UPPSALA
UNIVERSITET

UPTEC STS 19013
Examensarbete 30 hp
Juni 2019

Exploring Unsupervised Learning as a Way of Revealing User Patterns in a Mobile Bank Application

Elsa Bergman
Anna Eriksson

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Exploring Unsupervised Learning as a Way of Revealing User Patterns in a Mobile Bank Application

Elsa Bergman, Anna Eriksson

The purpose of this interdisciplinary study was to explore whether it is possible to conduct a data-driven study using pattern recognition in order to gain understanding of user behavior within a mobile bank application. This knowledge was in turn used to propose ways of tailoring the application to better suit the actual needs of the users.

In this thesis, unsupervised learning in the form of clustering was applied to a data set containing information about user interactions with a mobile bank application. By pre-processing the data, finding the best value for the number of clusters to use and applying these results to the K-means algorithm, clustering into distinct subgroups was possible. Visualization of the clusters was possible due to combining K-means with a Principal Component Analysis. Through clustering, patterns regarding how the different functionalities are used in the application were revealed. Thereafter, using relevant concepts within the field of human-computer-interaction, a proposal was made of how the application could be altered to better suit the discovered needs of the users. The results show that most sessions are passive, that the device model is of high importance in the clusters, that some features are seldom used and that hidden functionalities are not used in full measure. This is either due to the user not wanting to use some functionalities or because there is a lack of discoverability or understanding among the users, causing them to refrain from using these functionalities. However, determining the actual cause requires further qualitative studies. Removing features which are seldom used, adding signifiers, active discovery as well as conducting user-tests are identified as possible actions in order to minimize issues with discoverability and understanding. Finally, future work and possible improvements on the research methods used in this study were proposed.

Handledare: Jonas Sköld
Ämnesgranskare: Andreas Lindholm
Examinator: Elísabet Andrésdóttir
ISSN: 1650-8319, UPTEC STS 19013

Sammanfattning

Maskininlärning är den vetenskapliga studien av algoritmer och statistiska modeller som med hjälp av data kring ett specifikt fenomen kan lära sig av datamängden och genomföra specifika uppgifter utan att explicit vara programmerade för detta. Maskininlärning kan delas upp i övervakat och oövervakat lärande där oövervakat lärande ligger i fokus i denna studie. Oövervakat lärande handlar till stor del om att göra utforskande analyser för att hitta mönster där det inte finns någon specifik vetskap kring vilka resultat som kommer att nås.

Mönsteranalys är en del av maskininlärning som handlar om att hitta meningsfulla mönster i stora mängder data. Mönsteranalys kan appliceras på olika områden, och för att hitta dessa mönster kan olika klustermetoder som kategoriserar, eller med andra ord klustrar, datapunkter i olika subgrupper baserat på deras egenskaper användas.

I denna studie undersöktes möjligheten att använda oövervakat lärande för att identifiera användarmönster i en bankapplikation för mobiltelefon. Den datamängd som användes i denna studie bestod av information om personers användande av bankapplikationen. Totalt extraherades fyra månader av sparad data som beskrev vad användarna klickar på och hur de interagerar med bankapplikationen, vilket summerade till 800 Gigabyte data. K-means och HDBSCAN användes för att identifiera subgrupper av sessioner vars egenskaper liknade varandra. Efter detta genomfördes en analys av hur ofta olika funktionaliterer i applikationen triggades i olika kluster samt vilka av applikationens funktionaliteter som ofta triggades ihop med varandra. Slutligen presenterades en rekommendation för hur bankapplikationen kan omstruktureras för att bättre anpassas till de beteendemönster som finns hos bankapplikationens användare. Med hjälp av klusteranalysen skapades även en förståelse för vilka funktionaliteter som inte användes särskilt ofta eller inte användes i den utsträckning som är tänkt.

Innan klusteralgoritmerna kunde köras behövde datamängden konverteras och reduceras till en hanterbar mängd. Varje användarsession sparades som en rad och alla funktionaliteter sparades som kolumner i en DataFrame. Därefter fylldes varje rad med antalet gånger de olika funktionaliteterna hade blivit triggade under sessionen. Genom att plocka ut de viktigaste funktionaliteterna i bankapplikationen reducerades datamängden till 23 dimensioner och efter vidare reducering av datapunkter baserat på satta kriterier reducerades datamängden slutligen till sju Gigabytes.

Utöver de kvantitativa metoderna användes även diverse kvalitativa metoder vid genomförandet av denna studie. Exempelvis hölls intervjuer med designers för bankapplikationen för att skapa förståelse för vad en användare kan göra i bankapplikationen, hur den är uppbyggd idag och vilka funktionaliteter som anses vara viktigast.

Resultaten från klustermetoderna K-means och HDBSCAN visar att K-means med användandet av fyra kluster var det fördelaktiga valet av metod för detta problem men att även denna metod för med sig diverse svagheter vid applicering på den tillgängliga datamängden. Fyra distinkta kluster kunde hittas av K-means men visualisering av klusterna visar på att det finns förbättringspotential hos modellen angående vilka datapunkter som tillhör vilket kluster.

Resultaten visar att det är möjligt att urskilja användarmönster i den mobila bankapplikationen med hjälp av en klusteranalys. Det operativsystem som används visade sig vara av stor betydelse vid skapandet av kluster och resultaten visade även att vissa funktionaliteter inte används särskilt ofta samt att vissa inbyggda genvägar inte används i den utsträckning som är möjlig. Applikationen

erbjuder således mer än vad användarna verkar ta till vara på. Anledningarna till att funktionaliteterna inte används kan bero av tre olika anledningar. Den första är att användarna inte är intresserade av att använda vissa funktionaliteter, den andra är att användarna har svårt att hitta vissa funktionaliteter i applikationen och den tredje är att användarna inte förstår hur vissa funktionaliteter ska användas. I denna studie har vi kunnat redogöra för hur många funktionaliteter som inte används särskilt ofta, vilka funktionaliteter som ofta används tillsammans samt vilka de vanligaste använda funktionaliteterna är. För att säkerställa varför användarmönstrena ser ut som de gör, bör en vidare kvalitativ analys med användarna som respondenter utföras. Valet att utföra en kvantitativ studie har möjligjort ett resultat som är representativt för en större population än vad som varit möjligt med enbart en kvalitativ analys. Vidare anser vi att de kvantitativa metoderna som presenterats i denna studie även är applicerbara på andra mobilapplikationer som också har sparad data över användarinteraktioner.

Acknowledgements

This report is the result of our master thesis project, which was conducted in the spring of 2019 as the final thesis of the Master Programme in Sociotechnical Systems Engineering at Uppsala University. The thesis was conducted together with Bontouch AB and the Department of Information Technology at Uppsala University.

We would like to thank Jonas Sköld, our supervisor at Bontouch, for providing important material necessary for the completion of this thesis, helping us find valuable connections and always supporting us along the way. Thank you to everyone from Bontouch who have contributed with valuable information. We would also like to say thank you to our fellow master thesis students at Bontouch for all knowledge you have shared with us and the fun times we have had this spring. Lastly, this thesis would not have been possible without the help from our supervisor at Uppsala University, Andreas Lindholm. Thank you for always making time for us, sharing your knowledge within machine learning and giving us valuable insights and feedback throughout the entire project. We are very thankful for your help and patience.

Elsa Bergman and Anna Eriksson
Uppsala, June 2019.

Distribution of Work

This thesis has been written by Elsa Bergman and Anna Eriksson who, together, have worked on all areas covered in this thesis. Most code has been written using pair-programming, a technique where one person writes the scripts and one person observes and gives comments on the code, after which they trade places. However, some parts of the code were developed separately and in these cases, the person not programming was responsible of writing the corresponding theory in the thesis. For example, Elsa had the main responsibility of implementing K-means and HDBSCAN, while Anna implemented the Principal Component Analysis. Using this method ensured that both authors were included in all areas of the project. Furthermore, the qualitative research methods were also conducted together. When conducting the interviews, one person had the main responsibility of asking the questions and the other had the responsibility of taking notes. Lastly, reviewing the thesis has been done multiple times in order to satisfy both authors.

Table of Contents

Wordlist and Abbreviations

- An **active cluster** is a subgroup in which many different functionalities are triggered or in which the occurrence of triggers per functionality is high.
- An **event** is the name of an action performed by a user in the bank application.
- An **eventArray** is an array containing event and eventParameter pairs for each triggered action in one session. Each session has one eventArray.
- **EventParameters** are key:value pairs containing additional information about a triggered action in the application.
- A **JSON object** is an object of format JavaScript Object Notation.
- A **mockup** is a graphic visualization of a user interface.
- **Pattern analysis** refers to finding underlying similarities between data points in a data set.
- A **passive cluster** is a subgroup in which few functionalities are triggered or in which the occurrence of triggers per functionality is low.
- **Peer-to-peer mobile payment** is an instant money transfer between two users [1].
- **RAM** is short for Random Access Memory.
- **Raw data** is data collected directly from the source.
- A **user session/session** is a time period of using the application. A session starts when the user logs in and ends when the session has been inactive for more than ten seconds.
- **UX** is short for User Experience.
- **Web mining** is a data mining technique which entails analyzing the behavior of users on a web platform [2].

1 Introduction

In today's developed countries, most people own smartphones and are able to keep different applications for different purposes in their phones. These applications are used for all kinds of everyday tasks such as buying bus tickets, writing groceries lists and transferring money, where each application has its unique design and set of users. When launching a service or application, it is important to think about the user's perception of the application. User experience, within the field of human-computer interaction, is defined as "a person's perceptions and responses that result from the use or anticipated use of a product, system or service" [3, p. 1], according to ISO standards. In addition to this, user experience can also be referenced to as the feelings which a user gets when he or she is using a product [3]. User experience is important when designing applications as it is vital for companies and creators of applications that the users like their products, have positive responses to them and understand what they should do to get the most out of the product [3]. Otherwise, it can result in a company losing its competitiveness on the market [4]. However, understanding how users interact with technology is not always easy. To develop user-friendly products, it is important to understand this relationship. It is stated by Slob et al. [5] that technology and user behavior are always intertwined since technology influences human behavior and, conversely, the patterns in human behavior affect the development of technology. Although this approach may be difficult to grasp, there are some methods one can use to understand how the users of an application are using the product, for example by tracking the users' actions and movements [2].

Moreover, the usage of smartphones is increasing. In the United Kingdom, smartphone usage has increased in all ages from 2012 to 2017 [6]. For people in the ages 16-24, the increase was seven percentages and for users in the ages 55-64, the usages have increased even more. Furthermore, in May 2018 it was reported that Americans now spend more time using mobile applications than they do browsing the internet [7]. This shows that both smartphones and mobile applications are becoming a bigger part of people's daily lives. To study how people use these mobile applications to fulfill their goals and needs is therefore more interesting today than ever before.

