



SCHOOL OF COMPUTER SCIENCES
UNIVERSITI SAINS MALAYSIA

CDS513: Predictive Business Analytics
Semester 2, 2020/2021

GROUP PROJECT FINAL REPORT

GROUP 3: Improving Sales Performance of the Ecommerce Website for an Electronics Store using Predictive Business Analytics Techniques

Member Name	Matric No.
Lee Yong Meng	P-COM0012/20
Lee Kar Choon	P-COM0130/19
Lim Hang Thing	P-COM0143/20

Date of Submission

28 June 2021

Contribution of Each Member

Task	Member Name
Abstract	Lee Kar Choon
Problem Background	
Background of the Problem Domain	Lim Hang Thing
Issues and Problem Statement	Lee Kar Choon
Project Objectives	Lee Kar Choon
Motivations	Lee Kar Choon
Scope and Limitations	Lee Kar Choon
Literature Review	
Literature Review	Lim Hang Thing
Methodology	
Project Framework	Lee Yong Meng, Lee Kar Choon
Dataset Description and Preparation	Lee Yong Meng
Approaches and Algorithms	Lee Yong Meng
Experiments and Analysis	
Market Basket Analysis	Lim Hang Thing
Recommender Systems	Lee Yong Meng
Time Series Forecasting	Lee Kar Choon
Conclusion and Future Research	Lim Hang Thing
References	Lim Hang Thing

Table of Content

Contribution of Each Member	1
Table of Content	2
Abstract	1
1. Problem Background	2
1.1. Background of the Problem Domain	2
1.2. Issues and Problem Statement	2
1.3. Project Objectives	3
1.4. Motivations	3
1.5. Scope and Limitations	4
2. Literature Review	5
3. Methodology	6
3.1. Project Framework	6
3.2. Dataset Description and Preparation	6
3.3. Approaches and Algorithms	8
4. Experiment and Analysis	13
4.1. Market Basket Analysis	13
4.2. Recommender System	18
4.3. Time Series Forecasting	24
5. Conclusion and Future Research	32
References	34

Abstract

In recent years, the trend of purchasing items online through ecommerce websites have become a norm all around the world. In this project, a store owner selling electronics products is interested on customer buying behaviours in the ecommerce website for decision making such as product cross-selling and promotion. This is addressed via product affinity analysis by using market basket analysis (MBA) technique on customer transaction data to study the customer purchase patterns and discover items that are frequently bought together. In large ecommerce websites, there may exist thousands of products of various brands sold within a single product category. Without an effective search algorithm, customers might not be able to search for the products that best match their preferences from an ecommerce website. This issue also happens to the ecommerce website of the electronics store owner. Therefore, a recommender system (RS) is implemented in the ecommerce website. The goal of this implementation is to improve the customer experience when purchasing items on the website. RS is implemented using different types of collaborative filtering (CF) approaches, including User-to-User, Item-to-Item, and Bayesian Personalized Ranking Matrix Factorization (BPRMF), based on the user behaviour data on the ecommerce website in the past. In addition, the ability to identify the next sale trend of the products in the ecommerce website is also essential to the electronics store owner to plan for marketing spend to maximize the revenue. Therefore, time series (TS) forecasting is a relevant technique to help the store owner understand the factors contributing to the sales trend through sales forecasting. Two types of TS models are implemented, including ARIMA model and a machine learning-based TS model. Based on the analysis, television and smartphones are both frequently purchased together. The store owner can utilize this information when planning for marketing strategies to improve the sales of the electronics store. For RS, the best RS in recommending electronics products is the item-to-item CFRS with weighted k-NN ($k=100$), which can be implemented in the ecommerce platform to help recommend list of electronics products to the user. Finally, machine learning-based TS model outperforms ARIMA model in forecasting the smartphone sales and therefore, it prepares the electronics store owner to take several business actions such as marketing strategy planning and restocking.

Keywords: predictive business analytics, ecommerce websites, electronics store, market basket analysis, recommender system, time series forecasting

1. Problem Background

1.1. Background of the Problem Domain

Ecommerce has become a global phenomenon and the industry competition is mounting in various aspects. Multiple shopping items and background from home electrical appliances to large items like housing and properties have joined the trend as well as the players of different background from small scale of home-based to the giants of the multi-billions. This eventually affects the business stakeholder's interest in their respective profit margin and market share.

With the presence of growing big data, the effectiveness on the interpretation and analysis of the data with the aim to assist in the business stakeholders towards a profitable decision making is being tested. As from the past record, ecommerce has shown a substantial contribution towards a country economic growth [1].

Despite of the big data availability, the questions on the data types, its validity, and reliability is a challenge in order to achieve the data interpretation and prediction accuracy as well as the other factors arising from the consumers is also the highlights [2]. The concern ranges from understanding the user behaviours and preferences, improving online shopping experience, product recommendation, stock management, up to sales forecast and sales enhancement.

Ecommerce business stakeholders' decision in their next move in increasing sales and marketing strategies is equally important for their business sustainability and to remain competitive in the industry for the years to come. Certainly, this has to draw a tremendous amount of attention and effort to study to improve the sales performance to eventually benefit the stakeholders and industry as it is the way forward.

1.2. Issues and Problem Statement

Ecommerce has the large size of customer accounts and can have hundred transactions every minute depends on the business size of the platform. With the volume and velocity of daily transactions, it is difficult to understand customer personality and buying power per transaction. Hereby there is a need to use the proper computer-based technique to analyse the transactions to study customer behaviours. One of the most common techniques used for such analysis is the market basket analysis. The goal of using market basket analysis technique is to find the product affinity from the platform, to understand which product customer likes. This information is useful to the store owners to know the customer behaviours and urge them to provide promotion or cross-selling to drive customer purchase desires.

With the increasing merchant stores in ecommerce platform, the size of the items sold also increases dramatically. In this case, the usage of traditional methods to identify and sort the best items list to all customers equally has become less efficient and irrelevant. This is because the traditional methods are unable to recognize the fact that every

customer has their own preference and needs, hence the system fails to treat all customers equally. Hereby, the implementation of recommender system (RS) to the ecommerce platform based on the past event data of the customer might provide better insights that helps to generate personalized list of recommended items to each customer more efficiently. The system act like knowing your tastes, needs accurately and just in time to suggest the products that best match your preferences, sounds good?

Ecommerce market is also very competitive as the competitors always provide offers and advertisings to lure customer resources. To ensure constant growth of ecommerce sales, the sales trend prediction is a necessary technology, which helps to forecast the next hours, days sales trend timely. And management level personnel can make fact-based decision based on the predictive analysis such as marketing budget plans, merchant inventory restock early notification.

1.3. Project Objectives

Project Objective 1

To discover the associations and relationships between electronic products in the transactions by using market basket analysis (MBA) in terms of support, confidence and lift measures, so that it helps to determine electronic products item sets to be bundle together to maximize revenues generated from item combination in ecommerce.

Project Objective 2

To implement a RS that helps to generate personalized recommendations of the electronic products to every customer of the ecommerce platform of an electronics store, which indirectly improves the user experience of the platform to encourage user engagement and improve user loyalty towards the platform.

Project Objective 3

To discover sales trend and predict next seasonal sales using time series (TS) forecasting, so that the electronics store owner can plan for appropriate marketing actions, such as automated informative restock alert to merchants, align carrier schedule and resources to cater the marketing campaign, before the event happens through TS forecasting.

1.4. Motivations

There are few motivations to conduct this project. Firstly, we are excited to have the chance to conduct predictive business analytics on real world dataset by using appropriate data science techniques. For example, MBA is used to discover the customer frequent item set and urge to product cross-selling to boost maximum sales and encourage to sell the relevant products together to gain maximum profit.

Appropriate RS is bringing convenient personalized search experience to customers and suggest what products they like and increase possible purchase actions. Hence, it urges to retain the customer to keep continue using the excellent services.

We believe that throughout this study, we could know how to create and provide sales trend forecasting based on the past data and proper TS forecasting modelling. And the model may produce certain future trend and decision makers can assess and utilize the outcome as well as to have early preparation and reaction for the future.

1.5. Scope and Limitations

Project Scope

Scope one is to discover the customer behaviours from the electronic transactions dataset by identifying the frequent item sets in the transactions to know what customers like to purchase through the ecommerce platform.

Scope two is to apply different RS approaches to the ecommerce electronic transactions dataset to check which recommender is suitable for the platform.

Scope three is to apply TS forecasting on ecommerce electronic transactions dataset to check the trend and modelling performance in order to provide a reliable forecasting system for the electronic products sales.

Project Limitations

There are also some limitations we should know and to avoid under this project. Example, there might be limited information that not able to use different approaches for predictive analysis, such as the dataset has missing rating and transactions feedback.

The dataset is huge, but the date range is from January to November, it might not be useful if need to perform annual seasonal cycle sales trend prediction. Furthermore, sales sometimes might be affected by many factors that cannot be obtained from the dataset that makes the sales forecasting very challenging in this project.

2. Literature Review

Ecommerce is in an uptrend and gaining its popularity [1] which attracted numerous studies conducted to analyse its effectiveness [3], the development and market competitiveness especially by the adoption of different analytic methods [4]. This includes the challenges faced [1] especially in the electronic markets [5]. In ecommerce industry, customer preference were studied for product recommendation [6], analysing the user behaviour complexity [7] and user experience [8] by using RSs and further enhanced with collaborative filtering (CF) and content-based (CB) mechanisms [6] and even integration of RSs into Proactive Modelling Engine too shows increased usability and learnability [9]. This too assisted the business stakeholders to effectively predict the potential purchases during shopping festivals via online [10] especially with the amount of big data collected [2]. Besides that, it has also proven its usefulness by applying RS in new user identification [11]. Using the RS to value add customer shopping experience for example time saving via ecommerce website appreciated very much by the customers [12] as well as the real time input hence the recommendation by adopting the same system for the benefit of users [13]. To the online retail business owner, product price and profit is a tough decision but it could now be further assisted by RSs [14].

Nevertheless, retail industry has always been the spotlight for the application of MBA to study the importance and correlation of the items purchase frequency and sales, the customer shopping characteristics [15], which further enhanced by applying different algorithm [16] including the Apriori [17]. On the other hand, accurate sales forecast in ecommerce could now been materialized by the approaches like time-series decomposition and Arima [18].

In addition, different types of data too been studied by applying different analytic methods with the aim for new information discovery. MBA application through association rule method on the transaction data has shown a positive outcome [19]. Customers' buying and purchasing behaviour at supermarket has been studied in regards to its relationship by applying association rules and MBA has been identified to be useful in the analysing process [20]. In the most recent happening, buying pattern during covid-19 lockdown too has been explored and further analysed using MBA [21].

3. Methodology

This section starts with the discussion about the overall project framework which is adopted to achieve the objectives of this project as defined in Section 1.3, followed by the dataset used in this project. After that, the discussion is followed by the approaches and algorithms included in the project framework and implemented to achieve each objective of this project.

3.1. Project Framework

To accomplish the three objectives of this project, a project framework is designed to incorporate two different sets of data and three different algorithms. The two sets of data are purchase data¹ and user behaviour data² respectively, whereas the algorithms include MBA, RS and TS analysis and forecasting. Figure 3.1 illustrates the overall project framework designed to accomplish the three objectives of this project.

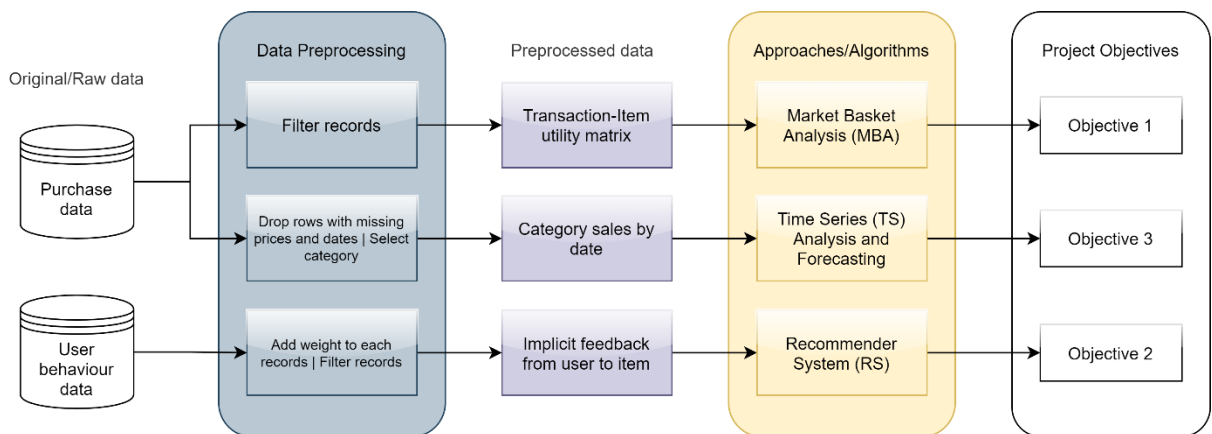


Figure 3.1: Overall project framework.

3.2. Dataset Description and Preparation

In this project, two sets of related data, namely the purchase data and the user behaviour data are used. These two datasets, which can be found in the Kaggle website, are open-source data collected by Open CDP (Customer Data Platform) project from the REES46, a company providing platform for ecommerce marketing automation.

¹ Purchase data: <https://www.kaggle.com/mkechinov/ecommerce-purchase-history-from-electronics-store>

² User behaviour data: <https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store>

The purchase data consists of the ecommerce purchase history data from an electronics store. This dataset contains more than 2.6 million records of purchase events for a period of 11 months (from January to November 2020) on the ecommerce website of the electronics store. Each record corresponds to the purchase event of only one unit of item. Therefore, multiple items purchased in a single transaction (which is recorded under the same event time and order ID) are separated into multiple rows. Some attributes in this dataset that are available for analysis include “event_time”, “order_id”, “product_id”, “category_id” and “user_id”.

The other set of data, the user behaviour data, is the user behaviour history data in the same ecommerce website of the electronics store. This dataset contains more than 100 million records of user behaviour data for October and November 2019 on the ecommerce website of the electronics store. In this project, only the records from October 2019 are used due to limitations in the computational resources available. Similar to the purchase data, each record corresponds to a user event of only one unit of item. Some attributes in this dataset include “event_time”, “event_type”, “product_id”, “category_id” and “user_id”.

Furthermore, these two datasets share most of the attributes such as “event_time”, “product_id”, “category_id”, and “user_id”, to name a few. A comprehensive data description of these two datasets is summarized in Table 3.1.

Table 3.1: Description of the datasets (purchase and user behaviour data) used in this project.

Attribute name	Attribute type	Attribute Description
event_time	Date	Recorded date and time when the user event occurs.
event_type ³	Categorical	Name of user events on the product, possible values: “view”, “cart”, and “purchase”.
order_id ⁴	Categorical	Order ID: Unique identifier of the ecommerce transaction.
product_id	Categorical	Product ID: Unique identifier of the product.
category_id	Categorical	Category ID: represents the category of the product.
category_code	Categorical	Product category code name (indicating the taxonomy of the category) corresponding to the category ID, only exists for meaningful categories.
brand	Categorical	Brand name of the product.
price	Numerical	Unit price of the product.

³ “event_type” attribute is only available in the user behaviour data.

⁴ “order_id” attribute is only available in the purchase data.

Attribute name	Attribute type	Attribute Description
user_id	Categorical	User ID: represents the user who creates the event
user_session ⁵	Categorical	Temporary session ID assigned to each user which will change at different event time.

