Team 10: Nigel Wang, Susie Li,Faustin Nzitonda, Jennifer Kim

Professor: RAJIV GARG

ISOM_671

6 December 2022

1. Executive Summary

This project is about building a collaborative filtering recommendation system based on sentiment reviews of users. Unlike other recommendations which are based on rating, our recommendation system is based on how users feel about books using reviews. To generate rating scores based on reviews, we used a pretrained model based on deep learning(from flair package) which is powerful compared to some other models(from nltk package) that are based on a bag of words. Due to how the sentiment scoring model used is built, it makes it perform better on previously unseen words. We not only limited our scope on model building, our project is completely based on the use of big data technologies like pyspark and hive for model building using tensorflow and generation of visualization by connecting PowerBI to hive. Dataset used for this project is from kaggle. It's open data from kaggle with about 100k books and 4118 users**.**

2. Introduction

We selected Goodreads because it is one of the world's largest cataloging websites that allows users to search its database of books, annotations, quotes, and reviews. With that said, this is a company with a large database and where we can widely analyze customer needs based on an individual's profile and features. Book recommendation is an important process since this process can be extended to other products as well such as recommendation for a customer on a product. For a possible hypothesis, it may be that a retail company would like to suggest their products based on categories online based on their previous purchase history. For the Goodreads company, the results of recommending books may be an indicator of selection of the type of books in their database to satisfy an individual's preference.

3. Data wrangling

A. Data Preprocessing for Visualization

The first step we took was data cleaning where we removed all commas and HTML tags such as <b>, < /b>, <i>, etc. in text format column to Book Name, Publisher, Description. We also removed all meaningless columns such as the dataframe index.

The second step is merging the tables. We created a standard column index to reorder the columns of each data frame and replaced it with "None" for the columns that were lacking data. In essence, we formatted and merged the data frames together.

The final part was feature engineering. We deleted any redundant contents such as the following.

RatingDistTotal	total:30313 →	RatingDist_Total	30313
RatingDist1	1:257 →	RatingDist_1	257

We then combined some columns together such as "PublishYear"-"PublishDay"-"PublishMonth" to a single column named “PublishDate”. Lastly, we changed the data type for the PySpark schema.

Summary statistics

summary	Rating	CountsOfReview	pagesNumber	Count of text reviews	RatingDist_1	RatingDist_2	RatingDist_3	RatingDist_4	RatingDist_5
count	1537949	1537949	802973	405953	1537949	1537949	1537949	1537949	1537949
mean	3.0716870607741655	13.549387528455105	280.74908371763433	6.954443002022401	106.29102720571359	234.77237996838647	856.606284733759	1483.7250565525906	1953.0721772958661
stddev	1.615125019934719	316.22454699314716	5647.286889498766	104.38462934796398	2239.255479184782	3738.106649560078	11934.618787506735	22123.08961042776	39757.00836448353
min	0.0	0	0	0	0	0	0	0	-2
max	5.0	154447	4517845	99	546262	539083	1002494	1912159	4543188

B. Data Preprocessing for Recommendation systems

To build recommendation systems we used a users dataset which has user id, book name and review. The review provided in our dataset was not compatible with what we want, therefore we decided to change review into sentiment score based on pretrained NLP model name flair[1]. Flair is a powerful NLP model with a lot of functionalities built in PyTorch. To get sentiment score from flair, we used TexClassiffication model with english language embeddings. There are a lot of models for sentiment classification, but flair was our choice because it’s more accurate and it can easily react well to previously unseen words. The model generates scores that are between 0 and 1 with labels(POSITIVE or NEGATIVE) associated with it. In order to allow the model to differentiate between positive and negative scores, the negative scores were multiplied by negative signs which resulted in scores between -1(more negative) and 1(more positive). For the purpose of effectiveness of model training, we used min-max normalization to scale scores between 0(more negative) and 1(more positive).

Summary statistics

summary	Name	Rating	User	Book	Scores
count	342591	342591	342591	342591	342583
mean	Infinity	null	1762.1904229382308	25775.206876722244	9.864896175528578
stddev	NaN	null	1023.5215778753176	28211.39340877142	718.5724204221948
min	آنچه سینما هست!	""The Friends of the Friends""	"" The Jolly Corner"""	""The Love of a Good Man"""	-0.99983644
max	ﷺ come Closerﷺ: Critical Perspectives on Theatre of the Oppressed	really liked it	really liked it	really liked it	93210.0

C. Challenges and limitation of data

The book's datasets were very messy. We had 23 files in the Books dataset with different schemas and commas in the description column which confused the pyspark read csv method. To solve those issues we removed commas from description using Python. Another main problem we had was files with different schemas which mixed entries from different columns. This was the hardest part because we spend almost 70% of our time figuring out how to make a standard schema for all books. We solved the issue by taking a file which has all columns and replacing a column that is missing in that table with null values.

