November, 2018

EB5202: Web Analytics

# RecSys Challenge 2015


*Submitted By:*

| | |
|---|---|
| Apurv Garg | (A0178205E) |
| Bhabesh Senapati | (A0178349M) |
| Daksh Gupta | (A0178644M) |
| Dibyajyoti Panda | (A0178271Y) |
| Tanmoy Chakraborty | (A0178252B) |
| Zhang Xiaoman | (A0178489A ) |

INSTITUTE OF SYSTEMS SCIENCE

# Table of Contents

*A. Explanatory Data Analysis* ..... 3

*B. Model Build and Test Processes* ..... 5

*C. Association and Sequence found* ..... 8

*D. Recommendations* ..... 10

*E. Performance and Results* ..... 12

*F. Appendix* ..... 12

# Executive Summary

In this challenge, YOOCHOOSE is providing a collection of sequences of click events; click sessions. For some of the sessions, there are also buying events. The goal is to make product and page recommendations. If a user viewing product/page(A) recommend them product/page(B)- product the user is likely to buy or page they would like to view. Such an information is of high value to an e-business as it can indicate not only what item to suggest to the user but also how it can encourage the user to become a buyer. For instance, to provide the user some dedicated promotions, discounts etc. The data represents six months of activities of big e-commerce businesses in Europe selling all kinds of stuffs such a garden tools, toys, clothes, electronics and much more.

Each record or line in the file has the following fields:

| Click Events | Explanations |
|---|---|
| **Session ID** | The ID of the session. In one session there are one or many clicks |
| **Time Stamp** | The time when click occurred |
| **Item ID** | The unique identifier of item |
| **Category** | The category of item. |

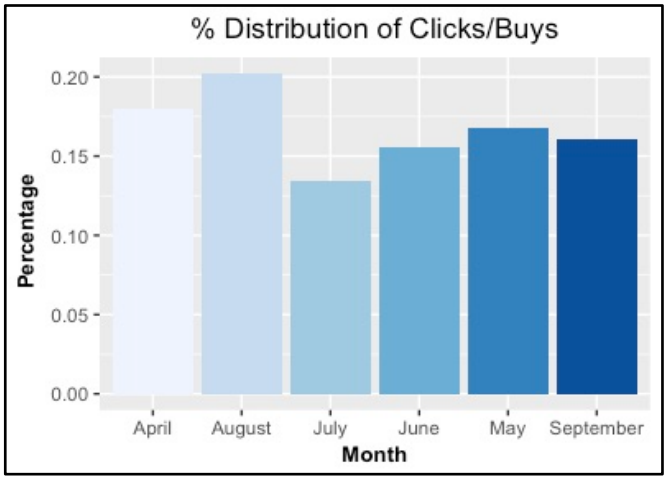
| Buy Events | Explanations |
|---|---|
| **Session ID** | The ID of the session. In one session there are how many buying events |
| **Time Stamp** | The time when the buy occurred |
| **Item ID** | The unique identifier of item |
| **Price** | The price of item |
| **Quantity** | How many of these items were brought |

As a solution to this challenge we plan to 1st prepare the data having complete session transactions. We aim at deriving accurate rules for a sequence of clicks to be a buy sequence. This will help the e-retail store get more insights into the patterns resulting in a buy, hence fetching them more business. Thereafter we extract only the buy sequences from the data and build association rules after breaking the entire data into 80-20 train test split, and test the derived rules on the 20% unseen training data. The rules formed are **72.6%** precise.

For sequencing we considered the entire session as a buy even if one of the clicks being buy. Similarly , we used the cspade algorithm under the R Package arulesSequences to search for frequently occurring combinations below the support level of 0.001. After that we did rule induction to find rules with confidence above the threshold of 0.4 and with lift above 1. The support and confidence are adjusted to investigate the effect of different thresholds.

We have also build a Recommendation system wherein we have considered Session IDs as User IDs. We have made 10 levels of various price ranges and have recommended the user to visit various levels if he/she visits a particular level.

# A. Explanatory Data Analysis




The Illustration shows the distribution of total clicks in each month irrespective of whether it is a buy or a no buy. We can observe an equally distributed online traffic across months – with the traffic being on a higher end for **August** and **April**.

*Illustration 1: Percentage distribution of web traffic/month*

This Illustrations shows the percentage distribution of the total web traffic across the days of the week. We can expect a high traffic on **Sundays** and **Mondays**, their conversion into a successful buyer is not certain though.