The data processing steps vary depending on the techniques and algorithms used to address different project objectives, which will be discussed in the following subsections. Alternatively, the Python code snippets for data preprocessing steps are attached in Appendix A – Appendix C at the end of this report.

3.3. Approaches and Algorithms

In this section, the predictive business analytics techniques and algorithms used to address different project objectives, and how these two sets of data are preprocessed prior to the implementation of these techniques, are discussed. The approaches and algorithms involved are MBA, RS, and TS forecasting.

Market Basket Analysis

MBA is a technique used in product affinity analysis to discover the associative relationships between items and products purchased in a single transaction. This technique is suitable to address the first objective of this project, which is to discover the correlations and patterns between products in the transaction from the purchase history. This information is useful for the electronics store owner to plan for strategies to increase the revenues through product cross-selling and promotions.

This technique is performed on the transaction data stored in a tabular format, in which, each row corresponds to a single transaction in the dataset, whereas each column represents a unique item available in the list of all items in the transaction data. The table stores binary values, which are “False” and “True”, indicating the absence or presence of an item in a transaction, respectively. Then, the transaction data stored in the tabular format is used to generate a list of frequent itemsets, which are sets of items frequently bought together by the customer. Two popular algorithms used for generating the frequent itemsets are Apriori algorithm and FP-Growth algorithm. Finally, interesting association rules can be extracted from these frequent itemsets, which are evaluated using different metrics such as support, confidence and lift.

To prepare the purchase data used for MBA, the first step is to derive new attribute “product_name” for each purchase record. This step is done by concatenating the

⁵ “user_session” attribute is only available in the user behaviour data.

“category_code” and the “brand” from each record. For purchase records with missing “category_code” or “brand”, the values are replaced with “unknown_category” or “unknown_brand” respectively. Then, data filtering step is performed to keep only the products with total sales amount not less than RM30,000 from January to October 2020, and the transactions with total amount not less than RM200. After the filtering process, the number of unique transactions and products are reduced from 1,435,266 and 496,738 to 37,631 and 478, respectively.

Next, the purchase records with the same “order_id” are grouped as one transaction. When multiple “product_names” are grouped into the same transaction, the unique values of “product_name” are stored, indicating the absence or presence of the items that correspond to the “product_name” in that transaction regardless of their purchased quantities.

After that, the transactions data with unique “order_id” and list of “product_name” are transformed into a transaction-item utility matrix. This utility matrix is represented in a tabular format, in which, each row represents a transaction with unique “order_id”, whereas each column consists of the “product_name” of all transactions used in the MBA. The data processing steps for MBA in this project is illustrated in Figure 3.2.

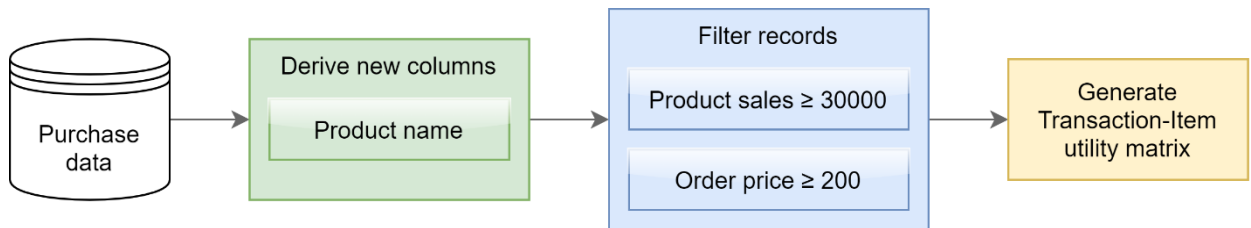


Figure 3.2: Data processing steps for MBA.

Recommender System

RS is an important application of information retrieval system, which aims to match a list of items to users in a system. The lists of recommended items are generated based on the predicted outcome of the user preferences towards items available in the system. This technique is widely used in different domains to recommend items to the users, such as video and programme recommendations in streaming services such as Netflix and YouTube, and product recommendations in Amazon online stores. It is also a suitable technique to address the second objective of this project, which is to generate lists of relevant and personalized content, which are the products, to customer. The process of generating lists of recommended items to user will improve the user experience when browsing the ecommerce website of the electronics store, which is an important aspect in encouraging user engagement towards the electronics store. This will indirectly retain the customer loyalty and slowly expand the ecommerce markets to increase the overall sales of the electronics store.

There are several types of information allowing the recommendation engine to “learn” the utility function to predict user preferences towards products and items in a system,

including the user feedbacks, item attributes and the context associated to which the recommendation is made. The most common types of RSs include collaborative filtering (CF) and content-based RSs (CBRSs). Both these types of RSs require user feedbacks (including explicit feedbacks such as user ratings, and implicit feedbacks such as user events) as important information to enable the recommendation engine to generate list of recommended items to the users. Under CF approach, the recommendation is generated by measuring the similarities between users or items in terms of feedbacks given to or received by the items. This involves the interactions between the target user with other users in the system. On the other hand, feedbacks of the target user are important in CB approach to evaluate user preference towards the items in the system when creating user profile to match item profiles during the recommendation process.

In this project, CF approach is selected for implementing the RS for the ecommerce platform in the electronics store. The relevant dataset used is the user behaviour data, which stores attributes such as “event_type”, “category_code”, “brand”, and “user_id”. For using this set of data for implementing different types of CFRSs, user preferences towards products in the electronics store is assumed to remain the same for an extensive period, which is at least one year from the date at which the user behaviour data is collected. Therefore, the list of recommended products generated from user behaviour data collected in October 2019 is still relevant for the analysis.

The “event_type” attribute is treated as the implicit feedback provided by the ecommerce website users to show their preferences towards specific items in the system. For this purpose, different types of events are assigned different weights, in which, “view”, “cart”, and “purchase” are assigned weights of 1, 3 and 10, respectively. These weights are arbitrarily assigned to each type of event to indicate different levels of preference of each user towards the products in the system.

Next, two new attributes, namely “product_name” and “new_product_id” are derived from the user behaviour data. Similar to MBA, the step to derive “product_name” is done by concatenating the “category_code” and the “brand” from each record. On the other hand, the “new_product_id” is a new attribute derived to uniquely identify each “product_name” from the user behaviour data. After that, the records are filtered to keep only the top 2,000 users with the highest cumulative preference over all products and top 1,000 products receiving the highest cumulative preference from all users, respectively. The user preference data is then generated by aggregating dataset by grouping the records with the same “user_id” and “product_id” into each row. As a result, there are 133,665 user preference records, split into training and test sets with the ratio of 80:20, used for generating RSs in this project. The data processing steps for RS in this project is illustrated in Figure 3.3.

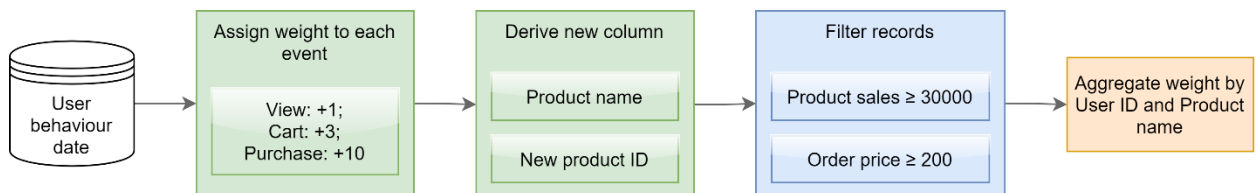


Figure 3.3: Data processing steps for RS.

Time Series Forecasting

Time series (TS) forecasting is a predictive analytics technique used for prediction and forecasting tasks in various domains. From the business perspective, this technique is widely used for predicting sales and demand of the customers so that the business decision maker can plan for predictive actions and marketing strategies on the targeted customers based on the predicted outcomes. The third objective of this project matches with the goal of incorporating TS forecasting technique to perform predictive analytics on the purchase data, which is to predict the future sales by analysing the sales trend. This helps the electronics store owner to stay at the competitive edge in the ecommerce market.

The four components of TS data include trend, seasonality, cyclical fluctuations, and irregular variables. Trend is defined as the general direction of the TS data, whether it is increasing or decreasing. Seasonality represents the changes of the continuous variable of the TS data within a period. Cyclical fluctuation captures the repeated patterns of changes within the TS data. Irregular variables mean the residual changes from unknown sources in the TS data that are unpredictable and thus treated as noises in the set of data. One of the popular techniques used to analyse the TS data and the four components is the ARIMA technique. The output of ARIMA technique is a mapping function which can be used to forecast the changes of a continuous variable for an extended period.

In this project, the TS data used is the data of the category sales by date. This TS data is generated from the purchase data through aggregating the unit price by each category into a total daily sales price. Therefore, several data preprocessing steps are performed to generate the TS data in this project. First, there are some rows from the purchase data with missing values in the “price” column and abnormal value, which is “1970-01-01”, in the “event_time” column, respectively. The rows with missing “price” value and abnormal “event_time” values are not helpful in our analysis because these might happen due to the system error during the data collecting process. To prevent misrepresentation of the results from the experiments in TS forecasting due to missing and abnormal values, the rows with missing “price” values and “event_time” recorded as “1970-01-01” are removed from the original purchase data. The number of records is reduced from 2,633,521 to 2,186,014.

The next data preprocessing step is to derive new attribute “category” for each purchase record. This step is done by concatenating the “category_id” and the “category_code” values (or “unknown_category” for records with missing “category_code”) from each record, to derive a more meaningful “category” naming to each product. For example, the “category_id” and “category_code” values of “2.268105428166509e+18” and “electronics.smartphone” are merged to become a unique value of “2.268105428166509e+18-electronics.smartphone”. Finally, the purchase data is analyzed using some statistical measurements such as mean, minimum, maximum and count values of the sales price by each “category”. The top ten categories sorted by the count value are shown in Figure 3.4.

category	price			count
	mean	min	max	
2.268105428166509e+18-electronics.smartphone	302.831564	46.04	2314.79	335709
2.2681054301629978e+18-electronics.audio.headphone	65.115374	0.23	810.16	65637
2.3744989140005924e+18-electronics.video.tv	458.729482	92.57	50925.90	58804
2.268105392070329e+18-appliances.environment.vacuum	155.693218	37.01	2474.51	58127
2.2681053899563999e+18-appliances.kitchen.washer	347.419603	90.02	4629.58	51527
2.268105442636858e+18-furniture.kitchen.table	32.931493	8.89	324.05	49491
2.2681054418567176e+18-appliances.kitchen.kettle	34.070708	4.61	405.07	47723
2.2681054072201554e+18-computers.notebook	646.389867	0.00	5370.35	47155
2.268105393848714e+18-appliances.kitchen.refrigerators	528.568818	92.57	9173.59	44634
2.2681054411017428e+18-appliances.kitchen.blender	66.642515	17.34	821.74	36540

Figure 3.4: Top 10 categories by count values with the statistical measurements such as mean, minimum and maximum values.

From Figure 3.4, the category with the highest count value, “2.268105428166509e+18-electronics.smartphone”, is selected and transformed into daily sales data used for the TS modelling and forecasting in this project. The data preprocessing steps for TS forecasting is illustrated in Figure 3.5.

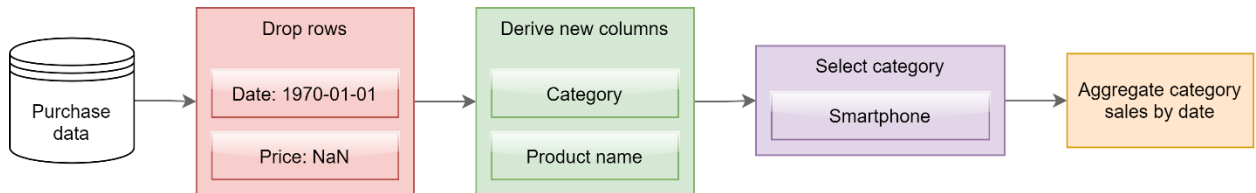


Figure 3.5: Data processing steps for TS forecasting technique.

4. Experiment and Analysis

In this section, the discussion focuses on the modelling process and experiments conducted using the data and algorithms introduced in the previous sections. Furthermore, it also includes the evaluation and analysis of the experiment results obtained from these modelling process and experiments. The discussion starts with MBA, then followed by RS and TS analysis and forecasting.

4.1. Market Basket Analysis

Implementing Market Basket Analysis using RapidMiner Studio

Market Basket Analysis is being deployed to achieve our first objective which is to discover the associations and relationships between electronic products purchased in the transactions under the electronic store. The result is further analysed by the performance metrics in terms of the support, confidence, and lift measures, to determine the electronic products items for which to be bundled together for further marketing strategies and commercial promotional activities via ecommerce platform to maximize the revenues generated.

Sequence of operators are utilized under the RapidMiner Studio to generate the results by implementing Market Basket Analysis. To calculate frequently purchased together items under a transaction, FP-Growth algorithm is used to mine. It then being further analysed with Create Association Rule operator to generate the association rules by taking the frequent items that helps to uncover the relationships for example if a customer buys 'AA', he is 80% likely to also purchase 'BB', in which the result will be further discussed.

The entire process is illustrated under Figure 4.1. The details of the RapidMiner processes and parameter settings for MBA are attached in Appendix D at the end of this report.

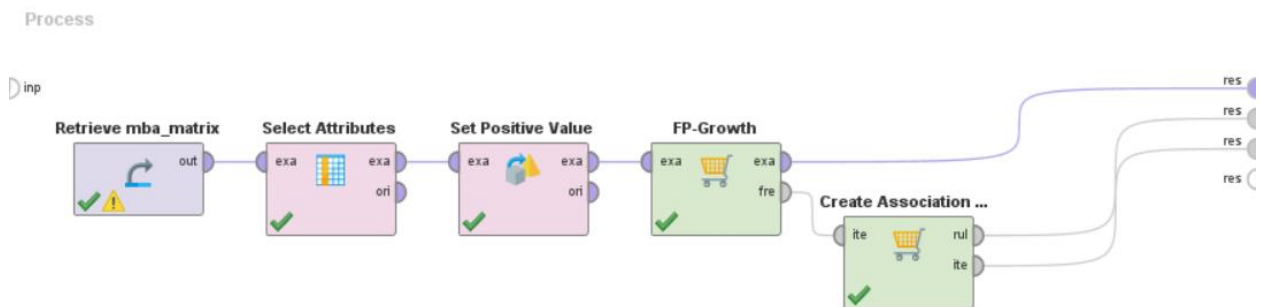


Figure 4.1: Market Basket Analysis by using RapidMiner Studio.

Experiments under Market Basket Analysis

Operator Create Association rule helps to analyse data for frequent patterns and different value of criteria support and confidence is tested to identify the most important relationships. The Support value is being tested with 0.001 to 0.2 or equals to 0.1% to 20%, which is the indication of how frequent the items appear under the transactions, while for Confidence that can indicate the number of times the statements found to be true which was set against the values of 0.05 to 0.8 or equals to 5%-80% of transactions. No association generated when the minimum support is above 0.1 but result is shown when minimum support and minimum confidence were both set at 0.05 or 5% of the transactions and below and being further discussed.

Result Analysis

When different minimum support and minimum confidence is set, respective result is being analysed and discussed. The following tables shown that there's a association derived when support value is set below 0.05 (i.e. at 0.05 & 0.001).

i. When Support Min: 0.05 & Confidence Min: 0.05

Under '1 item', electronics item shown a great response among the customer purchase preference especially the smartphones, among nearly 8-16% of the transactions. Figure 4.2 shows the summary for the top 10 of the preferred items.