4. Insights & analytics

A. Model Free Insights

We connected the two tables “books” and “users” from S3 and Hive in the metastore to Power BI through the ODBC driver in order to perform data analysis and extract findings through visualization of the dataset. Since we performed modeling on the review ratings to get the rating scores, we have broken down the visualization as pre-modeling and post-modeling. Our focus is to explore the type of books that received the most rating from users and the distribution of ratings. For the pre-modeling dashboard, we created four visualizations. The first dashboard shows books with the most reviews. The second is the count of total reviews over time. The third is the distribution of ratings. The fourth is looking at authors with the most books in the database with counts of books in different languages.

Post-Modeling Dashboard:

For our post modeling dashboard, we show deeper into the rating scores. For the score, +1 signifies that the user really liked the book while -1 score signifies that the user really disliked the book. On one graph, we portrayed the distribution of the rating scores. For the second graph, it shows all of the users in the database and the distribution of how each user rated the books. The users on the top of the chart are users who gave the most number of ratings. For example, user 1473 (very top user) gave the most number of reviews and she/he liked most of the books she read.

In conclusion, there are few insights we can derive from the visualizations. The most common rating users provide for the book is 4 out of 5. We can also see that roughly 90% of users liked the book while 10% of users expressed dislike of the book.

From visualization

B. Modeling

Overview of Neural Network Layers used for building Our recommendation System

Linear Layer

Linear is defined as a dot product between weigits(parameters) and independent variable, the output of linear layer is a response variable.

$\boldsymbol{Y = Xw + b,}$ where **Y** is the response variable, **w** is a vector if parameters and **b** is the bias

Sigmoid Layer

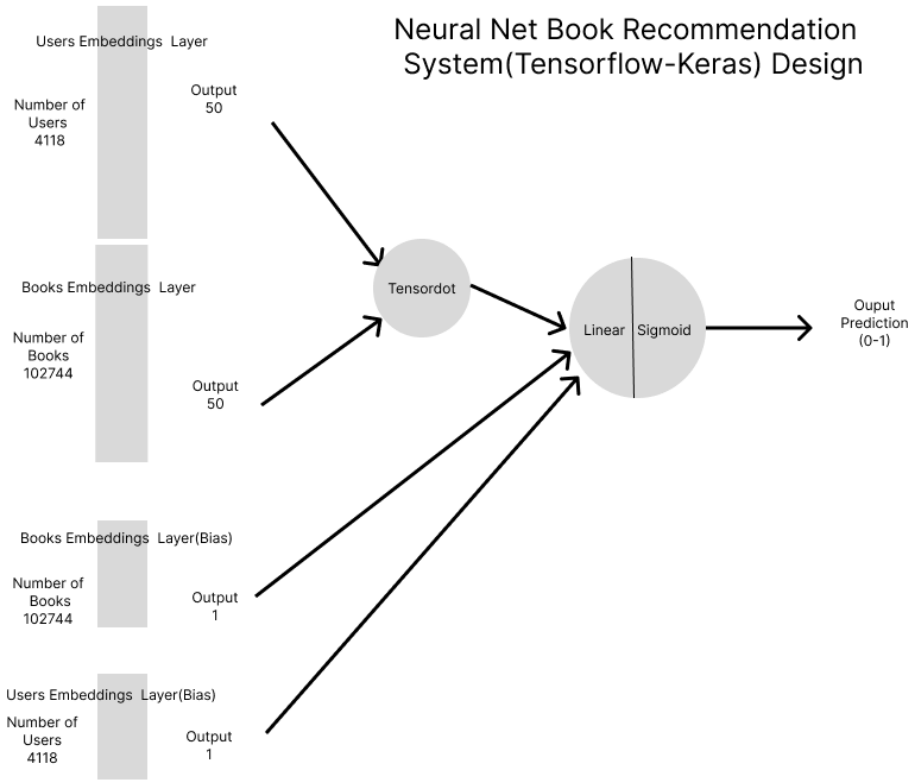
Sigmoid is an activation function that scales real real numbers between -∞ and ∞ to 0 to 1

$$sigmoid(x) = 1/(1 + e^{-x})$$

Embeddings Layer

Embeddings layer is a trainable network that transforms categories into a continuous vector. It works as one-hot encoding, but the distance between categories in case of one-hot encoding is the same. This makes one-hot encoding fail to group similar categories together. One-hot encoding also fails when the number of categories increases because it tends to create variables of each category. For the case of embeddings, it projects all categories to a vector which groups similar categories together and reduces the number of variables in the network. Embedding parameters that project a categorical feature into a vector, is a part of neural network which means that embedding parameters are learned using gradient descent[2]. Embeddings layer is mostly used in speech recognitions, language models, languages translation, recommendation systems etc.

