Predicting a customer's e-commerce purchase

A study that combines sequential pattern mining with machine learning

R.J.A.W.M. van
Heesch
ANR: U890116

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
DEPARTMENT OF COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
TILBURG UNIVERSITY

Supervisor:	dr. A. Hendrickson
Second reader:	dr. M. Louwerse

Tilburg University
School of Humanities & Digital Sciences
Department of Cognitive Science & Artificial Intelligence
Tilburg, The Netherlands
January 2022
Word count: 8,642

Preface

This thesis is written in the context of the growth in e-commerce. In times with restrictions to go outside due to COVID-19, e-commerce became more important. However, it is hard for businesses to get a high conversion rate. For this reason, businesses must understand the purchase behavior of their customers. I hope this thesis gives insight into the purchase behavior of e-commerce customers. With knowledge about purchase behavior, businesses can create better marketing strategies and increase conversion rates.

I would like to thank my supervisor for the guidance of the thesis process, and the enjoyable thesis meetings. Even in times when we could not meet in person, the thesis meetings always kept me motivated. I would also like to thank my fellow students for the great support, feedback, and educational times in the last months.

Finally, I want to thank my parents, boyfriend, and friends who supported me during this thesis period, and for always having a listening ear.

Robin van Heesch

Tilburg, January 14th, 2022

Abstract

In the past two years, COVID-19 has been the antecedent for different developments. The growth of e-commerce is one of those developments. E-commerce businesses need to understand their customers' behavior in order to have competitive advantages and increase conversion rate. This study focuses on the prediction of a customer's e-commerce purchase.

Most research regarding the prediction of e-commerce purchase concern the creation of recommendation systems. This study uses the knowledge of customer behavior to predict whether a customer will commit to a purchase. These predictions were accomplished by using sequential pattern mining (SPM) techniques combined with machine learning algorithms.

Using SPM algorithms PrefixSpan, Closed Sequential Patterns (ClaSP), and Vertical Mining of Maximal Sequential Patterns (VMSP), several frequent sequences were extracted using data from an online electronic store. Those frequent sequences will be the features for the machine learning algorithms.

Extreme Gradient Boosting (XGBoost) and K-nearest neighbor (KNN) were used to predict whether a customer will purchase. This study compared those two models with three different feature subsets. Namely, the top five frequent sequences from PrefixSpan, ClaSP and VMSP.

Based on F1-scores, four out of six models outperformed the baseline model. Overall, XGBoost performed best. The features that were obtained from the ClaSP algorithm resulted in the best overall performance. The machine learning algorithms with features subtracted from the VMSP algorithm resulted in the worst overall performance.

Keywords: e-commerce, conversion rate, sequential pattern mining, machine learning

Table of contents

1. Introduction

1.1 Context

In today's economy, businesses are more digital-focused. As a result of the COVID-19 pandemic, countries have put a lot of restrictions on people's behavior. One of those restrictions is to stay at home as much as possible. Therefore, people tend to do more online. The same goes for shopping. As a consequence, businesses are more focused on online activities (Putranto et al., 2021). This results in the growth of electronic commerce, abbreviated as e-commerce. E-commerce is known as online transactions consisting of goods and services. Nguyen et al. (2021) explain that 80% of their respondents purchased more products through e-commerce during the pandemic than they did before. Several studies predict that this growth will remain after the pandemic (Kim, 2020; Sheth, 2020).

However, COVID-19 is not the only reason for this perceived increase in e-commerce. Traditional businesses expand the number of customers by shifting towards online stores (Pantelimon et al., 2020). Therefore, online shopping is a widely known concept nowadays (Hamed & El-Deeb, 2020). Also, online stores offer a larger collection of products and consumers experience lower purchase costs as they do not need to travel to a store anymore to acquire new items (Dolfen et al., 2019).

Due to the high number of online retail stores, it is difficult to convert visitors into customers (Soonsawad, 2013). Even though e-commerce has grown tremendously in the last few years, Yeo et al. (2017) explain that only 2% of e-commerce customers make a purchase. In addition, only 8% of 98% of window shoppers will eventually return for a purchase. This extremely low conversion rate suggests that online retailers would benefit from understanding the purchasing patterns of their (potential) customers. Understanding these patterns can make retargeted marketing more effective, and can ultimately contribute to growth in conversion rate. According to Droomer & Bekker (2020), knowledge about customers' purchase patterns is highly important because it will result in competitive advantages.

From an economic point of view, knowledge of the next e-commerce purchase can be employed to avoid economic collapse. Booming e-commerce will continue to grow in the future (Ingaldi & Ulewicz, 2018). A large number of studies already examined the improvement of conversion rate, but since e-commerce has grown tremendously during the COVID-19 pandemic, it is important to reconsider the view on customers' preferences. When the conversion rate stays low and e-commerce grows faster than brick and mortar retailers, consumer spending on e-commerce will automatically decrease (Yu et al., 2017). Further research can focus on the improvement of retargeting to get higher consumer spending and stimulate the economy.

A lot of studies already investigated the purchase behavior of e-commerce customers, which are discussed in Chapter 2. However, the scientific motivation of this study is to examine if machine learning algorithms, with sequences extracted from SPM algorithms as features, are also suitable to predict e-commerce purchases.

1.2 Research questions

In this study, it is investigated to what extent it can be predicted if an e-commerce customer will purchase a product. With this, businesses can better understand the behavior of their customers and use this information to get a higher conversion rate and eventually get more profit. Accordingly, the research question is: *To what extent can a customer's e-commerce purchase be predicted?*

To answer the research question, three sub-questions are formulated. The first part of this study is about feature extraction. Feature extraction is done with the use of three SPM algorithms. The SPM algorithms result in frequent sequences which will be the features for the machine learning algorithms. The frequent sequences that result in the highest F1-score will be evaluated as the best set of features. From this, the first sub-question can be formulated as follows:

1) *Which SPM algorithm results in the best set of features?*

The extracted sequences are the input features for two machine learning algorithms that are used for prediction. This results in several models with different features for the two machine learning algorithms. From this, the second sub-question can be formulated as follows:

2) *Which machine learning algorithm, in combination with SPM algorithms, results in the highest F1-scores?*

To check whether all three models of the best performing machine learning algorithm make the same predictions, which model performed best, and why that model performed best, confusion matrixes are made and compared. From this, the third sub-question can be formulated as follows:

3) *Which model, from the best performing machine learning algorithm, has the highest F1-score, and why does it have the highest F1-score?*

1.3 Findings

Four out of six models in this study outperform the baseline. Among the SPM algorithms, ClaSP performs best and VMSP performs worst. Among the machine learning algorithms, XGBoost performs overall better than KNN. An overview of the results is given in Chapter 5.

2. Background

In this chapter, relevant literature regarding the research question is discussed. At first, general e-commerce purchase prediction will be discussed. After that, relevant studies regarding SPM and machine learning algorithms are discussed.

2.1 E-commerce purchase prediction.

Several studies already predicted customers' e-commerce purchases in different ways. For example, one way to predict purchases is with Market Basket Analysis (MBA). MBA uses historic data for items that were bought together. Businesses use this information to understand and make predictions about online purchase behavior (Yang et al., 2013; Gangurde et al., 2017). Another example for predicting e-commerce purchases is with the use of demographic information like age, gender, and income from the customers (Bhatnagar et al., 2000; Ratchford et al., 2001). This study will make use of SPM algorithms to predict a customer's e-commerce purchase.

Clickstream data describes the path a customer takes through an online journey. According to Jia et al. (2017) clickstreams contain a lot of information about the customer's purchase preferences, and are, therefore, optimal to use in this study. Also, results from Van den Poel & Buckinx (2005) show that clickstream variables are the most important variables when it comes to e-commerce purchase behavior. Bucklin & Sismeiro (2009) conclude that clickstreams are most useful for accurate predictions. Sequences can be created with the use of these clickstreams. The frequent sequences of the dataset are found with the use of SPM algorithms. Those frequent sequences have the most interesting patterns about customers' purchase preferences. With, for example, machine learning algorithms it is possible to make predictions about these patterns (Lee et al., 2021).

2.2 Sequential pattern mining in e-commerce purchase prediction

Several studies used SPM algorithms on e-commerce data to predict customers' purchase patterns. Trivonanda et al. (2020) used SPM algorithm PrefixSpan to find frequent patterns and build a recommendation system with these patterns. They used PrefixSpan because they concluded that PrefixSpan outperforms other apriori-based and pattern-growth algorithms. In their study, sequences were made with the use of historical clickstream data from two different datasets. To make sequences they sorted the data by user ID and transaction time and included invoice ID and product ID in the sequences. The frequent patterns obtained with

PrefixSpan were used to generate predictions for the recommendation system. Two scenarios for both datasets were investigated, one scenario with product category and one without. Their results show that the use of product categories has better performances.

Pitman & Zanker (2010) designed a recommendation system to predict an *add-to-basket* event of a user. They studied several scenarios with different complexity of sequences and compared the results of those scenarios. For example, one of the scenarios they investigated to predict a customer’s *add-to-basket* action was the use of sequences that ended with *add-to-basket*. The sequences were extracted with a closed SPM algorithm on the clickstream data and were used as rules for the recommendation system. The sequences were made with users’ IP addresses and events like view, or add to cart. They conclude that the increasing complexity of sequences will lead to less applicability of those sequences and that mining closed patterns is most useful for building the highest performing recommender system.

Liu et al. (2017) investigated four different levels of granularity in data. Among those levels they investigated clickstream data. They conclude that clickstream data can contain a lot of information and not all information is needed. They presented in their research that maximal frequent patterns are most efficient when businesses want to extract information from their clickstream data. For businesses to show only the most important sequences, maximal frequent sequences can be used.