Size	Support	Item 1
1	0.163	electronics.smartphone-samsung
1	0.096	electronics.smartphone-apple
1	0.076	electronics.smartphone-huawei
1	0.054	electronics.video.tv-samsung
1	0.052	electronics.smartphone-oppo
1	0.045	appliances.kitchen.washer-samsung
1	0.043	electronics.video.tv-lg
1	0.036	appliances.kitchen.washer-lg
1	0.031	appliances.environmentvacuum-samsung
1	0.030	computers.notebook-asus

Figure 4.2: The summary for the top 10 of the preferred items, under '1 item'.

Under 2 items, smartphones under the categories of electronic items implies the purchase of smartphones and TV in which both bought together in 1.3% & 1.6% of the transactions, respectively, and 58% & 92% of the times that smartphones is bought, TV is bought which as well only 45% of the time, smartphone is bought with another smartphone. Figure 4.3 shows the 2 items bought together.

No.	Premises	Conclusion	Support ↓	Confidence	LaPlace	Gain	p-s	Lift
3	electronics.smartphone-awax	electronics.video.tv-megogo	0.016	0.578	0.988	-0.040	0.016	32.971
4	electronics.video.tv-megogo	electronics.smartphone-awax	0.016	0.924	0.999	-0.019	0.016	32.971
2	electronics.smartphone-awax	electronics.smartphone-samsung	0.013	0.451	0.985	-0.043	0.008	2.764

Figure 4.3: The summary for the two items bought together with their respective support and confidence level.

ii. When Support Min: 0.001 & Confidence Min: 0.05

Results for the top two on the list is the same as illustrated above, however, the listing is expanded as it accommodates more pairs that have lower support level, and the bundles are up to three items (i.e., Figure 4.5) and four items (i.e., Figure 4.6) purchased in a transaction. Besides the purchase pairing of smartphones with smartphones and TV of different brands respectively, results too showed purchase of computer notebook with its peripherals is likely to happen as well as apparel with computer peripherals, which both recorded 0.5% each that both bought together in a transaction. Figure 4.4 shows the summary for the two items bought together with their respective support and confidence level.

Size	Support ↓	Item 1	Item 2
2	0.016	electronics.smartphone-awax	electronics.video.tv-megogo
2	0.013	electronics.smartphone-samsung	electronics.smartphone-awax
2	0.008	electronics.smartphone-samsung	electronics.video.tv-megogo
2	0.007	electronics.smartphone-apple	electronics.smartphone-awax
2	0.006	electronics.smartphone-awax	electronics.smartphone-unknown_brand
2	0.005	electronics.smartphone-samsung	electronics.smartphone-huawei
2	0.005	electronics.smartphone-huawei	electronics.smartphone-awax
2	0.005	computers.notebook-asus	computers.peripherals.mouse-logitech
2	0.005	electronics.video.tv-megogo	electronics.smartphone-unknown_brand
2	0.005	apparel.glove-sony	computers.peripherals.joystick-sony

Figure 4.4: The summary for two items bought together, when Support Min:0.001, Confidence Min: 0.05.

Bundles of three items purchased together because of lower support min and confidence min set, and findings shown the purchase of smartphones with smartphones and smartphones with TV remained the most popular bundles, as illustrated under Figure 4.5.

Size	Support ↓	Item 1	Item 2	Item 3
3	0.008	electronics.smartphone-samsung	electronics.smartphone-awax	electronics.video.tv-megogo
3	0.005	electronics.smartphone-awax	electronics.video.tv-megogo	electronics.smartphone-unknown_brand
3	0.004	electronics.smartphone-apple	electronics.smartphone-awax	electronics.video.tv-megogo
3	0.003	electronics.smartphone-samsung	electronics.smartphone-awax	electronics.smartphone-unknown_brand
3	0.003	electronics.smartphone-samsung	electronics.video.tv-megogo	electronics.smartphone-unknown_brand
3	0.003	electronics.smartphone-huawei	electronics.smartphone-awax	electronics.video.tv-megogo
3	0.002	electronics.smartphone-oppo	electronics.smartphone-awax	electronics.video.tv-megogo
3	0.001	electronics.smartphone-samsung	electronics.smartphone-awax	electronics.video.tv-okko
3	0.001	electronics.smartphone-apple	electronics.smartphone-awax	electronics.smartphone-unknown_brand
3	0.001	electronics.smartphone-apple	electronics.smartphone-awax	electronics.video.tv-okko
3	0.001	electronics.smartphone-apple	electronics.video.tv-megogo	electronics.smartphone-unknown_brand

Figure 4.5: The summary for three items bought together, when Support Min: 0.001, Confidence Min: 0.05.

Nevertheless, results remained where four items' bundles purchased still show smartphones purchased with smartphones and TV recorded a 0.3% & 0.1% in the transactions, as summarised under Figure 4.6.

Size	Support	Item 1	Item 2	Item 3	Item 4
4	0.003	electronics.smartphone-samsung	electronics.smartphone-awax	electronics.video.tv-megogo	electronics.smartphone-unknown...
4	0.001	electronics.smartphone-apple	electronics.smartphone-awax	electronics.video.tv-megogo	electronics.smartphone-unknown...

Figure 4.6: The summary for four items bought together, when Support Min:0.001, Confidence Min: 0.05.

Under these values, purchase of smartphones with smartphones and TV in which both bought together respectively showed the similar result as illustrated above but posing different confidence level when the combinations under different brands of each category. Figure 4.7 shows the summary of the two items bought together with the respective support and confidence level. This is also illustrated under Figure 4.8 & Figure 4.9 with a better visualized graph.

No.	Premises	Conclusion	Support ↓	Confidence	LaPlace	Gain	p-s	Lift
148	electronics.smartphone-awax	electronics.video.tv-megogo	0.016	0.578	0.988	-0.040	0.016	32.971
164	electronics.video.tv-megogo	electronics.smartphone-awax	0.016	0.924	0.999	-0.019	0.016	32.971
132	electronics.smartphone-awax	electronics.smartphone-samsung	0.013	0.451	0.985	-0.043	0.008	2.764
133	electronics.video.tv-megogo	electronics.smartphone-samsung	0.008	0.452	0.991	-0.027	0.005	2.767
102	electronics.smartphone-awax	electronics.smartphone-samsung, electronics.vid...	0.008	0.280	0.980	-0.048	0.008	35.349
131	electronics.video.tv-megogo	electronics.smartphone-samsung, electronics.sm...	0.008	0.448	0.990	-0.027	0.008	35.384
136	electronics.smartphone-awax, electronics.video.tv-...	electronics.smartphone-samsung	0.008	0.484	0.992	-0.025	0.005	2.966
150	electronics.smartphone-samsung, electronics.sm...	electronics.video.tv-megogo	0.008	0.620	0.995	-0.017	0.008	35.384
173	electronics.smartphone-samsung, electronics.vid...	electronics.smartphone-awax	0.008	0.991	1.000	-0.008	0.008	35.349
100	electronics.smartphone-awax	electronics.smartphone-apple	0.007	0.251	0.980	-0.049	0.004	2.621

Figure 4.7: The summary for the two items bought together with their respective support and confidence level.

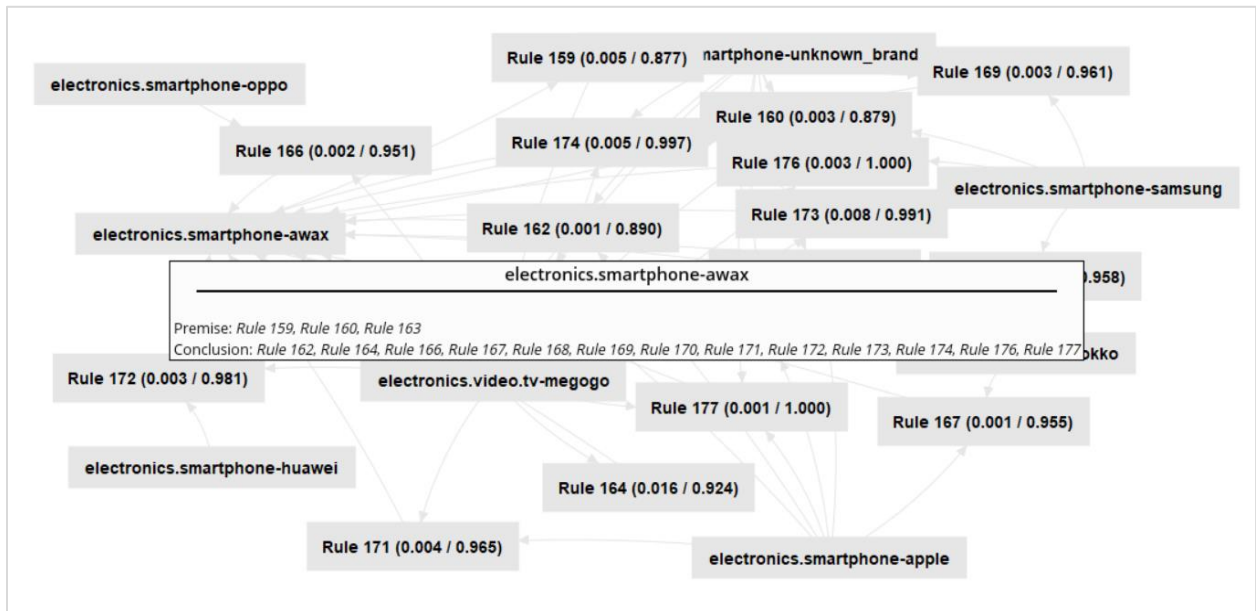


Figure 4.8: Graph showed the association rules for the example – smartphone.

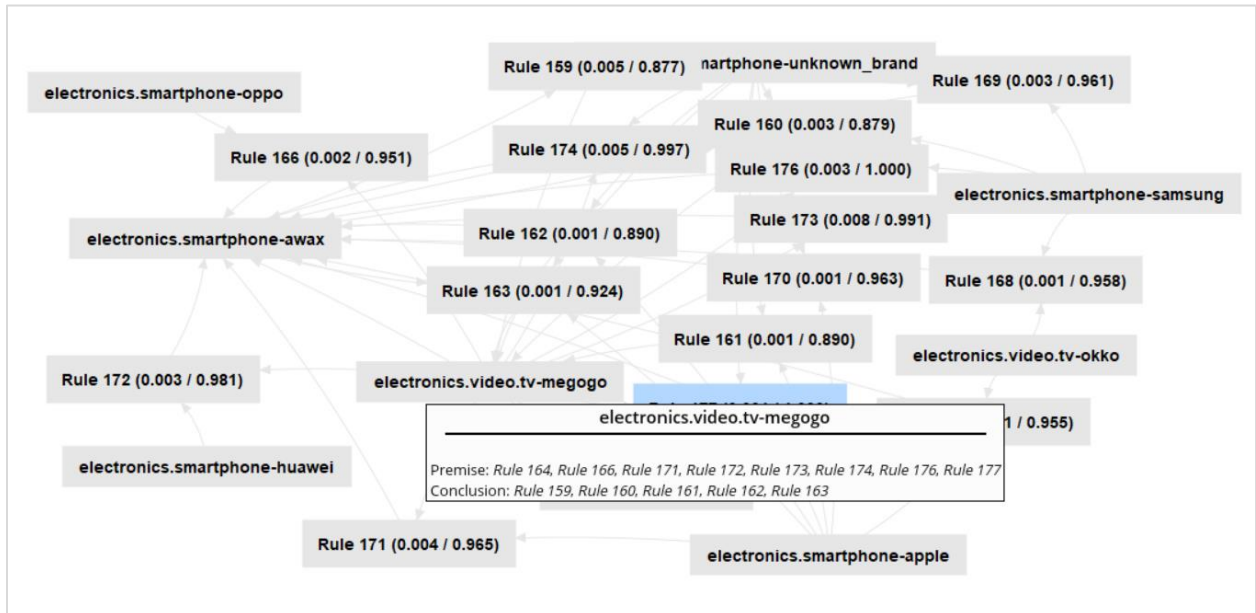


Figure 4.9: Graph showed the association rules for the example – TV.

In short, higher level of support is preferred. Similarly, higher the confidence, greater the likelihood that the two itemset is accepted. Lift summarises the strength of association between the transaction which implies larger the lift, greater the link between the two. On the other hand, it will take longer time for the process to compute especially when it involves large amounts of data and lower minimum confidence level is chosen. However, it was not significant under this MBA as computed by RapidMiner due the processes took relatively short time, within seconds to deliver the results.

4.2. Recommender System

Implementing Recommender Systems using RapidMiner Studio

The second objective of this project is to implement a RS that helps generate personalized recommendations of the electronic products to the customer of the ecommerce platform of the electronics store. Therefore, an experiment is conducted to identify the best RS for the electronics store by comparing the performance of different RSs implemented using the RapidMiner Studio version 9.9 with the user preference data generated from the user behaviour data. In RapidMiner Studio, RS can be implemented with the operators introduced in the Recommender Extension.

As mentioned in Section 3, collaborative filtering (CF) approach is used to implement RS for this project. The three types of CFRS implemented using RapidMiner Studio are User-to-User, Item-to-Item and Bayesian Personalized Ranking Matrix Factorization (BPRMF) CFRS. Figures 4.1 – 4.1 illustrate the implementation of different RS using RapidMiner studio.

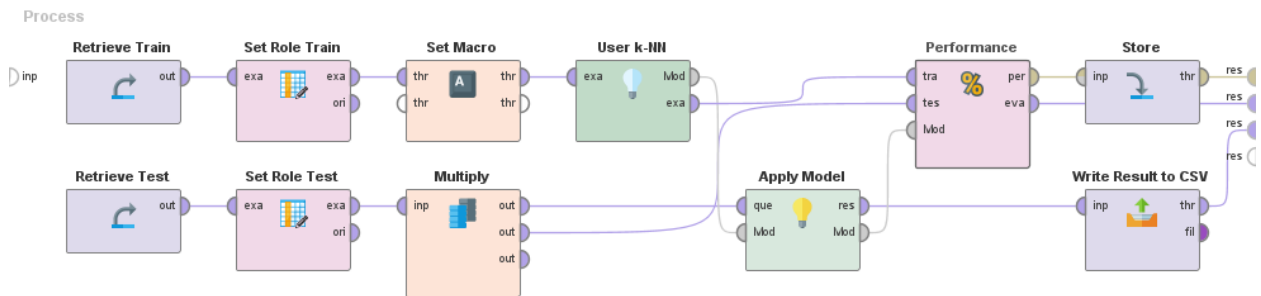


Figure 4.10: User-to-User CFRS implemented using User k-NN operator.

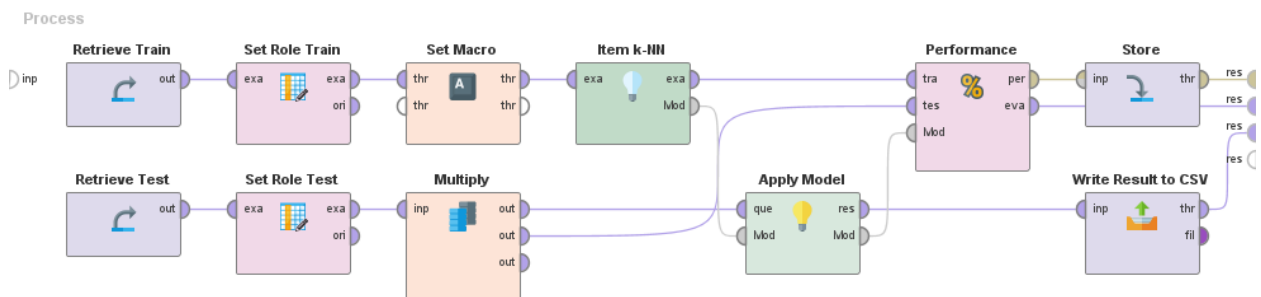


Figure 4.11: Item-to-Item CFRS implemented using Item k-NN operator.