As it gets more important to understand the users of a product or service, companies are often collecting different kinds of data from their users [8]. This data can later be analyzed and used to identify new insights about users and to make predictive analyses. Furthermore, data of user behavior can also help companies make recommendations of products which the user should buy based on what other users with similar interests have bought [9]. To be able to make these types of analyses, there is often a need for massive quantities of data. This amount of data is usually referred to as big data [10].

By using data collected from users of a website or mobile application, it is possible to train machine learning models to make analyses regarding user behavior. Some of these models can for example help with finding patterns and create subgroups within a data set [11]. The more users an application has, the more data is collected and can be used to train the models.

Identifying patterns of user behavior on web pages or in mobile applications makes it possible to gain further understanding of how users are using a platform. Knowledge of whether users are using the implemented features in the way intended or if they are having difficulties understanding how to fulfill their goals and needs can help developers and designers in making decisions regarding future implementations. This type of knowledge can also be used to customize applications for different types of users or optimize them to better suit the needs of the users.

1.1 Purpose

The purpose of this study is to explore whether it is possible to conduct a data-driven study in order to gain understanding of user behavior within a mobile application. More specifically, we wish to cluster the sessions of the application into subgroups based on what functionalities users are using in one session. We hope to gain insight into how the functionalities in the application are used and to lay a foundation for how the application can be altered to better fit the identified needs of the users. In order to fulfill the purpose, we have chosen to use a mobile bank application as our subject of study.

1.1.1 Research Questions

1. How can clustering algorithms be used to understand how functionalities are used within a mobile bank application?
2. What patterns can be identified from clustering analysis of user actions in the mobile bank application?
3. In what way can pattern analysis provide a foundation for making recommendations of how the mobile bank application can be altered to better suit the actual needs of the users?

1.2 Disposition

The following thesis is divided into eleven chapters. The remaining section in chapter one consists of the delimitations of the study. Chapter two covers the topic of machine learning with some relevant terminology, pre-processing models and algorithms which are used. In chapter three, relevant concepts in the field of human-computer interaction are presented. In chapter four, related works in the fields of web mining, pattern analysis in user behavior, clustering and the importance of keeping the user in mind when creating applications are presented. Chapter five contains an explanation of the bank application's functionalities which are in focus in this study. In chapter six, the data which was used is presented. In chapter seven, explanations of the qualitative and quantative research methods that were used, such as semi-structured interviews, pre-processing methods and clustering models are given. In chapter eight, the results of the study are presented and this is followed by a discussion of the results in chapter nine. Chapter ten consists of the conclusive findings. Finally, future research, which is outside the scope of this study, is presented in chapter eleven.

1.3 Delimitations

For this study we have chosen to work with a mobile bank application developed by the mobile application development bureau Bontouch, the company at which this study was conducted. This application was chosen since it is well established on the Swedish mobile application market, it has a wide variety of usage among different users and the collected set of data of user actions in the application is of considerable size. The study is based on data which has been gathered during the year of 2018. To narrow down our scope, we have chosen to only work with data from four different months: March, May, September and October. The reason for this is further explained in section 6. Furthermore, when analyzing the results in order to find different patterns in user behavior, we do not take the sequence in which events happen into account. This means that we only consider what functionalities are used during a session but not in the order they are used. We do not take the unique users into consideration and will not investigate the user patterns of specific users. Instead we only consider what functionalities are triggered in one session, and not which user who triggered them. This means that multiple sessions can be launched by the same user. Lastly, because of the large number of actions a user can trigger within the application, this study focuses on a few selected functionalities. These functionalities are considered to be most important within the application and are explained in section 5.

2 Machine Learning

Machine learning is the scientific study of algorithms and statistical models which rely on a collection of data of some phenomenon [12]. According to Burkov [12], machine learning can be defined as the process of gathering a data set and using algorithms to build a statistical model based on the gathered data set in order to solve a practical problem. Furthermore, Burkov [12, p. i] also defines machine learning as a "...universally recognized term that usually refers to the science and engineering of building machines capable of doing various useful things without being explicitly programmed to do so". The decisions that machine learning models can make are made based on statistical models trained on massive amounts of data and there are two primary fields of machine learning, unsupervised and supervised learning [13]. In this thesis we focus on unsupervised learning.

Unsupervised learning is explained by James et al. [13] as a branch of machine learning in which data is not labeled. In other words, we have a set of observations but no response variables to the observations. Unsupervised learning algorithms do not solve any classification problems and are therefore not making any predictions. Instead, the goal is to making interesting discoveries about the observations and find patterns within data sets. As explained by James et al., unsupervised learning problems are often used for exploratory analysis and it can be difficult to verify the results obtained from these types of problems. This is because there is no universally accepted method of checking the accuracy of the work. According to James et al., the problem lies in the fact that we cannot measure the accuracy of the model and confirm whether our model performed well or not since we have no way of comparing the results and predictions against a true answer. This is due to the fact that with unsupervised problems there are no true answers.

In unsupervised learning we are also often interested in visualizing the data in different ways and one way we can do this is by *clustering*. Clustering is described by James et al. as an unsupervised learning method that refers to finding subgroups, or clusters, within a larger data set. Data points with characteristics that are similar to each other are clustered within the same subgroup and data points with differing characteristics are assigned to different subgroups. Clustering can be applied to the field of pattern recognition. In this thesis, we use clustering to outline which types of actions are most frequently triggered as well as often triggered together in the same session.

The first part of this chapter explains the machine learning terminology which is important to understand in order to comprehend the technical discussions in the report. Secondly, the field of pattern recognition is introduced in section 2.2, with an explanation of how pattern recognition is used in machine learning. Furthermore, a brief explanation of the connection between big data and machine learning is made in section 2.3. Later, in section 2.4 the importance of pre-processing is deliberated on and in section 2.5, an explanation of different pre-processing methods and important factors to take into consideration when working with machine learning algorithms are presented. This is followed by an explanation of the clustering methods which are used in this study, and how to find the best number of clusters to use. This is explained in 2.6 and 2.7 respectively.

2.1 Machine Learning Terminology

The following terminology, as defined by Han et al.[11], James et al. [13] , Kuhn et al. [14] and Brownlee [15], is considered important for the reader to comprehend in order to grasp the technical aspects of this study:

- An **attribute** or a **feature** is a characteristic of a data point. Each data point can be assigned several attributes/features. The words attribute and feature are used interchangeably in this report.
- **Binary variables** are nominal attributes and have values which can only take on one of two variables, such as 0 or 1, or true or false.
- **Categorical** or **nominal variables** take on values which do not have a scale but instead represent different categories or a certain state. Examples of categorical data are a person's gender (male or female) or a purchased product (product A, B or C).
- A **centroid** is a center point around which a cluster is formed.
- A **data point** is a single unit of data.
- A **data set** is a collection of data points.
- **Numerical variables** are quantitative, meaning they are represented by real numbers and can be measured. Examples of numerical variables are income, age and height.
- A **sparse data set** is a data set with a high percentage of zeros.

2.2 Pattern Recognition

In this thesis, we base our work on the definition of pattern recognition as "...the scientific discipline whose goal is the classification of objects into a number of categories or classes". [16, p.1] Here, classification refers to the categorization of objects into categories based on the characteristics of the objects, not the supervised learning task. Depending on the application, these objects can be images or sounds or any type of measurement that should be classified [16].

Pattern recognition is used in several different fields, for instance within computer engineering and medicine [16]. For example when a doctor is going to make a diagnostic decision if a patient has cancer or not, she can be assisted through image analysis. The image analysis gives indications of whether a tumor is benign or malignant by comparing the characteristics of the tumor to other tumors already classified as benign or malignant. In this example the doctor can both save time and be more certain of her diagnostic decision. Besides images, there are also other data formats that can be used for detecting patterns, such as sounds and text. Another popular area to use pattern recognition is in speech or character recognition [16].

Furthermore, pattern recognition is also used in web mining, defined by Singh et al. [2] as a data mining technique which entails analyzing the behavior of users on a web platform. In web mining, data containing information about the users' actions on a specific web platform is collected in order to analyze the behaviors of the users. Singh et al. state that by analyzing customer patterns, companies are hoping to learn more about the users' goals and needs. They describe that this technique has brought the end customer closer to the company providing the service and has enabled companies to customize their web platforms to the needs of different types of users.

Analyzing different patterns is possible by the help of clustering [13]. In web mining, clustering can be used to find subgroups of customers performing similar actions on a web application, or to find pages on a web application which users perceive as being related to one another [17]. There are multiple clustering algorithms which can be used in unsupervised learning but in this study we mainly use partitioning clustering explained further in section 2.6.1. Furthermore, we use density-based clustering as an additional method.

2.3 Big Data in Machine Learning

The foundation of machine learning is having access to data, and preferably a large data set. It is therefore important to understand the concept of big data. Data can be gathered from many different sources, such as web pages on computers and from mobile applications. Today's smartphones are equipped with a variety of sensors which are used to collect large sets of data [10]. It is stated by Cheng et al. [10, p.1] that "the purpose of big data processing is to piece together such data fragments so as to gain insights on user behaviors, and to reveal underlying routines that may potentially lead to much more informed decisions.". To be able to call a data set big data, the data set must be of a tremendous size [10].

Furthermore, Cheng et al. describe that big data is often defined by the five Vs: *volume*, *velocity*, *variety*, *veracity* and *value*. These refer to the enormous size of the data, the fast streaming of the data, the heterogeneity of the data, the quality of different sources of data, which may be inconsistent with one another and contain noisy data that must be removed, and lastly the economic value of the data.

According to Brownlee [18], there are several different approaches one can take when working with larger data sets. One method is to use a cloud solution and divide the work on different nodes. Another approach is to use progressive loading and read the data files in batches. What options you have when choosing how to work with your large data set is according to Brownlee, dependent on the computer your code is running on. If the computer has a large RAM, the processing of data will be quite efficient and thus enable the usage of progressive loading. On the other hand, if the computer's RAM is small, then using the cloud solution could be preferable. Furthermore, the format in which you choose to store the data files also matters when working with larger sizes of data.

2.4 Pre-processing Methods

Before a data set is ready to be used for modelling by machine learning algorithms, it needs to be pre-processed in order to improve the quality of the data. According to Han et al. [11], some of the factors which make up high quality data are *accuracy*, *completeness*, *believability*, *interpretability* and *consistency*. Accurate data does not contain any errors or noise, meaning there are no values which greatly deviate from what is expected. Furthermore, data is counted as complete when the attributes and values which are of interest to study are present in the data set. Believability and interpretability mean that the data should be trusted by the users and easy to interpret. Consistent data has attributes which are populated in the same way for all data points, meaning there are no discrepancies in how the data is stored for each category. One example of inconsistent data described by Han et al. is having dates stored in different ways, such as YYYY/MM/DD for some data points and DD-MM-YYYY for others.