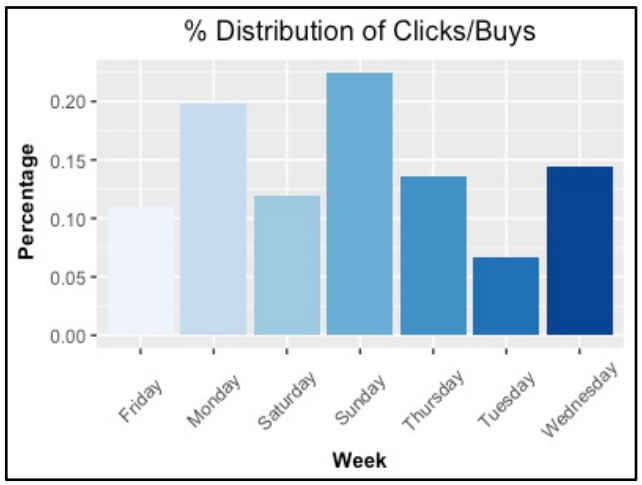
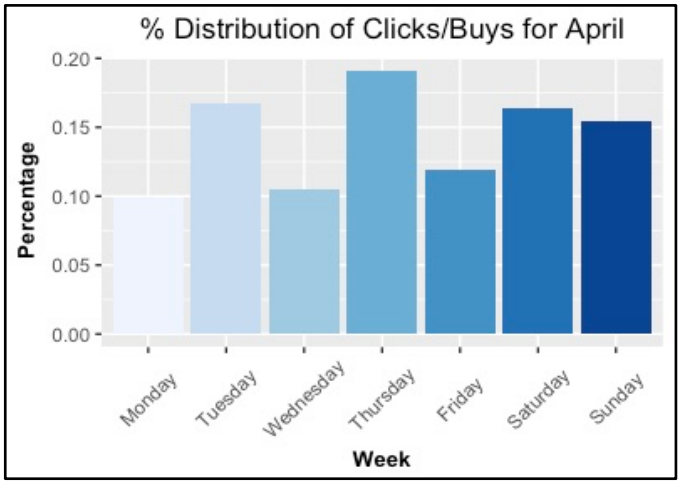



*Illustration 2: Percentage distribution of web traffic/day of week*

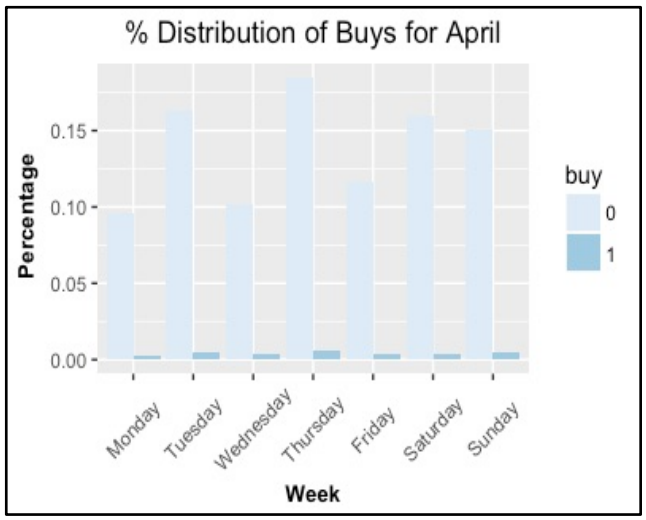



This is the percentage distribution of the total clicks per day of the week for our **sample data** (limited data used for making association rules) i.e just for the month of **April**. We observe the traffic increasing as the weekend approaches.

*Illustration 3: Percentage distribution of web traffic/day of week for April*

Out of the total traffic we can see a comparison between the successful converts and the unsuccessful converts for the retail store. Figures have dropped significantly for Monday.