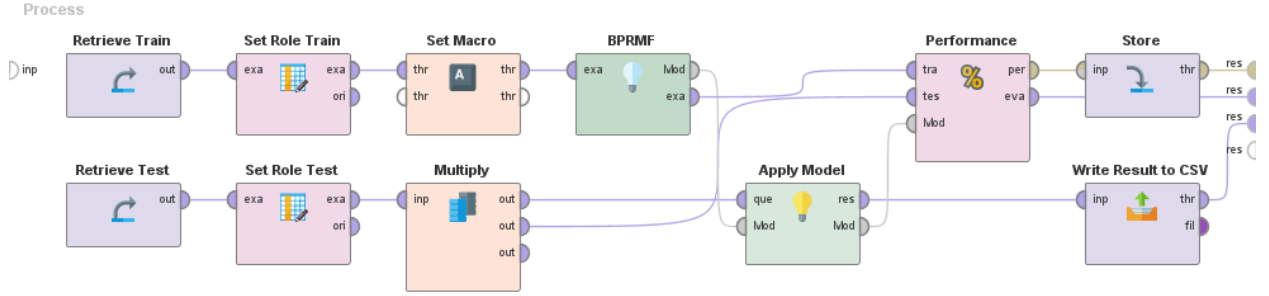


Figure 4.12: BPRMF CFRS implemented using BPRMF operator.

The details of the RapidMiner processes and parameter settings are attached in Appendix E at the end of this report.

Experiments for different Recommender Systems

For both User-to-User and Item-to-Item CFRS, the “Weighted Knn” parameter of the User k-NN and Item k-NN operators are ticked, indicating that weighted k-NN algorithm is used to generate the list of recommended items to a selected user in the system. This means, when calculating the user preference towards an item in the system, the preferences provided by similar users or received by similar items are given more weights. Then, list of recommendation is generated by sorting the items according to their preference score in descending order.

The experiments for both User-to-User and Item-to-Item CFRS are repeated using different values of k parameters, which are 100, 200, 300, 400 and 500, respectively. For BPRMS CFRS, the experiment is repeated using different values of “learn rate” parameter, which are 0.005, 0.01, 0.025, 0.05, 0.1, 0.25 and 0.5, respectively. The details of the parameter of the RapidMiner processes, including the parameter settings can be found in Appendix E.

Result Analysis

Different metrics are used to compare the performance between different RSs implemented in this project. The metrics implemented in Recommender Extension in RapidMiner include area under curve (AUC), precision metrics such as mean average precision (MAP), prec@k for k = 5, 10 and 15, and normalized discounted cumulative gain (NDCG). On the other hand, the processing times (in second) to implement each RS are also recorded. The performances of different CFRSs measured using different metrics and their respective processing times are summarized in Tables 4.1 – 4.3 respectively.

Table 4.1: Performance and processing time of User-to-User CFRS on user preference data.

k	AUC	prec@5	prec@10	prec@15	NDCG	MAP	Processing time (s)
100	0.858	0.345	0.260	0.216	0.582	0.223	26
200	0.861	0.333	0.252	0.210	0.574	0.215	41
300	0.859	0.317	0.244	0.203	0.565	0.206	61
400	0.857	0.304	0.236	0.198	0.558	0.198	83
500	0.854	0.294	0.228	0.195	0.552	0.192	109

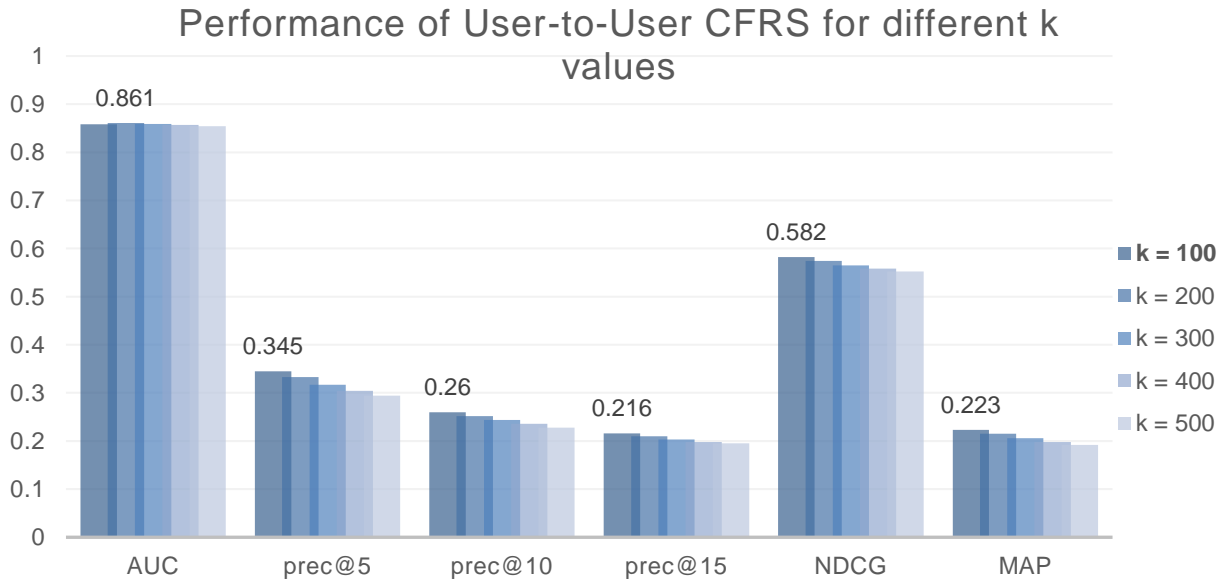


Figure 4.13: Bar chart showing performance of User-to-User CFRS on user preference data.

From Table 4.1 and Figure 4.13, it shows that the AUC values of different User-to-User CFRS are very close to each other for different values of k. The difference between the largest and the smallest AUC values is only 0.007. On the other hand, when increasing the value of k from 100 to 500, the performance of the User-to-User CFRS measured using other metrics gradually decreases. This indicates that the User-to-User CFRS performs best when the value of k is set to 100. Furthermore, when comparing the processing time taken to implement different recommender systems, User-to-User CFRS with k set to 100 takes the shortest time, which is 26 seconds. As the value of k increases, it takes longer processing time to complete the implementation of User-to-User CFRS. In conclusion, the best performing User-to-User CFRS is the one with value of k set to 100.

Table 4.2: Performance and processing time of Item-to-Item CFRS on user preference data.

k	AUC	prec@5	prec@10	prec@15	NDCG	MAP	Processing time (s)
100	0.882	0.360	0.275	0.230	0.594	0.238	24

k	AUC	prec@5	prec@10	prec@15	NDCG	MAP	Processing time (s)
200	0.876	0.350	0.264	0.222	0.586	0.228	44
300	0.872	0.341	0.256	0.216	0.581	0.223	65
400	0.869	0.335	0.251	0.211	0.577	0.219	87
500	0.867	0.330	0.248	0.208	0.574	0.215	109

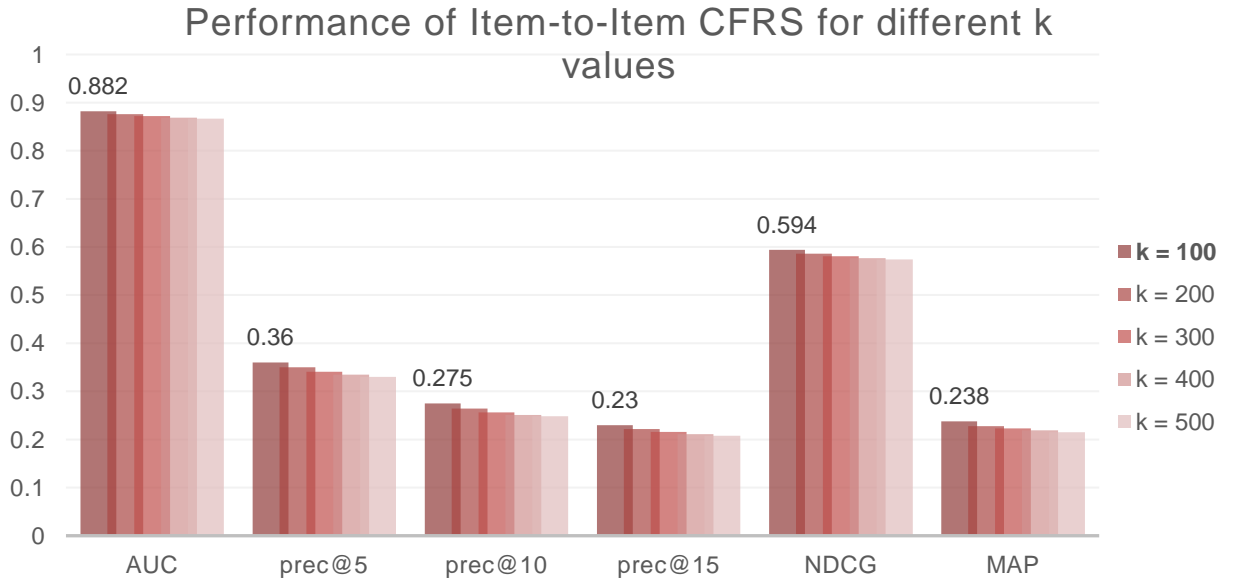


Figure 4.14: Bar chart showing performance of Item-to-Item CFRS on user preference data.

From Table 4.2 and Figure 4.14, the performances of the Item-to-Item CFRS measured using different metrics gradually decrease with increasing values of k from 100 to 500. This trend is similar to those of the User-to-User CFRS, except that the Item-to-Item CFRS with k set to 100 yields the highest AUC value. Furthermore, Item-to-Item CFRS generally performs better than User-to-User CFRS of the same values of k. In terms of processing time, as the value of k increases, the time taken to complete the implementation of each Item-to-Item CFRS also increases. In general, the processing time between different values of k for both User-to-User and Item-to-Item CFRS are similar, in which, the Item-to-Item CFRS with k set to 100 takes the shortest processing time for implementation. In conclusion, the best performing Item-to-Item CFRS is also the one with value of k set to 100.

Table 4.3: Performance and processing time of BPRMF CFRS on user preference data.

Learn rate	AUC	prec@5	prec@10	prec@15	NDCG	MAP	Processing time (s)
0.005	0.733	0.177	0.131	0.110	0.450	0.106	6
0.01	0.809	0.219	0.173	0.148	0.493	0.140	6

Learn rate	AUC	prec@5	prec@10	prec@15	NDCG	MAP	Processing time (s)
0.025	0.851	0.269	0.208	0.178	0.534	0.177	6
0.05	0.858	0.277	0.216	0.185	0.542	0.184	5
0.1	0.854	0.256	0.205	0.173	0.532	0.174	6
0.25	0.825	0.213	0.165	0.144	0.492	0.139	5
0.5	0.598	0.040	0.036	0.034	0.333	0.036	6

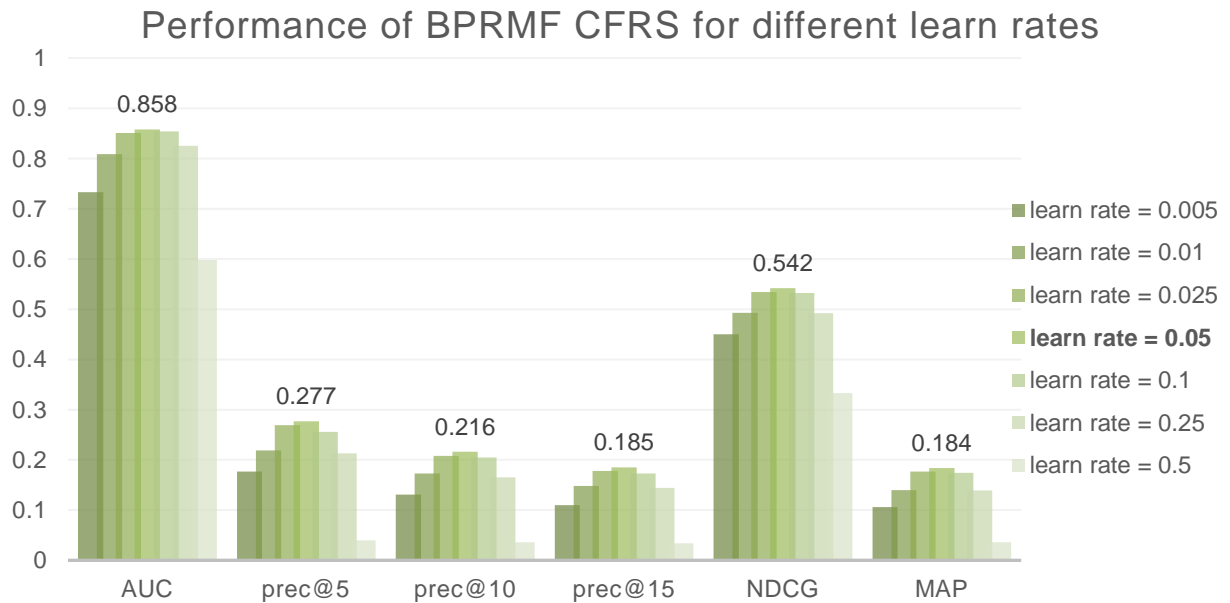


Figure 4.15: Bar chart showing performance of BPRMF CFRS on user preference data.

From Table 4.3 and Figure 4.15, the impact of learn rate on the performance of BPRMF CFRS is significant. The optimal learn rate is 0.05, in which, the AUC, precision metrics and NDCG are the highest among different BPRMF CFRS implemented using other values of learn rate. For smaller values of learn rate, the performance of BPRMF CFRS gradually improves as the value of learn rate approaches 0.05. After that, the performance of BPRMF CFRS drops significantly as the value of learn rate further increases from 0.05. This shows that it is important to obtain an optimal learn rate to implement a BPRMF CFRS through experimentations. Interestingly, the processing time to implement BPRMF CFRS does not change much regardless of the value of learn rate. In conclusion, the best performing BPRMF CFRS is the one with learn rate set to 0.05.

The CFRS which yields an overall best performance in recommending the products to the customer of the ecommerce platform of the electronics store is the Item-to-Item CFRS with value of k set to 100. It yields the highest AUC, prec@5, prec@10, prec@15, NDCG and MAP values of 0.882, 0.360, 0.275, 0.230, 0.594, and 0.238, respectively.

The comparison between the performance of different CFRSs in the product recommendation task is illustrated in Figure 4.16.