Model Design and Training Process



Recommendation systems design

We built a book recommendation system neural network model based collaborative filtering. It takes the total number of users and books, and uses embeddings of output of 50 to group users that like similar

books and books that tend to be similar. Books and users embedding are multiplied together to generate the aggregated results. The final layer is the sigmoid activation which generates scores and makes sure that the output is in range of 0 and 1. Regarding data preprocessing, We generated a Sentiment score using the NLP model called flair which generates the positivity or negativity score of a sentence. For our case, Sentiment score was generated based on reviews feedback from users(short sentences review which show their interests).

Big Data Technology and Model Training Process

Data used in this project are stored on S3. Data read from s3 into EMR cluster to do preprocessing discussed in the data preprocessing section using PySpark. The model was trained on the Namenode(Spark Master node) because we had difficulties in installation of distributed machine learning libraries like elephas. The model took about 1 hour to be trained for 5 epochs. After training the model, we generated the top 10 predicted sentiment for random users for books they didn't watch.

Recommendation Examples For 3 Random users

```
============TOP  10 RECOMMENDED BOOK================
Becoming
The BFG
Binti (Binti, #1)
Zeitoun
رياضيات خيام
Maus II: A Survivor's Tale: And Here My Troubles Began (Maus, #2)
The Cider House Rules
There There
Hyperion (Hyperion Cantos, #1)
The Martian
============TOP  5 BOOKS READ BY USER===============
The Book Thief
Good Night Stories for Rebel Girls: 100 Tales of Extraordinary Women
The Big Book of Birth
Pride and Prejudice
Catching Fire (The Hunger Games, #2)
```

```
============TOP  10 RECOMMENDED BOOK================
Between the World and Me
The Lord of the Rings (The Lord of the Rings, #1-3)
Becoming
The Nightingale
The BFG
Zeitoun
Pachinko
When Breath Becomes Air
Wonder (Wonder, #1)
The Things They Carried
============TOP  5 BOOKS READ BY USER===============
One Hundred Years of Solitude
Jude the Obscure
Far from the Madding Crowd
A Soldier of the Great War
The House of the Spirits
```

```
============TOP  10 RECOMMENDED BOOK================
Between the World and Me
Becoming
Pachinko
Maus II: A Survivor's Tale: And Here My Troubles Began (Maus, #2)
The Things They Carried
Maus I: A Survivor's Tale: My Father Bleeds History (Maus, #1)
Commonwealth
The Lord of the Rings (The Lord of the Rings, #1-3)
The Nickel Boys
The Nightingale
============TOP  5 BOOKS READ BY USER===============
The Girl Who Played with Fire (Millennium, #2)
Jitterbug Perfume
Blood Red Road (Dust Lands, #1)
The Dovekeepers
Fever Pitch
```

5. Discussion

After creating the tables, storing all relevant information from the books database, and analyzing customers' preference and ratings for different books, we can generate personalized content for our current customers. Everytime the current customers making a new purchase or renting of books will update their information in our database, and this helps us improve the quality of future recommendations.

In addition, after we train the recommender system to be applicable, we can collect more potential clients data to come up with better marketing strategies by providing higher-quality promotions and suggestions to the potential clients. In another dimension, it greatly saves the cost of time and increases the accuracy. Also our recommendation system is based on collaborating filtering. After testing the performance of this recommendation system, it might be better in future to combine collaborating filtering and content based recommendations to build hybrid recommendations.

The two perspectives above implies that the model we built and analysis we did will gain more competitiveness and reputations in the markets. As customers' satisfaction rate goes up, the retention rate will also increase to help us retain more profitability.

The weakness of our database and model is that they really rely on accurate and abundant information about books and clients. If the future data is not clear and structured, the cost of analyzing it and updating it to the database might be higher, and it may also result in less accurate recommendations. On the other hand, our database is limited to the books we have, which subsequently limits our database update. It will be practical if we can get part streaming data from online book orders with more flexible database schema. The only threat we have to confront is the cost of maintaining the database, therefore, the tradeoff between the ‘latitude’ of our database and economical aspects would be an important question to focus on in the implementation of model and analysis.

Reference

1. A. Akbik, D. Blythe, and R. Vollgraf, “Contextual String Embeddings for Sequence Labeling,” *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1638–1649, Aug. 2018.
2. W. Koehrsen, “Neural network embeddings explained,” *Medium*, 02-Oct-2018. [Online]. Available: https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526. [Accessed: 06-Dec-2022].