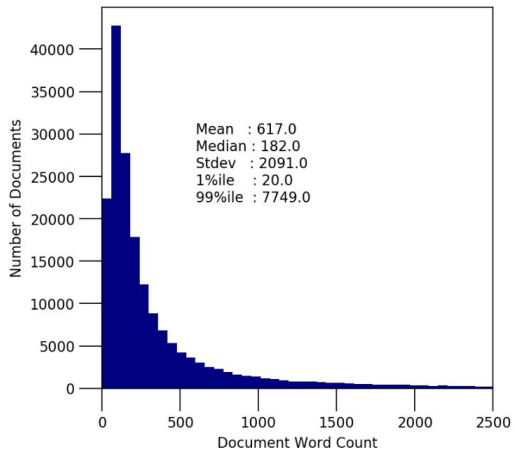

Fig. 5. Distribution of Document word counts

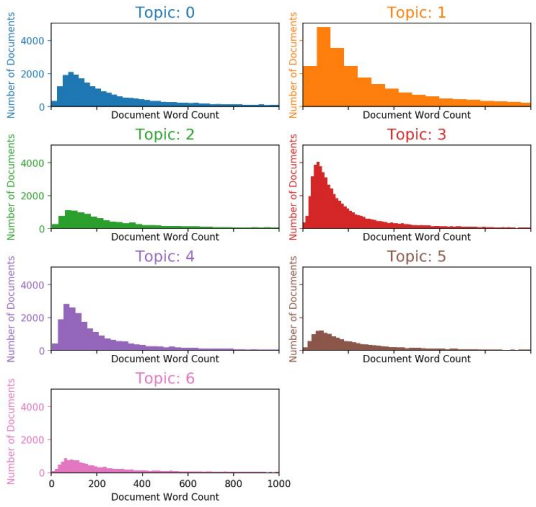

Fig. 6. Distribution of Document word counts by Dominant topic

Figure 5, shows the distribution of Document and word token counts

2) **Training LDA** 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. The gensim LDAmodel offers an option called 'auto' for these two hyper parameters which automatically adjusts based on No.of topics and documents. For this project we will be using the baseline model with a number of topics 7.

Figure 6, shows the distribution of Document and word token counts by dominant topic. Figure 7, shows the top 10 words in each of the topics generated.

3) **Recommendation based on Topics**
Inference is run on all the reviews for each product to find the most representing topic. After the products are represented by probabilities of topics Cosine similarity is calculated to find similar items based on topics. Cosine similarity is measured between bag of words which gives

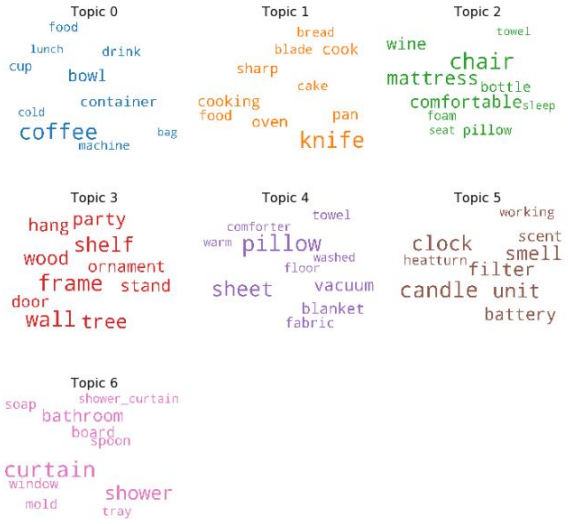

Fig. 7. Wordcloud of Top 10 words

similarities in the range of -1 to 1 (the greater, the more similar) Similarity = cos() = (AB)/(AB), X being its norm. Order those outputs for recommendations based on given item id One way to validate the recommender is to back test the data to check if the user has bought the recommended product.

B. Collaborative Filtering using ALS:

Collaborative filtering is a technique used for recommendation of items based on user-item data. It is also known as the Nearest Neighbourhood Algorithm. There are many types of collaborative filtering, user-based, item-based and model based collaborative filtering techniques. These techniques help to fill in the missing entries of a user-item matrix. 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 alternating least squares (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 userID and productID 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 data is loaded into sparks RDD dataframe.
- Drop review text and rev date columns as they are not needed for the analysis.
- 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.
- Final dataframe with userId, itemId and ratings is generated and given as input to the ALS algorithm.