According to Han et al., pre-processing consists of a few primary steps: *data cleaning*, *data integration*, *data reduction* and *data transformation*. Data cleaning consists of filling in any missing values, solving inconsistencies and removing outliers. Data integration generally consists of several smaller steps but in this study, the primary step of data integration is handling redundancy in the data. Attributes are redundant if they explain the same scenario or if they can be derived from other attributes. Data reduction includes reducing the size of a large data set into a smaller size which can represent the large data set and produce approximately the same results. Reducing data can be done through dimensionality reduction using feature selection. The third step, data transformation consists of converting the data to another form, for example by standardizing the data, in order to improve the modelling results. In this study, these steps are performed to ensure that the data holds the highest possible standard. The complete process is further explained in section 7.2.3.

2.4.1 Feature Selection

Due to the structure and workings of unsupervised learning, Yao et al. [19] state that feature selection using unsupervised data is considered a more difficult problem than it is using supervised data. In addition to this, when clustering high dimensionality data it is often challenging to find meaningful connections between data points amongst a large number of features, many of which are often irrelevant. According to Yao et al., clustering algorithms are often sensitive to data of high dimensionality and feature selection is a way of selecting attributes in order to reduce dimensionality of the data set and computational complexity. There are several definitions of what feature selection entails but in this study we refer to feature selection as the process of removing redundant and irrelevant attributes from the original data set [11].

However, the topic of how to find relevant attributes and performing feature selection using unsupervised data is relatively untouched. Dash et al. [20] states that there are many feature selection algorithms one can use when working with supervised data sets which provide class information. On the other hand, in the field of unsupervised learning, there are no available class labels and therefore it is difficult to measure how the selection of different attributes actually affects the outcome of the clustering. Furthermore, it is also described by Cai et al. [21] that the possible correlations between different attributes make the task of finding relevant attributes for the unsupervised problem even more complex.

Although it is stated that it is quite difficult to find relevant attributes, a framework for efficient feature selection is suggested by Yu et al. [22] The framework is divided into two parts: relevance analysis and redundancy analysis. Furthermore, an algorithm to effectively perform feature selection according to this framework is also presented by Yu et al. However, in this study we use the framework suggested by Yu et al. as inspiration, without using the algorithm presented in their study. This is due to the algorithm requiring supervised data, which is not used in this study.

The first part of Yu et al's framework consists of extracting strongly relevant, weakly relevant and irrelevant attributes. In the second part, the relevant subset is divided into redundant and non-redundant attributes. By following the framework, one can obtain a satisfying subset in an efficient way. The relevance analysis helps determine a relevant subset from the original data set, after which the redundancy analysis helps eliminate redundant attributes from the relevant subset and output the final subset [22]. This general process is visualized in figure 1. There are similarities between Yu et al's relevance analysis and Han et al's definition of data reduction as both include reducing the size of the data set. Furthermore, Yu et al's redundancy analysis is also similar to Han et al's definition of data integration, as both highlight the importance of removing redundant attributes.

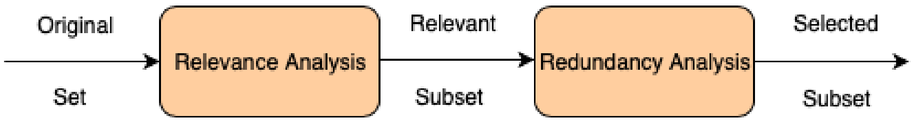


Figure 1: Efficient Feature Selection [22].

Yao et al. [21], Dash et al. [20], Cai et al. [21] and Yu et al. [22] all propose different methods of feature selection for unsupervised data. This goes to show that the process of feature selection for unsupervised learning is not agreed on among scientists and researchers within the field.

2.4.2 Outlier Detection

One way of reducing the data set is to identify outliers and remove them. An outlier can be described as ”a pattern which is dissimilar with respect to the rest of the patterns in the dataset” [23, p.24]. Furthermore, another description of an outlier is a ”...data object that deviates significantly from the rest of the objects, as if it were generated by a different mechanism” [11, p.544]. This means that outlier data points have characteristics which greatly differentiate them from the rest of the data points. Due to this, outliers can lower the accuracy of the clustering algorithm [24]. Furthermore, some clustering algorithms are sensitive to outliers, which means that outlier data points may have a negative impact on the cluster groupings [24]. For instance, this can depend on many clustering algorithms requiring all data points in the data set to be forced into a cluster [13]. A small number of outlier data points can greatly affect the mean values and therefore also the positions of the cluster centroids [24]. This will in turn affect the cluster formations.

There are several ways of detecting outliers within data sets and in this study we use the number of actions that every user triggers in a session as a reference point to detect outliers. This procedure is explained in section 7.2.3.

2.4.3 Standardization

Standardization, part of the pre-processing step data transformation [11], is often used before clustering in order to ensure that the entire data set has a particular property [25]. In a data set consisting of several attributes with different units and magnitudes, standardizing to standard normal distribution ensures that all variables are transformed to take on a mean of zero and a standard deviation of one. This results in no attribute having a dominating impact on the outcome of the clustering [25].

James et al. [13] state that the decision of choosing to standardize or not can strongly affect the obtained results. However, when handling unsupervised learning problems, James et al. explain that there is often no single correct solution to whether methods such as standardization should be used or not. Instead one can try different options in order to find the one which reveals some interesting patterns of the data.

2.4.4 One-hot-encoding

One-hot-encoding is another useful method for pre-processing the data. This method converts categorical attributes into numerical attributes [26]. This is a convenient approach when working with clustering models, since they are often based on calculations of distances and are therefore in need of numerical values. For example, if one attribute is *Color* and the values of the data points differ between *red* and *blue*, the encoder derives the categories into unique attributes and assigns the value 1 for true and 0 for false. Figure 2 visualizes one-hot-encoding.

Data point	Color
1	Red
2	Blue

–>

Data point	Red	Blue
1	1	0
2	0	1

Figure 2: Description one-hot-encoding.

One disadvantage with this approach is that it increases the number of attributes in the data set, and therefore also the number of dimensions, which can lead to the curse of dimensionality [27], further explained in section 2.5.

2.4.5 Principal Component Analysis

Large data sets often contain large amounts of attributes. These attributes might not always contribute to the data analysis and therefore, the data set can sometimes be represented by a smaller amount of variables which represent most of the variability in the original set [13].

In order to reduce the dimensionality, James et al. [13] refer to the use of Principal Component Analysis, or PCA. PCA is part of the pre-processing step data reduction [11] and works as a tool which takes data of high dimensionality and finds a low-dimensional representation of the data set. James et al. state that PCA also ensures that we keep as much of the information as possible by finding a small number of dimensions which are considered to be the most interesting, where ”interesting” is measured by the amount that the observations vary along each dimension. Each dimension, which PCA helps find, is then represented as linear combinations of the attributes in the data set, called principal components.

Visualization of observations consisting of a large set of attributes is difficult and although it is possible to reduce the dimensions by creating multiple two-dimensional scatter plots with two attributes in each plot, it is likely that none of them will be informative. As explained by James et al., this is because each plot will contain a very small proportion of the total information which the data set contains. By instead reducing the dimensionality with PCA, we can keep most information in the data set. Figure 3 gives a visual representation of how a three dimensional space can be converted into a two dimensional space with PCA transformation.

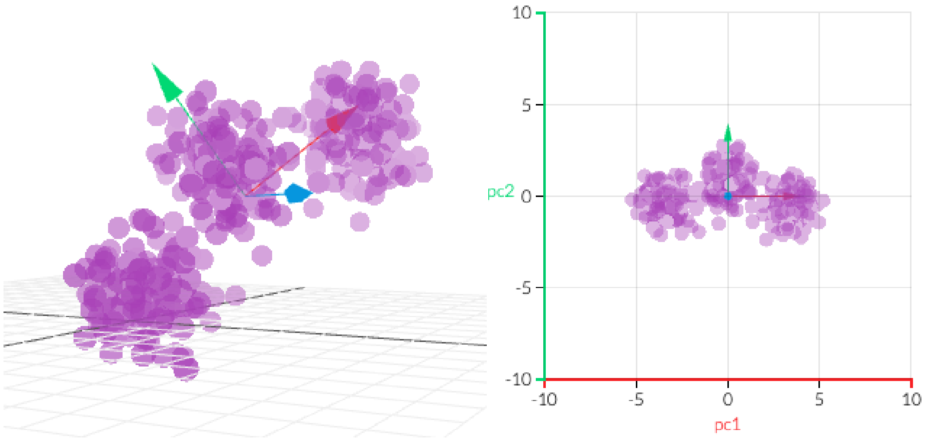


Figure 3: PCA transformation from three dimensional space (left) to a two dimensional space (right), with the first and second principal components represented on the x-axis and y-axis [28].

The first principal component

$$Z_1 = \phi_{11}x_1 + \phi_{21}x_2 + ... + \phi_{p1}x_p \tag{1}$$

is explained by James et al. as the normalized linear combination of attributes with the highest variance. The attributes x_1,...,x_p are assumed to have mean zero. The fact that we have a normalized linear combination means that the sum of squares of all coefficients ϕ_{11},...,ϕ_{p1} should add up to one. These coefficients are referred to by James et al. as the *loadings* of the first principal component and ϕ_1, the vector for the loadings of the first principal component, is called the first loading vector. The loading vectors assign *weights* to each attribute x in the principal components. The weights determine which attributes the components mostly correspond to. If the loading vector places equal weights on all attributes, James et al. state that the component corresponds equally to all attributes. If the loading vector places most of its weight on a few attributes, it means that the component mostly corresponds to these few attributes. If the loading for an attribute x_i is zero, it means that the attribute is not included in the principal component.

As explained by James et al., the first principal component loading vector solves the optimization problem

$$\max_{\phi_{11},...,\phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \Big(\sum_{j=1}^{p} (\phi_{j1} x_{ij}) \Big)^2 \right\} \text{ subject to} \sum_{j=1}^{p} \phi_{j1}^2 = 1 \tag{2}$$

This means that we wish to find the principal component Z_i which has the largest variance while still keeping the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

Furthermore, the second principal component is the normalized linear combination

$$Z_2 = \phi_{12}x_1 + \phi_{22}x_2 + ... + \phi_{p2}x_p \tag{3}$$

with maximal variance, out of all linear combinations which are uncorrelated with Z_1 [13]. By keeping Z_1 and Z_2 uncorrelated, James et al. state the loading vectors ϕ_1 and ϕ_2 for each of the linear combinations become orthogonal to each other. Knowing this, we can compute ϕ_2 with the help of ϕ_1 (see [29] for further explanation). By keeping the principal components uncorrelated, we are also able to maximize the variance of the data points and ensure that the principal components represent different aspects of the data points [30]. The first principal component explains the largest amount of the variation in the data set, the second principal component explains the second to largest amount of the variation in the data set and so on [13].