This study will use SPM algorithms to extract frequent sequences, and use those as input features for several machine learning algorithms.

2.3 Machine learning algorithms in e-commerce purchase prediction

Clickstream data from e-commerce businesses is also used several times as input for machine learning algorithms. A study by Bharathi et al. (2018) used historical clickstream data to discover different patterns based on different models. The KNN model was used to identify the interest of customers. A Naïve Bays model and Decision Tree were applied to find patterns and identify potential customers with those patterns. Patterns obtained from a Markov Model were used to predict what next page a user is likely to view. They concluded that these models are effective to find patterns in clickstream data, but that combining a cognitive model with these data mining models will improve knowledge about customers’ behavior. Koehn et al. (2020) also use clickstream data for predicting e-commerce purchase behavior. The sequences from their data are used as input for a deep learning Recurrent Neural Network (RNN) algorithm, in order to predict a purchase or a non-purchase. To make the predictions, they used several features like the number of page views, the session duration, or the counts or values of items that were put into baskets. They measured the performance of the RNN algorithm against several supervised machine learning algorithms and concluded that the RNN algorithm predicted the sequence outcomes most accurately.

Furthermore, there are several studies that investigated which machine learning algorithm is most suitable for predicting online consumer behavior. Lee et al. (2021) investigated eight different machine learning algorithms. They used different variables to predict whether a customer would purchase. Their results show that XGBoost is the best suitable algorithm, among eight others, for the prediction of online consumer behavior. Droomer & Dekker (2020) developed a Next Purchase Date (NPD) predictor with the use of two deep learning algorithms and two machine learning algorithms. The algorithms they used are a Neural Network, Recurrent Neural Network, Linear Regression, and XGBoost. From the machine learning algorithms, XGBoost performed best. Adeniyi et al. (2016) used six machine learning algorithms to produce classifications on the clickstream data and predict future product requests. Their results show that KNN outperformed all the algorithms for classification based on the F1-score and is most suitable for predictions on clickstream data.

2.4 Gap in current work

Several studies used clickstreams to predict consumer behavior. Sequences extracted with SPM algorithms are often used to build rules for recommendation systems as presented in paragraph 2.2. Also, sequences are used in machine learning algorithms as input, but not as features. In this study, sequences from historic clickstream data will be input for machine learning algorithms. Also, the top five most frequent sequences, extracted with SPM algorithms, will be used as features for the machine learning algorithms.

3. Methods

In this chapter, the general approach for the SPM algorithms and machine learning algorithms that are used will be discussed. At first, the three SPM algorithms that are used in this study will be elaborated. After that, the two machine learning algorithms and the evaluation method are explained.

3.1 Sequential pattern mining algorithms

In this study, a customer's e-commerce purchase is predicted based on historical purchase patterns from clickstream data. Frequent patterns from the clickstream data were obtained through SPM algorithms.

SPM is a data mining approach that aims to find frequent patterns in a database. Given a set of sequences and a minimum support threshold, the SPM algorithm finds all frequent patterns (Abbasghorbani & Tavoli, 2015). The minimum support threshold is the minimum percentage a pattern needs to be present in the dataset to be considered as frequent. In general, there are two different techniques related to SPM, which are the apriori-based technique and the pattern-growth technique (Mabroukeh & Ezeife, 2010). The apriori-based technique discovers the dataset in a breadth-first search approach. A breadth-first search approach starts at the top node and searches through the database layer by layer. The pattern-growth technique discovers the dataset in a depth-first search approach (Achar et al., 2013). With the depth-first search approach, each node searches the database as far as possible. Graph 1 in Appendix A shows a visualization of the differences between breadth-first search and depth-first search. All three SPM algorithms applied in this study make use of the depth-first search technique.

3.1.1 Pattern-growth

Apriori-based techniques are very inefficient. Those techniques require scanning the entire database multiple times and keeping a lot of sequences in memory that are not supported by the minimum support. The PrefixSpan technique is much more efficient because it captures only the sequences that satisfy the minimum support (Pitman & Zanker, 2010). Besides, several studies show that PrefixSpan outperforms apriori-based techniques (Jian Pei et al., 2004; Trivonanda et al., 2020). Therefore, this study will use PrefixSpan.

PrefixSpan is short for Prefix-projected Sequential Pattern Growth and makes use of the concepts prefix and suffix. To explain the concept of prefix and suffix, several sequences are defined in Table 1.

Table 1 *Example sequences*

SID	Sequence
10	<view(view-add to cart-purchase)(view-purchase)view>
20	<(view-view)purchase(add to cart-purchase)(view)>
30	<(add to cart-purchase)(view-add to cart)(view-purchase)purchase>

A prefix is a pattern that exists at the beginning of several sequences. The suffix is the residue of the sequence after the prefix. The prefix is replaced with a placeholder in the suffix. Table 2 shows several prefix and suffix from the first sequence of Table 1.

Table 2 *Prefix and Suffix*

Prefix	Suffix (prefix-based projection)
<view>	<(view-add to cart-purchase)(view-purchase)view>
<view-view>	<(_add to cart-purchase)(view- purchase)view>
<view-add to cart>	<(_purchase)(view-purchase)view>

To find frequent patterns, PrefixSpan starts with finding sequential patterns of length 1, for example the sequences <view> or <add to cart>. When all frequent patterns of length 1 are found, the dataset is divided into several projected databases. A projected database only captures the sequences with relation to the length 1 frequent pattern. All the sequences from Table 1 are presented with the length 1 sequence <view> as prefix in Table 3. This results in shorter sequences because all the events before <view> are replaced with an underscore.

Table 3 *Projected database*

SID	<view> Projected database
10	<(view-add to cart-purchase)(view-purchase)view>
20	<(_view)purchase(add to cart-purchase)(view)>
30	<(_add to cart)(view-purchase)purchase>

When all the frequent patterns of length 1 are found, the algorithm starts over with frequent patterns of length 2 in the projected databases, for example <view-add to cart>. The algorithm keeps on doing this until there is no suffix available for each prefix (Trivonanda et al., 2020).

The major strength of PrefixSpan is that no candidate sequences need to be generated. A candidate sequence is a possible frequent sequence. Furthermore, the projected databases of PrefixSpan keep decreasing because adding a prefix will shrink the sequences as represented in Table 3, and the sequences that do not contain the prefix are not included.

3.1.2 The redundancy problem

The result of pattern-growth algorithms can contain many similar sequences. For example, when the sequence <view (add to cart-purchase) view> is frequent, all of the following sequences must also be frequent: <view>, <add to cart>, <purchase>, <view-purchase>, etc. This is called the redundancy problem. There are two different approaches to reduce the number of sequences and overcome the redundancy problem. Before explaining the two approaches it is necessary to know the difference between sub-sequences and super-sequences. Given the following two sequences A: <(view-add to cart) view> and B: <(view-add to cart-purchase) view (view-add to cart) > it can be stated that that sequence A is a subsequence of sequence B. Every element of sequence A needs to be a subset of sequence B with a preserved order. Therefore, sequence B is the super-sequence of sequence A.

As mentioned, there are two different approaches to reduce the redundancy problem. The first approach is with maximal frequent patterns. A frequent sequence is maximal if no frequent super-sequences of that sequence can be found (Vo et al., 2017). The second approach is with closed frequent patterns. A frequent sequence is closed if no super-sequence has the same support. Support is the amount that a sequence occurs (Gomariz et al., 2013).

The output of a maximal pattern algorithm is smaller than pattern-growth and closed pattern algorithms. To explain this relation several sequences with their support are given in Table 4.

Table 4 *Sequences with support*

SID	Sequence	Support
10	<view-add to cart>	60%
20	<view-add to cart-purchase>	60%
30	<view-add to cart-purchase-view >	40%
40	<view-add to cart-purchase-view-purchase>	20%

Now assume that the minimum support is set to 40%. Mining the frequent patterns with a pattern-growth technique result in three frequent sequences, namely sequences 10, 20, and 30. The closed frequent sequences are sequences 20 and 30. Sequence 10 is not a closed frequent sequence, because it has a super-sequence (20) with the same support (60%). The result of the maximal frequent sequence is only one sequence, 30. Sequence 20 is not maximal frequent, because it has a super-sequence (30) that is frequent (40%).

Besides PrefixSpan, this thesis will also use a closed and a maximal pattern algorithm to see what set of sequences will result in higher F1-scores to answer sub-question one.

3.1.2 ClaSP

Pitman & Zanker (2010) concluded that mining closed patterns is most useful. Several algorithms find closed patterns in a database. This study will use ClaSP as a closed pattern algorithm. The ClaSP algorithm is computationally fast because it uses a vertical pattern strategy. Vertical mining is the same as depth-first search mining, like the PrefixSpan algorithm. Furthermore, ClaSP outperforms state-of-the-art closed pattern algorithms (Gomariz et al., 2013).

ClaSP has two main steps to find frequent closed patterns. At first, the algorithm finds all the frequent sequences of length 1, like PrefixSpan. When all frequent sequences of length 1 are found, the algorithm uses a depth-first search strategy to find Frequent Closed Candidates (FCC) and makes a subset from those sequences. At the second step, the algorithm eliminates all the non-closed sequences from the FCC subset and results in all closed sequences (Gomariz et al., 2013).

3.1.3 VMSP

The sequences in this study are created from clickstream data in order to predict purchases with it. Maximal frequent patterns are most efficient to extract information from clickstream data (Liu et al., 2017). However, mining maximal patterns is computationally expensive. To reduce computational time, VMSP was introduced, which is used in this study. It is the first maximal algorithm that uses vertical mining. To make fair comparisons, three algorithms with the same search technique are compared. Fournier-Viger et al. (2014) conclude that VMSP is faster than the current state-of-the-art algorithms. A fast algorithm is important in this study because of the large dataset that is used.

