Amazon Co-purchasing Analysis

Randolph Zhao (yxz1648)

CSDS 234 (Fall 2021)

Case Western Reserve University

***Abstract*—In this project, we have built a recommendation system that integrates a prediction model of co-purchasing patterns. We organized the unstructured Amazon review data set into three tables and developed a query engine for these tables. We used logistic regression to develop a co-purchasing pattern prediction model and achieved an accuracy of 92.7%. Using this prediction model, we designed a recommendation system that provides ranked potential products or customers and adapts well to new products. We believe that our recommendation system will benefit from the prediction model and help increase the company's sales revenue.**

Keywords—query engine, co-purchasing analysis, logistic regression, recommendation system

I. Introduction

Recommendation systems are a key functionality of various websites and companies to increase their revenues. To further improve the performance of recommendation systems, we need to analyze the co-purchasing patterns. Co-purchasing patterns are patterns of products that if customers buy one, they will buy the other. In other words, if we can successfully recognize the link and the strength of the link of the co-purchasing pattern, we can improve the performance of the recommendation system. As machine learning has been paid increasing attention, we want to explore how machine learning models of co-purchasing patterns can help to improve recommendation systems. In this project, we introduced the Amazon Reviews dataset and built a query engine, co-purchasing prediction model, and a sample recommendation system integrated with the trained model.

II. Related Work

A. Amazon Reviews Datasets

The Amazon Reviews dataset [1] includes a text document that includes unstructured information about products, reviews, and co-purchasing edges. In this dataset, there are 548,552 products, 7,781,990 reviews, and 1,788,725 co-purchasing edges. As a data record example showed in Fig. 1., *ASIN* represents the Amazon Standard Identification Number, *similar* represents ASINs of co-purchased products, and other attributes are self-explained.

ASIN: 1559362022
title: Wake Up and Smell the Coffee
group: Book
salesrank: 518927
similar: 5 1559360968 1559361247 1559360828 1559361018 0743214552
categories: 1
|Books[283155]|Subjects[1000]|Literature & Fiction[17]|Drama[2159]|United States[2160]
reviews: total: 8 downloaded: 4 avg rating: 4
2002-5-13 customer: A2IGOA66Y6O8TQ rating: 5 votes: 3 helpful: 2
2002-6-17 customer: A2OIN4AUH84KNE rating: 5 votes: 2 helpful: 1
2003-1-2 customer: A2HN382JNT1CIU rating: 1 votes: 6 helpful: 1
2003-6-7 customer: A2FDJ79LDU4O18 rating: 4 votes: 1 helpful: 1

Fig. 1. Example of data records in Amazon Reviews Dataset

B. Logistic Regression Model

Logistic Regression is a frequently used linear classification method. It trains coefficients of each feature to minimize its loss function and adapts L2 regularization to reduce overfitting as Eq. (1).

$$w, b = argmin\ \frac{1}{2}\|w\|^2 + C \times loss\ function \quad (1)$$

Compared to other machine learning models, it trains faster and easier, makes no assumptions about data distribution, and has relatively good accuracy compared to other linear models.

In this project, we will compare two optimization algorithms: Stochastic Average Gradient (SAG) [2] and SAGA [3], the optimized variant of SAG. They both support binary classification and, most importantly, converge faster on large datasets. SAG saves a gradient $f'(\cdot)$ for each sample, select randomly one sample to update step size d, and uses step size to update variable x (Fig.2.). Based on SAG, SAGA stores the gradient of previous iterations and pick randomly to calculate the update direction (Fig.3.).

Algorithm 1 Basic SAG method for minimizing $\frac{1}{n}\sum_{i=1}^{n} f_i(x)$ with step size α.