When reducing the space to two or three dimensions in order to visualize the data points, the data points are described as projections along the directions of the first two or three principal components [13]. The x-axis will represent the first principal component vector, the y-axis will represent the second principal component vector and the z-axis will represent a potential third principal component.

Deciding on how many principal components to use can be done with the help of a *scree plot*, visualized in figure 4. In the scree plot, the smallest number of principal components which together explain a substantial amount of the variation in the data are chosen [13]. These components are found by looking for the point in the scree plot where the difference between the bars is the largest. This is referred to by James et al. [13] as the elbow, as the point resembles an elbow when the entire graph is compared to the shape of an arm. However, as highlighted by James et al., there is no universally accepted way of determining the number of principal components to use. It will depend on the specific data set and the area of application. In order to be able to visualize the data in one graph, the number of principal components need to be reduced to two or three. Furthermore, a scree plot can also be used to understand how much of the variation each principal component explains. This is helpful when analyzing how much information was lost when projecting the observations onto two or three principal components [13].

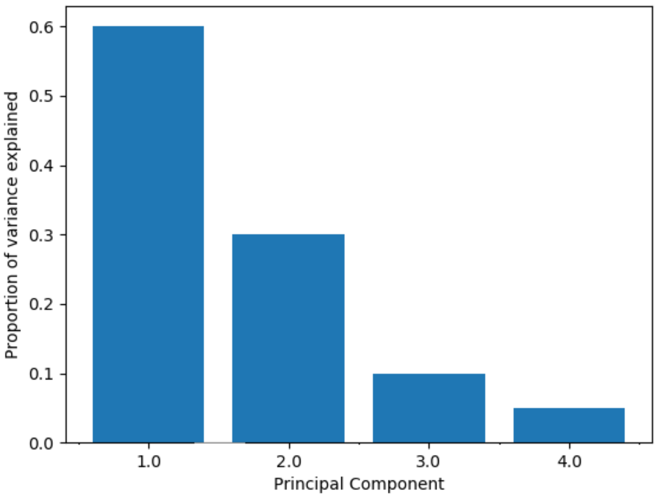


Figure 4: Scree plot, where the axes and data points are retrieved from [13].

Furthermore, James et al. explain that the result of a PCA depends on the format of the data. If the data is non-standardized, there may be a big difference in variance among the features. The feature with the highest variance is given the largest loading score and this feature has the greatest impact on the principal component. On the contrary, if the data is standardized it is more likely that the features will have approximately the same importance and the loading scores will therefore be more equally distributed.

2.5 Curse of Dimensionality

The curse of dimensionality problem was first introduced by Bellman [27] in order to describe the complications which arise with exponential increase in volume when adding extra dimensions to the Euclidean space. Feature selection can help against curse of dimensionality problems, as dimensionalities in data sets decrease when the number of features decrease [19].

When working with a large set of features, some clustering methods may show a decrease in performance due to redundant features and noisy data [31]. According to Steinbach et al. [32], problems with high dimensionality usually occur due to a fixed number of data points becoming more sparse as the dimensions increase. Lastly, Steinbach et al. refer to the findings

$$\lim_{dim \to \infty} \frac{MaxDist - MinDist}{MinDist} = 0 \tag{4}$$

which state that the relative difference in distance between the closest and the farthest data point of an independently selected point converges to zero as the number of dimensionalities increase, especially if the data points are identically and independently distributed. This goes to show that clustering based on distance becomes less meaningful as the dimensions increase.

2.6 Clustering Algorithms

As mentioned by James et al. [13], clustering is a common method within unsupervised learning, used to find subgroups, or clusters, within a data set. By the use of clustering, we are able to partition a data set in such way that data points with similar characteristics belong to the same subgroup and that data points with dissimilar characteristics belong to different subgroups. With clustering, we aim to find an underlying structure in the data set, which has previously been unknown. There are several clustering methods in the field of unsupervised learning, usually divided into one of the categories partitioning methods, hierarchical methods, density-based methods and grid-based methods [11]. In this study, we focus on partitioning clustering and density-based clustering and more specifically the clustering algorithms K-means and DBSCAN [11].

2.6.1 K-means

K-means is one of the most common and widely used clustering methods which can be used to find interesting patterns in data sets based on characteristics of data points [13]. Han et al. [11] calls K-means a partitioning clustering method in which we partition the data set into a pre-defined number of clusters, K, by first choosing a fixed number of cluster centroids which clusters are formed around. K-means has time complexity $\mathcal{O}(nKt)$ where n is the total number or data points, K is the number of chosen clusters and t is the number of iterations needed to reach convergence. Usually $K << n$ and $t << n$, which makes K-means scalable and a good choice when working with large scale data sets. However, K-means is not a suitable method for finding clusters of considerably different sizes and non-convex shapes [11]. Furthermore, K-means is also sensitive to outliers [13].

James et al. [13] explains that when running the K-means algorithm, we compute the pairwise distance between every data point and cluster centroid in order to identify which cluster centroid has the shortest distance to each data point. Each data point will thereafter be assigned to the cluster centroid which has the shortest distance to that data point. When all data points have been assigned to a cluster, the value of the cluster centroid in each cluster will be re-calculated. This is done by calculating the mean of the data points within the clusters, and this becomes the new cluster centroid. After this, each data point will be re-assigned to the cluster centroid which is located closest to the data point. The process is iterative and iterates until the cluster assignments stop changing, as this means that the final clusters have been formed.

Euclidean distance, defined as

$$d(x_i, x_{i'}) = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = \|x_i - x_{i'}\|^2 \tag{5}$$

is a common way of computing the distance between data points and centroids when using the K-means algorithm [33].

K-means finds a local, not a global, optimum and the cluster results will depend on the initial centroids that are set in the beginning [13]. As a result of this and mentioned by James et al., it is important to run the algorithm multiple times with different initial cluster assignments, after which the best solution is selected. The best solution is the one for which

$$\min_{C_1,\ldots,C_K} \left\{ \sum_{k=1}^{K} \frac{1}{\mid C_k \mid} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\} \tag{6}$$

is fulfilled. The optimization problem which defines K-means clustering thus consists of finding the cluster constellations which make the within-cluster variation summed over all clusters as small as possible, [13] where the within-cluster variation decreases with every iteration [11]. The within-cluster variation for cluster K, C_K, is the sum of pairwise least squared Euclidean distances between the data points in C_K divided by the total number of data points in cluster K [13].

One of the most challenging problems when conducting K-means clustering is finding the value for the number of clusters, K, which should be used [13]. According to Han et al. [11], it is important to use an appropriate number of clusters, as this will affect the balance between the compressibility and accuracy of the clusters. If the number of clusters are equal to the number of data points, we achieve great accuracy due to the distance between each centroid and the cluster point being zero. However, this defeats the purpose of clustering into subgroups. If we choose K=1, the compression of the data points into a smaller representation is maximized but the accuracy would probably be very low and again, we lose the purpose of clustering.

As explained by Arthur et al. [34], finding the initial centroids for the K-means algorithm can be done using the kmeans++ algorithm. This is an algorithm which helps in finding centroids by choosing the initial cluster centroids according to the value of $D(x)^2$, and not at random, which results in a better clustering. The algorithm works as follows:

1. Choose one centroid at random from the data set.
2. Choose a new data point and assign it as a new center. The new data point x in the data set X is chosen with probability
$$\frac{D(x)^2}{\sum_{x \in X} D(x)^2} \tag{7}$$
where $D(x)^2$ is the squared shortest distance from a data point x to the closest center which has already been chosen.
3. Repeat step 2 until we have the number of centers we wish to use for K-means clustering. Then proceed with K-means clustering.

2.6.2 Mini Batch K-Means

Due to of its efficiency, K-means is a popular choice when it comes to clustering [13]. However, the run time always depends on the size of the data. An alternative approach called Mini Batch K-means is suggested by Sculley [35]. This is a variant of K-means that lowers the computational complexity on large data sets. The main idea behind this algorithm, according to Béjar [36], is to use small random batches of samples of a fixed size which can fit into memory. With each iteration, a new random sample, also called a mini batch, is obtained from the data set and used to update the clusters until convergence is reached. As the number of iterations are increasing, the ability of new mini batches to change the clusters is reduced. Convergence is reached when no changes occur in the clusters, exactly like in K-means.

2.6.3 DBSCAN

In addition to partitioning clustering methods, there are density based clustering algorithms such as Density-Based Spatial Clustering of Application of Noise, henceforth referred to as DBSCAN. Density based clustering algorithms are much less sensitive to outliers compared to partitioning clustering methods and are used to find arbitrary shapes [37]. This kind of clustering algorithms can find non-spherical shapes since its strategy is to cluster dense regions and classifying the sparse regions as outliers [11]. Furthermore, DBSCAN does not need a pre-defined number of clusters as a parameter [37]. DBSCAN is very useful for clustering large data sets since it does not classify all data points into clusters. Han et al. [11] explains that the idea of DBSCAN is that the algorithm continues to increase the size of a given cluster as long as the number of data points in the "neighborhood" exceeds the chosen threshold. If the number of data points in the neighborhood exceeds the chosen threshold, the area is considered dense. The neighborhood is defined by Han et al. as the space around a specific object located within a chosen radius.

Han et al. explains that DBSCAN finds core objects, or in other words, objects that have dense neighborhoods. Connecting the core objects to their neighborhood enables the formation of clusters. To be able to run DBSCAN, Han et al. state that the two variables *minimum points* and *epsilon* have to be pre-defined. Epsilon specifies the radius of the neighborhood for each object. Due to epsilon being pre-defined, the density of a neighborhood can be measured by the number of objects in the neighborhood. The minimum points variable determines whether the neighborhood is dense or not. If the number of data points in the neighborhood of a core object exceeds the chosen threshold for minimum points, the area is considered dense. Han et al. refer to a core object as an object in which the object's epsilon neighborhood contains at least as many core points as specified by the minimum points parameter. Furthermore, Han et al. also discuss border points and noise. A border point is the point which is reachable from a core object but is not considered a core object itself. This point is also part of the dense area. Lastly, a data point is considered to be noise if the neighborhood of the data point does not contain at least the number of data points specified by minimum points. If the data point is marked as noise, it will not be considered part of any cluster. In figure 5, points m, p, o and r are core points when minimum points $= 3$. Data points q and s are border points and the points not included within a circled are marked as noise. The circled areas have the radius epsilon.