VMSP consists of three steps. The first step is called Efficient Filtering of Non-maximal patterns (EFN). This step checks if a sequence is a super-sequence or a sub-sequence and adds all super-sequences to another subset. After that, the depth-first search technique is applied with Forward-Maximal Extension (FME). All the patterns that were added to the subset in step one will grow by adding items to it one item at a time. This way, only maximal patterns remain in the subset because items will be added to every sequence and thus no smaller sequences exist. Lastly, Candidate Pruning with a Co-occurrence map (CPC) is used. It prunes the sequences from the subset that are not supported by the minimum support threshold (Abbasghorbani & Tavoli, 2015).

3.2 Imbalanced data

Graph 2 in Appendix A shows the different event types and the percentage of appearance in the used dataset. From all the events in the dataset, only 4.2% are purchase events. This is also reflected in the sequences. There are 488,374 sequences in total, and only 22,320 contain a purchase. Thus, only 4.79% of the sequences contain a purchase. From this, it can be concluded that the data is imbalanced. A class imbalance can result in very high accuracy scores by simply predicting the majority class and must therefore be addressed. This class imbalance is tackled by downsampling the majority class. This involves randomly removing observations from the majority class. This way, the majority class will not dominate the data set. The dataset contains a lot of data and with downsampling there are still enough data points to train the model on. Oversampling the minority class would mean that a lot of data points are needed to be created artificially which does not reflect reality.

3.3 Machine learning algorithms

This study uses two machine learning algorithms for binary classification. With the use of these two algorithms, it will be predicted whether a customer will purchase a product. Several studies used different machine learning algorithms to predict online consumer behavior. Among most studies, XGBoost resulted as most suitable to predict online consumer behavior (Lee et al., 2021; Droomer & Dekker, 2020). KNN is proven to be most effective with the usage of clickstream data, which is the input for the machine learning algorithms in this study (Adeniyi et al., 2016). Therefore, this study will use XGBoost and KNN.

3.3.1 Extreme Gradient Boosting

Boosting is an ensemble learning method. Ensemble concerns the use of several models together. To improve the performance of the single model, a boosting algorithm has several iterations. At every iteration, the incorrectly predicted features from the previous iteration are getting a higher weight than the correctly predicted features. Graph 3 in Appendix A shows a visual explanation of a boosting algorithm. The first line shows the data points for the boosting algorithm and the decision tree with which the predictions were made. The second line shows the same data points, but some are larger displayed. Those data points were incorrectly predicted in the previous iteration and thus got a higher weight. The boosting algorithm keeps iterating through the data until it made a number of trees, that were indicated in advance, or until it has a perfect fit (González et al., 2020).

Boosting algorithms have some flaws which are improved by gradient boosting algorithms. Unlike boosting, gradient boosting scales the tree after every iteration with the use of a learning rate. Besides, gradient boosting makes iterations until it makes the number of trees indicated in advance, or until it cannot reduce the size of the residuals (Natekin & Knoll, 2013). XGBoost is an extreme gradient boosting algorithm that can be used for regression and classification. The main idea of XGBoost is that it optimizes the objective function. Given k amount of trees, the optimization of the objective function is mathematically given as follows (Li et al., 2019):

$$\hat{y}_i = \sum_{k=1}^{k} f_t(x_i), f_t \in F$$

f_t is every independent tree and F is the entire space of the tree. x_i represents the features in the dataset. Every tree, f_t, in XGBoost tries to minimize the loss function. The loss function of XGBoost can be derived from the following equation (Chen & Guestrin, 2016):

$$L(f_t) = \sum l(\hat{y}i, y_i) + \sum \Omega\ (f_t)$$

Where $\hat{y}i$ is the predicted outcome and yi is the true outcome. Ω is the regularization term for every tree f_t which ensures that the model is not overfitting (Ma et al., 2020). The extreme part of XGBoost is that it has a lot of different hyperparameters that can be tuned. The hyperparameters used in this study are listed with a short explanation in Table 5 in Appendix A. The parameters that will be used, are the parameters that prevent the algorithm from overfitting. This is important because of the high level of variety in the data.

3.3.2 K-nearest neighbors

KNN is a method that effectively classifies the data (Guo et al., 2003). The algorithm assumes that similar data points are near each other in the dataset. The KNN algorithm calculates the distance between two data points to find their similarity. This distance can be calculated with several different distance metrics, which is a hyperparameter. As a result of hyperparameter tuning, this study uses the Minkowski metric to calculate the distance. The hyperparameter tuning is further discussed in Chapter 4. It is a generalized metric and is calculated as follows (Singh et al., 2013):

$$Dist_{XY} = \left(\sum_{K=1}^{d} |X_{ik} - X_{jk}|^{\frac{1}{p}}\right)^{p}$$

Where x_{ik} is the data point in the training set and x_{jk} is the data point in the test set. This metric is a generalized metric because of p. When $p = 2$, the distance becomes the euclidean distance. When $p = 1$ the distance becomes the so-called city block distance and with $p = \infty$ the distance is called Chebyshev (Singh et al., 2013). The number of K is to define how many nearest neighbors the algorithm takes to define to which cluster the test set data point belongs. When $K = 2$, the algorithm takes the two nearest neighbors to classify the test set data point. When K gets smaller, the model is more likely to overfit. Unlike XGBoost, KNN is a lazy learning algorithm because it does not learn from the data, but memorizes the training data (Guo et al., 2003).

3.4 Baseline

The dummy classifier was provided as a baseline for measuring the performance of the models. The dummy classifier makes predictions without trying to find patterns in the data, and always predicts the most frequent label present in the training data (Barletta et al., 2017). The machine learning models in this study are trying to find patterns in the data. To be useful for prediction, the evaluation scores need to be higher than the baseline model.

3.5 Evaluation

By applying the machine learning model to the test set, it can be measured how effective the model is. The evaluation score reflects this effectiveness (Junker et al., 1999). As an evaluation metric, F1-score is used as this is one of the most used evaluation metrics when the data is imbalanced. F1-score is the harmonic mean between precision and recall and is computed as follows (Lipton et al., 2014):

$$F1\ score = \frac{2\ (Precision \cdot Recall)}{Precision + Recall}$$

A further explanation of precision and recall is given in Appendix B. The F1-score was calculated for all three XGBoost models and all three KNN models. The evaluation scores are presented in Chapter 5.

To evaluate the models accurately and prevent overfitting of the models, the dataset is split into training and test set. The most common split for large datasets is an 80/20 split. However, a 70/30 split was chosen because of the highly imbalanced data. 70% of the data is used to train the models on and 30% is used to test the models. To preserve a good reflection of reality, only the training data is balanced and the test data is not. Because the test set is still highly imbalanced, it has a higher percentage for the split so it has more data to test on.

4. Experimental setup

This chapter describes all the procedures that were taken to get the results. First, the dataset is described. After that, pre-processing is discussed. Then, the feature selection method is described. Lastly, the machine learning algorithms and their evaluation methods are explained.

4.1 Dataset description

The dataset that is used in this study is retrieved from Kaggle (Kaggle.com) and is originally from REES 46 technologies. REES 46 provides online stores knowledge and helps businesses with their development (REES46, n.d.). On Kaggle it is mentioned that the dataset is publicly available, and thus no license is required. Therefore, this dataset is appropriate to use in this study. The dataset contains data obtained from a large home appliances and electronics online store with purchases from October 2019 to February 2020. The dataset contains nine features and 885,129 observations with each observation representing an event from a user. An event can be *view*, *add to cart,* or *purchased*. The number of unique users is 407,283, and the number of unique user sessions is 490,398. The features contain the event time, event type, product ID, category ID, category code, brand, price, user ID, and the user session. A description of the features is represented in Table 6 in Appendix C.

4.2 Pre-processing

First of all, the data was pre-processed. In the dataset for this study, every row contains one event from a particular user session. The sequences that are extracted from the dataset will contain all the events from one user session in order of time. To accomplish this, the data was grouped by *user_session* and *event_type*, and sorted by *event_time.* Contrary to the literature in Chapter 2, a lot of features like brand, product ID, and price are left out by making the sequences. This study investigates if an e-commerce purchase can be predicted, regardless of the type of product, the brand, or the price of the product. The input file for the SPM algorithms requires a particular format which is presented in Table 7 in Appendix C. With the label encoder from the sklearn package, the event type *view* was encoded to integer *2*, *add_to_cart* to *0,* and *purchase* to *1.* With the join and apply functions from the pandas package, *-1* was placed between every integer, and *-2* was applied at the end of every sequence. Lastly, having *purchase* in the sequence will lead to perfect prediction so every *purchase* was deleted from the sequences. Eventually, a data frame was created with 488,374 sequences, which was used as input for the SPM algorithms (see Table 8, Appendix C).

The input data for the machine learning algorithms also require preprocessing. The input variables for the machine learning algorithms consist of the created sequences and a label. For the label, a list was created with *1* for every sequence containing purchase and *0* for every sequence without purchase. The list was reframed to a column named *target*.

4.3 Feature selection

Frequent sequences are extracted from the dataset using three SPM algorithms, PrefixSpan, ClaSP, and VMSP. The SPM algorithms are implemented with the SPMF package in Python which is an open-source data mining library written in Java (Fournier-Viger, 2008).

At first, the PrefixSpan algorithm is applied. This algorithm requires three arguments, namely the minimum support, the maximal pattern length, and whether the sequence ID will be shown. The minimum support is a number between 0 and 1 representing a percentage. This percentage is set to 2% which means that a sequence needs to be in the data 9,768 times, to be marked as frequent. When this percentage was increased, very few or no sequences were marked as frequent. Thus, there are a lot of different sequences in the dataset which means the dataset has a lot of variety. The second argument, maximal pattern length, was set to a very high number, *200,* to make sure all the sequences were included. The last argument was set to *True* to see the sequence ID from all the frequent sequences. The output of the algorithm results in six frequent sequences.