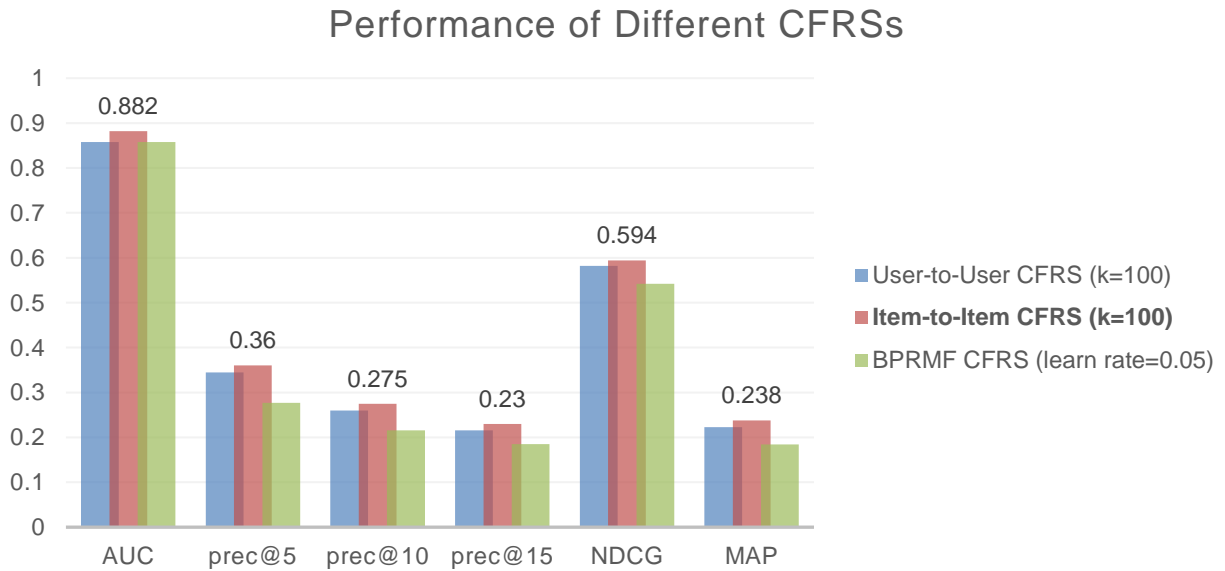


Figure 4.16: The performance of different CFRSs in recommending products for customers on the ecommerce platform of the electronics store.

To further analyse the performance of different RSs in recommending products from the ecommerce platform of the electronics store, a study is conducted to examine the list of top 10 recommended products to a selected customer (ID: 551211823). Table 4.4 shows the configurations of three different RSs used for this study.

Table 4.4: List of RSs and their corresponding parameters to generate list of top 10 recommended products.

No	Recommender Systems	Parameters
1	User-to-User CFRS	k = 100
2	Item-to-Item CFRS	k = 100
3	BPRMF CFRS	learn rate = 0.05 (keep other parameters as default values)

From the list of recommended products generated to this customer, it is observed that 8 out of the top 10 products from the list are identical for these 3 RSs. The list of these 8 products with their average rankings are summarized in Table 4.5. The list of top 10 recommended products by these RSs are attached in Appendix F.

Table 4.5: List of products in the list of top 10 recommended products by all 3 RSs and their corresponding average rankings.

No	Product ID	Product Name	Average Ranking
1	2681	electronics.smartphone-xiaomi	1.33
2	2641	electronics.smartphone-apple	1.67
3	2655	electronics.smartphone-huawei	4.33
4	2571	electronics.clocks-apple	4.67
5	2668	electronics.smartphone-oppo	5.33
6	2425	electronics.audio.headphone-sony	5.67
7	2375	electronics.audio.headphone-jbl	6.33
8	2325	electronics.audio.headphone-apple	9.00

From Table 4.5, it is observed that four of the top 10 products recommended to the selected customer by all 3 RSs are smartphones of different brands, namely: Xiaomi, Apple, Huawei, and Oppo. The smartphones with specified brands are also among the products with very high rankings in the recommended list of each RS.

Specifically, Xiaomi smartphone is the most recommended product by both User-to-User CFRS and Item-to-Item CFRS, and the second most recommended product by BPRMF CRFS. From the training set (see Table F-1), this customer has also interacted with other electronics products of Xiaomi brand, such as electronic clocks, headphones, and electric kettle. Furthermore, this customer shows the most frequent interactions with Xiaomi smartphone (with the highest preference score) among all products in the test set (see **Error! Reference source not found.**). This shows that different RSs can generate list of recommended products by predicting customer preference on selected products based on their past interaction on similar products on the ecommerce platform. Interestingly, this customer has never interacted with Apple, Huawei, and Oppo smartphones in both training and test sets. This shows that the recommended products generated by CFRSs are not too specialized and there might be times the recommended products might not be known to the customer in the past. Therefore, the preference scores for these products are zero because the customer has never interacted with these products.

4.3. Time Series Forecasting

Auto-Regressive Integrated Moving Average (ARIMA)

As discussed in Section 3, the purchase data from the selected category “2.268105428166509e+18-electronics.smartphone” is aggregated into daily sales data used for TS modelling and forecasting in this project. Two types of TS models are implemented to fit the sales data for sales forecasting, including Auto-Regressive

Integrated Moving Average (ARIMA) and machine learning models. The implementation of an ARIMA model starts with the visualization of the daily sales data of the selected category, as illustrated in the TS plot in Figure 4.17.

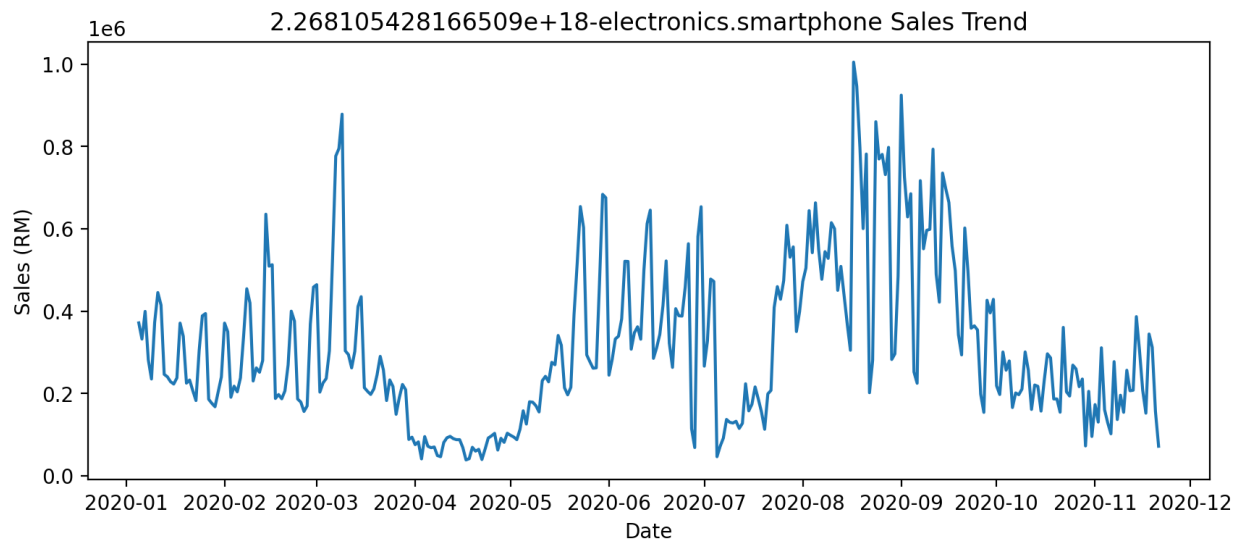


Figure 4.17: Time series plot showing the sales of products from “2.268105428166509e+18-electronics.smartphone” category.

From Figure 4.17, the presence of trend and seasonality component can be observed from the TS plot. This indicates that the daily sales data for the selected category is a non-stationary TS data. This claim can be verified by performing the Augmented Dickey-Fuller Test (ADF Test) on the daily sales data for the selected category to test the inference. In Python, the ADF Test is performed using “adfuller” function implemented in the “statsmodels” library. The result of the ADF Test performed on the daily sales data is shown in Figure 4.18.

```

ADF TEST:

The test statistic: -2.164866
p-value: 0.219243
Critical Values:
1%: -3.451
5%: -2.871
10%: -2.572

```

Figure 4.18: The result of ADF test on the original smartphone category sales data.

The null (H_0) and the alternate hypotheses (H_1) of the ADF Test are listed below:

- H_0 : There is a unit root in the TS data.
- H_1 : The TS data is stationary.

The test statistic shows a negative value, but the p -value is 0.219243 which is higher than the significance level of 0.05. Hence, we fail to reject H_0 . This verifies that the daily sales data of the selected category is not stationary and thus, differencing is required to

obtain stationary data. Data stationarity is important in TS analysis and forecasting to ensure the validity of the results obtained from the standard techniques used to analyze TS data.

Next, differencing is performed by subtracting the previous observation from the current observation. Figure 4.19 shows the result of the ADF test performed on the differenced data.

```
ADF TEST:

The test statistic: -8.085826
p-value: 0.000000
Critical Values:
1%: -3.451
5%: -2.871
10%: -2.572
```

Figure 4.19: The result of ADF test on the differenced smartphone category sales data.

From Figure 4.19, the test statistic shows -8.085826, which is more negative value than before, and the p -value having 0.000000, which is lower than the significance level of 0.05. Therefore, there is strong evidence to reject H_0 , and the differenced data is said to be stationary. Since differencing is performed only once on the daily sales data of the selected category to achieve stationarity, the value of the integration order, d is set as 1.

The next step is to determine the parameter of ARIMA model. partial autocorrelation function (PACF) and autocorrelation function (ACF) are used to identify the values of autoregressive component, p and the moving average component, q , respectively. The ACF and PACF plots for the first 40 lags for the daily sales data after first differencing are shown in Figure 4.20.

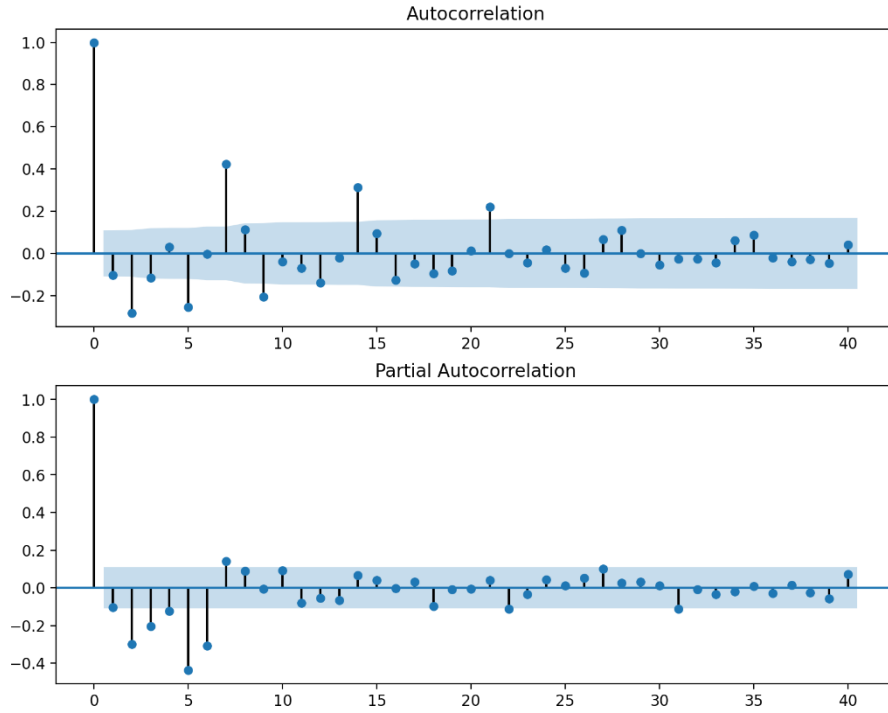


Figure 4.20: The result of the ACF and PACF plot of the daily sales data after first differencing.

The parameters of the ARIMA model can be identified from ACF and PACF plots in Figure 4.20. The parameter selection is justified in Table 4.6.

Table 4.6: The setup of the ARIMA parameters.

Parameter	Order	Explanation
Autoregressive component	$p = 7$	There are seven significant spikes in the first 7 lags in the PACF plot.
Differencing component	$d = 1$	Data stationarity is achieved after differencing the daily sales data once.
Moving-average component	$q = 0$	There is horizontal S trend throughout the ACF plot as it shows no significant spike for the q .

With the known values of ARIMA parameters, the ARIMA model can be modelled, trained, and tested on the daily sales data for the selected category to verify the performance of the ARIMA model.

Machine Learning: XGBoost Regressor

In this project, machine learning approach is also used to predict and forecast the daily sales of the electronic category. The selected machine learning algorithm is XGBoost

Regressor, which is an algorithm involving gradient boosted trees designed for performance.

Firstly, the daily sales data is transformed into the format consisting of several features and target attributes, which can be fit into a supervised problem. In this case, the target attribute is the category sales for each recorded day, and feature attributes are the value of daily sales from previous days that affect the sales for each recorded day. As such, rolling windows technique is applied to the daily sales data to determine the number of previous days, in which, the sales are most correlated to the sales of the recorded days. For this purpose, the daily sales amount from the previous 10 days are set as the list of features associated to the sales data of each recorded day. Then, the correlation between the list of feature attributes (daily sales of the previous 10 days) and the target attribute (daily sales of the recorded day) can be identified. As a result, the sales data from the previous 8 days are most correlated to the sales of the recorded days (with correlation value above the threshold value, 0.45), forming a 9-day lags used for building a machine learning model for TS forecasting.

Next, hyperparameters tuning process is performed to identify the hyperparameter values of the XGBoost regressor used for forecasting task to a specific TS data. During the modelling process, the daily sales data must be sorted by date as it is a sequential problem. There are different ways to split the data used for machine learning modelling into training and test sets, such as percentage split and splitting according to specific number of days. In this case, the sales data from the last 30 days (from 23 October to 21 November 2020) is set as the test set, while the sales data from the remaining days are set as the training set.

In Python, “GridSearchCV” function implemented in the “sklearn” library is used to identify the best hyperparameters for the XGBoost regressor model. The “cv” parameter is set to an instance of TimeSeriesSplit with 10 splits on the training set, whereas the “random_state” parameter is set to 42. The hyperparameters are used to find the best model in Table 4.7.

Table 4.7: The parameter values used for hyperparameter tuning in XGBoost regressor using GridSearchCV.

Hyperparameter	Tuning values
n_estimator	[100, 1000, 2000]
learning_rate	[0.01, 0.1, 0.3, 0.5]
max_depth	[6, 7, 8, 10]

The best XGBoost regressor model is obtained from the process of GridSearchCV, and the best hyperparameters values are obtained as below:

- n_estimators: 100
- learning_rate: 0.1
- max_depth: 10

Until this step, it is good to do the prediction on the train set and test set to observe the model performance readings.

Result Analysis

The result analysis between the ARIMA and XGBoost regressor model implemented for TS forecasting are conducted and presented in terms of different aspects, namely the model performance metrics, prediction plot and the forecasting plot. For an ARIMA model, the model performance is commonly measured using the AIC and BIC readings. These readings must be as low as possible to produce a good model for the TS analysis and forecasting tasks. In Python, “auto_arma” function implemented in the “pmdarima” library is used to obtain the best values for the order of p , d , and q respectively. By using the daily sales data, the “auto_arma” function identifies an ARIMA model with p , d , and q order of 7, 1, 0 respectively with the AIC reading of 8398.94.

With the setting of ARIMA(7,1,0) fitted with the dataset, it yields the reading of AIC and BIC at 8396.91 and 8427.08. Then, residuals diagnostic conduct on the ARIMA model to ensure that no more information is left for extraction. The ACF and PACF of the residuals are plotted in Figure 4.21.