The time complexity for DBSCAN is $\mathcal{O}(n^2)$ [38]. The metric used to calculate the distance between data points is often Euclidean distance but other metrics such as Manhattan distance can also be used [39]. As these require numeric attributes [11], DBSCAN also requires numerical values.

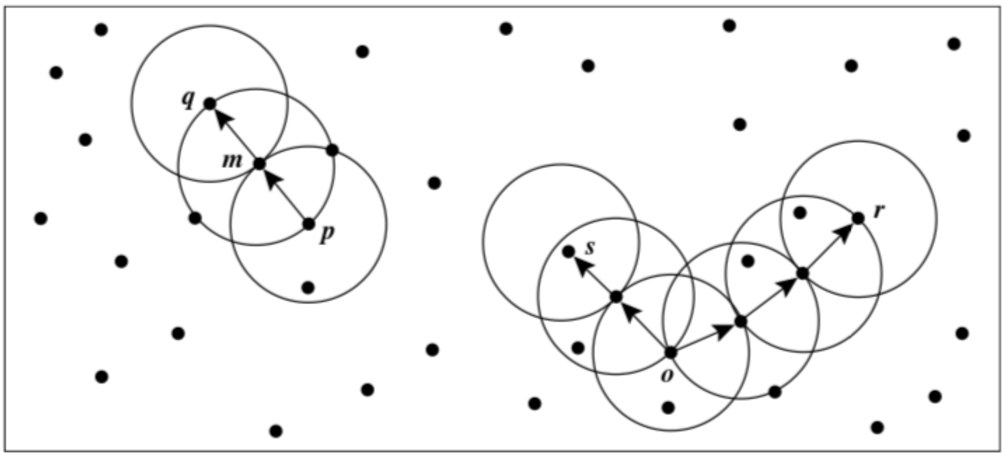


Figure 5: Visual representation of how data points are classified as core points, border points or noise. [11]

2.6.4 HDBSCAN

Hierarchical DBSCAN, or HDBSCAN, functions in a similar way to DBSCAN. However, the goal of using HDBSCAN is to allow clusters of varying density [40]. The HDBSCAN documentation [40] explains that the algorithm transforms the space according to density and applies single linkage clustering, exactly as DBSCAN. Furthermore, the HDSBSCAN documentation states that the algorithm has several parameters such as *minimum cluster size* which decides how many data points must be included in a cluster in order for the cluster to be classified as a cluster. This parameter is not present in the DBSCAN algorithm. Furthermore, HDBSCAN is also dependent on choosing a *minimum sample*, which sets a constraint as to how conservative the clustering should be [40]. The larger minimum sample used, the more conservative the clustering will be. This is because more points will be classified as noise if minimum sample increases. The time complexity of HDBSCAN is $\mathcal{O}(an^2)$ where a is the number of attributes in the data set and n is the number of data points [39]. HDBSCAN is considered to be efficient when well implemented using suitable parameter values [40].

2.7 Finding K

In order to be able to use K-means, one must first specify the number of clusters which should be used in the algorithm [13]. What actually is the appropriate number of clusters is difficult to define as it depends on the data set and its distribution of data points [11]. There are several methods which calculate the number of clusters, K, for K-means. Examples of methods are the Elbow Method and calculating the Silhouette Scores, both explained below.

2.7.1 Elbow Method

The Elbow Method is a method in which we analyze the within-cluster variation as a function of the number of clusters [41]. To be able to choose the right number of clusters, a plot is created where the number of clusters are plotted against the within-cluster sum of squares [42]. The within-cluster sum of squares explains how far individual data points are from the mean within each cluster [13]. The Elbow Method must have a pre-defined range of clusters. Bholowalia et al. [42] explain that the best number of clusters to use is the number for which adding another cluster does not give much better modelling of the data. The plot shows a graph where the first cluster will explain much of the variation, but then it will continue to decrease. This decrease will form an angle in the graph, similar to the angle created when bending your elbow. The correct number of clusters, K, also called the *elbow criterion*, is found where the graph starts to curve. The idea is to start with K=2 and continue to increase K one step at a time. At some value for K the variance will rapidly drop and the graph will flatten out. This drop responds to the K which give the best number of clusters to use in the clustering method. An example of the Elbow Method can be seen in figure 6.

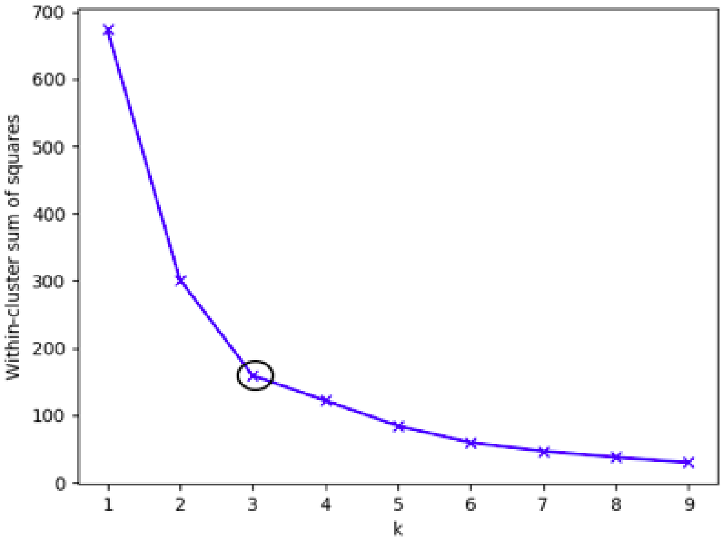


Figure 6: Elbow Method with the elbow criterion circled.

2.7.2 Silhouette Score

Another possible method when deciding the number of clusters is to use Silhouette Scores. The following explanation of the method is made according to Cordeiro de Amorim et al. [43]. They state that these scores are based on how well every data point fits into the assigned cluster, by comparing the within-cluster cohesion. The cohesion is based on the distance to all other data points within the same cluster and the Silhouette Score for each data point is calculated by

$$S(x) = \frac{b(x) - a(x)}{max(a(x), b(x))} \tag{8}$$

where $a(x)$ is the mean distance between a sample and all other data points in the same cluster and $b(x)$ is the distance between a sample and all other data points in the next nearest cluster. This results in the Silhouette Score interval $S(x) \in$ [-1,1]. If the Silhouette Score is around zero, the data point could be assigned to another cluster without making the cluster cohesion or separation any worse. However, if the $S(x)$ is negative, then the data point is incorrectly assigned and is damaging the cohesion in the current cluster. If the Silhouette Score is positive, then the data point is correctly assigned. Plotting the average Silhouette Scores of the data points in each cluster against the number of clusters, results in a line graph. Looking at this graph, the best value of K is the maximum value of the average Silhouette Scores. This is because this K results in the average Silhouette Score of all data points being closest to one, or 100 percent. Figure 7 below shows an example of the best value of K found using a Silhouette Score plot.

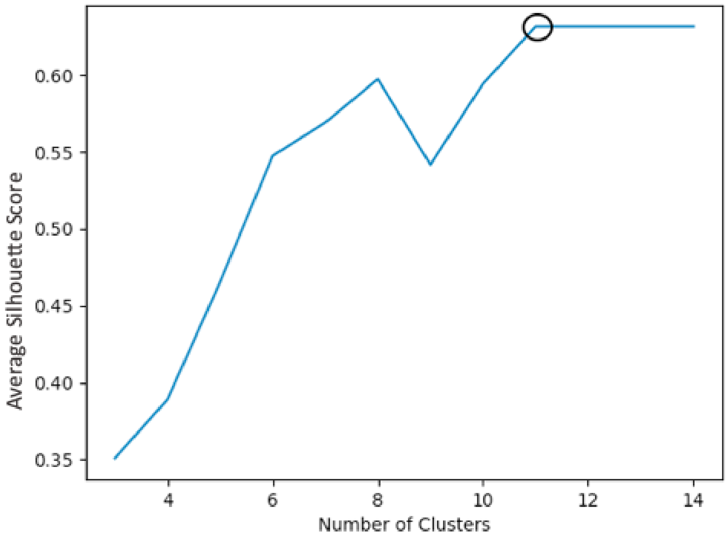


Figure 7: Average Silhouette Score plotted against number of clusters with the best value of K circled.

3 Human-Computer Interaction

Since this interdisciplinary research study combines the fields of machine learning and human-computer interaction, it is important to define important concepts in both areas. In addition to the machine learning terminology presented in section 2, we explain important theories of human-computer interaction in sections 3.1 to 3.4. Using the models within machine learning presented in section 2 to conduct pattern recognition, the concepts in human-computer interaction are used as tools to analyze the revealed patterns.

The purpose of this study is not to create a completely new product or to redesign an application from scratch. Our goal is instead to suggest smaller changes in hopes of improving the user experience. Changing small details will not add any new functionalities, but may improve the core parts of the existing product. Kraft [3] considers this a more effective method of improving the user experience, rather than adding completely new functionalities. Furthermore, bigger changes may take longer for users to understand and learn, while smaller changes can be acknowledged and appreciated immediately.

In order to fulfill the purpose of this study, it is also necessary to briefly explain the concept of interaction design. Interaction design focuses on how people interact with technology and different systems. As explained by Norman [44], the goal of interaction design is to give people a better understanding of what is happening when they are using a system, what can be done from now on and give feedback on what has previously occurred. Furthermore, Cooper et al. [45, p. xxvii] define interaction design as ”the practice of designing interactive digital products, environments, systems and services” and describe interaction design as knowing what the user wants. This is knowledge which will help a designer create more successful products.

3.1 Fundamental Principles of Design

According to Norman [44], the two most important characteristics of good design are *discoverability* and *understanding.* Discoverability is the characteristic which sheds light on whether it is possible for the user to figure out what actions are possible to perform and how to perform them. Understanding has to do with the user understanding how the product should be used and what the different controls in the system mean. This definition of understanding will work as a foundation for analyses in this study.

Discoverability is also the first of the seven fundamental principles of design which are presented by Norman. In addition to discoverability, the other fundamental principles of design are *affordances*, *signifiers*, *constraints*, *mappings*, *feedback* and *conceptual* models.