The second SPM algorithm that was used is ClaSP. This algorithm requires two arguments, minimum support, and show of the sequence ID. Those arguments are set to the same values as for the PrefixSpan algorithm. This resulted in fourteen closed frequent sequences.

VMSP is the third SPM algorithm that is used in this study. This algorithm requires four arguments, namely minimum support, minimum pattern length, maximum gap, and show sequence ID. The first two and the last arguments are set to the same value as for the PrefixSpan and ClaSP algorithms. The maximum gap argument specifies if gaps are allowed in the sequences. In this study, the value *1* is used. This means that no gaps are allowed because the exact purchase behavior from an e-commerce customer needs to be measured. This algorithm results in three maximal frequent sequences. Table 9 in Appendix C gives an overview of the values for the arguments of the SPM algorithms. The results of frequent sequences from all three algorithms are listed in Chapter 5.

The features that were used in the machine learning algorithms to predict, are the top five frequent sequences that result from the SPM algorithms. The choice to only include the top five frequent sequences is based on the number of frequent sequences. VMSP results in three frequent sequences, PrefixSpan in six and ClaSP in fourteen. When the top fourteen frequent sequences are used to include all sequences, there would be a major difference in number of features for VMSP and PrefixSpan algorithms. When only three frequent sequences were used, there were a lot of sequences not included from ClaSP and PrefixSpan. Because the sixth frequent sequence of PrefixSpan and ClaSP are already included in one of the other SPM algorithms, the choice was made to use only the top five frequent sequences. In order to create the correct input for the machine learning algorithms, new data frames were created. Every feature consists of a column with zeros and ones. If the frequent sequence feature is present in the user session, the result is 1, otherwise 0. Lastly, the *target* column was merged with every data frame with an inner merge based on user session to match the right target to the right sequence.

4.4 Imbalanced data

The sequence input for the machine learning algorithms needs to be downsampled. Downsampling is done with the RandomUnderSampler from imblearn. To downsample the majority class, the strategy was set to *majority*. The data is split into 70/30 training and test data. Only the training data is downsampled to preserve a good reflection of reality in the test set. The training data for every algorithm consist of 326,172 sequences without purchase and 15,689 sequences with a purchase before balancing the data, which is represented in Graph 4 in Appendix C. After balancing, the training data consist of 15,689 sequences with a purchase and 15,689 sequences without purchase which is represented in Graph 5 in Appendix C.

4.5 Implementation of machine learning algorithms

Two machine learning algorithms were used to predict a customer's next e-commerce purchase. These are XGBoost and KNN.

4.5.1 Extreme Gradient Boosting

The first algorithm that was applied is XGBoost and was created with the xgboost package in Python. XGBoost was created three times with different features from the SPM algorithms. The label for XGBoost is the *target* column and the features are the frequent sequences. The data was split into 30% test data and 70% training data to estimate the performance using train_test_split from sklearn.model_selection. After that, the training data was balanced. XGBoost has a lot of different hyperparameters. For the hyperparameter tuning of XGBoost, the hyperopt module is used. Hyperopt assumes that if one value gives a bad result, all surrounding values also give bad results. Therefore, hyperopt gets more points from the regions with high results (Komer et al., 2019). Because of the many hyperparameters from XGBoost, the fast hyperopt optimization function was used. Table 10 in Appendix C shows the range per hyperparameter for the hyperparameter optimization. XGBoost also requires an objective parameter. The multiclass objective for 2 classes is used with the softmax function. Also, the input for XGBoost needs to be in DMatrix format. After the hyperparameter tuning, the train and test set were transformed into a DMatrix and the number of epochs was set to ten. The model was trained with the best hyperparameters obtained from hyperopt. The trained model predicted on the test data and evaluation scores and confusion matrixes were obtained, which are presented in Chapter 5.

4.5.2 K-nearest neighbors

The second machine learning algorithm is KNN and was created with the KNeighborsClassifier from the sklearn.neighbors package. The input for the KNN algorithm is the same as the input for the XGBoost algorithm and the data is split again into 30% test set and 70% training set. After the split, the training data was balanced. For KNN, hyperparameter tuning was done using grid search with the GridSearchCV method from the sklearn.model_selection package. Grid search is a computationally expensive method, but since KNN only has two important hyperparameters to tune, this method can be used. The explanation of these hyperparameters can be found in Chapter 3. The parameters that were tuned were the distance metric and the number of neighbors (K). The results are further discussed in Chapter 5. After the hyperparameters were tuned, the algorithm was fit on the training data. Lastly, the algorithm predicted on the test data and the evaluation scores and confusion matrixes were obtained, which are presented in Chapter 5.

4.6 Dummy classifier

The Dummy classifier is used as a baseline model in this study. The data for the baseline was also split into 70% training data and 30% test data. For the evaluation of the machine learning models, the training data is balanced. To get a fair comparison, the training data is also balanced for the baseline. The dummy classifier was created with the DummyClassifier method from sklearn.dummy. The strategy was set to *most_frequent*, because of the imbalanced dataset.

After that, the dummy classifier was fit on training data and predicted on the test data. Lastly, the evaluation metrics and confusion matrixes were obtained. The results of the dummy classifier are presented in Chapter 5.

4.7 Evaluation

This study uses the F1-score as the evaluation metric. For every machine learning algorithm, the F1-score was calculated with f1_score method from sklearn.metrics. To get a more detailed explanation of the F1-score, confusion matrixes were obtained for every model with sklearn.metrics.

5. Results

This chapter provides all the results derived from the previous chapters and is separated into three parts. At first, the results of the baseline are presented. The second part contains the results from the SPM algorithms. Lastly, the results from the machine learning algorithms are presented.

5.1 Baseline

The results of the baseline are in line with the expectations. The confusion matrix and evaluation scores for the baseline are presented in Graph 6. In all the confusion matrixes in this chapter, 0 represents *no-purchase* and 1 represents *purchase*. According to Barletta et al. (2017), the dummy classifier will find the majority class and it classifies all data to the value of that class. Since the data is highly imbalanced, almost all data points belong to the majority class. This results in a high accuracy score of 0.955. Because the baseline predicts no data point as *purchase*, there are no true and false positives. This results in precision and recall of 0.0, and thus also F1-score of 0.0 since this is the harmonic mean between precision and recall.

Graph 6 *Confusion matrix and evaluation scores baseline*

	Predicted 0	**Predicted 1**
Actual 0	139,882	0
Actual 1	6,631	0

Accuracy: 0.955
Precision: 0.0
Recall: 0.0
F1-score: 0.0

5.2 Sequential pattern mining

To answer the first sub-question, three SPM algorithms were conducted as described in Chapter 4. From those three algorithms, several frequent sequences were found. The results of all the frequent sequences are presented in Table 11, 12 and 13 in Appendix D. The top five frequent sequences and their support are given in Table 14.

Table 14 *Top-5 frequent sequences*

	PrefixSpan	*Support*	**ClaSP**	*Support*	**VMSP**	*Support*
1	View	*488,360*	View	*488,360*	View-view-add to cart	*15,629*
2	View-view	*125,161*	View-view	*125,161*	View-view-view-view-view-view	*10,324*
3	View-view-view	*54,301*	View-view-view	*54,301*	View-add to cart-view-view	*9,881*
4	View-view-view-view	*30,136*	Add to cart	*41,270*	-	-
5	View-view-view-view-view	*19,151*	View-add to cart	*41,232*	-	-

The event *purchase* is deleted from these sequences to prevent perfect predictable sequences. The columns *support* present the number of occurrences of that sequence in the dataset.

PrefixSpan makes use of prefix and suffix, and eliminates sequences that do not contain the prefix. The event *add to cart* is not present in PrefixSpan, which means that when *add to cart* is a prefix, there are not many frequent sequences. This results in sequences that only contain *view* events. This is in line with the expectations because most people will first view a product before buying it. Also, the elimination of sequences without the prefix, results in less frequent sequences than ClaSP (see Table 11 and 12, Appendix D). This is not in line with the redundancy problem explained in paragraph 3.2.1, which states that closed pattern techniques result in fewer sequences than pattern-growth techniques.

On the contrary to PrefixSpan, the ClaSP algorithm does contain *add to cart*. For the ClaSP algorithm, the sequence *add to cart* need to be present somewhere in the sequence to be counted as frequent. The support for this sequence is higher than the support for the fourth PrefixSpan sequence. Thus, the event *add to cart* is more present than sequences that start with four views. Besides, closed patterns have higher support than maximal patterns. This is in line with the redundancy problem. Maximal patterns discarded all the sequences that have a super-sequence with minimum support.

The VMSP algorithm represents the redundancy problem very well. All the sequences from PrefixSpan and the first three from ClaSP are summarized in the second frequent sequence from the VMSP algorithm. As a result, VMSP only has three frequent sequences. This is also in line with the literature. Liu et al. (2017) showed in their study that the use of maximal patterns will decrease the number of frequent sequences. Besides, the support of the patterns from

VMSP is small in contrast to the support from PrefixSpan and ClaSP. This is because no gaps are allowed in the sequences as explained in Chapter 4. That means that de sequences need to be present in the same format as presented in Table 14 to be counted as frequent.

5.3 Machine learning algorithms

To answer sub-question two, two machine learning algorithms were each three times conducted, as described in Chapter 4. At first, the hyperparameters were chosen based on the outcome from hyperparameter tuning. Table 15 in Appendix D shows the best hyperparameters for every XGBoost algorithm. KNN has two hyperparameters. For all three algorithms, the Minkowski distance resulted as the best metric. Graph 7, 8, and 9 in Appendix D show the difference between the training set score and test set score for every K. The outcome from grid search resulted in K = 3 for every KNN algorithm.