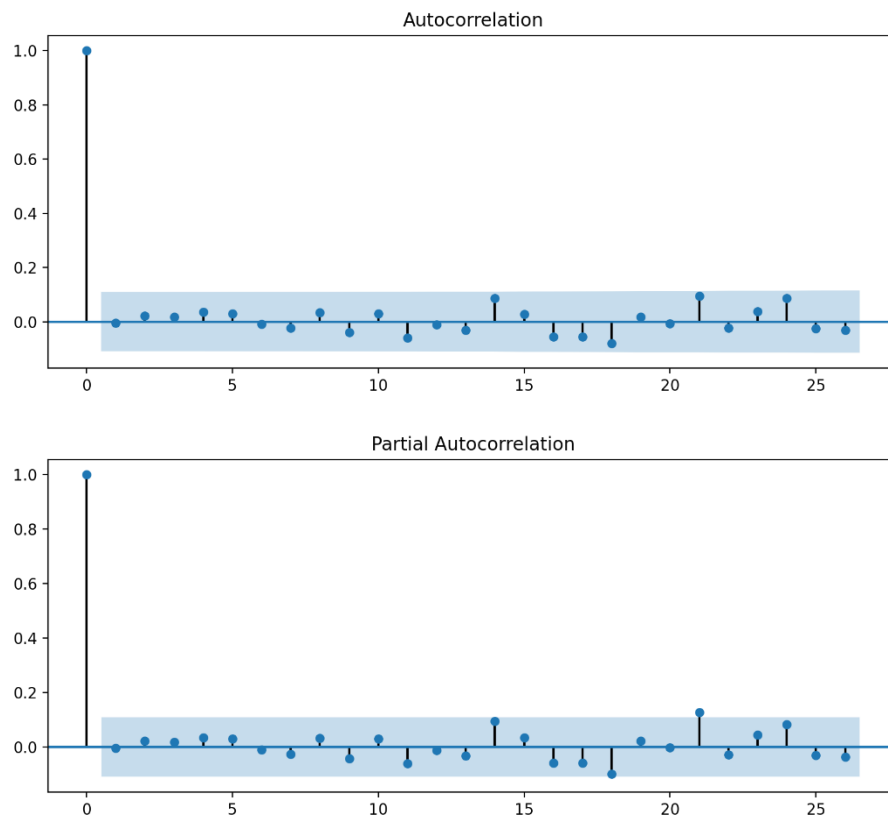
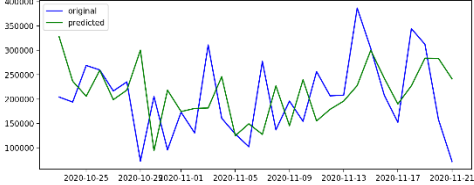
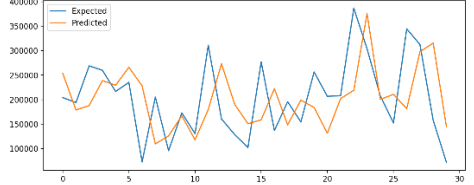
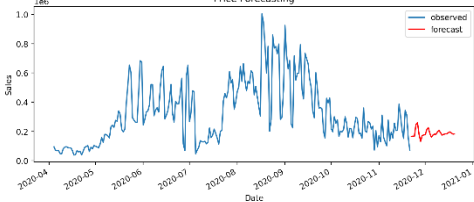
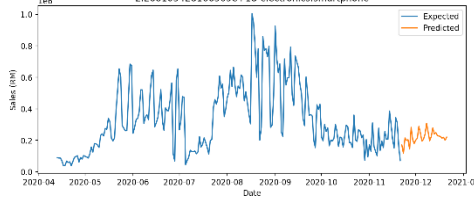


Figure 4.21: The ACF and PACF plot of the residuals.

The residual diagnostics looks fine as almost 100% of the sample autocorrelations are between the limit boundaries and it meets the criteria for a good predict and forecast.

The comparison between ARIMA(7,1,0) and the XGBoost model in terms of analysis is summarized in Table 4.8.

Table 4.8: Comparison of ARIMA(7,1,0) model and XGBoost Regressor in TS analysis and forecasting.

	ARIMA(7,1,0)	XGBoost Regressor
Parameters	$p = 7$ $d = 1$ $q = 0$	n_estimators: 100 learning_rate: 0.1 max_depth: 10
Performance metrics	MSE: 8954281994.78 RMSE: 94627.07 MAE: 74608.42 MRE: 0.50	MSE: 7061066569.06 RMSE: 84030.15 MAE: 67641.38 MRE: 0.40
In-sample 30-days prediction		
Out-of-sample 30-days forecast		

From the table above, it shows that machine learning model which is XGBoost regressor model outperforms ARIMA model in the in-sample 30-days prediction task as XGBoost regressor model has lower errors in every performance metric such as mean squared error, root mean squared error, mean absolute error and mean relative error.

Furthermore, the in-sample 30-days prediction of the daily sales data using the XGBoost regressor model shows a pattern that is very close to original daily sales data. On the other hand, there are some over predicted and under predicted from original values the prediction using the ARIMA model.

When come to forecasting out-of-sample 30-days comparison, it shows that XGBoost model is forecasting multiple-step ahead with reasonable sales range and trend whereas ARIMA model is forecasting the value in narrow range, and it trends to forecast in contraction at the end.

5. Conclusion and Future Research

In short, multiple approaches have been analysed and studied to achieve the project objectives and result are discussed. It is a wise step taken by the business owner to utilize the available data for his next strategy plan and to stay competitive in this challenging market.

Market Basket Analysis is a simple approach, to analyse, to generate the outcome and to be implemented for different retail items and widely practiced by different industry including retail and ecommerce platform. It can be categorised to match with the business owners' target and preference, and hence to be analysed under Market Basket Analysis to compute and conclude the likelihood of electronic items being purchased together for a better decision making in marketing and commercial promotional activities, which include cross-selling and bundles sales, as well as to assist in better inventory management and customer satisfaction improvement. Plenty of such applications in the real business world for both ecommerce platform like Amazon website in recommending 'frequently bought together' items as well as by bricks-and-mortar stores where strategy to merchant display is derived from the result of Market Basket Analysis. These are the references for the electronic store owner on his next business strategy plan with higher confidence to boost the sales and revenues online as part of his expansion plan and potentially offline.

On the other hand, recommender system can be implemented in various businesses and platforms to generate a list of recommended products to the users or customers. Implementing recommender systems in a business platform brings several benefits to both the business owner and its customers. First, the implementation in the ecommerce platform helps increase the sales revenue to the electronics store owner by selling more products, including those recommended products that might not have known by the customers. Furthermore, customers can save time by quickly navigating to those products from the list of recommendations generated based on their previous interactions and activities in the ecommerce platform. In this project, different types of recommender systems are implemented using collaborative filtering approach to generate list of products to the customers. During the implementation, customer preferences towards products in the electronics store is assumed to remain the same for an extensive period. The result shows that Item-to-Item collaborative filtering recommender systems implemented using weighted k-NN algorithm with $k=100$ yields the best performance in generating list of products relevant to the electronics store customer. This provides another option to the electronics store owner to boost the sales of the electronics products using the information of the customers' previous activities on the electronics platform. For future research, recommender systems can also be implemented using more advanced techniques such as deep learning and reinforcement learning.

From the time series forecasting experiments and modelling process, ARIMA and XGBoost models are able to be trained successfully within the electronic category sales. Based on the performance of the models, XGBoost model is performing better than ARIMA with lower errors, and XGBoost model can be implement for the ecommerce

provide forecast services to all their store owners. So that the store owners can plan for the next actions such as restock planning, and market retreat planning before the event happens.

Currently, the time series forecast model only for univariate data, so that it can be enhanced with multivariate data forecasting in the futures as it provides multiple categories sales analysis and forecasting in single view. In addition, it can be enhanced to have auto model parameter tuning by feed in the different dataset based on the end user selection. With the auto tuned model, it could provide more customized forecasting function than using static model for all kind of dataset.

References

- [1] O. Novikova and K. Zhang, "Analyses of the E-Commerce Development in the World and China," *Modern Economics*, vol. 23, p. 155–160, 2020.
- [2] T. He, H. Yin, Z. Chen, X. Zhou and B. Luo, "Predicting Users' Purchasing Behaviors Using Their Browsing History," *Lecture Notes in Computer Science Databases Theory and Applications*, p. 129–141, 2015.
- [3] J. B. Schafer, J. A. Konstan and J. Riedl, "E-Commerce Recommendation Applications," *Applications of Data Mining to Electronic Commerce*, p. 115–153, 2001.
- [4] F. Karimova, "A Survey of e-Commerce Recommender Systems," *European Scientific Journal, ESJ*, vol. 12, p. 75, 2016.
- [5] Z. Song, Y. Sun, J. Wan, L. Huang and J. Zhu, "Smart e-commerce systems: current status and research challenges," *Electronic Markets*, vol. 29, p. 221–238, 2017.
- [6] Y. Zhang, H. Abbas and Y. Sun, "Smart e-commerce integration with recommender systems," *Electronic Markets*, vol. 29, p. 219–220, 2019.
- [7] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun and L. Cao, "Intention Nets: Psychology-Inspired User Choice Behavior Modeling for Next-Basket Prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, p. 6259–6266, 2020.
- [8] J. A. Konstan and J. Riedl, "Recommender systems: from algorithms to user experience," *User Modeling and User-Adapted Interaction*, vol. 22, p. 101–123, 2012.
- [9] A. Nair, X. Ning and J. H. Hill, "Using recommender systems to improve proactive modeling," *Software and Systems Modeling*, 2021.
- [10] M. Zeng, H. Cao, M. Chen and Y. Li, "User behaviour modeling, recommendations, and purchase prediction during shopping festivals," *Electronic Markets*, vol. 29, p. 263–274, 2018.
- [11] T. Geurts, S. Giannikis and F. Frasincar, "Active learning strategies for solving the cold user problem in model-based recommender systems," *Web Intelligence*, p. 1–15, 2021.
- [12] G. Faggioli, M. Polato and F. Aiolli, "Recency Aware Collaborative Filtering for Next Basket Recommendation," *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*, 2020.

- [13] X. Xie, F. Sun, X. Yang, Z. Yang, J. Gao, W. Ou and B. Cui, "Explore User Neighborhood for Real-time E-commerce Recommendation," *arXiv preprint arXiv:2103.00442*, 2021.
- [14] D. Jannach and G. Adomavicius, "Price and profit awareness in recommender systems," *arXiv preprint arXiv:1707.08029*, 2017.
- [15] M. Yuan, Y. Pavlidis, M. Jain and K. Caster, "Walmart online grocery personalization: Behavioral insights and basket recommendations," in *International Conference on Conceptual Modeling*, 2016.
- [16] G. Kronberger and M. Affenzeller, "Market basket analysis of retail data: supervised learning approach," in *International Conference on Computer Aided Systems Theory*, 2011.
- [17] Y. Kurnia, Y. Isharianto, Y. C. Giap, A. Hermawan and others, "Study of application of data mining market basket analysis for knowing sales pattern (association of items) at the O! Fish restaurant using apriori algorithm," in *Journal of Physics: Conference Series*, 2019.
- [18] Y. U. Jian-hong and L. E. Xiao-juan, "Sales forecast for amazon sales based on different statistics methodologies," *DEStech Transactions on Economics, Business and Management*, 2016.
- [19] F. Kurniawan, B. Umayah, J. Hammad, S. M. S. Nugroho and M. Hariadi, "Market Basket Analysis to identify customer behaviours by way of transaction data," *Knowledge Engineering and Data Science*, vol. 1, p. 20, 2018.
- [20] R. Khandelwal, H. Kanwar and others, "Analysing Customer's Purchasing Pattern by Market Basket Analysis," *i-Manager's Journal on Computer Science*, vol. 7, p. 43, 2019.
- [21] A. Sinha, "Implying Association Rule Mining and Market Basket Analysis for Knowing Consumer Behavior and Buying Pattern in Lockdown-A Data Mining Approach," 2021.

Appendix A. Important Source Code: Data Preprocessing for Market Basket Analysis

The Python code snippet below shows the important source code to generate the data for MBA. The complete Python code can be found in the submitted notebook “Data_Preprocessing_MBA.ipynb”.

```
# =====
# Load purchase data
# =====

df_purchase = pd.read_csv("src/kz.csv")

# =====
# Derive new columns
# =====

# Convert 'event_time' to date type
df_purchase['event_time'] = pd.to_datetime(df_purchase['event_time'])
df_purchase['event_time'] = df_purchase["event_time"].dt.date

# Derive new columns: `product_name`
df_purchase['category_code'] =
df_purchase['category_code'].fillna('unknown_category')
df_purchase['brand'] = df_purchase['brand'].fillna('unknown_brand')
df_purchase['product_name'] = df_purchase['category_code'] + '-' +
df_purchase['brand']

# =====
# Filter purchase data
# =====

df_purchase_with_price = df_purchase[~df_purchase['price'].isna()]
df_purchase['unit_sold'] = 1

# -----
# Generate set of product (total sales >= MIN_PRODUCT_SALES)
# -----

total_sales_by_product = (df_purchase
                           .groupby('product_name')
                           .sum()
                           .sort_values(by='price', ascending=False)[['price',
'unit_sold']])

MIN_PRODUCT_SALES = 30_000
```

```

product_sales = total_sales_by_product[total_sales_by_product['price'] >=
MIN_PRODUCT_SALES]

set_product = set(product_sales.index)

# -----
# Generate set of order ID (amount >= MIN_ORDER_PRICE)
# -----

total_sales_by_order = (df_purchase
                        .groupby('order_id')
                        .sum()
                        .sort_values(by='price', ascending=False)[['price',
'unit_sold']])

MIN_ORDER_PRICE = 200

order_price = total_sales_by_order[total_sales_by_order['price'] >=
MIN_ORDER_PRICE]

set_order = set(order_price.index)

# -----
# Define conditions for filtered records
# -----

df_purchase_mba = df_purchase[['order_id', 'product_name']]

df_purchase_mba['popular_product'] =
df_purchase_mba['product_name'].apply(lambda x: x in set_product)
df_purchase_mba['order_over_min_sales'] =
df_purchase_mba['order_id'].apply(lambda x: x in set_order)

CONDITION = ((df_purchase_mba['popular_product'] == True) &
              (df_purchase_mba['order_over_min_sales'] == True))

# -----
# Group filtered records by `order_id` and `product_name`
# -----

df_purchase_group_by_order_n_product = (df_purchase_mba[CONDITION]
                                         .groupby(['order_id', 'product_name'])
                                         .count())

# =====
# Generate pivot table for MBA
# =====

```



```

# Convert grouped records into pivot table
df_purchase_mba_matrix = (df_purchase_group_by_order_n_product
                           .pivot_table(index='order_id',
                                         columns='product_name',
                                         values='popular_product'))

# Convert cell values into Boolean:
# ... False: missing value; True: otherwise.
df_purchase_mba_matrix = ~df_purchase_mba_matrix.isna()

# =====
# Save data
# =====

df_purchase_mba_matrix.to_csv('output/mba_matrix.csv')