*Illustration 4: Percentage distribution of Buy vs No-Buy sessions of the total traffic/day of week in April*



# B. Model Build and Test Processes

## B1. Details of data cleaning and pre-processing performed

1) The two different sets of data files provided had the log details for every click and every session in the specific format. All the clicks corresponding to a buy were stored in the file names buy.dat and the rest of clicks were stored in clicks.dat.
2) Both the data files were combined in order to get all the events (irrespective of buy/no buy) in every session. A dummy variable was created which indicates whether the session is a buy/no-buy session. A buy session considered having at least one buy click
3) There were duplicate events observed in the data which were removed
4) The size of data files cumulatively exceeded ~ 1.5GB which made the computations difficult within the R tool. Hence sampling was done on the data based on Time stamp. The available data was for 6 months starting from April till September. The number of sessions were found to be almost equally distributed among all the months, as seen in the Illustration below:

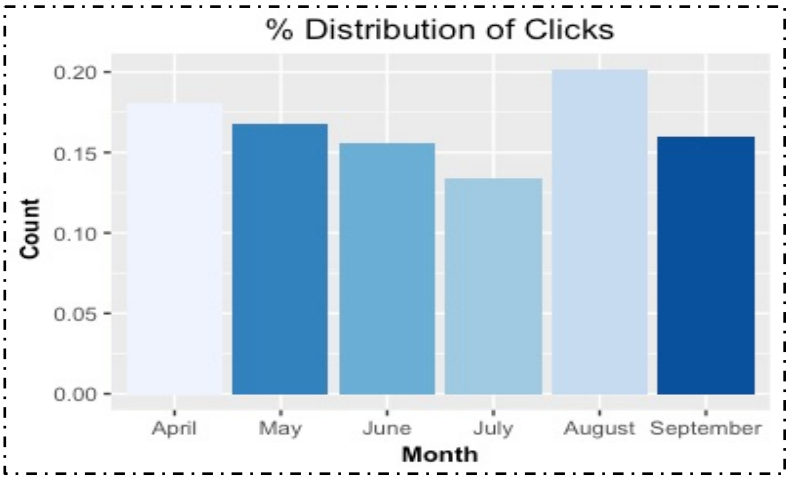



*Illustration 5: Monthly distribution*

5) The click sessions for the month of April were considered to form rules and sequences for the Product/ Pages
6) Our objective is to find most frequent rules for the sessions resulting in a buy. The retailers will get an insight into the sequence of sessions which are favorable for their portal. Hence, we divided the data sets into sets of two different sessions.
    a) All session sequences resulting in a buy (Used to derive rules)
    b) All session sequences resulting in a no-buy

## B2. Tools and algorithm used? Problems(if-any) faced

Tools Used: **R**

Algorithms used:

The main algorithm that we used in the whole project is association mining. Treat the basic training and testing set of "buys" and "no buys" Item IDs in a shopping basket, then use Market Basket Analysis (MBA) techniques

1) **Apriori Algorithm** for forming association rules for buy and Non buy transactions
2) **cSpade Algorithm** for mining frequent sequential patterns. This algorithm utilizes temporal joins along with efficient lattice search techniques and provides for timing constraints.

   *Syntax:* *cspade(data, parameter = NULL, control = NULL, tmpdir = tempdir())*

3) **ruleInduction algorithm** which provides the generic function and the needed S4 method to induce all rules which can be generated by the given set of item sets from a transactions dataset. This method can be used to create closed association rules.

   *Syntax:* ruleInduction*(x, transactions, confidence = 0.8, control = NULL)*

## B3. What settings did you use, e.g. state if you reduced or raised minimum rule confidence and support before model build?

After getting all the possible associations from the transactions we filter these associations based on the selected value of support and confidence. Our intent is to select the most optimal values for these parameters. To start off we set support as 0.5 and confidence as 0.1 and failed to get any set of rules from the association set.

Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the antecedents appear in the database, therefore we assumed that greater the total number of items in the database lesser will be the support value for every basket of association rule formed. Hence a smaller value of support will help us filter rules.

Confidence is the ratio of the number of transactions that include all items in the consequent as well as the antecedent to the number or transactions that include all the items in the antecedent. Therefore, we could interpret confidence as the authenticity of an association rule. Hence higher the value of confidence more significant the rule is.