$d = 0$, $y_i = 0$ for $i = 1, 2, \ldots, n$
for $k = 0, 1, \ldots$ **do**
  Sample i from $\{1, 2, \ldots, n\}$
  $d = d - y_i + f_i'(x)$
  $y_i = f_i'(x)$
  $x = x - \frac{\alpha}{n} d$
end for

Fig. 2. Pseudo-code of SAG [2]

Algorithm 5.2 SAGA Method for Minimizing an Empirical Risk R_n

Choose an initial iterate $w_1 \in \mathbb{R}^d$ and stepsize $\alpha > 0$.
for $i = 1, \ldots, n$ **do**
  Compute $\nabla f_i(w_1)$.
  Store $\nabla f_i(w_{[i]}) \leftarrow \nabla f_i(w_1)$.
end for
for $k = 1, 2, \ldots$ **do**
  Choose j uniformly in $\{1, \ldots, n\}$.
  Compute $\nabla f_j(w_k)$.
  Set $g_k \leftarrow \nabla f_j(w_k) - \nabla f_j(w_{[j]}) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_{[i]})$.
  Store $\nabla f_j(w_{[j]}) \leftarrow \nabla f_j(w_k)$.
  Set $w_{k+1} \leftarrow w_k - \alpha g_k$.
end for

Fig. 3. Pseudo-code of SAGA [4]

III. Search Engine

To simplify the data management and access, we organized the raw data as three Python Pandas Dataframes.

A. Data Transformation

For different purpose, we organized the raw data into three tables:

- Products (**product_id**, title, group, salesrank, co_purchased_num, categories, reviews_num,

avg_rate, first_rating_time, last_rating_time, highest_rate, lowest_rate)

- Reviews (**product_id, customer_id**, votes_num, helpful_num, time, rating)
- Co-purchased (**first, second**)

, which can be represented as Entity-Relation Diagram as Fig.4. The tables of products and reviews are self-explained. The two attributes of the Co-purchased table are both referred to product_id. In other words, the Co-purchased table includes all co-purchasing product pairs.

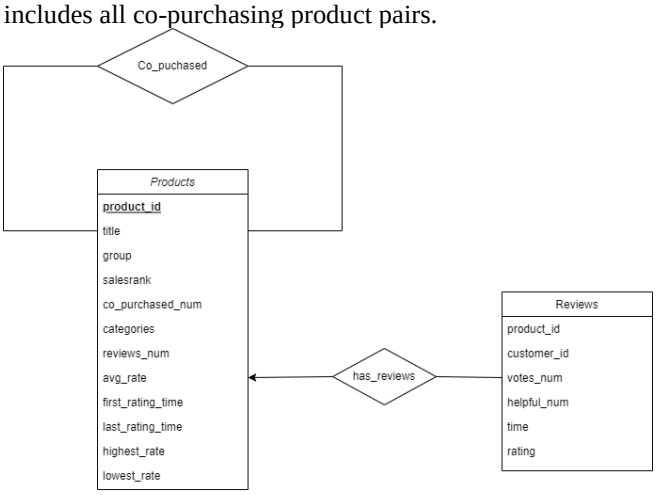

Fig. 4. Entity-Relation Diagram

B. *Search Engine*

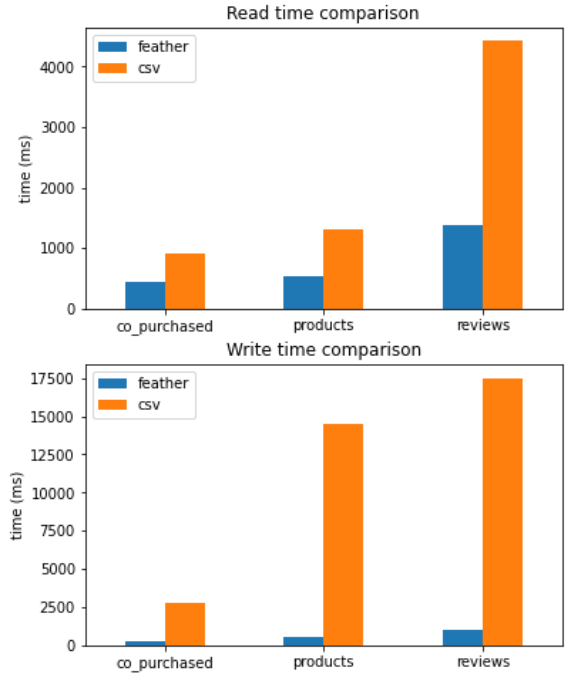

Fig. 5. Time Comparison

Python Pandas library provides flexible and fast access to Dataframes, which supports various operations such as selecting attributes, joining tables, and filtering by conditions.