Affordances have to do with the user understanding what a product or feature is used for. For example, people know that a chair is meant for sitting on, meaning the chair affords sitting. Whereas affordance explains what actions are possible, the second fundamental principle of design, signifiers, explains where the action should take place. As explained by Norman, good design requires clues which help people understand what is happening in the product and what the controls are used for. The signifiers provide these clues and communicate how the product or a specific control should be used. An example of a simple signifier is the word ”push” on a door. The third principle is constraints. Constraints are clues which limit the number of possible actions and thus help people decide how they should proceed when using products or features. The fourth principle, mapping, explains the relationship between different elements. An example of mapping is specifying which light switch is connected to which lamp in a room. The fifth principle is feedback. Feedback is letting the user know that his or her actions are acknowledged by the system as well as communicating the results of the action back to the user. It is important for feedback to be immediate and informative. Furthermore, it is important that feedback is planned in such a way that all actions must be confirmed without being annoying or too discrete. The final principle is a conceptual model, which is a simplified model of how something works. Examples of conceptual models are icons of folders and files on your computer giving you an idea of a structure where files are placed into folders, making it easy to understand where to find different documents. There are no actual folders inside a computer but the made-up folder structure makes it easier for the user to use the system.

3.1.1 Discoverability

The first fundamental principle of design, discoverability, is one of the most important characteristics of good design according to Norman [44]. This concept can be further analyzed by looking into the First Principles of Interaction Design, presented by Tognazzini [46], who describes important guidelines to follow in order to design and implement effective interfaces. Some of the principles in the discoverability category of the first principles of interaction design are presented below.

- *Any attempt to hide complexity will serve to increase it.*

  This means that while it is important to design nice-looking products, one should never hide the controls needed to use a product in order to make it look more simple to use. By hiding necessary controls in order to maintain a minimalistic design, the product often becomes more difficult for the user to use.

- *If the user cannot find it, it does not exist.*

  This principle sheds light on the fact that users generally do not want to have to search for a functionality within a product; the functionalities should instead be easily accessible to them. Hiding functionalities is therefore usually not a good idea according to Tognazzini.

- *Use active discovery to guide people to more advanced features.*

  Active discovery is about helping users find features they need within a system by offering them to them. This can be implemented by mentioning to the user that a specific functionality exists at the time you think he or she might need it. This can be done by creating message hints to the users which pop up when the user launches the product or visits a specific page. However, if most users are turning off the hint functionality, Tognazzini explains that this is an indication of the hint being displayed too soon, before the user needs it.

- *Controls and other objects necessary for the successful use of software should be visibly accessible at all times.*

  This principle is quite self-explanatory, but there are two exceptions mentioned by Tognazzini. The first one is that the screen is so small that it is impractical to follow the principle, and the second is that it is impossible for users to avoid finding this functionality by accident, meaning they will eventually find it themselves.

- *User-tests for discoverability.*

  In order to ensure that the product successfully enables discoverability among users, Tognazzini explains that it is important to conduct user-tests covering this principle of design on a regular basis.

3.2 Hidden Treasures and Wow Factors

All small changes might not always be appreciated by all kinds of users. The *Wow factor* is defined by Kraft [3] as a change that will bring a smile to the user, exceed their expectations, is in the open and causes no pain for the user. This can in turn help users accept new changes. There are different ways to create these improvements, for example through visual layouts and animations, placing small *treasures* in the product or creating intelligent solutions that will surprise the users. Hidden treasures can be shortcuts to often-used functionalities. Finding a quicker way to perform a task is often considered to be a positive experience. However, as stated by Kraft, the designer must be careful when hiding treasures. One should never hide functions that are needed to accomplish a core task, in order to avoid the risk of creating a pain point for the user.

3.3 User Needs

The importance of understanding the users' needs is also pointed out by Kraft [3]. He distinguishes three types of user needs: *immediate*, *perceived* and *latent* user needs. The immediate user needs are needs that are most wanted. These are needs that the user expects a product to supply. However, this kind of need can also easily change. Furthermore, immediate needs are important to take into consideration when users are visiting a new website and deciding on whether to stay on the website or not. On the other hand, when it comes to perceived user needs, the designer has to be careful. A perceived need is often superficial and created by the market, making users think they have a need for a special product when they actually do not. Perceived needs may set the user's expectations too high, which in turn can ruin the first impression of the product and also the long-term user experience. Lastly, latent user needs are both the kinds of needs that the user already has, but he or she is not aware of, and the needs that do not exist yet. It is explained by Kraft that latent needs are rarely expressed by the user; they are instead captured by the observer. Latent needs are essential for user experience innovation and they can make a company more competitive on the market.

3.4 Contextual Inquiry

A contextual inquiry is a data gathering technique where the researcher observes how the user uses the system in order to get a better understanding of how the system works. It is stated by Holtzblatt et al. [47] that people are often able to explain in general terms how they use a system but that it is difficult for them to explain in greater detail why they interact in a certain way with the system and what workarounds they use to overcome any issues that may occur. The things that people do everyday easily become habits which are done automatically, and according to Holtzblatt et al., this may lead to people experiencing difficulties in explaining these processes to a researcher. Contextual inquiry is a good method to use in these situations as it helps the researcher understand the system by observing how the user interacts with the system, which is something that the user sometimes finds problematic to explain with words.

4 Related Work

In this section, we present work which has been published in the field of pattern recognition, user behavior and interaction design. We discuss similarities and differences between our study and the presented articles.

Tracking user behavior in softwares and applications is not new but has primarily been performed in web applications. The method of analyzing customers' patterns in web applications is, as previously mentioned, called web mining. Web mining is a procedure for extracting information from websites on the internet [48]. Web mining may consist of *web structure mining*, *web content mining* and *web usage mining*. Web usage mining, which is relevant to the present study, is the web mining process which involves extraction of activities of the users from web log files and click streams. Web usage mining can be divided into three phases: data pre-processing, pattern discovery and pattern analysis. Data pre-processing is an essential phase and must be performed before implementing data mining methods. The second phase, pattern discovery, consists of identifying patterns in the pre-processed data by using data mining methods such as clustering. Lastly, the final phase includes inspection of the patterns and extraction of meaningful conclusions. In our study, we apply the method for web usage mining explained by Sathya et al. [48] to user data from a mobile bank application. This method can be visualized in figure 8 below.

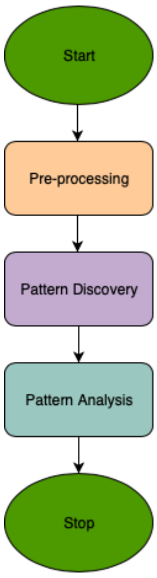


Figure 8: Process for web usage mining [48].

Work on user behavior with a similar purpose to the one in this study has previously been published by Zhang et al. [49]. The authors cover the topic of creating data-driven personas based on hierarchical clustering. In their study, they gathered data from user click streams on a platform in order to create data-driven personas which are based solely on user behavior. Taking our inspiration from to this approach, we wish to conduct a data-driven study based on user behavior within an application. In contrast to the procedure explained by Zhang et al., we use data in order to gain knowledge about what functionalities are being used within a mobile application and not for the purpose of creating data-driven personas. Furthermore, Zhang et al. investigate which types of click streams are common amongst different users, and the users with similar click streams are represented as a particular persona. However, in this thesis we instead focus on understanding *how* these functionalities are used together in order to find patterns in user behavior and give recommendations regarding altering the application based on the revealed user patterns. Thus, we do not take the individual users into consideration and are not tracking which user triggered what actions, but instead we focus on what actions are triggered within the same session.

Whilst there are some studies covering the topic of user behavior analysis in mobile applications, several of these deal with understanding which applications are downloaded in different countries [50] or analyzing user reviews of a mobile application to understand their preferences [51]. There are not as many studies focusing on the user behavior within mobile applications. The articles found when researching the topic of pattern recognition in user behavior on a platform cover the topic specifically for web platforms. As a result of this, a need for a study based on web mining techniques for mobile applications has been identified.

Published works in the field of interaction design, such as Cooper et al. [45] and Garret [52], are relevant to the present study in that they discuss the importance of keeping the user in mind when designing products. Cooper et al. highlight the fact that the design target depends on the user; in other words it is important to pay attention to the users and keep in mind who the users are, what they are doing and what their goals are. The authors propose qualitative research methods such as interviews and user observations to help understand attitudes and behaviors of users. These research methods can help reveal what functionalities and actions users find necessary and easy to use as well as pain points which they identify when using the product. In contrast, in this thesis we explore the possibility of using a quantitative approach in order to find patterns in user behavior and gain understanding of how users are perceiving the mobile bank application.

5 Explanation of Chosen Functionalities

The mobile bank application consists of many different functionalities and to narrow down the scope of this study, we have chosen to focus on a few selected functionalities. The chosen functionalities of the bank application are visualized in the site map seen in figure 9. The site map shows how the different elements and functionalities of the application are connected. There are two versions of the bank application, one for the iOS operating system and one for the Android operating system. The site map and the explanation of the application below were made based on the iOS version of the application. However, the Android version is similar to the iOS version and holds the same functionalities. The few noticeable design differences between the iOS version and the Android version will be pointed out below. Appendix A contains mockups of the iOS version of the application in order to facilitate the comprehension of the description which follows below.

When logging into the bank application, the user will see the ”accounts” page, where his or her accounts are listed together with the cards that the user has connected to an account. The user will also see the balance of each account (see Appendix A: View over accounts). Furthermore the user can choose to see a custom image on the ”accounts” page, or they can swipe left to see a bar chart of the user's expenses categorized into different categories (see Appendix A: Bar chart of expenses). The user can click on this bar chart in order to see a more detailed view of their expenses and how the expenses have changed during the passed months. From the ”accounts” page, the user can click on each account to see the transactions that have been made from or to that account, together with upcoming transactions which will be processed within the upcoming 30 days. The user can also search among all transactions in a specific account (see Appendix A: Account details). Furthermore, on the ”accounts” page, the user can also click on a plus sign to create a new transaction. We call this the *standard way* of transferring money. When the user has clicked the plus sign, he or she is given the opportunity to scan an invoice or to manually fill the input fields (see Appendix A: Scan an invoice.). The user can also partially scan an invoice and fill in the rest manually. The user will also need to add the date on which he or she wants the transaction to be processed. Here, the user has the option to use the smart date-picker and choose from one of two buttons: ”as soon as possible” or ”28th”, both which act as short-cuts, or to fill in the date manually. The ”as soon as possible” button is pre-filled. The user can also do a long-press on ”28th” to change this short-cut button to the date of the user's choice. When the user has filled in all fields and clicks ”add”, the invoice can either be added to what is called the ”bundle”, or the user can choose to sign it directly. The ”bundle” contains all invoices which are yet to be signed by the user (see Appendix A: The bundle). From the ”accounts” page, the user can click on an icon next to the plus sign in order to view the ”bundle”. In the ”bundle”, the user can click on all transactions which he or she wishes to sign and sign them all at once.