2) **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: [1, 5, 10,15,30]
- maxIter: [10]
- regParam: [0.05, 0.1, 0.5,1]

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.

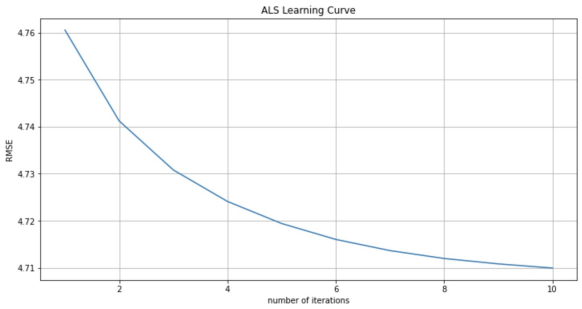

Fig. 8. Learning Curve ALS

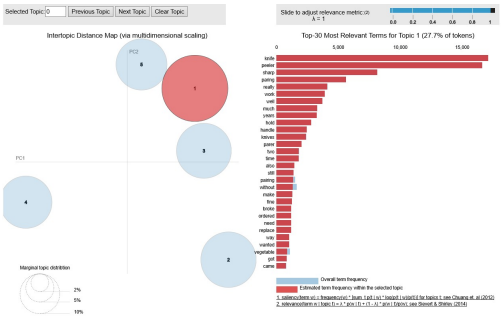

Fig. 9. Topic Modeling for Sentiment Analysis

3) **Recommendations for users**

Recommendations for a particular user is evaluated to test the recommendation system. A user ID is selected at 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.

C. Sentiment Analysis on Review Topics from LDA:

Topic modeling and sentiment analysis is applied on the review texts to identify the topics discussed about a product and analyse if the review is discussing positively or negatively and then finally assigning ratings ti the products accordingly.

1) **Data Preparation and Preprocessing**

As the data is huge in our dataset, we are considering only first 50 unique productId reviews from the dataset. All the preprocessing steps that are explained earlier in this paper under topic modeling are applied for these 50 review texts. The topics that were generated and top 30 frequent words in each topic are shown in figure 9

2) **Sentiment Analysis**

- Find the Word2vec embeddings for the words in the document collection of reviews.
- Passing the Word2vec list to K means clustering algorithms with k as 2.
- Manually go through the words listed on using gensims most similar method for both the clusters to identify the positive and negative clusters.
- Sentiment scores are calculated for the words in the cluster as inverse of the distance from centroid for positive words and negated inverse of distance from centroid for negative words.
- TF-IDF values are calculated for the words in the review text document.
- The words in list of lists which maintains the review text in sentences are replaced with their corresponding sentiment scores as one array

Fig. 10. Frequent words

- Similarly another array stores the TF-IDF score maintaining the review text document order.
- A dot product is calculated between Sentiment scores from k means and TF-IDF values to get the final Sentiment scores for sentences.
- The most frequent words discussed in a review are plotted using word cloud shown in figure 10

3) **Topic based rating for reviews**

- The LDA model gives the topic distribution for a sentence.
- Sentiment Scores are used to determine if the sentences are positive.
- For a given topic, take the percentage of positive topics to assign a rating for that topic.

VII. Task Contribution

A. Harika

- Data preparation
- Data Preprocessing
- Data Visualizations (Tableau)
- Topic modeling with sentiment Analysis
- Report work and researching

B. Shravanthi

- Data preparation
- Data Preprocessing
- Data Visualizations (Tableau)
- Recommender systems using ALS
- Report work and researching

C. Thanmai

- Data preparation
- Data Preprocessing
- Data Visualizations (Tableau)
- Topic modeling with sentiment Analysis
- Report work and researching

D. Thasleem

- Idea proposal, Data Collection and Preparation
- Data Cleaning (Drop unverified reviews, removing urls from reviewText)
- Data Visualizations (Tableau, Matplotlib)
- LDA Topic modeling and Item recommendation using topic similarity
- Report section I,II,III,IV,VI A

References

[1] Taneja and A. Arora, " Cross domain recommendation using multidimensional tensor factorization," Expert Systems With Applications, vol. 92, pp. 304-316, 2018.
[2] https://blog.insightdatascience.com/topic-modeling-and-sentiment-analysis-to-pinpoint-the-perfect-doctor-6a8fdd4a3904
[3] https://ieeexplore.ieee.org/document/6913637
[4] https://nijianmo.github.io/amazon/index.html
[5] https://monkeylearn.com/sentiment-analysis/
[6] https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f