Since these three tables are well-structured, they will be more space-efficient and faster-to-save-or-load using Feather file format. As our results showed in Fig.5. and Fig.6., the Feather format is more space-efficient and faster compared to the CSV format when storing our three tables: co_purchased, products, and reviews.

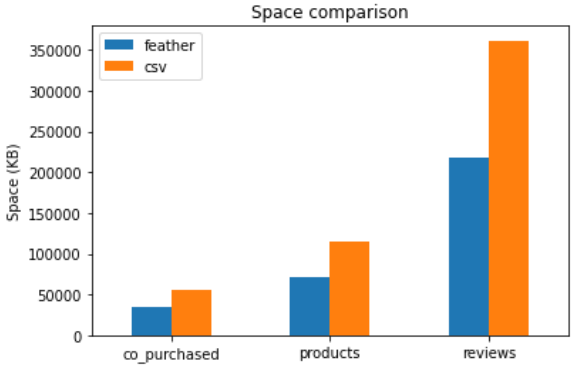

Fig. 6. Space Comparison

C. *Example Queries*

1) Select the IDs of products which has avg_rate>4 and lowest_rate>3.

- Python Codes:

products.loc[(products['avg_rate'] > 4) & (products['lowest_rate'] > 3), 'product_id']

- Results: Fig.7.

```
0         0827229534
2         0486287785
5         B00000AU3R
8         0871318237
15        3895780812
             ...
402716    B000065AHM
402718    1930519206
402719    B000059TOC
402722    B00008DDST
402723    B00005MHUG
Name: product_id, Length: 175092, dtype: object
```

Fig. 7. Results of 1st example query

2) Select the group of the products that are bought by the customer 'A11NCO6YTE4BTJ'.

- Python Codes:

products.loc[products['product_id'].isin(reviews.loc[reviews['customer_id'] == 'A11NCO6YTE4BTJ', 'product_id']), 'group']

- Results: Fig.8.

```
1         Book
29010     Book
94978     Book
102197    Book
127105    Book
171963    Book
Name: group, dtype: object
```

Fig. 8. Results of 2nd example query

IV. Co-purchasing Pattern Identifier

A. *Data Preparation*

For each product, we used features salesrank, co_purchased_num, categories, reviews_num, avg_rate, first_rating_time, last_rating_time, highest_rate, lowest_rate.

From categories, we further calculated their similarity as the fraction of shared common categories shown in Eq. (2).

$$Categ\_similarity = \frac{|category(p1) \cap category(p2)|}{|category(p1) \cup category(p2)|} \quad (2)$$

For numeric features, we calculated as differences of features between two products, including sales_diff, co_puchased_diff, reviews_num_diff, avg_rating_diff, first_time_diff, last_time_diff, highest_rating_diff, and lowest_rating_diff.

Predicting whether having a co-purchasing pattern is a binary classification task, with co-purchasing as label=1 and non-co-purchasing as label=0. Since we only have co-purchasing pattern records, we randomly generated non-co-purchasing pairs. We also make sure that there are 50% co-purchasing pairs in the training dataset so that the training dataset is class balanced.

B. *Data Preprocessing*

After preparing our training dataset, there are 2,013,128 records, each of which has 28 attributes. Among 28 attributes, there are two categorical features, twenty-three numerical features, and our binary class label. For categorical features, we transformed them with a one-hot encoder for our model. For numerical features, we rescaled them into the range [0,1] using MinMaxScaler.

C. *Modeling*

After preprocessing, there are 45 features and a binary class label in the training dataset. We have not filtered useless or not very important features yet. Thus, we implemented feature selection using Recursive Feature Elimination (RFE). RFE will generate combinations with different numbers of features and calculate the corresponding performance in order to find the best feature combination. According to Fig. 9., the combination of 20 features results in the highest accuracy scores.

