CS-GY 6513 Big Data Final Project

Sentiment Analysis and Recommendation System Based on Amazon Reviews

Group members:

Jackson Qu - hq20xx

Runze Li - rl50xx

1 Introduction

In the age of e-commerce, user-generated content, especially reviews, has become a great source of valuable insights. Understanding the intent, preferences, and purpose behind these reviews can greatly help enterprises customize their services, improve user experience, and optimize their marketing strategies. In this report, we implement a project to utilize Amazon review data for sentiment analysis and recommendation systems. The main goal of the project is to develop a recommendation system that extracts meaningful insights from massive user reviews, applies sentiment analysis, and provides personalized product recommendations to improve the competitiveness of companies in the e-commerce industry. By employing data preprocessing, multithreading, and random sampling, we applied key big data principles to ensure that our system can process large amounts of data efficiently. These methods allowed us to work within a single-node environment to improve recommendation coverage, diversity, and accuracy.

2 Problem Statement

In the e-commerce industry, user reviews contain a lot of valuable information about user needs, preferences, and product quality. However, with the rapid growth of review data, how to extract useful information from massive reviews, identify user emotions accurately, and provide users with personalized recommendations has become a hot topic. At the same time, it is crucial for companies to capture market trends and optimize the recommendation system to improve user experience, strengthen customer relationships, and optimize marketing strategies. Therefore, it is important to explore the value of user reviews through big data technology and sentiment analysis for the business improvement and competitiveness enhancement of e-commerce platforms.

3 Problem Goal

This project aims to develop a big data e-commerce user review analysis system to extract user insights from massive reviews through modules such as data cleansing, sentiment analysis, personalized recommendation, and trend prediction. The system will be designed by using big data methods such as distributed computing and data preprocessing. Moreover, our project combines recommendation algorithms and models to achieve efficient processing and analysis within the constraints of a single-node environment. Meanwhile, through clear visualization of analysis results, our project will provide key decision support for enterprise management, help enterprises optimize product recommendation, improve users' experience, and provide the basis for market trend prediction.

4 Data Description

The web-Amazon dataset from Stanford's SNAP (Stanford Network Analysis Project) consists of data collected from the Amazon e-commerce platform, including product reviews and social networks. This dataset is valuable for various analyses in big data contexts, such as sentiment analysis, recommendation systems, and network analysis.

Web data: Amazon reviews

Dataset information

This dataset consists of reviews from amazon. The data span a period of 18 years, 2013. Reviews include product and user information, ratings, and a plaintext review duplicates, due to products whose reviews Amazon merges. A file has been added identify products that are potentially duplicates of each other.

Note: A new-and-improved Amazon dataset is available here, which corrects the ab more complete data/metadata.

Dataset statistics	
Number of reviews	34,686,770
Number of users	6,643,669
Number of products	2,441,053
Users with > 50 reviews	56,772
Median no. of words per review	82
Timespan	Jun 1995 - Mar 2013

Source (citation)

- J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding RecSys, 2013.

5 Data Preprocessing in Big Data

Data preprocessing is one of the most important steps in big data projects, particularly in e-commerce data analysis. Given the large volume of user reviews and product information, it is essential to use big data processing methods to clean, transform, and prepare the data for analysis. In our project, we utilize several big data techniques to preprocess millions of reviews efficiently and ensure that the data is well-structured and ready for recommendation models.

In our project, we faced the challenges of processing a large dataset of Amazon product reviews efficiently. Although MapReduce and Spark are common choices for distributed computing on large data, we implemented batch processing with memory optimization techniques, simulating the effect of multithreading for our data preprocessing tasks. Specifically, by leveraging panda and gc.collect() for chunked data loading and garbage collection, we optimized memory usage on a single machine. This approach allowed us to achieve significant processing efficiency without requiring a distributed cluster of machines. However, this approach may not scale well for significantly larger datasets beyond memory constraints. With the help of multithreading, we can utilize a single machine within the system to process data concurrently, ensuring efficient handling of large datasets in memory. Overall, this method can satisfy our requirements.