```

Appendix B. Important Source Code: Data Preprocessing for Recommender System

The Python code snippet below shows the important source code to generate data for RS. The complete Python code can be found in the submitted notebook “Data_Preprocessing_RS.ipynb”.

```
# =====
# Load user behaviour data
# =====

df_oct = pd.read_csv('src/2019-Oct.csv.zip')
df_oct = df_oct[['event_type', 'category_code', 'brand', 'user_id']]

# =====
# Derive new columns: `product_name` and `product_id` (new),
# ... and `preference`
# =====

# Fill missing values with "unknown_{category/brand}"
df_oct['category_code'] = df_oct['category_code'].fillna('unknown_category')
df_oct['brand'] = df_oct['brand'].fillna('unknown_brand')

# Create `product_name`
df_oct['product_name'] = df_oct['category_code'] + '-' + df_oct['brand']

# Generate product ID (new) - this product ID helps in identifying
# ... which products are recommended by RS in RapidMiner.
prod_dict = {prod: i for i, prod in enumerate(list_prod)}
df_oct['product_id'] = df_oct['product_name'].apply(lambda x: prod_dict[x])

# Assign weight to each event
weight_dict = {'view': 1, 'cart': 3, 'purchase': 10}
df_oct['preference'] = df_oct['event_type'].apply(lambda x: weight_dict[x])

# =====
# Filter user behaviour data
# =====

# -----
# Generate set of users (preference >= MIN_USER_PREF)
# - approximately 2000 users
# -----

df_group_user = (df_group_user_prod
                  .groupby(level=0)
```

```

        .count()
        .sort_values(by='preference', ascending=False))

total_user = df_group_user.shape[0]
top_n_user = 2000

MIN_USER_PREF = df_group_user.quantile(1-(top_n_user/total_user)).values[0]
df = df_group_user
df_group_user = df[df['preference'] >= MIN_USER_PREF]

set_user_rs = set(df_group_user.index.values)

# -----
# Generate set of products (preference >= MIN_PROD_PREF)
# - approximately 1000 products
# -----

df_group_prod = (df_group_user_prod
                  .groupby(level=1)
                  .count()
                  .sort_values(by='preference', ascending=False))

total_prod = df_group_prod.shape[0]
top_n_prod = 1000

MIN_PROD_PREF = df_group_prod.quantile(1-(top_n_prod/total_prod)).values[0]
df = df_group_prod
df_group_prod = df[df['preference'] >= MIN_PROD_PREF]

set_prod_rs = set(df_group_prod.index.values)

# -----
# Define conditions for filtered records
# -----

df_group_user_prod = (df_oct[['user_id', 'product_id', 'preference']]
                      .groupby(['user_id', 'product_id']).sum())

df = df_group_user_prod

# Convert index (`user_id` and `product_id`) back into columns.
df = df.reset_index()

# Create two new columns to help filter process
df['keep_user'] = df['user_id'].apply(lambda x: x in set_user_rs)
# df['keep_product'] = df['product_name'].apply(lambda x: x in set_prod_rs)
df['keep_product'] = df['product_id'].apply(lambda x: x in set_prod_rs)

CONDITION = (df['keep_user'] & df['keep_product'])

```

```

# -----
# Filter user preference records
# -----

df = df[CONDITION]
df = df.reset_index(drop=True)

df_group_user_prod = df[['user_id', 'product_id', 'preference']]

# =====
# Split user preference data into training and test sets
# - Train:test = 80:20
# =====

from sklearn.model_selection import train_test_split
df_train, df_test = train_test_split(df_group_user_prod,
                                     stratify=df_group_user_prod['user_id'],
                                     test_size=0.20,
                                     random_state=42)

# =====
# Save data
# =====

df_train.to_csv('output/train_oct.csv', index=False)
df_test.to_csv('output/test_oct.csv', index=False)

```

Appendix C. Important Source Code: Data Preprocessing and Implementation of ARIMA Model

The Python code snippet below shows the important source code to preprocess the data and implement ARIMA model. The complete Python source code, including the implementation of helper functions and machine learning models for time series analysis and forecasting, can be found in the submitted notebook “Time_Series_Forecasting.ipynb”.

```
# =====
# Load purchase data
# =====

df_purchase = pd.read_csv('src/kz.csv')
df_cleaned = df_purchase.copy()

# =====
# Data Preprocessing
# =====

# -----
# Derive new column: `product_name`, `category`
# -----

df_cleaned['category_code'] =
df_cleaned['category_code'].fillna('unknown_category')
df_cleaned['brand'] = df_cleaned['brand'].fillna('unknown_brand')
df_cleaned['product_name'] = df_cleaned['category_code'] + '-' +
df_cleaned['brand']
df_cleaned['category'] = df_cleaned['category_id'].astype(str)
df_cleaned['category'] = df_cleaned['category'] + '-' +
df_cleaned['category_code']

# -----
# Convert `event_time` to date type
# -----

df_cleaned = df_cleaned[['event_time', 'category', 'price']]
df_cleaned['event_time'] = pd.to_datetime(df_cleaned['event_time'])
df_cleaned['event_time'] = df_cleaned["event_time"].dt.date

# -----
# Drop rows with missing price and date == "1970-01-01"
# -----
```

```

df_cleaned = df_cleaned[~df_cleaned['price'].isna()]
display(df_cleaned.shape)

df_cleaned = df_cleaned[df_cleaned['event_time'] > dt.date(1970, 1, 1)]
display(df_cleaned.shape)

# -----
# Explore the data to select category
# -----

df_category_all = df_cleaned.groupby('category').agg({ 'price':
['mean', 'min', 'max', 'count']})
df_category_all = df_category_all.sort_values([('price', 'count')],
ascending=False)
display(df_category_all.head(10))

# -----
# Generate daily sales data (time series data) for
# ... selected category
# -----

cat = '2.268105428166509e+18-electronics.smartphone'
display(df_cleaned[df_cleaned['category'] == cat])

df_category = df_cleaned[df_cleaned['category'] == cat]
df_category = df_category.groupby(['event_time'], sort = True).agg( {'price':
'sum'})

period_end_date = df_cleaned['event_time'].max()
cat_start_date = df_category.index.min()
cat_end_date = df_category.index.max()

date_range = []

i_date = cat_start_date
while i_date <= period_end_date:
    #print(i_date)
    if (i_date not in df_category.index):
        date_range.append(i_date)

    i_date = i_date + dt.timedelta(days=1)

df_cat_zero = pd.DataFrame(index = date_range, columns = df_category.columns)
df_cat_zero = df_cat_zero.fillna(0.00)

df_cat_ = df_category.copy()

pdList = [df_cat_, df_cat_zero]

```

```

df_cat_result = pd.concat(pdList)
df_cat_result = df_cat_result.sort_index()

# =====
# ARIMA Model
# - using helper functions
# =====

# -----
# Step 1: Check stationarity
# -----

PlotCategorySaleLineChart(df_cat_result, 'price', cat+' Sales Trend')
CheckStationarity(df_cat_result.price)

# -----
# Step 2: Differencing
# -----

df_shift = Differencing(df_cat_result, 'price')
plt.plot(df_shift)
CheckStationarity(df_shift['shift'].dropna())

# -----
# Step 3: Plot ACF and PACF to obtain p, d and q order.
# -----

PlotACFPACF(df_shift, 'shift')

auto_arima_fit = pm.auto_arima(df_cat_result.price, start_p=1, start_q=1,
                               max_p=13, max_q=13, m=12,
                               seasonal=True,
                               trace=True,
                               error_action='ignore',
                               suppress_warnings=True,
                               stepwise=True)

# -----
# Step 4: Build ARIMA model and check on residual diagnosis
# -----

arima_model = sm.tsa.arima.ARIMA(df_cat_result.price, order=(7,1,0))
arima_model_fit = arima_model.fit()
print(arima_model_fit.summary())

residuals = pd.DataFrame(arima_model_fit.resid)

```

```

display(residuals.plot())
display(residuals.plot(kind='kde'))
display(residuals.describe())

plot_pacf(residuals)
plot_acf(residuals)

# -----
# Step 5: Check test set output performance on ARIMA(7,1,0)
# -----

test_days = 30
train = df_cat_result.iloc[:-test_days]
test = df_cat_result.iloc[-test_days:]
test_start = len(train)

y_pred_ = arima_model_fit.predict(test_start, test_start +test_days-1)
y_true_ = test.price

rmse1 = rmse(y_true_, y_pred_)
mse1 = mean_squared_error(y_true_, y_pred_)
mae1 = mean_absolute_error(y_true_, y_pred_)
mre1 = mean_relative_error(y_true_, y_pred_)

print('ARIMA(7,1,0)')
print('MAE: {0}'.format(str(round(mae1,2))))
print('RMSE: {0}'.format(str(round(rmse1,2))))
print('MSE: {0}'.format(str(round(mse1,2))))
print('MRE: {0}'.format(str(round(mre1,2))))

plt.figure(figsize=(10,5))
plt.plot(test,label = 'original', color='b' )
plt.plot(y_pred_,label = 'predicted', color='g' )
plt.legend()

# -----
# Step 6: Multi-Step Out-of-Sample Forecast
# -----

forecast_days = 30
view_days = 90

forecast = arima_model_fit.forecast(steps=forecast_days)
df_forecast =
pd.DataFrame({'forecast':forecast},index=pd.date_range(start=period_end_date,
periods=forecast_days, freq='D'))
#print(df_forecast.head(forecast_days))

```



```
ci = df_forecast.values
ax = df_cat_result[view_days:].price.plot(label='observed', figsize=(15, 6))
df_forecast.plot(ax=ax, label='Forecast', color='r')
ax.fill_between(df_forecast.index,
                ci[:, -1],
                ci[:, 0], color='b', alpha=.25)
ax.set_xlabel('Date')
ax.set_ylabel('Sales')
ax.set_title(cat + ' Price Forecasting')
plt.legend()
plt.show()
```

Appendix D. Market Basket Analysis: Data Details and Screenshot of RapidMiner processes

Dataset has been pre-processed to fit to use in RapidMiner Studio for the purpose of Market Basket Analysis study, as shown in Figure D-1, where data indicated if found in the basket with true/false.

Open in		Filter (56,604 / 56,604 examples): all											
Turbo Prep		Auto Model											
Row No.	apparel.glov...	apparel.glov...	apparel.glov...	apparel.shir...	apparel.tshi...	appliances...	appliances...	appliances...	appliances...	appliances...	appliances...	appliances...	appliances...
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False	False	False	False	False	False
10	False	False	False	False	False	False	False	False	False	False	False	False	False
11	False	False	False	False	False	False	False	False	False	False	False	False	False
12	False	False	False	False	False	False	False	False	False	False	False	False	False
13	False	False	False	False	False	False	False	False	False	False	False	False	False
14	False	False	False	False	False	False	False	False	False	False	False	False	False
15	False	False	False	False	False	False	False	False	False	False	False	False	False
16	False	False	False	False	False	False	False	False	False	False	False	False	False
17	False	False	False	False	False	False	False	False	False	False	False	False	False

Figure D-1: Dataset loaded in RapidMiner & indicated with True/False for each attribute.

Selected attributes for the project purpose has been identified and selected, as performed and summarised under Figure D-2.

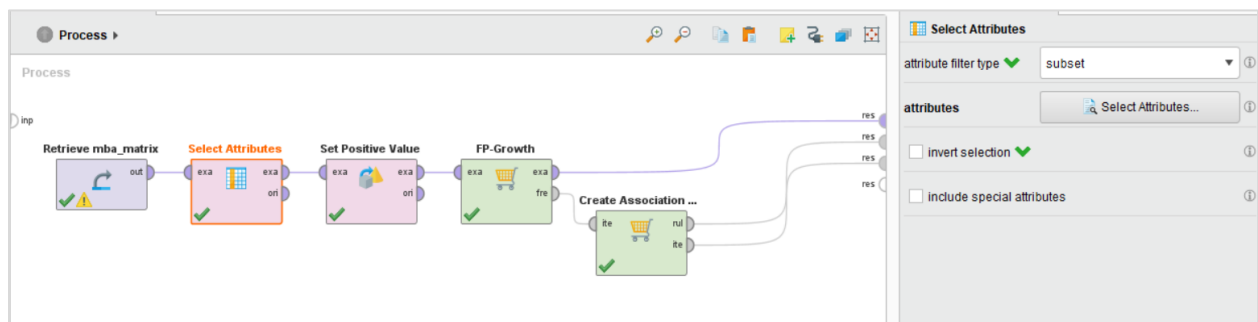


Figure D-2: 'Select Attributes Operator' – to select subset of attributes to be analysed

Under input format, 'items in dummy coded columns' is selected, which described items in the dataset has its own column with the column name as the item name. The binomial values under each transaction indicates if the item can be found in the basket/transaction. Positive value parameter has been set and values identified prior to this although data was processed to be binomial. Subsequently, the support min values has been tested from 0.2 to 0.001 to observe and record the result generated. This is shown under Figure D-3, as part of the process of MBA in RapidMiner Studio.

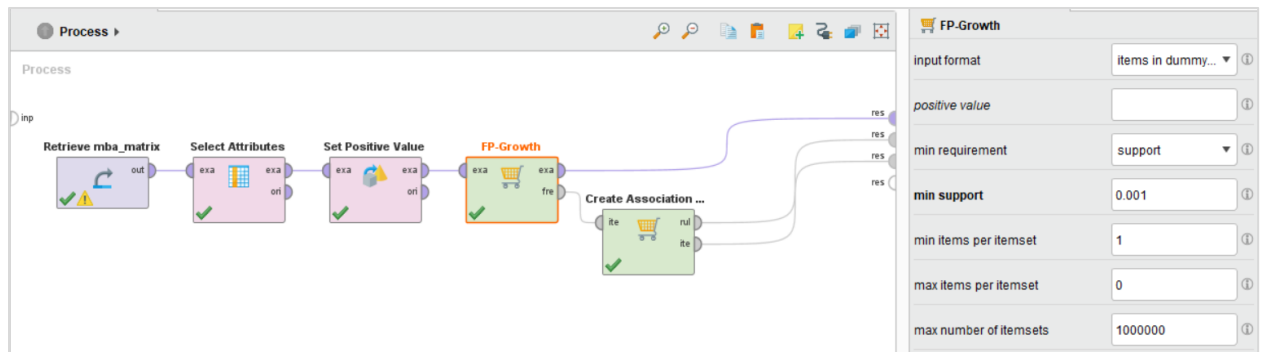


Figure D-3: 'FP-Growth' – input format and support min filled as choice

No association rule could be generated when the support min is set at 0.2 and hence it is further set at 0.05 to test the outcome, as illustrated under Figure D-4.

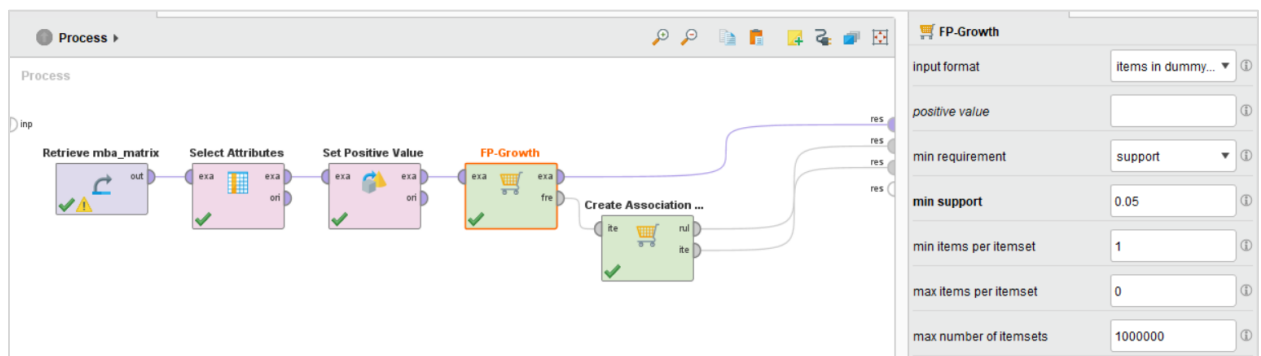


Figure D-4: 'FP-Growth' Operator – input format and support min filled as choice

Following the above, the operator 'create Association Rule' has been added (as shown under Figure D-5), with the min confidence level set at 0.05 and the result is observed and analysed.