Based on F1-scores, four models outperformed the baseline. Table 16 shows the F1-scores per model.

Table 16 *F1-scores*

	PrefixSpan	**ClaSP**	**VMSP**
XGBoost	0.393	0.516	0.333
KNN	0.0	0.009	0.0

In general, the algorithms with features obtained from ClaSP performed best. Algorithms with features obtained from VMSP performed worst. Furthermore, XGBoost performed overall best. This outcome was expected as XGBoost was also best performing in the literature (Lee et al., 2021; Droomer & Dekker, 2020).

5.3.1 Confusion matrixes

To answer sub-question three, confusion matrixes have been created. At first, the confusion matrixes for XGBoost are presented in Graph 10, 11, and 12.

Graph 10 *Confusion matrix and evaluation scores XGBoost with PrefixSpan*

	Predicted 0	**Predicted 1**
Actual 0	130,240	9,642
Actual 1	2,656	3,975

Accuracy: 0.916
Precision: 0.292
Recall: 0.599
F1-score: 0.393

Graph 11 *Confusion matrix and evaluation scores XGBoost with ClaSP*

	Predicted 0	**Predicted 1**
Actual 0	136,374	3,508
Actual 1	3,106	3,525

Accuracy: 0.955
Precision: 0.501
Recall: 0.532
F1-score: 0.516

Graph 12 *Confusion matrix and evaluation scores XGBoost with VMSP*

	Predicted 0	**Predicted 1**
Actual 0	138,370	1,512
Actual 1	5,003	1,628

Accuracy: 0.956
Precision: 0.518
Recall: 0.246
F1-score: 0.333

The confusion matrixes are obtained from the test set, which contains 139,882 no-purchases and 6,631 purchases. The accuracy scores of every model are very high. This means that the models predicted no-purchase very often since the dataset is highly imbalanced. This is like the baseline. To draw better conclusions, confusion matrixes have been compared.

For XGBoost with PrefixSpan, the false positives are very high which results in low precision. On the other hand, the false negatives are low which results in a higher recall. Since F1-score is the harmonic mean of precision and recall, and precision is very low, F1-score is also low. For XGBoost with VMSP the opposite counts. Recall is very low because of the high false negatives compared to true positives . Precision is higher, because of the false positives compared with true positives are lower. This also results in a low F1-score. For XGBoost with ClaSP, the model predicted just as good on precision as on recall. Out of all the purchases the model picked up to predict, half of them were actually purchases, and half of the purchases were correctly predicted. This results in an F1-score of 0.5.

The confusion matrixes for the KNN algorithms are represented in Graph 13, 14, and 15.

Graph 13 *Confusion matrix and evaluation scores KNN with PrefixSpan*

	Predicted 0	**Predicted 1**
Actual 0	139,882	0
Actual 1	6,631	0

Accuracy: 0.955
Precision: 0.0
Recall: 0.0
F1-score: 0.0

Graph 14 *Confusion matrix and evaluation scores KNN with ClaSP*

	Predicted 0	**Predicted 1**
Actual 0	139,856	26
Actual 1	6,599	32

Accuracy: 0.955
Precision: 0.552
Recall: 0.005
F1-score: 0.009

Graph 15 *Confusion matrix and evaluation scores KNN with VMSP*

	Predicted 0	**Predicted 1**
Actual 0	139,882	0
Actual 1	6,631	0

Accuracy: 0.955
Precision: 0.0
Recall: 0.0
F1-score: 0.0

For the KNN models, the same holds regarding the accuracy score. In general, the KNN models predicted none or almost none purchases. The models with PrefixSpan and VMSP predicted exactly the same as the baseline. For the model with ClaSP, the precision is much higher than the other two models because the proportion of false positives towards true positives is higher. However, recall is still very low because the false negatives are still very high compared to the true positives, which results in a low F1-score.

6. Discussion

The goal of this study is to combine SPM algorithms with machine learning algorithms and predict a customer’s e-commerce purchase. The combination of the SPM algorithm and machine learning algorithm that results in the highest performance will be used for prediction.

6.1 Sequential pattern mining

People often start with viewing products one or more times before adding a product to their cart, as shown in Table 14. In this study, product IDs are not included in the sequences, and it is not clear if the views are for the same product. Trivonanda et al. (2020) conclude that the use of product categories has better performances. In further research, the product ID can be included to see if the views in one sequence are for the same product or not. This way, a more targeted prediction can be made, to see which product a customer is likely to purchase.

Pitman & Zanker (2010) concluded that mining closed patterns are most useful for building recommendations. This study shows that the same holds for machine learning algorithms. To answer sub-question one, closed frequent sequences result in the best predictive performances since the ClaSP algorithm performed overall best.

6.2 Machine learning algorithms

To answer sub-question two, F1-scores were obtained per model. F1-scores of the KNN models are much lower than the F1-scores of the XGBoost models. Since KNN is a simple model with only two hyperparameters and XGBoost is a very complex model with a lot of hyperparameters, more complex models may be better in predicting e-commerce purchases with sequences as features. Koehn et al. (2020) predicted e-commerce purchases with sequences as input variables and concluded that RNN models outperformed other machine learning models. RNN is a complex model. To check the assumption, further research can be done with sequences as features for RNN models to check if those models have higher F1-scores. Besides, a very simple model like logistic regression can be used to check if simple models do perform worse.

The KNN models have very low scores because the models did not learn from the training data. In Graph 7, 8, and 9 in Appendix D the training accuracy score is 0.5 and the test accuracy is 0.9 for every possible value of K. Since the training data is balanced and only 50% is majority class, and test data is not balanced, it shows that the models only predict the majority class. This is represented in no purchase prediction at all for the models with PrefixSpan and

VMSP. The models have a high bias and are underfitting. To try to overcome this problem, more complex models can be used and further research can be done with the use of k-fold cross validation. Besides, this study did not tune the hyperparameters on a separate validation set, which could also avoid underfitting of the models.

For the third sub-question, the confusion matrixes of XGBoost are compared to see why ClaSP performed best in combination with XGBoost. The combination with PrefixSpan predicted the most purchases but has very high false positives. This possibly is because of the very basic patterns PrefixSpan results in. Many sequences contain those patterns, and thus many purchases are made with those patterns, and the model is more likely to predict purchases. This reflects in low precision and high recall. On the contrary, VMSP predicts fewer purchases than ClaSP which results in high false negatives. This possibly is because the maximal frequent patterns are more complex and have low support. Besides, VMSP has fewer features to predict purchases on. However, the purchases are more correctly predicted than PrefixSpan. This results in low recall and high precision. ClaSP is exactly in the middle of VMSP and PrefixSpan. The model has basic sequences like PrefixSpan which results in higher purchase predictions. However, the complexity of the last two sequences makes sure the model does predict the purchases more correctly.

This study shows that the higher the complexity of sequences, that serve as features, the fewer purchases predictions are made. This is in line with Pitman & Zanker (2010) who concluded that increasing complexity of sequences leads to less applicability of those sequences. However, sequences that are complex result in more correct predictions. To make sure this also holds when VMSP has more features, further research could lower the minimum support and create more features for VMSP and check if the same results come up.

7. Conclusion

To answer the research question, three sub-questions were determined. The first sub-question *Which SPM algorithm results in the best set of features?* is answered with feature extraction. In the previous chapter, it is discussed that ClaSP results in the sequences that have the highest F1-scores. From this study, it can be concluded that closed patterns are most useful when they are used as features for machine learning algorithms for predicting purchases.

The second sub-question is *Which machine learning algorithm, in combination with SPM algorithms, results in the highest F1-scores?* To answer this sub-question, F1-scores were compared from both machine learning algorithms. Two of the KNN models did not outperform the baseline. XGBoost performed overall better than KNN. It can be concluded that the more complex XGBoost algorithm performed best in combination with SPM algorithms, based on F1-scores.

The last sub-question is *Which model, from the best performing machine learning algorithm, has the highest F1-score, and why does it have the highest F1-score?* ClaSP in combination with XGBoost resulted in the highest F1-score. To see why, confusion matrixes were made and compared. From this, it can be concluded that the more complex the sequence features are, the less the model makes purchase predictions, but more predictions were correct. Besides a model with more basic sequences predicts more purchases, but predicts them wrong.

The combination XGBoost with ClaSP is used to answer the RQ *To what extent can a customer's e-commerce purchase be predicted?* It can be concluded that with 95.5% accuracy, a customer's e-commerce purchase can be predicted with sequences as features for machine learning algorithms. However, these predictions only have 50% precision. Businesses can use this knowledge for creating better marketing strategies to make the customer purchase and eventually, increase their conversion rate.