5.1 Data Loading and Batch Processing

In traditional data processing, loading and processing a massive amount of data can easily make system memory crash and lead to performance issues. Therefore, in our big data environments, we handle these challenges by using Pandas and TQDM, which can allow us to process large data efficiently within the constraints of our environment. Specifically, we use Pandas' chunksize argument and TQDM library for progress tracking. In fact, our project is implemented in Google Colab. Although this is a convenient environment for our project, Colab has limitations in running complex distributed systems such as Apache Spark or MapReduce. For example, Spark requires a cluster of machines or a cloud-based distributed infrastructure, which is not feasible in a single node environment of Colab. What's more, chunksize in Pandas allowed us to load the data in chunks, process each chunk separately, and then discard it before moving on to the next. By processing smaller pieces of data sequentially, we can reduce memory usage and improve efficiency without the need for more complex distributed systems.

5.2 Handling Missing and Inconsistent Data

Missing and inconsistent data is often a common problem in large datasets, especially in user-generated content such as reviews. However, in a big data environment, invalid data can severely impact a model's ability to make accurate predictions. Therefore, effective data cleaning is necessary. During data processing, we use some techniques to deal with missing data to ensure that the dataset remains complete. For example, replacing missing comment text with empty strings ensures the continuity of the data processing pipeline, a critical step in big data workflows where error handling for missing values can introduce significant overhead. Additionally, to ensure data quality, we filter out some irrelevant reviews, such as those with low ratings (less than 4) or review text that is too short (less than 50 characters). These thresholds were chosen based on the assumption that higher ratings reflect more meaningful user engagement and that short reviews lack sufficient semantic content for embedding models. This ensures that only high-quality data is used to improve the effectiveness of the recommendation model.

5.3 Text Processing

Actually, e-commerce reviews are unstructured. This process is challenging when working with large datasets because traditional methods aren't feasible due to processing time and memory constraints. In order to utilize these data in recommendation models, we first need to transform the text into a structured format. Specifically, we remove stop-words and tokenize the text of each review into individual words. Secondly, We apply text normalization such as lowercasing to ensure normalization across the data. This preprocessing step is especially important in big data systems, where the volume of text can create a large number of features.

5.4 Random Sampling

In the big data project, processing the entire dataset for testing purposes is often computationally infeasible due to the large volume of data. Therefore, random sampling plays a crucial role in ensuring that models are evaluated efficiently without having to process the entire dataset, which would take a lot of time and resources. This is especially important when dealing with massive datasets such as Amazon reviews, which consists of terabytes or petabytes of data.

6 Model Training and Evaluation

In the process of model training, we use libraries such as surprise for the analysis of big data recommendation systems. To generate product recommendations, we employ several recommendation models as follows:

6.1 Word2Vec

We apply the Word2Vec model to represent words in review text as the vectors, and we use the skip-gram algorithm to capture semantic relationships between vectors. Actually, in the following chapter, we find that Word2Vec has some limitations in handling unseen words and rare terms, which prompts us to adopt FastText for better vector representation and enhance recommendation diversity.

6.2 SVD (Singular Value Decomposition)

SVD is a matrix decomposition technique used to extract latent factors from the user-item rating matrix. The main role of SVD is to extract important features in the matrix by dimensionality reduction, which can improve the effectiveness of the model. In our recommendation systems, by mapping users and items to a low-dimensional latent factor space, SVD can capture the relationship between users and items and improve recommendation accuracy.

6.3 Recommendation Model

Recommendation models are tools that can provide personalized recommendations to users based on their historical behavior, preferences and products. Here, we use three main recommendation models:

- Content-based recommendation model relies on product features, such as product descriptions, categories, and so on. This model recommends products to users that are similar to their historical preferences.
- Collaborative filtering recommendation model recommends products to users based on their behavioral data, such as user ratings, clicks and purchase history. It analyzes the historical interaction data and infers the similarities between users and commodities.

- Combined recommendation model combines the advantages of content recommendation and collaborative filtering, and this model provides more diversity and accuracy by combining the advantages of different methods.