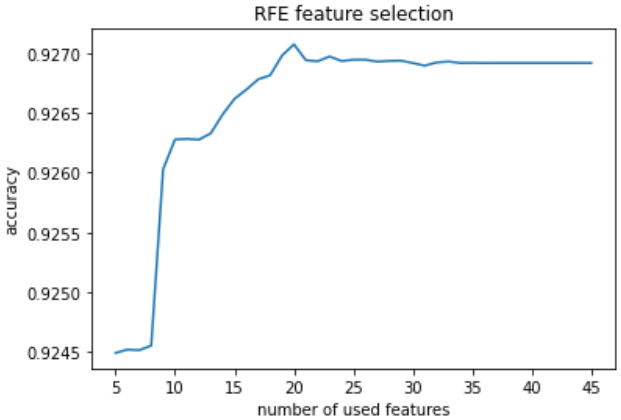

Fig. 9. Recursive Feature Elimination with Accuracy

To further improving performance, we also fine-tuned hyperparameters, including the regularization term (C), max iterations (max_iter), solver (solver), and error tolerance (tol). Using GridSearchCV, we resulted in {'C': 0.1, 'max_iter': 1000, 'solver': 'sag', 'tol': 1e-06}. Thus, we will use SAG as the optimization algorithm with regularization term=0.1 amd error tolerance=1e-06.

D. *Performance Measure*

We first trained on 90% data and tested on the rest 10% data. The resulting accuracy is 92.8%.

We further measure the accuracy scores based on 5-fold cross-validation. As the Fig.10. shows, the model results in 92.7% mean accuracy with 0.03% standard deviation.

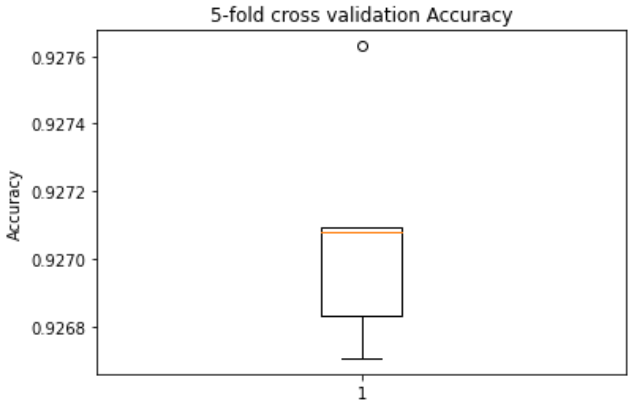

Fig. 10. 5-fold cross-validation accuracy

Since our training dataset is balanced (50% True, 50% False), 92.7% accuracy is much higher than random choice (50% accuracy). Thus, we believe our model fits and predicts the co-purchasing pattern very well.

E. *Model Analysis*

We also analyzed our prediction model further. As plotted in Fig.11., we can see several highly important features, including category similarity, salesrank difference, and co-purchasing numbers.

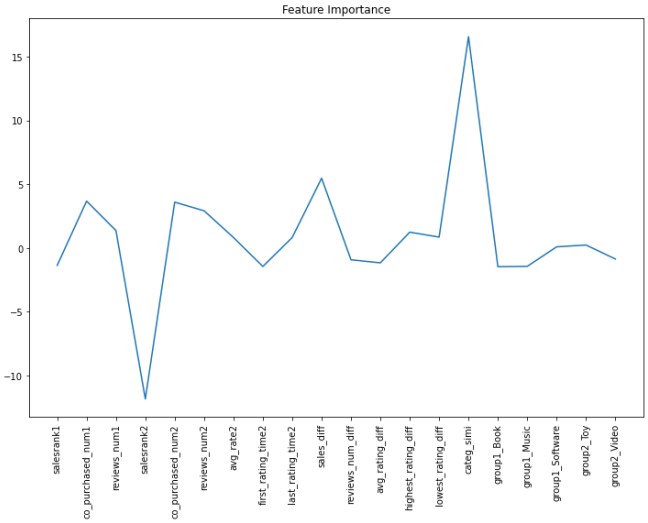

Fig. 11. Feature Importance Analysis

V. Recommendation System

Since our main purpose is to improve the performance of our recommendation system by using the co-purchasing pattern prediction model, here is a proposed sample recommendation system that uses our prediction model.

A. *Ranked Recommendations*

In traditional solutions, there are no rankings in the set of recommended products or customers. However, if we can know whether the co-purchasing pattern is strong or weak, we can put more recommendation resources to attract those

customers for stronger links, which may increase the probability of success in recommendations.