8. References

Abbasghorbani, S., & Tavoli, R. (2015). Survey on sequential pattern mining algorithms. *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*. https://doi.org/10.1109/kbei.2015.7436211

Achar, A., A, I., & Sastry, P. (2013). Pattern-growth based frequent serial episode discovery. *Data & Knowledge Engineering*, *87*, 91–108. https://doi.org/10.1016/j.datak.2013.06.005

Adeniyi, D., Wei, Z., & Yongquan, Y. (2016). Automated web usage data mining and recommendation system using K-Nearest Neighbor (KNN) classification method. *Applied Computing and Informatics*, *12*(1), 90–108. https://doi.org/10.1016/j.aci.2014.10.001

Barletta, L., Giusti, A., Rottondi, C., & Tornatore, M. (2017). QoT Estimation for Unestablished Lighpaths using Machine Learning. *Optical Fiber Communication Conference*. https://doi.org/10.1364/ofc.2017.th1j.1

Bharathi, A. V., Rao, J. M., & Tripathy, A. K. (2018). Click Stream Analysis in E-Commerce Websites-a Framework. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. https://doi.org/10.1109/iccubea.2018.8697475

Bhatnagar, A., Misra, S., & Rao, H. R. (2000). On risk, convenience, and Internet shopping behavior. *Communications of the ACM*, *43*(11), 98–105. https://doi.org/10.1145/353360.353371

Buckland, M., & Gey, F. (1994). The relationship between Recall and Precision. *Journal of the American Society for Information Science*, *45*(1), 12–19. https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/(SICI)1097-4571(199401)45:1%3C12::AID-ASI2%3E3.0.CO;2-L

Bucklin, R. E., & Sismeiro, C. (2009). Click Here for Internet Insight: Advances in Clickstream Data Analysis in Marketing. *Journal of Interactive Marketing*, *23*(1), 35–48. https://doi.org/10.1016/j.intmar.2008.10.004

Chen, T., & Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Dolfen, P., Einav, L., Klenow, P., Klopack, B., Levin, J., Levin, L., & Best, W. (2019). Assessing the Gains from E-Commerce. *National Bureau of Economic Research*. https://doi.org/10.3386/w25610

Droomer, M., & Bekker, J. (2020). Using Machine Learning to predict the next purchase date for an individual retail customer. *South African Journal of Industrial Engineering, 31*(3). https://doi.org/10.7166/31-3-2419

Fournier-Viger, P. (2008). *SPMF: A Java Open-Source Data Mining Library*. SPMF. Retrieved on 25 november 2021, from http://www.philippe-fournier-viger.com/spmf/

Fournier-Viger, P., Wu, C. W., Gomariz, A., & Tseng, V. S. (2014). VMSP: Efficient Vertical Mining of Maximal Sequential Patterns. *Advances in Artificial Intelligence*, 83–94. https://doi.org/10.1007/978-3-319-06483-3_8

Gangurde, R., Kumar, B., & Gore, S. D. (2017). Building Prediction Model using Market Basket Analysis. *Int. J. Innov. Res. Comput. Commun. Eng*, *5*(2), 1302–1309. https://doi.org/10.15680/IJIRCCE.2017.0502136

Gomariz, A., Campos, M., Marin, R., & Goethals, B. (2013). ClaSP: An Efficient Algorithm for Mining Frequent Closed Sequences. *Advances in Knowledge Discovery and Data Mining*, 50–61. https://link-springer-com.tilburguniversity.idm.oclc.org/content/pdf/10.1007%2F978-3-642-37453-1.pdf

González, S., García, S., Del Ser, J., Rokach, L., & Herrera, F. (2020). A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Information Fusion*, *64*, 205–237. https://doi.org/10.1016/j.inffus.2020.07.007

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN Model-Based Approach in Classification. *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, 986–996. https://doi.org/10.1007/978-3-540-39964-3_62

Hamed, S., & El-Deeb, S. (2020). Cash on Delivery as a Determinant of E-Commerce Growth in Emerging Markets. *Journal of Global Marketing*, *33*(4), 242–265. https://doi.org/10.1080/08911762.2020.1738002

Ingaldi, M., & Ulewicz, R. (2018). Evaluation of Quality of the e-Commerce Service. *International Journal of Ambient Computing and Intelligence*, *9*(2), 55–66. https://doi.org/10.4018/ijaci.2018040105

Jia, R., Li, R., Yu, M., & Wang, S. (2017). E-commerce purchase prediction approach by user behavior data. *2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*. https://doi.org/10.1109/cits.2017.8035294

Jian Pei, Jiawei Han, Mortazavi-Asl, B., Jianyong Wang, Pinto, H., Qiming Chen, Dayal, U., & Mei-Chun Hsu. (2004). Mining sequential patterns by pattern-growth: the PrefixSpan approach. *IEEE Transactions on Knowledge and Data Engineering, 16*(11), 1424–1440. https://doi.org/10.1109/tkde.2004.77

Junker, M., Hoch, R., & Dengel, A. (1999). On the evaluation of document analysis components by recall, precision, and accuracy. *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR '99 (Cat. No.PR00318)*. https://doi.org/10.1109/icdar.1999.791887

Kaggle: Your Machine Learning and Data Science Community. (2021). Kaggle. Retrieved on 26 November 2021, from https://www.kaggle.com/

Kim, R. Y. (2020). The Impact of COVID-19 on Consumers: Preparing for Digital Sales. *IEEE Engineering Management Review, 48*(3), 212–218. https://doi.org/10.1109/emr.2020.2990115

Koehn, D., Lessmann, S., & Schaal, M. (2020). Predicting online shopping behaviour from clickstream data using deep learning. *Expert Systems with Applications, 150*, 113342. https://doi.org/10.1016/j.eswa.2020.113342

Komer, B., Bergstra, J., & Eliasmith, C. (2019). Hyperopt-Sklearn. *Automated Machine Learning*, 97–111. https://doi.org/10.1007/978-3-030-05318-5_5

Lee, J., Jung, O., Lee, Y., Kim, O., & Park, C. (2021). A Comparison and Interpretation of Machine Learning Algorithm for the Prediction of Online Purchase Conversion. *Journal of Theoretical and Applied Electronic Commerce Research, 16*(5), 1472–1491. https://doi.org/10.3390/jtaer16050083

Li, C., Chen, Z., Liu, J., Li, D., Gao, X., Di, F., Li, L., & Ji, X. (2019). Power Load Forecasting Based on the Combined Model of LSTM and XGBoost. *Proceedings of the 2019 the International Conference on Pattern Recognition and Artificial Intelligence - PRAI '19*. https://doi.org/10.1145/3357777.3357792

Lipton, Z. C., Elkan, C., & Naryanaswamy, B. (2014). Optimal Thresholding of Classifiers to Maximize F1 Measure. *Machine Learning and Knowledge Discovery in Databases*, 225–239. https://doi.org/10.1007/978-3-662-44851-9_15

Liu, Z., Wang, Y., Dontcheva, M., Hoffman, M., Walker, S., & Wilson, A. (2017). Patterns and Sequences: Interactive Exploration of Clickstreams to Understand Common Visitor Paths. *IEEE Transactions on Visualization and Computer Graphics, 23*(1), 321–330. https://doi.org/10.1109/tvcg.2016.2598797

Ma, B., Meng, F., Yan, G., Yan, H., Chai, B., & Song, F. (2020). Diagnostic classification of cancers using extreme gradient boosting algorithm and multi-omics data. *Computers in Biology and Medicine*, *121*, 103761. https://doi.org/10.1016/j.compbiomed.2020.103761

Mabroukeh, N. R., & Ezeife, C. I. (2010). A taxonomy of sequential pattern mining algorithms. *ACM Computing Surveys*, *43*(1), 1–41. https://doi.org/10.1145/1824795.1824798

Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, *7*. https://doi.org/10.3389/fnbot.2013.00021

Nguyen, M. H., Armoogum, J., & Nguyen Thi, B. (2021). Factors Affecting the Growth of E-Shopping over the COVID-19 Era in Hanoi, Vietnam. *Sustainability*, *13*(16), 9205. https://doi.org/10.3390/su13169205

Pantelimon, F. V., Georgescu, T. M., & Posedaru, B. S. (2020). The Impact of Mobile e-Commerce on GDP: A Comparative Analysis between Romania and Germany and how Covid-19 Influences the e-Commerce Activity Worldwide. *Informatica Economica*, *24*(2/2020), 27–41. https://doi.org/10.24818/issn14531305/24.2.2020.03

Pitman, A., & Zanker, M. (2010). Insights from Applying Sequential Pattern Mining to E-commerce Click Stream Data. *2010 IEEE International Conference on Data Mining Workshops*. https://doi.org/10.1109/icdmw.2010.31

Putranto, S., Putranto, I., Saputra, J., & Rialmi, Z. (2021). A Qualitative Study of E-Commerce Growth During Corona Virus Disease (COVID-19) Pandemic in Indonesia. *IEOM Society International*, 3208–3216. https://www.researchgate.net/profile/Jumadil-Saputra/publication/353306374_A_Qualitative_Study_of_E-Commerce_Growth_During_Corona_Virus_Disease_COVID-19_Pandemic_in_Indonesia/links/60f282549541032c6d464c2c/A-Qualitative-Study-of-E-Commerce-Growth-During-Corona-Virus-Disease-COVID-19-Pandemic-in-Indonesia.pdf

Ratchford, B. T., Talukdar, D., & Lee, M. S. (2001). A Model of Consumer Choice of the Internet as an Information Source. *International Journal of Electronic Commerce*, *5*(3), 7–21. https://doi.org/10.1080/10864415.2001.11044217

REES46. (n.d.). REES46. Retrieved on 26 November 2021, from https://rees46.com/

Sheth, J. (2020). Impact of Covid-19 on consumer behavior: Will the old habits return or die? *Journal of Business Research*, *117*, 280–283. https://doi.org/10.1016/j.jbusres.2020.05.059

Singh, A., Yadav, A., & Rana, A. (2013). K-means with Three different Distance Metrics. *International Journal of Computer Applications*, *67*(10), 13–17. https://doi.org/10.5120/11430-6785

Soonsawad, P. (2013). Developing a New Model for Conversion Rate Optimization: A Case Study. *International Journal of Business and Management*, *8*(10). https://doi.org/10.5539/ijbm.v8n10p41

Trivonanda, R., Mahendra, R., Budi, I., & Hidayat, R. A. (2020). Sequential Pattern Mining for e-Commerce Recommender System. *2020 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. https://doi.org/10.1109/icacsis51025.2020.9263192

Van den Poel, D., & Buckinx, W. (2005). Predicting online-purchasing behaviour. *European Journal of Operational Research*, *166*(2), 557–575. https://doi.org/10.1016/j.ejor.2004.04.022