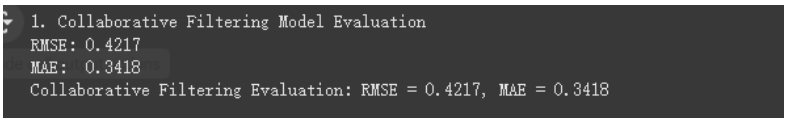
6.4 Evaluation Metrics

We evaluate the models using key metrics:

- RMSE is mainly used to measure the predictive accuracy of a model. It is sensitive to large errors.
- MAE represents the average of the absolute errors between the actual values and prediction values and MAE is used to measure the accuracy of the model in prediction.
- Coverage measures the proportion of items recommended by the recommendation model. In our project, we consider diversity by analyzing the distribution of recommended items across product categories, aiming to find out how well the model promotes various recommendations.

7 Results and Analysis

We use the collaborative filtering model for testing in our project. The results show that RMSE = 0.4217 and MAE = 0.3418, which indicate strong recommendation ability and prediction accuracy. However, we find that the low coverage is 0.0003, reflecting the model's strong reliance on popular items, which is a limitation of collaborative filtering model in the dataset.



While the RMSE and MAE values indicate good performance, low coverage will have an impact on the user experience, because low coverage value leads to repeated recommendations and a lack of diversity. In real life, if users receive the same popular items repeatedly, the recommendation system will reduce users' satisfaction. Users are less likely to discover other new products which may meet their needs better. Therefore, we need to make some improvements to the project based on this issue.

8 Improvements

While our recommendation models in the project can demonstrate strong prediction accuracy, there are several issues for improvement in the system. Word2Vec has some weakness like high complexity when dealing with millions of reviews, so training on the large datasets will be time-consuming and computationally expensive, which makes the model less efficient for updated data. Therefore, we plan to use FastText to improve the recommendation system.

FastText can effectively capture unseen words, allowing the model to better handle rare words. This ability to handle rare words is useful in updated data in real life, where products with limited review data can still be represented through limited topics and sentences. Moreover, FastText can generate more accurate word vectors, enhancing the relevance and diversity of recommendation models.

```
fasttext_model = FastText(
        vector_size=200,    # Higher dimension for better semantic representation
        window=5,               # Context window size
        min_count=3,            # Lower frequency threshold
        sg=1,                       # Skip-gram model
        epochs=20,              # More training epochs
        workers=4               # Multithreading
)
```

By testing the improved recommendation system, we can get improved results: RMSE = 1.1914, MAE = 0.6856 and coverage = 0.0248.

```
1. Collaborative Filtering Model Evaluation
RMSE: 1.1914
MAE:  0.6856
Collaborative Filtering Evaluation: RMSE = 1.1914, MAE = 0.6856
```

```
Recommendation Coverage: 0.0248
```

Using FastText, the evaluation metrics slightly decreased (RMSE = 1.1914, MAE = 0.6856), but the coverage increased significantly to 0.0248. This result highlights improved recommendation diversity, which enhances user experience by providing them a wider range of products.

9 Conclusion and Future work

This project successfully applies big data technologies to build a scalable recommendation system that can handle large amounts of data in the e-commerce field. By using big data technologies in the single-node machine of Google Colab, we can efficiently preprocess and analyze massive datasets. The implementation of models such as Word2Vec, SVD, and FastText helps show the accuracy and diversity of recommendations.

In the future, we need to improve the recommendation system. One of the challenges in our recommendation system is the low coverage, with the system currently only covering about 2% of the entire data. We need to explore hybrid models combining collaborative filtering and content-based models with additional features such as sentiment scores and user profiles. This method will enable the system to balance popular recommendations with personalized and diverse options effectively. The comprehensive models can balance recommendations between widely popular items and less-known but potentially relevant items. What's more, Pandas can be replaced with Spark for truly distributed computing, which overcome platform limitations to achieve faster processing of larger datasets. This method forms the foundation for making better decisions, optimizing product recommendations, and enhancing user experience.