The minimum value of **support** was dropped to **0.001** (due to large number of total items) and that of **confidence** was set to **0.5**. These values of the filtering parameters helped us fetch **703 significant rules** for the buying sequence of clicks. The scatter plot of the rules can be seen in the illustration below

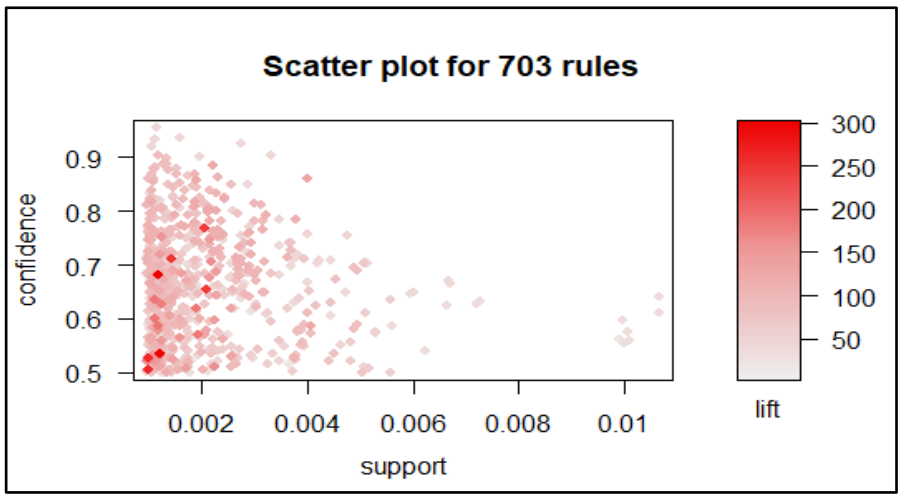



*Illustration 6: Scatter plot of buy rules with support, confidence and lift parameters*

Coming to Sequence Rule mining, it finds typical sequences of events in the data. Sequence rule models contain various sequence rules. A sequence consists of a previous item or item set in the rule body that leads to a consecutive item set in the rule head. The consecutive item set occurs after a particular period of time.

The minimum value of **support** was dropped to **0.001** (due to large number of total items) and that of **confidence** was set to **0.4**. These values of the filtering parameters helped us fetch **300 significant rules** for the buying sequence of clicks

### B4. Details of how you split the data into training and test tests and how you performed testing?

The data was divided in a 80-20 split, training the model on 80 percent data and testing on the remaining unseen 20 percent data. A set of significant rules were generated from the train data (having buy sequences only). Predictions for the test data were made by the model taking the item ids in a sequence as input as antecedent and giving the output as the consequent.

The predictions were recorded in a data frame and were compared with the actual values to get the correct number of predictions. There were 7965 correct predictions out of 10974 total predictions. The precision was calculated by the following formula:

Precision = (Correct Predictions *100)/(Total Predictions)

The precision was calculated to be ***72.58 %.***

# C. Association and Sequence found

## C1. Top Association found

1. Find rules of Lift ranking

```
> inspect(ordered_rules_buy[1:10])
     lhs             rhs          support     confidence lift     count
[1]  {214553533} => {214716671} 0.001203597 0.6793893  300.8230  89
[2]  {214716671} => {214553533} 0.001203597 0.5329341  300.8230  89
[3]  {214826994} => {214826921} 0.001014267 0.5244755  262.0429  75
[4]  {214826921} => {214826994} 0.001014267 0.5067568  262.0429  75
[5]  {214716707} => {214716710} 0.001433498 0.7114094  244.6752 106
[6]  {214829045} => {214829336} 0.002082629 0.7700000  241.2612 154
[7]  {214829336} => {214829045} 0.002082629 0.6525424  241.2612 154
[8]  {214709685} => {214829737} 0.001190074 0.5866667  186.9874  88
[9]  {214718169} => {214821022} 0.001135979 0.6363636  185.9917  84
[10] {214821390} => {214718203} 0.001149503 0.5985915  185.2002  85
```

2) Find rules of Confidence ranking

```
> inspect(ordered_rules_buy[1:10])
     lhs                                 rhs          support     confidence lift     count
[1]  {214839995,214839999}            => {214839313} 0.001135979 0.9545455  42.54603  84
[2]  {214839997,214839999}            => {214839313} 0.001622828 0.9375000  41.78628 120
[3]  {214839999,214840001}            => {214839313} 0.001149503 0.9340659  41.63322  85
[4]  {214839995}                      => {214839313} 0.002785854 0.9279279  41.35963 206
[5]  {214839997,214840001}            => {214839313} 0.001054838 0.9176471  40.90139  78
[6]  {214698446,214837442,214837490} => {214837485} 0.001163027 0.9052632  89.73148  86
[7]  {214840001}                      => {214839313} 0.001987964 0.9018405  40.19686 147
[8]  {214839999}                      => {214839313} 0.003353844 0.9018182  40.19587 248
[9]  {214717877,214826621}            => {214826615} 0.001311786 0.8981481  71.18281  97
[10] {214698446,214837442,214837487} => {214837485} 0.001271215 0.8867925  87.90063  94
```