Vo, B., Pham, S., Le, T., & Deng, Z. H. (2017). A novel approach for mining maximal frequent patterns. *Expert Systems with Applications*, *73*, 178–186. https://doi.org/10.1016/j.eswa.2016.12.023

XGBoost Parameters — xgboost 1.6.0-dev documentation. (2021). Dmlc XGBoost. Retrieved on 26 November 2021, from https://xgboost.readthedocs.io/en/latest/parameter.html#general-parameters

Yang, Y. C., Liu, H., & Cai, Y. (2013). Discovery of Online Shopping Patterns Across Websites. *INFORMS Journal on Computing*, *25*(1), 161–176. https://doi.org/10.1287/ijoc.1110.0484

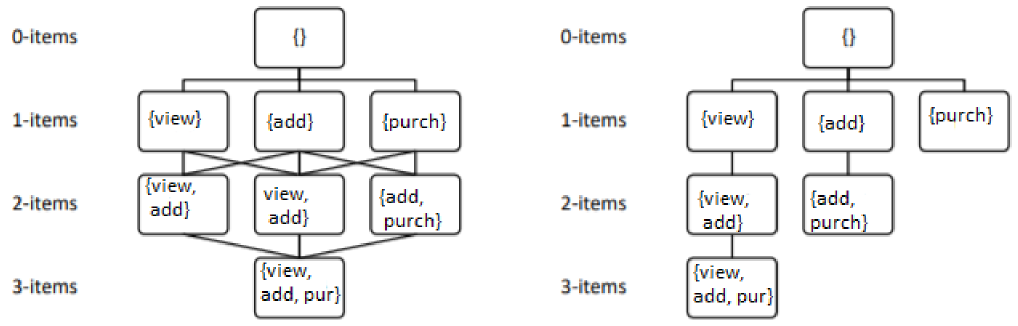
Yeo, J., Kim, S., Koh, E., Hwang, S. W., & Lipka, N. (2017). Predicting Online Purchase Conversion for Retargeting. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. https://doi.org/10.1145/3018661.3018715

Yu, Y., Wang, X., Zhong, R. Y., & Huang, G. (2017). E-commerce logistics in supply chain management. *Industrial Management & Data Systems*, *117*(10), 2263–2286. https://doi.org/10.1108/imds-09-2016-0398

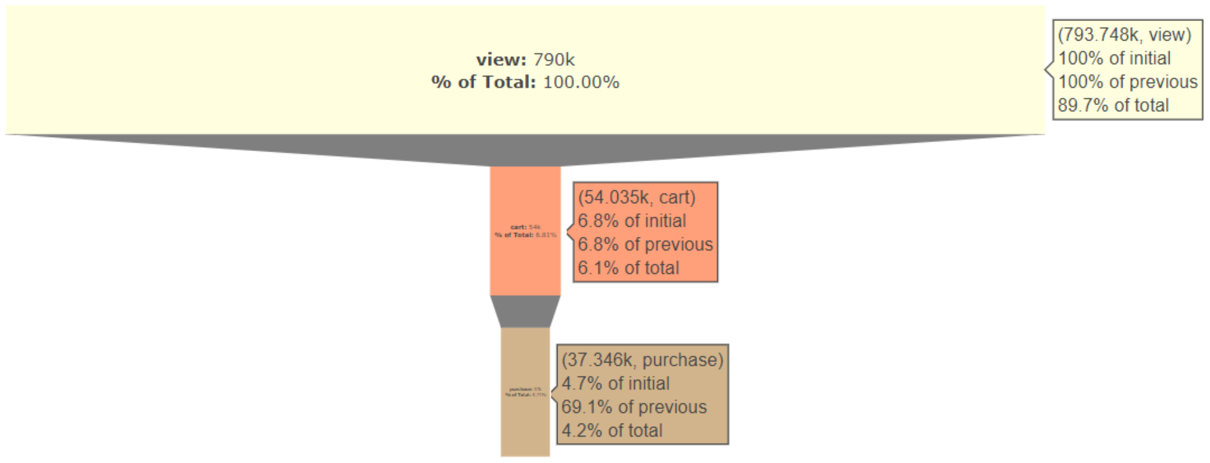
Appendix

Appendix A. Methods

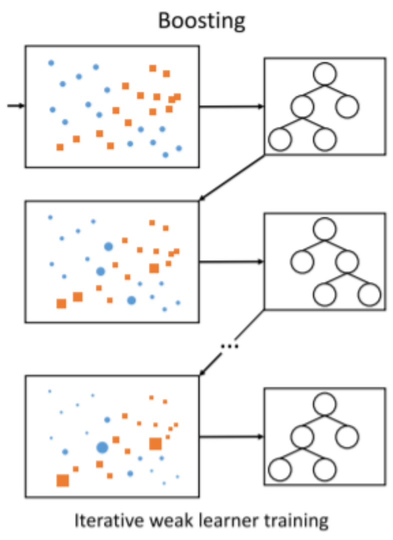
Graph 1 *Difference between breadth-first search (left) and depth-first search (right)*



Graph 2 *Percentages of events in the dataset*



Graph 3 *Explanation boosting*



Note. Source: González et al. (2020)

Table 5 *Used hyperparameters for XGBoost*

Hyperparameter	**Description**	**Default**
Booster	Which booster to use, tree-based or linear	Gbtree
Eta	Learning rate	0.3
Gamma	Used to prune the tree. It is the minimum value required to make a further split in the nodes of the tree.	0
Max_depth	Maximum depth of the tree. A higher value results in overfitting.	6
Min_child_weight	The minimum sum of weights that need to be present in a child. This is used to control overfitting.	1
Colsample_bytree	The ratio of subsampling columns with the construction of a tree.	1
Reg_lambda	L2 regularization parameter. The model is more conservative with a higher lambda value because it prunes more with a higher value.	1
Reg_alpha	L1 regularization parameter. Increasing this value will make the model more conservative	0

Note. Source: XGBoost Parameters – XGBoost 1.6.0-Dev documentation (2021)

Appendix B. Explanation precision/recall

Precision is the proportion of correctly predicted samples. So, out of all the samples the classifier predicts as purchase, what proportion of it was correctly predicted. Recall is, out of all the purchases, what proportion did the model pick up. There is a trade-off between precision and recall. With precision, there is a possibility that the model miss out on positive predictors. So the model could miss a purchase prediction. In that case, recall is low. With recall, to get as much as positive predictors, the model could also pick up negative predictors. So in that case recall is high, but precision is low. This is generally known as the precision-recall trade-off (Buckland & Gey, 1994). To combine the advantages of precision and recall, the harmonic mean can be calculated. This is known as the F1-score.

Appendix C. Experimental setup

Table 6 *features in the dataset*

Feature	**Data type**	**Description**
Event_time	Object	Time when event happened (in UTC).
Event_type	Object	Only one kind of event: purchase, view, added to cart, or remove from cart.
Product_id	Int64	ID of a product.
Category_id	Int64	Product's category ID.
Category_code	Object	Product's category taxonomy (code name).
Brand	Object	String of brand name.
Price	Float64	Price of a product.
User_id	Int64	Permanent user ID
User_session	Object	Temporary user's session ID. Changed every time a customer comes back from a long pause.

Table 7 *Input format sequential pattern mining*

```
1 -1 1 2 3 -1 1 3 -1 4 -1 3 6 -1 -2
1 4 -1 3 -1 2 3 -1 1 5 -1 -2
5 6 -1 1 2 -1 4 6 -1 3 -1 2 -1 -2
5 -1 7 -1 1 6 -1 3 -1 2 3 -1 -2
```

Note. Source: Fournier-Viger (2008)

Table 8 *Dataframe extracted sequences*

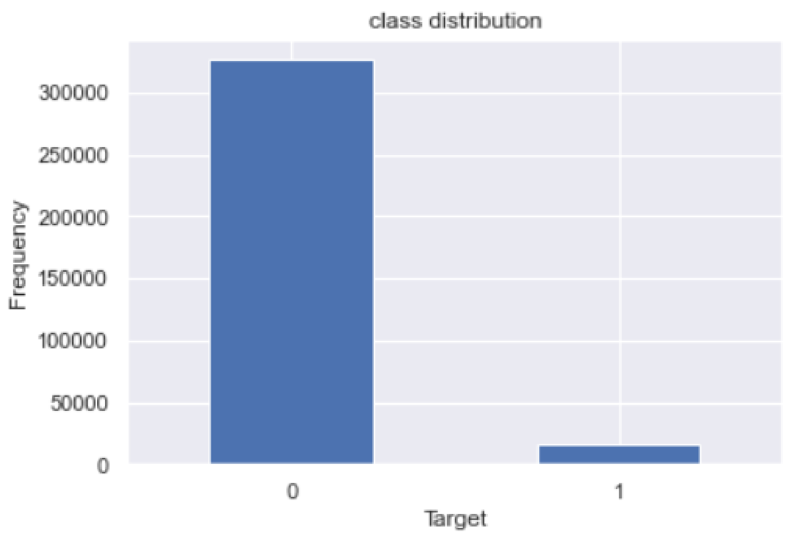
User_session	**Sequences**
000MhYaQu	2 -1 -2
001HttdHUk	2 -1 -2
001O7IK0Pt	2 -1 2 -1 -2
001RxUtFJa	2 -1 -2
…	…
zzxngTdVaG	2 -1 -2
Zzu0qXtxYX	2 -1 -2
Zzy6W7KyIP	2 -1 -2
zzzKOuAubK	2 -1 2 -1 -2
zzzUMiLcf7	2 -1 -2