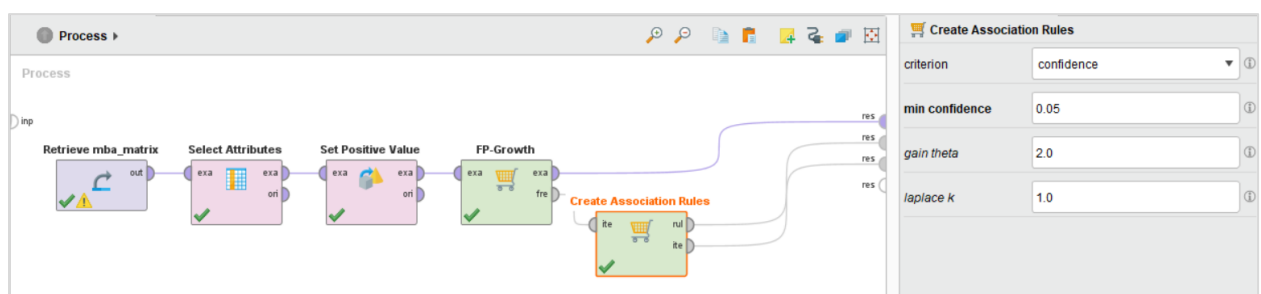


Figure D-5: 'Create Association Rules' Operator – input min confidence level at 0.05

Appendix E. Recommender System: Data Details and Screenshot of RapidMiner processes

The first 20 rows of the user preference data used for training and testing the CFRSs are shown in Table E-1 and Table E-2. The full set of data is submitted and can be retrieved from the files named “train_oct.csv” and “test_oct.csv” respectively.

Table E-1: Training set of the user preference data used for implementing CFRSs.

user_id	product_id	preference
512691530	4225	1
519209046	2443	9
514377205	2698	2
513670761	5584	6
515172430	1304	2
513874398	1487	6
512980445	3836	4
543316845	1420	1
546277040	2962	1
512955066	2769	20
513757665	1476	7
512462542	2678	14
512823199	1418	8
560585156	4086	3
558216853	5584	41
516000047	5133	4
561102380	3893	5
542048657	535	33
520990592	2774	4
520568355	4353	3

Table E-2: Test set of the user preference data used for implementing CFRSs.

user_id	product_id	preference
512636403	1728	1
556658830	2992	3
512858746	4030	2
519780241	4896	3
514127132	1351	2
513107821	3203	1
557356803	494	4
560196242	4785	6
525006081	2693	1
522051113	3939	18
514629713	1420	1
563428177	4905	4
512367874	2633	1
536456050	780	8
512633977	5592	6
512613383	2653	1
559464217	3963	1
539429414	1197	15
514171614	5762	1
519929517	2767	8

The 10 samples of product details with the matching (new) “product_id” in training and test sets in these tables are shown in Table E-3. The full set of data is submitted and can be retrieved from the files named “product_list.csv”.

Table E-3: Samples of (new) product_id with the corresponding product_name in the training and test sets of the user behaviour data.

product_id	product_name
2653	electronics.smartphone-honor

product_id	product_name
2655	electronics.smartphone-huawei
2664	electronics.smartphone-motorola
2665	electronics.smartphone-nokia
2667	electronics.smartphone-oneplus
2668	electronics.smartphone-oppo
2672	electronics.smartphone-samsung
2673	electronics.smartphone-sony
2680	electronics.smartphone-vivo
2681	electronics.smartphone-xiaomi

The parameter settings for different RSs are illustrated below. The “Set Macro” operator is used to set different parameter values of the “User k-NN”, “Item k-NN” and “BPRMF” operators for the repeated experiments.

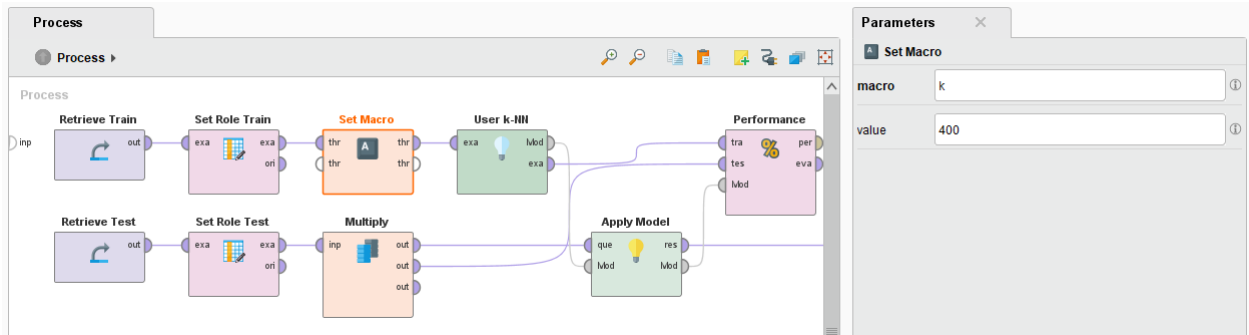


Figure E-1: User-to-User and Item-to-Item CFRS: Set Macro

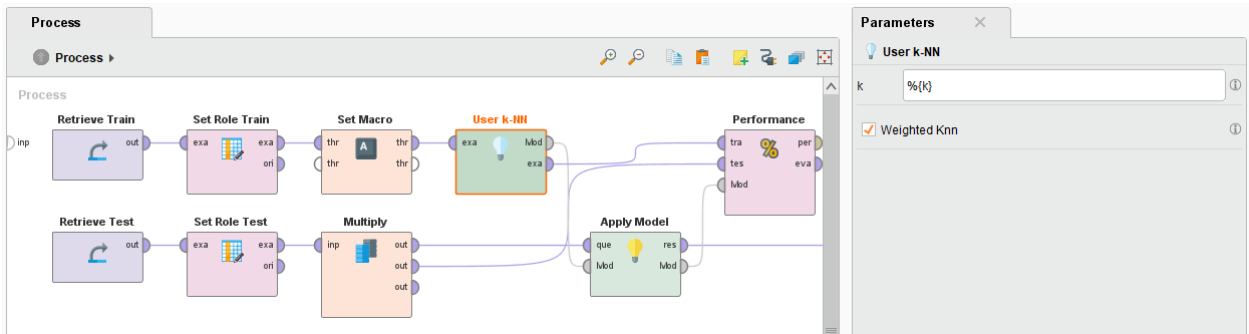


Figure E-2: User-to-User CFRS: User k-NN

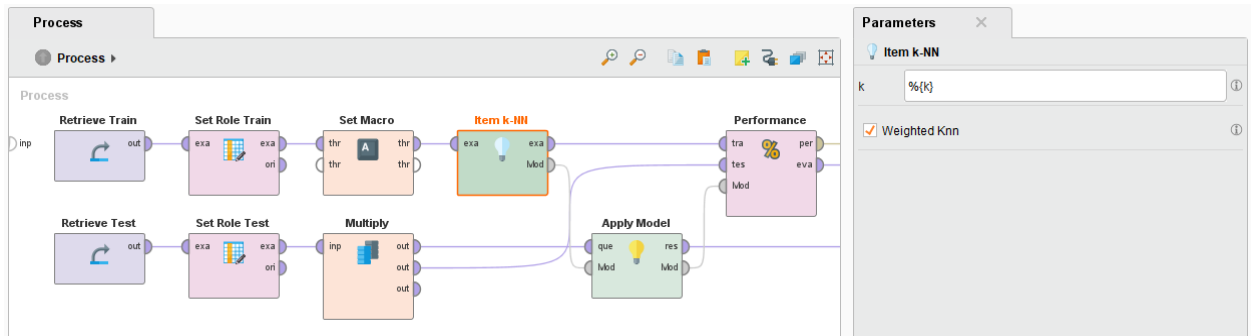


Figure E-3: Item-to-Item CFRS: Item k-NN

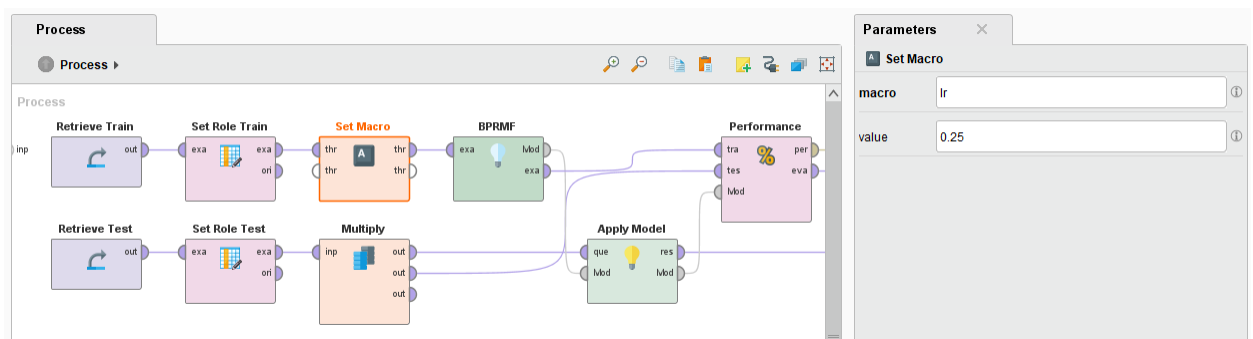


Figure E-4: BPRMF CFRS: Set Macro

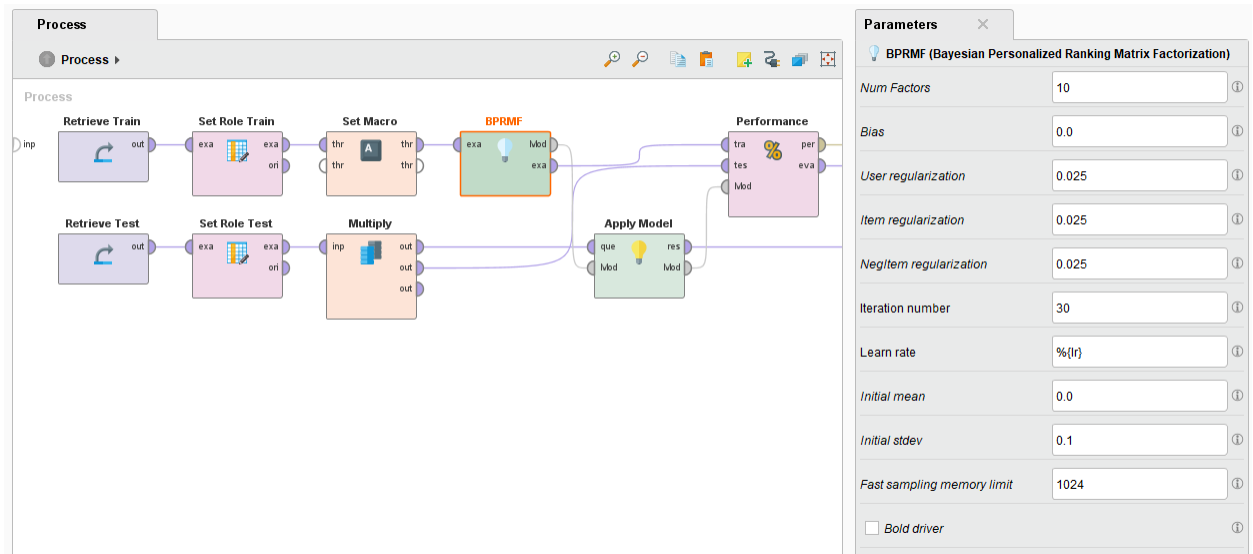


Figure E-5: BPRMF CFRS: BPRMF

Appendix F. Recommender System: User Preference Data and List of Top 10 Recommended Products to a Selected Customer (ID: 551211823)

Table F-1: User preferences on the electronics products in the training set.

No	Product ID	Product Name	Preference
1	2580	electronics.clocks-casio	1713
2	2633	electronics.clocks-unknown_brand	45
3	2638	electronics.clocks-xiaomi	38
4	5931	unknown_category-xiaomi	27
5	2672	electronics.smartphone-samsung	20
6	2443	electronics.audio.headphone-xiaomi	13
7	2623	electronics.clocks-samsung	12
8	2593	electronics.clocks-garmin	11
9	978	appliances.kitchen.kettle-tefal	9
10	2763	electronics.video.tv-samsung	9
11	936	appliances.kitchen.kettle-dauscher	8
12	2722	electronics.telephone-panasonic	6
13	4726	unknown_category-kingston	6
14	943	appliances.kitchen.kettle-galaxy	5
15	5584	unknown_category-sony	5
16	40	accessories.bag-lenovo	3
17	987	appliances.kitchen.kettle-xiaomi	3
18	1730	computers.desktop-nettechnics	3
19	2343	electronics.audio.headphone-defender	3
20	2579	electronics.clocks-canyon	3
21	5136	unknown_category-nintendo	3
22	5236	unknown_category-panasonic	3

No	Product ID	Product Name	Preference
23	5411	unknown_category-ritmix	3
24	5479	unknown_category-samsung	3
25	934	appliances.kitchen.kettle-bosch	2
26	2603	electronics.clocks-jet	2
27	2678	electronics.smartphone-unknown_brand	2
28	3964	unknown_category-canyon	2
29	5816	unknown_category-vega	2
30	32	accessories.bag-hp	1
31	1053	appliances.kitchen.microwave-samsung	1
32	2402	electronics.audio.headphone-panasonic	1
33	4558	unknown_category-huawei	1
34	5793	unknown_category-unknown_brand	1

Table F-2: User preferences on the electronics products in the test set.

No	Product ID	Product Name	Preference
1	2681	electronics.smartphone-xiaomi	12
2	968	appliances.kitchen.kettle-redmond	10
3	1733	computers.desktop-pulser	6
4	984	appliances.kitchen.kettle-vitek	5
5	4838	unknown_category-lg	4
6	971	appliances.kitchen.kettle-scarlett	3
7	2630	electronics.clocks-tissot	3
8	2589	electronics.clocks-elari	2
9	2767	electronics.video.tv-sony	2

Table F-3: List of top 10 products recommended by User-to-User CFRS (weighted k-NN, k=100) to the selected user.

Rank	Product ID	Product Name
1	2681	electronics.smartphone-xiaomi
2	2641	electronics.smartphone-apple
3	2375	electronics.audio.headphone-jbl
4	2655	electronics.smartphone-huawei
5	2571	electronics.clocks-apple
6	2425	electronics.audio.headphone-sony
7	2668	electronics.smartphone-oppo
8	2420	electronics.audio.headphone-samsung
9	2598	electronics.clocks-huawei
10	2325	electronics.audio.headphone-apple

Table F-4: List of top 10 products recommended by Item-to-Item CFRS (weighted k-NN, k=100) to the selected user.

Rank	Product ID	Product Name
1	2681	electronics.smartphone-xiaomi
2	2641	electronics.smartphone-apple
3	2655	electronics.smartphone-huawei
4	2425	electronics.audio.headphone-sony
5	2668	electronics.smartphone-oppo
6	2571	electronics.clocks-apple
7	2375	electronics.audio.headphone-jbl
8	1752	computers.notebook-acer
9	2325	electronics.audio.headphone-apple
10	2598	electronics.clocks-huawei

Table F-5: List of top 10 products recommended by BPRMF CFRS (learn rate=0.05) to the selected user.

Rank	Product ID	Product Name
------	------------	--------------

Rank	Product ID	Product Name
1	2641	electronics.smartphone-apple
2	2681	electronics.smartphone-xiaomi
3	2571	electronics.clocks-apple
4	2668	electronics.smartphone-oppo
5	2420	electronics.audio.headphone-samsung
6	2655	electronics.smartphone-huawei
7	2425	electronics.audio.headphone-sony
8	2325	electronics.audio.headphone-apple
9	2375	electronics.audio.headphone-jbl
10	5268	unknown_category-philips