The co-purchasing pattern prediction model provides the information needed to identify the strength of the co-purchasing patterns. The logistic regression model will calculate learned coefficients and result in two probabilities for each class label as Eq. (3) and Eq. (4). The probability of the positive class calculated by the model shows how strong the model thinks there is a co-purchasing link, i.e., the link strength. Thus, we can estimate the strength of the co-purchasing link using the probability of the positive class *y=1* (i.e., having co-purchasing links). For example, the larger the probability of positive (i.e., having a co-purchasing pattern), the stronger the co-purchasing link is.

In Section IV. D. *Performance Measure*, we have measured the accuracy of the model on predicting/identifying a co-purchasing pattern between two products. Thus, we believed the model can provide accurate probability/link strength measurement for our recommendation system and there is no need to validate it again.

$$p(y = 1|x) = \frac{1}{1+e^{-(wx+b)}} \quad (3)$$

$$p(y = 0|x) = 1 - p(y = 1|x) \quad (4)$$

1) Find Potential Products

Problem Statement:

- The website wants to recommend several potential products to a specific customer.
- Given customer A, find the potential products that customer A will be interested in.

Solution:

- Find all products bought by A as a set B, find all co-purchasing products of B as C
- For every pair (b*i*, c*i*) in (B, C), use the pre-trained model to calculate the probability of having co-purchasing link p*i*.
- Rank C by p*i* and return the ranked C.

2) Find Potential Customers

Problem Statement:

- The website wants to recommend a product to some potential customers.
- Given a product A, find the potential customers who have a high probability to buy product A

Solution:

- Find all co-purchasing products of A as B, find all customers who bought B as C, delete the customers who bought A from C as C'.
- For every pair (a*i*, b*i*) in (A, B), use the pre-trained model to calculate the probability of having co-purchasing link p*i*.
- Rank (C', B) by p*i* and return the ranked C'.

B. Recommendations for new products

Another common difficulty of traditional recommendation systems is that they cannot make recommendations for new-coming products since there are no historical links. However, the co-purchasing pattern prediction model perfectly solved this problem. Using the product information, the prediction model can provide the probability of having co-purchasing links between the new product and all existing products, which we can select the highest products as recommendations.

Since we want to make sure the model can rank all products to provide the most likely co-purchased products, we expected there is an explicit difference in the probabilities of having co-purchasing links between co-purchased pairs and non-co-purchased pairs. Therefore, I randomly selected 10 products, paired them with the whole products dataset, and compared the mean probability of having co-purchasing links between co-purchased pairs and non-co-purchased pairs as shown in Fig.12. Due to the incomplete products information, several mean probabilities of having the co-purchasing link of co-purchased pairs are not very determined (not greater than 0.5). However, compared to the mean probability of having co-purchasing links of non-co-purchased pairs, we can see there is an explicit difference of probability (or link strength). Thus, we believe our model can be used to predict potential co-purchased products for new-coming products.

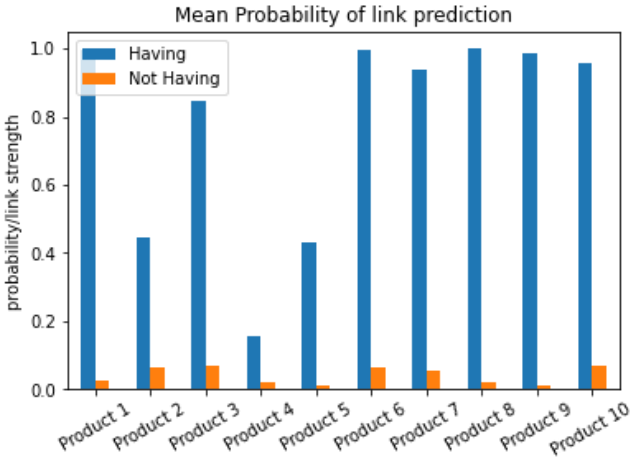

Fig. 12. Mean Probability Comparison

1) Find Potential Customers for new-coming products

Problem Statement:

- The website wants to recommend a new-coming product to some potential customers.
- Given a new-coming product A, find the potential customers who have a high probability to buy product A

Solution:

- Let all existing products as set B.
- For every pair (A, b*i*) in (A, B), use the pre-trained model to calculate the probability of having co-purchasing link p*i*.
- Rank (B, A) by p*i*. Select top 5 products in sorted B and find all customers of these products as C.
- Return C.