3) Find rules of Support ranking

```
> inspect(ordered_rules_buy[1:10])
     lhs                          rhs          support     confidence lift     count
[1]  {214834880}               => {214826610} 0.010683616 0.6391586  36.60928 790
[2]  {214826610}               => {214834880} 0.010683616 0.6119287  36.60928 790
[3]  {214826610}               => {214834877} 0.010048009 0.5755229  31.97373 743
[4]  {214834877}               => {214826610} 0.010048009 0.5582269  31.97373 743
[5]  {214834880}               => {214834877} 0.009980391 0.5970874  33.17177 738
[6]  {214834877}               => {214834880} 0.009980391 0.5544703  33.17177 738
[7]  {214821302}               => {214821305} 0.009926297 0.5620214  20.80014 734
[8]  {214833755}               => {214834877} 0.007289201 0.6296729  34.98209 539
[9]  {214833755}               => {214826610} 0.007248631 0.6261682  35.86523 536
[10] {214826610,214834880} => {214834877} 0.006680641 0.6253165  34.74006 494
```

4) Find rule for a specific item e.g. 214834880 with high confidence

```
> specificrules <- subset(rules_buy, items %in% c("214834880")& confidence > 0.7)
> inspect(specificrules[1:10])
     lhs                                rhs         support     confidence lift     count
[1]  {214829366,214834880}           => {214826610} 0.002596524 0.7032967  40.28294 192
[2]  {214829387,214834880}           => {214826610} 0.003989452 0.7583548  43.43652 295
[3]  {214829387,214834880}           => {214834877} 0.003746027 0.7120823  39.56042 277
[4]  {214833755,214834880}           => {214826610} 0.004719724 0.7537797  43.17447 349
[5]  {214833755,214834880}           => {214834877} 0.004422206 0.7062635  39.23716 327
[6]  {214821412,214826610,214834877} => {214834880} 0.001041314 0.7333333  43.87244  77
[7]  {214826589,214833755,214834880} => {214826610} 0.001027791 0.7600000  43.53075  76
[8]  {214826589,214833755,214834880} => {214834877} 0.001027791 0.7600000  42.22254  76
[9]  {214829366,214829387,214833755} => {214834880} 0.001230644 0.7109375  42.53258  91
[10] {214829366,214829387,214834880} => {214833755} 0.001230644 0.7109375  61.41387  91
```

## C2. Top Sequences found

1) Rules of Lift Ranking

```
> lift
                                          rule     support confidence     lift
1  <{214826705},{214826608}> => <{214826705}> 0.001015774  0.4755245 47.37130
2              <{214821290}> => <{214821290}> 0.004346917  0.4497682 46.53675
3  <{214717003},{214717003}> => <{214717003}> 0.001792543  0.4137931 45.56080
4              <{214826705}> => <{214826705}> 0.004346917  0.4330357 43.13861
5              <{214826700}> => <{214826801}> 0.001613289  0.4060150 42.46917
6              <{214826803}> => <{214821285}> 0.002136114  0.4121037 41.54800
7              <{214839313}> => <{214839313}> 0.008574331  0.5700099 37.89349
8              <{214821277}> => <{214821277}> 0.009963552  0.4812410 23.24401
9  <{214821277},{214821285}> => <{214821277}> 0.001374283  0.4742268 22.90522
10 <{214821277},{214821292}> => <{214821277}> 0.001090464  0.4562500 22.03694
```

2) Rules of Confidence Ranking

```
> confidence
                            rule     support confidence      lift
1  <{214827000}> => <{214826925}> 0.001717854  0.4121864  88.72477
2  <{214826803}> => <{214821285}> 0.002136114  0.4121037  41.54800
3  <{214826816}> => <{214826816}> 0.001583413  0.4108527 106.60513
4  <{214601040}> => <{214601040}> 0.002136114  0.4097421  78.59535
5  <{214832655}> => <{214832655}> 0.001389221  0.4078947 119.76362
6  <{214601042}> => <{214601042}> 0.001374283  0.4070796 120.58203
7  <{214826700}> => <{214826801}> 0.001613289  0.4060150  42.46917
8  <{214717007}> => <{214717007}> 0.001822419  0.4039735  89.54835
9  <{214837485}> => <{214837485}> 0.002434871  0.4034653  66.85541
10 <{214826925}> => <{214826925}> 0.001867232  0.4019293  86.51689
```