On the ”accounts” page, the user can also choose to make ”quick transactions” to one of the user's own accounts. On an iOS device, this is done by swiping left on the account and on an Android device, this is done by clicking on the three vertical dots, called an overflow menu, on the right side on the account. This will reveal three circles with different amounts on them. One circle has 100 SEK on it, one has 500 SEK and the last one says ”free-select amount”. By dragging one of the circles to another account and dropping it on the preferred account, the user can easily transfer money to one of its own accounts (see Appendix A: Quick transfer). When the money has been transferred, the balance of the account to which money has been transferred turns green and shows the new balance.

On the "tools" page (see Appendix A: View of tools page), the user can choose to create a savings target and connect it to one of the user's accounts. On an iOS device, the tools page is accessible from the menu bar placed at the bottom of the screen. On an Android device, the same page is accessible by clicking on three horizontal dots, another type of overflow menu, in the upper right corner of the start page. A savings target is a goal which the user can set to get motivated to save money. The user can also create a savings target by first clicking on an account from the "accounts" page and then clicking the overflow menu in the upper right corner. After this, the user can click "create savings target" (see Appendix A: Creating savings target and send account details). When the user has created a savings target, a pie chart will pop up next to the balance of the account on the "accounts" page to which the savings target is connected, indicating how far away the user is from reaching the savings target. The same pie chart is also presented to the user if the user clicks on the account which the savings target is connected to. Furthermore, the user can also use the overflow menu to share an account number. This is done by clicking "send account details" to send the account number and clearing number by text message to a phone number of the user's choice.

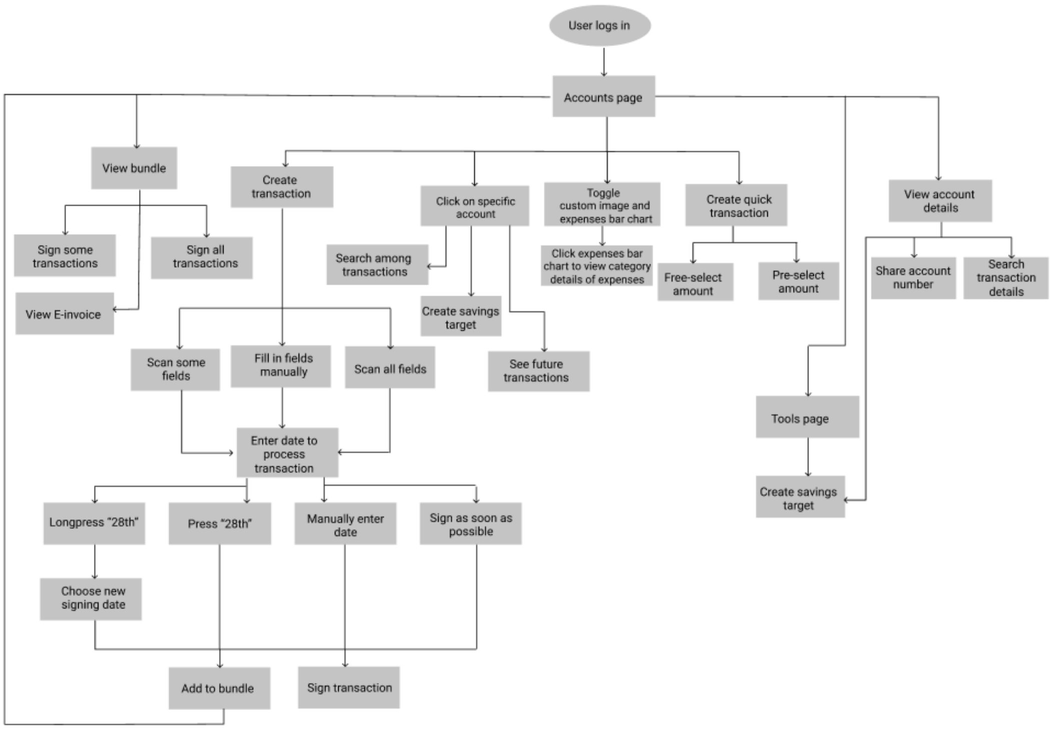


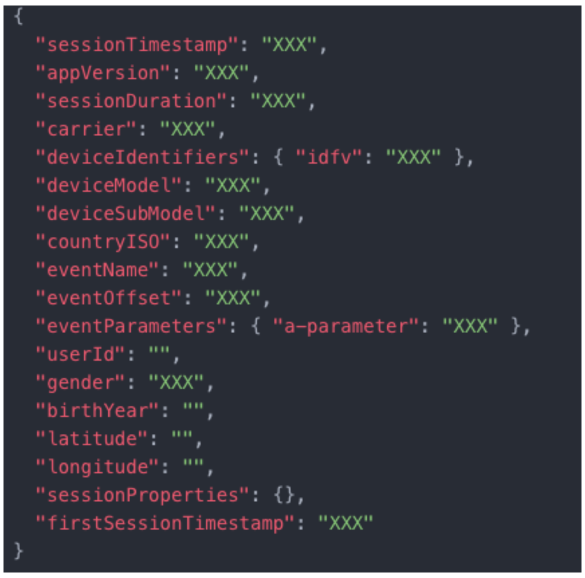
Figure 9: Site map of application.

6 Data

In this section we explain the structure and size of the data that was used in this study. The raw data was collected by the use of Flurry Analytics, an online tool which can be used to track user actions in applications. With the help of Flurry Analytics, one can keep track of how a specific application is used in real time by for example looking at how many sessions and devices are active at the same time, what actions users are triggering, how long the sessions in a set time period have been and much more.

When retrieving raw data from Flurry Analytics, the data is stored as JSON files where each file consists of one month of data containing usage of the application on an Android or iOS device. Each of the Android files are the size of approximately 50 Gigabytes and each of the iOS files are the size of approximately 150 Gigabytes. We chose to work with four months of data from each device from the year of 2018, which resulted in a total amount of eight files of raw data. Data from the months of March, May, September and October were used for each device. These months were chosen because they all are considered to be relatively similar to each other, since there are no big holidays, vacations or other major events happening in either of these months. Furthermore, this makes these months beneficial to use as there are no major happenings in the months which might affect people's spending habits and which therefore may also result in abnormal usage of the application compared to the other months of the year. The choice of only using eight files out of the 24 files from 2018 was based on the fact that 24 files would have generated a data set too large for the scope of this study. Even after reducing the number of files, we still had a large data set, in total consisting of around 800 Gigabytes of raw data. The size of the data implied that we were working with a big data set. Because of this, we had to take some extra measurements in order to be able to pre-process the data and run the clustering algorithms. How we went about this is explained further in the section 7.2.2.

The mobile bank application has multiple functionalities, as previously explained in section 5. When triggered by a user, these functionalities are logged and available for retrieval as JSON objects in JSON files. The user can perform tasks such as paying his or her bills, choosing to make transactions to other users or viewing his or her expenses within categories such as food, transport and shopping. Each action the user triggers within the application is logged together with other types of information about the user and its device. This results in one session being represented as multiple JSON objects if a user has triggered more than one action. Figure 10 gives a visualization of the structure of a JSON object. The data in each object is stored as key:value pairs. Each object consists of a *sessionTimestamp* of when the session was first launched. All objects belonging to the same session have the same sessionTimestamp, meaning that this attribute can work as a unique identifier for each session. Furthermore, the length of the session is also stored as the attribute *sessionDuration.* The version of the application which the user is using is also logged together with the user's telecom *carrier*, the user's *deviceModel* and the *deviceSubModel.* The deviceModel can either be an Android device, an iPhone or an iPad and the deviceSubModel gives further information of the device used. The *deviceIdentifiers* work as identification for every unique user using the application. The deviceIdentifiers attribute can thus assist in keeping track of all sessions belonging to the same user. The *countryISO* is a country code of the country from which the user is using the application. *EventOffset* is the number of milliseconds between the sessionTimestamp and the occurrence of the triggered action. Figure 10, shows a JSON object where the X's represent data and the empty strings indicate that this type of information is not being stored for this particular application. As visible in figure 10, information regarding the user's *birthyear* is not stored and neither is the geo-location represented in *latitude* and *longitude*. The *sessionProperties* key can hold information about the context of the session, but this is not stored either.



```
{
  "sessionTimestamp": "XXX",
  "appVersion": "XXX",
  "sessionDuration": "XXX",
  "carrier": "XXX",
  "deviceIdentifiers": { "idfv": "XXX" },
  "deviceModel": "XXX",
  "deviceSubModel": "XXX",
  "countryISO": "XXX",
  "eventName": "XXX",
  "eventOffset": "XXX",
  "eventParameters": { "a-parameter": "XXX" },
  "userId": "",
  "gender": "XXX",
  "birthYear": "",
  "latitude": "",
  "longitude": "",
  "sessionProperties": {},
  "firstSessionTimestamp": "XXX"
}
```

Figure 10: Structure of one JSON object in the raw data.

Since the purpose of this thesis is to understand the user behavior in a mobile bank application and what functionalities are being used in what way, the most interesting part in the data is the attribute called *eventName*. In the application, all actions that a user triggers are stored. It can for example be clicking on a specific button or visiting a specific page. The site-map in figure 9 in section 5 shows all possible actions which are of interest in this study. Moreover, each eventName has corresponding *eventParameters* which provide additional information about the action being triggered. Together, the values for the eventName and eventParameter keys provide information of what functionality the user has used in the application. A user can trigger multiple actions in one session, making it possible for the same session being represented by a large number of objects. This, together with the large number of customers who are using the application, are the reasons for why each JSON file consists of such a large quantity of data. Furthermore, the application has more iOS users than Android users, resulting in the iOS raw data file for one month of activity being three times the size of an Android file.

7 Method

In this study, we have worked with both qualitative and quantitative research methods. The quantitative methods are the primary methods used in the study whereas the qualitative methods act as a complement to the quantitative methods. The qualitative methods were conducted before the quantitative methods to get a better understanding of the workings of the application and because of this, the qualitative methods are presented first. The qualitative methods consisted of drawing inspiration from the contextual inquiry method to gain understanding of how the application is used, conducting two semi-structured interviews and testing of the chosen application in order to get acquainted with the application and its main functionalities. Section 7.1 clarifies the usage of these methods more thoroughly. Moreover, the quantitative methods, presented in section 7.2, consisted of data cleaning in the form of outlier detection, data integration with the purpose of removing redundant attributes, data reduction in the form of feature selection, as well as data transformation by standardizing the values in the data set. This was followed by the use of PCA, the primary clustering method K-means and the additional clustering method HDBSCAN.