448372 rows x 1 columns

Table 9 *Values for the arguments of the SPM algorithms*

	PrefixSpan	**ClaSP**	**VMSP**
Min support	0.02	0.02	0.02
Min pattern length	200	-	200
Max gap	-	-	1
Show sequence ID	True	True	True

Graph 4 *Class distribution before balancing*



Graph 5 *Class distribution after balancing*

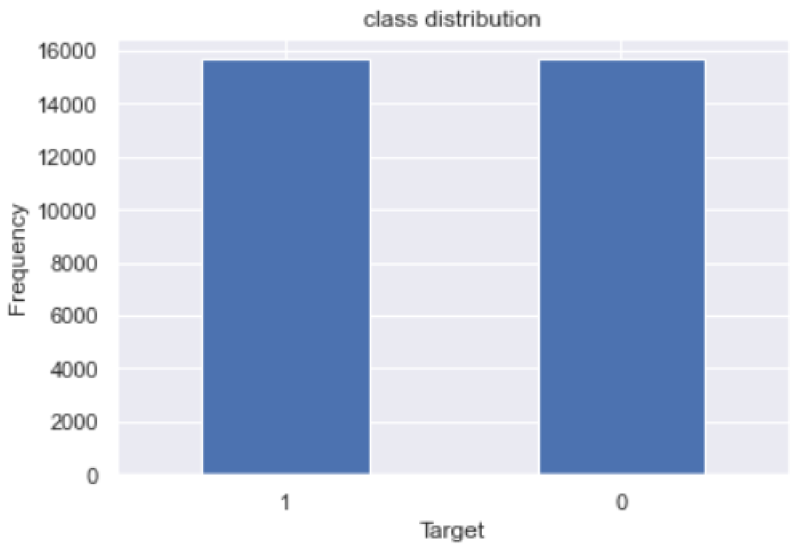


Table 10 *Range for hyperparameter tuning XGBoost*

Hyperparameter	**Range (start, end, stepsize)**
Max_depth	3, 18, 1
Gamma	1, 9
Reg_alpha	10, 180, 1
Reg_lambda	0, 1
Colsample_bytree	0.5, 1
Min_child_weight	0, 10, 1
N_estimators	180
Seed	0

Appendix D. Results

The numbers in the results are the encoded event types. Number *0* is *add to cart*, and number *2* is *view*.

Table 11 *Results PrefixSpan algorithm*

	Pattern	SUP
0	[2]	488,360
1	[2, 2]	125,161
2	[2, 2, 2]	54,301
3	[2, 2, 2, 2]	30,136
4	[2, 2, 2, 2, 2]	19,151
5	[2, 2, 2, 2, 2, 2]	13,149

Table 12 *Results ClaSP algorithm*

	Pattern	SUP
0	[2]	488,360
1	[2, 2, 2, 2, 2, 2]	13,149
2	[2, 2, 2]	54,301
3	[0, 2]	21,160
4	[2, 0, 2]	21,135
5	[2, 0]	41,232
6	[2, 2, 2, 2]	30,136
7	[2, 2, 2, 2, 2]	19,151
8	[0, 2, 2]	10,909
9	[2, 2]	125,161
10	[2, 2, 0, 2]	10,363
11	[2, 0, 2, 2]	10,899
12	[0]	41,270
13	[2, 2, 0]	17,537

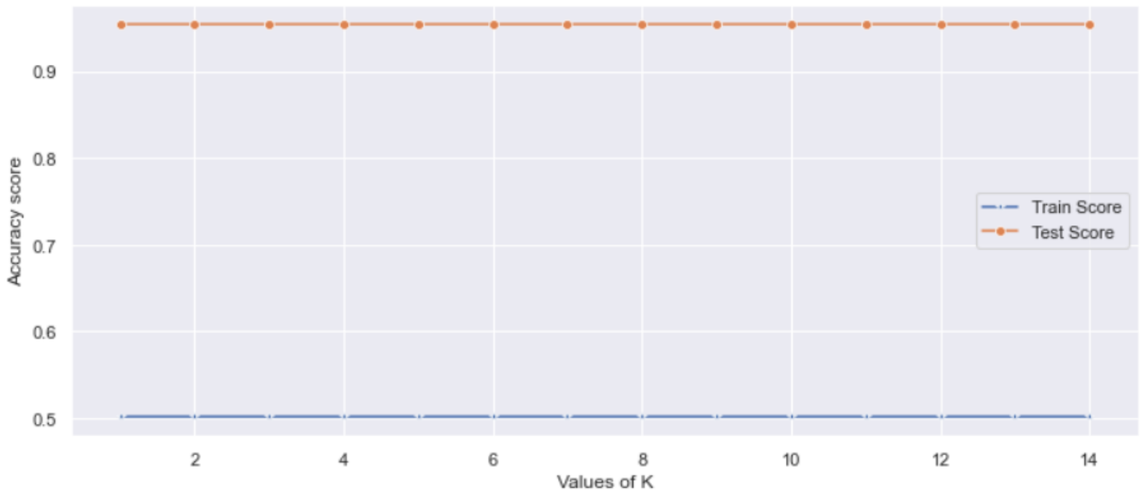
Table 13 *Results VMSP algorithm*

	Pattern	SUP
0	[2, 2, 0]	15,629
1	[2, 0, 2, 2]	9,881
2	[2, 2, 2, 2, 2, 2]	10,324

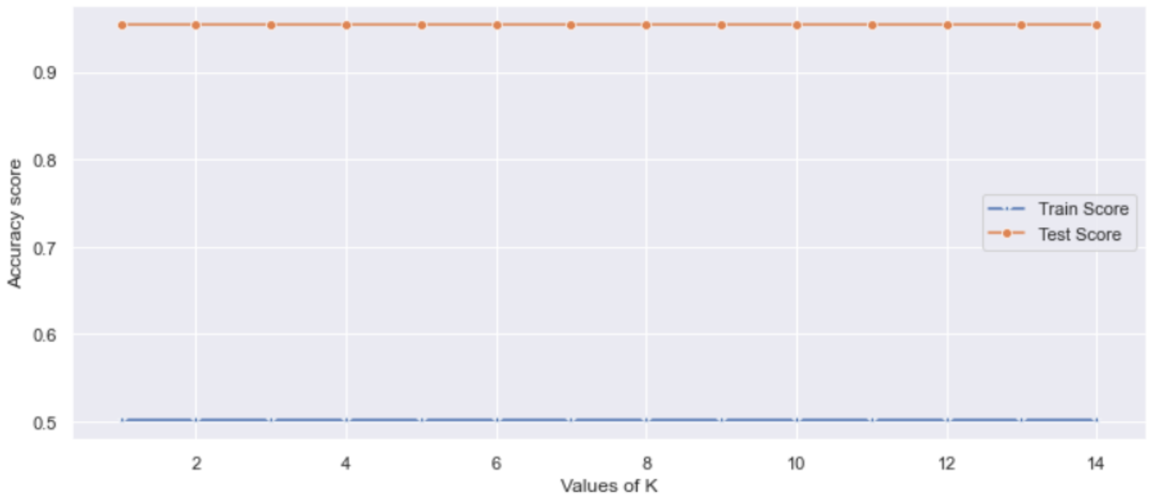
Table 15 Hyperparameters XGBoost

	PrefixSpan	**ClaSP**	**VMSP**
Colsample_bytree	0. 6274	0.7865	0.5977
Gamma	4.8539	1.0903	8.7229
Max_depth	14	12	15
Min_child_weight	8	3	6
Reg_alpha	160	143	150
Reg_lambda	0.3849	0.6293	0.0769

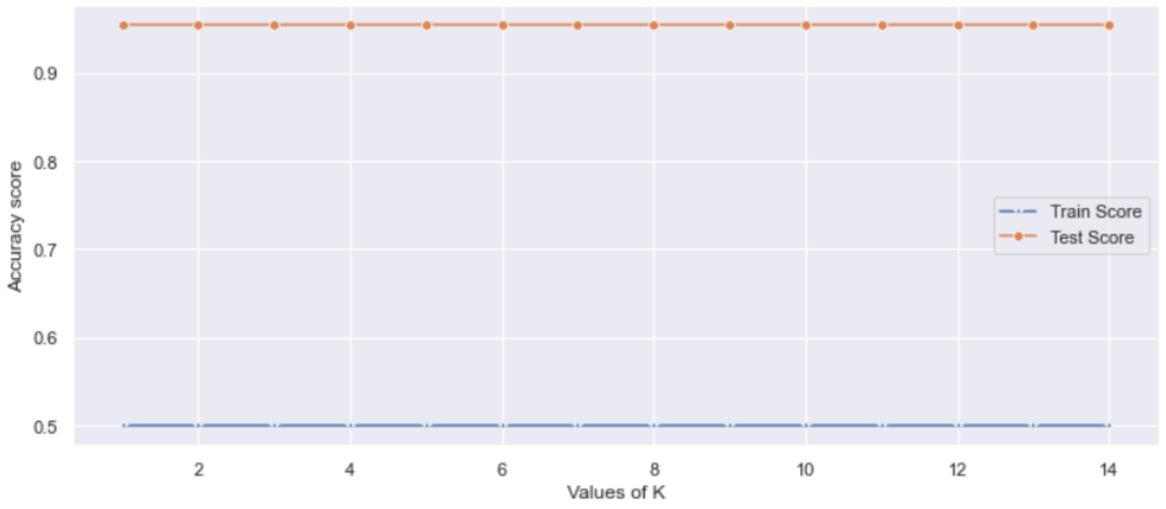
Graph 7 *Train and test score for PrefixSpan for different values of K*



Graph 8 *Train and test score for ClaSP for different values of K*



Graph 9 *Train and test score for VMSP for different values of K*



Appendix E. Code

The code for this thesis can be found on github: https://github.com/robinvanheesch1994/thesis