3) Rules of Support Ranking

```
> support
                                        rule     support confidence      lift
1  <{214826803},{214821285}> => <{214821285}> 0.001150215  0.5384615  54.28730
2              <{214594678}> => <{214594678}> 0.001105402  0.5441176 267.83391
3              <{214716937}> => <{214716937}> 0.001105402  0.4277457 165.52026
4  <{214826803},{214821285}> => <{214826803}> 0.001105402  0.5174825  99.83386
5  <{214821272},{214821272}> => <{214821272}> 0.001090464  0.5748031 209.12838
6  <{214821277},{214821292}> => <{214821277}> 0.001090464  0.4562500  22.03694
7              <{214718203}> => <{214718203}> 0.001060588  0.4671053 205.72299
8              <{214716977}> => <{214716977}> 0.001030712  0.5000000 242.55072
9  <{214827000},{214827000}> => <{214827000}> 0.001030712  0.4662162 111.86516
10 <{214826705},{214826608}> => <{214826705}> 0.001015774  0.4755245  47.37130
```

# D. Recommendations

Two methods have been used here to recommend Items to User. We have categorized the Items according to their prices and levelled them separately making different buckets as shown below. We have then calculated the Error (RMSE, MSE, MAE) of the recommendation method.

1. **Item Based Collaborative Filtering (IBCF)** recommends items on the basis of the similarity matrix. This algorithm is efficient and scalable.
2. **User Based Collaborative Filtering (UBCF)** recommends items by finding similar users to the active user (the person we want to recommend item).

We have used "***yoochoose-buys***" data here and considered the Session ID as User ID. And we classify the Item by categorizing them into various levels of Price range.

| Price | Level |
|---|---|
| **Less than 500** | 1 |
| **[500,1000]** | 2 |
| **[1000,1500]** | 3 |
| **[1500,2000]** | 4 |
| **[2000,2500]** | 5 |
| **[2500,3000]** | 6 |
| **[3000,3500]** | 7 |
| **[3500,4000]** | 8 |
| **More than 4000** | 9 |

*Table 1:Categorising Levels to various price ranges*

We have creatd a User-level matrix and have split the data into train and validation dataset (80% train, 20% validation). Then we have calculated the error of both IBCF & UBCF. The RMSE, MSE and MAE of the two-customer recommendation algorithm are shown at the table below.

| | RMSE | MSE | MAE |
|---|---|---|---|
| **UBCF** | 8.089916 | 80.8194330 | 6.356864 |
| **IBCF** | 0.658149 | 0.4331601 | 0.658149 |

*Table 2: Comparing RMSE, MSE and MAE of UBCF & IBCF*

**The Error of IBCF is significantly lower than UBCF**. The potential reason is we use Session ID as User ID which may cause some bias of training the model of User Based Collaborative Filtering (UBCF) algorithm.

Based on the UBCF, we can also recommend items (based on the price range) to specific users. Below we list the User's top 3 item recommendation.

| Levels if User would land | Recommendation Item (Level) |
|---|---|
| **1** | {5,8,6} |
| **2** | {5,8,6} |
| **3** | {5,8,4} |
| **4** | {5,8,6} |
| **5** | {5,8,3} |
| **6** | {1,2,4} |
| **7** | {1,2,6} |
| **8** | {1,2,4} |
| **9** | {1,2,4} |
| **10** | {1,2,4} |

*Table 3: Top 3 Recommendation for Users for a particular Level*

***Sample Case***: If User 1 would land on Level 1 then he would be recommended to buy Items having price ranges according to levels {5,8,6}

# E. Performance and Results

### D.1 Performance of basic data by Association

Precision: **72.58%**
Total number of recommendations on the test data: **10,974**
The number of correct recommendations on the test data: **7965**
The ratio of number of correct recommendations to the number of testing transactions: **0.211**

From the above results, that we can find all the precision are above 70% which can be regarded as feasible results. In addition, from the testing data the number of recommendations are all above 500 which is meaningful and critical for the recommendation system. Because if the recommendations are not enough despite of a good precision, the recommendation performance will be affected.

# F. Appendix

### F.1 Attached R code submitted to IVLE

- Association Code
- Sequencing Code
- Recommendation Code
- Association and Sequencing Data Image Files