C. Example Scenarios

1) Find potential products for the customer 'A2JW67OY8U6HHK'.

- Results: Fig.13.

2) Find potential customers for the product '0385411472'.

- Results: Fig.14.

3) Find potential customers for the new-coming product '1930519206'.

- Results: Fig.15.

	potential_products	strength
0	0805415505	99.99%
1	0687023955	99.99%
2	0804215715	97.64%
3	156101074X	1.36%
4	082721619X	0.89%
5	0802842623	0.00%
6	title0687074231	0.00%

Fig. 13. Results of 1st example scenario

	potential_customers	strength
0	A9Y05AGWAVY9V	100.00%
1	A31TVU8SBCCYHQ	100.00%
2	A1ZXXCE6SRQM9I	100.00%
3	A37I2PJG90RAZO	100.00%
4	A11D59W5BBPJPK	100.00%
...	...	...
56	AID6X4N2TXYGE	90.85%
57	A32EH1093C06O7	0.00%
58	A140XH16IKR4B0	0.00%
59	A2ZZM7Q55ZZEXY	0.00%
60	A3QDZOPT4HGS9S	0.00%

61 rows × 2 columns

Fig. 14. Results of 2nd example scenario

	potential_customer	strength
0	A1VCFMW5VP3JXN	99.82%
1	A3H48OXCO0UE7L	99.82%
2	A1PUJ0NFE9BIZC	99.82%
3	A3195MG8D8VKQJ	99.82%
4	A3807IWWRU1V0D	99.82%
5	ATVPDKIKX0DER	99.82%
6	A298SQ7KBSOMKC	99.82%
7	A1T1MKDHCNCG06	99.82%
8	A2WWGDZYCHOAEB	99.80%
9	A1C19UVH4PAFX8	99.80%
10	A3CASRHRFQANPT	99.80%
11	A2CEXF8C3BLIX4	99.80%
12	A20LFEBJE5ENIC	99.80%
13	AKPOHXIGYE2XI	99.80%
14	AW2J7RAZJ7IRC	99.80%
15	A145221D4IA2XT	99.80%
16	A852WJH90DK24	99.79%
17	A22W4JHPMB3RMI	99.79%
18	A2FRKEXDXDN1KI	99.79%

Fig. 15. Results of 3rd example scenario

VI. Conclusions

In this project, we successfully built a query engine, a co-purchasing link prediction model, and a sample recommendation system. We first transformed a large-scaled unstructured Amazon Reviews dataset into three tables and used the Python Pandas Dataframe to access them. We developed a fine-tuned logistic regression model based on preprocessed training dataset to predict co-purchasing links. With the prediction model, we further built a recommendation system that uses the prediction model to estimate the strength of co-purchasing links in order to rank the importance of the recommendations.

Although the recommendation system and the model are very naïve, it proves the success of our idea to combine machine learning models for co-purchasing pattern analysis with the recommendation system. It shows these pre-trained models can provide various useful information which is able to help to improve the performance of recommendation systems and to further increase the company revenue.

Appendix

All codes and results can be accessed in my GitHub repository https://github.com/vanity-lost/academic_projects/tree/main/CSDS%20234%20Projects .

Acknowledgment

This project is completely designed and conducted by Randolph Zhao.

References

[1] J. Leskovec, L. Adamic and B. Adamic. The Dynamics of Viral Marketing. ACM Transactions on the Web (ACM TWEB), 1(1), 2007.

[2] Schmidt, M., Le Roux, N. & Bach, F. Minimizing finite sums with the stochastic average gradient. Math. Program. 162, 83–112 (2017).

[3] Defazio, A., Bach, F., and Lacoste-Julien, S., “SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives”, 2014.

[4] Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. SIAM Review, 60(2), 223-311. https://doi.org/10.1137/16M1080173