7.1 Qualitative Research Methods

Two qualitative research methods were performed to understand how the application was built, and how data from users using the application is collected. Observing how developers use the application and asking questions to designers and a project manager, who have worked on designing the application and who decided what data should be collected, was deemed the most efficient way of learning the workings of the application.

7.1.1 Contextual Inquiry

Inspiration was drawn from the traditional contextual inquiry method explained by Holtzblatt et al. [47] in order to better understand how the bank application is designed and how it is meant to work. Since we had no previous experience in using the bank application, it was important for us to obtain as much information as possible regarding how the user interacts with the system, both the habitual tasks and workarounds users may use. For this, we met with a developer of the mobile bank application, whom we observed while he was using the application to perform actions in order to fulfill different goals.

7.1.2 Testing the Bank Application

In addition to the contextual inquiry we tested the bank application on our own to get acquainted with the application. Before we tested the application, we had received a document containing all attributes of the data set, meaning we knew about all actions that a user can trigger and which Flurry Analytics stores as raw data. When testing the application we tried as many functionalities as we could to get a good overview of what the main functionalities of the application are.

7.1.3 Semi-Structured Interviews

Furthermore, two semi-structured interviews were conducted. The first interview was held at the Bontouch office with a UX-designer for the bank application and a project manager. The second interview was held with the Design Lead for the bank application. The questions which were asked during the interviews can be found in Appendix B and Appendix C. Both interviews were conducted in hopes of gaining even more insight as to how the application works and how to interpret the data being collected from the application. In preparation for the interviews, we wanted to formulate a few general questions which could act as guidelines for the conversations but at the same time we wanted to remain open minded in regards to unexpected topics and discussions which may arise during the conversation. Bryman [53], discusses the fact that semi-structured interviews allow the interviewer to ask follow-up questions regarding things that the person being interviewed might have said. As the semi-structured interview method would allow us to prepare a few questions in advance but also to ask follow-up questions which could arise during the conversation, it was considered the superior interview method to use in this study.

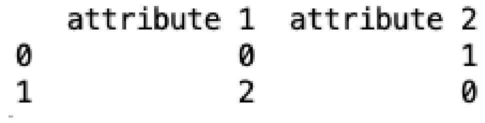
The participants for the interviews were found using the snowball method. The snowball method is a way of finding people to interview, by asking already chosen interviewees for recommendations of people who they know and think could be of interest to interview as well [53]. In our study, the UX-designer suggested that we should interview the Design Lead in order to get all our questions answered. The snowball method helped us broaden our networks and find new people to contact for the purpose of gathering information for this study. This method does not lead to a representative sample of an entire population [53]. However, as the purpose of the conducted interviews was to gain information about the bank application from a few knowledgeable people, all belonging to the same network, the snowball method was considered a good approach.

7.2 Quantitative Research Methods

The main research methods used in this study are all quantitative methods. These methods were used to fulfill the purpose of this thesis: to explore whether it is possible to conduct a data-driven study in order to gain understanding of user behavior within a mobile application. In the following sections we deliberate on the tools which were used in the quantitative research methods are presented. Furthermore, we explain how we used the data and took the handling of large quantities of data into consideration. After this, we clarify how the data has been thoroughly pre-processed. Lastly, we describe how PCA and two clustering algorithms were performed in order to gain knowledge about how the different functionalities are being used together in the application.

7.2.1 Tools Used

All code used in this study was written in the programming language Python. The machine learning library for Python, called Scikit-learn, was used in order to enable running machine learning algorithms such as PCA and one of the clustering algorithms used in this study. The library Matplotlib was used to enable visualization of PCA and the clustering algorithms, as well as plots needed for pre-processing. Furthermore, we used the Python library Pandas which provides data structures called DataFrames. DataFrames are used for data manipulation and are also considered fast and efficient to use [54]. Figure 11 shows the structure of a DataFrame, where a row corresponds to one data point and the columns correspond to a data point's characteristics. A DataFrame also has integrated indexing.



```
   attribute 1  attribute 2
0            0            1
1            2            0
```

Figure 11: Structure of a DataFrame.

With Pandas, it is also possible to read JSON files to in-memory DataFrames. In order to find some algorithms which were needed for clustering but were not available in Scikit-learn, we used the Python Package Index, PyPI, which is a repository for Python Software.

7.2.2 Handling Large Data Sets

Because of the considerable size of the original data set, we have had to be cautious when working with it. Firstly, the four months of data from both Android and iOS users were stored in eight files which were divided up into two groups. Because of the computer's storage space not being large enough to store eight files at once, the files were stored and pre-processed four at a time.

Moreover, as the computer's RAM was not sufficient for reading each file in the memory at once and parsing it into Python as a JSON object, we used the Pandas library for Python. With Pandas, we could do progressive loading and read our files incrementally using chunks, a technique better suited for larger data sets. The chunk size determines how many lines in a file we read at a time. Using chunks of 10 000 lines enabled us to read our large data files more efficiently and it took up less memory than reading each file all at once.

Another approach we used during this study was working with a smaller subset of the raw data before handling the complete raw data files. This was done in order to work efficiently and get results quickly, something which would not have been possible using the raw data files due to the long run time. This helped us understand if our pre-processing models did what they were intended to do. When reasonable results were achieved, we applied the pre-processing models on the complete raw data files.

7.2.3 Pre-processing of Data

Before clustering was possible, the data had to be pre-processed in order to ensure high quality within the data set. The pre-processing consisted of feature selection and handling of redundancy, outlier detection and standardization, thus covering the four steps of pre-processing explained by Han et al. [11]: data reduction, data integration, data cleaning and data transformation. In this section, we explain how we went about pre-processing the data before applying the two clustering algorithms onto our data set. Pre-processing was conducted using DataFrames. The initial step of pre-processing was to read the raw JSON data files using Pandas and storing the information in DataFrames. As we read our files incrementally, we used one DataFrame per chunk.

Feature Selection

Feature selection was conducted to avoid issues with curse of dimensionality, reduce run time and only keep attributes that were relevant for this study. As there is no universally accepted method for feature selection within the field of unsupervised learning, we mainly relied on knowledge of the data which had been gained during interviews with designers of the application when selecting relevant attributes. Furthermore, when performing feature selection we used the framework for relevance and redundancy analysis by Yu et al. [22]. A description of how this framework was used follows below.

Performing a relevance analysis is the first step of feature selection [22]. Interviews with designers and developers for the application were held to learn more about which features were relevant for this study. After the interviews we learned that, in addition to information about the users' actions, other information was also stored in Flurry. This included information about the camera in the users' devices, the country from which the user is using the application, as well as information about the number of accounts and cards that are connected to each user. These attributes were removed due to us considering them irrelevant in this study.

Originally, all triggered actions by users were stored in the eventName column in the DataFrame, with the corresponding parameters stored in the eventParameters column. However, not all values within these columns were relevant for this thesis. During the interviews, we gained knowledge regarding the main functionalities of the application which might be interesting to study further. Based on this knowledge, we could reduce the number of values in the both the eventName and eventParameters columns, focusing on the most frequently triggered actions, as well as the actions we knew were part of the key functionalities within the application, all mentioned in section 5. These functionalities are making a money transfer, scanning and paying an invoice, using the available short-cut functionality such as using the smart date-picker when signing an invoice and making a quick transfer to the user's own account, signing all invoices at a time as well as tracking what pages the user visits within the application. By choosing these key features, several irrelevant attributes could be removed, a procedure also in accordance with the relevance analysis in the framework explained by Yu et al.

Furthermore, it was decided to only use data from Android devices and iPhones, and remove sessions which were launched by iPad users. The choice to remove iPad data was made in order to reduce the large data set further. The actions triggered from iPads had eventNames only valid for iPad, meaning that these eventNames could not be triggered from an Android device or an iPhone. Keeping the iPad eventNames would have resulted in a large increase in dimensionality and sparsity in the total data set as the sessions not launched from an iPad would be populated with zeros for all iPad attributes. To avoid issues with curse of dimensionality and sparse data, it was decided to not use iPad data and keep the dimensionality as low as possible.

After a relevant subset had been extracted, a redundancy analysis was performed. This is the next step in the framework by Yu et al., and an important part of data integration. We created a new column in the DataFrames called eventArray, which contained the concatenation of each triggered event and its corresponding eventParameter. The eventName did not always provide much information about what action was actually triggered by the user and thus, the eventParameters helped provide additional information about the action. In some cases the data set contained multiple triggered actions with the same eventNames, but with different eventParameters. In these cases the eventParameters helped differentiate between eventNames of the same kind. After this, the original eventName and eventParameter attributes could be removed from the DataFrames. Handling redundancy in the attributes is an important part of pre-processing [22], and removing these attributes after concatenation was done in order to avoid redundancy issues.

The eventArray contained the attributes considered interesting in this study. When going over these attributes, it was noticeable that some attributes were very similar to each other. In these cases, the only differences between the attributes were the values of the eventParameters. However, keeping these similar attributes separated was not always necessary. It was sometimes deemed more beneficial to store some or all attributes with the same eventName as one attribute, despite the eventParameters being different. This choice was made due to the eventParameters not contributing with beneficial information in these cases but instead causing redundancy in the data. One example of this is that some actions had the same eventName but different eventParameters depending on if the action was triggered from an iPhone or an Android device, resulting in two attributes explaining the same action. In the cases where this happened, the Android and iPhone attributes for the same action were concatenated into one to avoid redundancy. The final chosen attributes make up what Yu et al. refer to as the final selected subset and we also consider this a complete data set.

Data Transformation

Using progressive loading and DataFrames, we wanted to convert the values of the eventArray into columns in the DataFrames. By creating new empty DataFrames both containing all attributes in the eventArray and the additional identified relevant columns as headers, we were able to maintain DataFrames in which each unique session was represented by one row. For each session, the attributes were populated with the number of times that each attribute had been triggered during that session. This allowed us to get a good visualization of which attributes had been triggered during each session and ensured that the data was consistent. However, modifying the data in this manner generated quite a sparse data set.

The data we received also contained information regarding the type of device each user was using during a session. As the application looks slightly different depending on whether the user is using it from an Android device or an iPhone, it was considered interesting to investigate whether the choice of device would affect other actions that the user triggers within the application. Therefore, we wanted to save this information for each session. However, in order to be able to use this information when clustering, we had to convert the information. The information that was originally stored as categorical values had to be converted to binary values with the help of one-hot-encoding. We created two new attributes within each chunk, named Android and iPhone. After this we added the number "1" as a value to the attribute that the user was using in each session and "0" to the other. Thus, we could keep track of the devices the users were using during each session.

Finally, each DataFrame chunk was added to a file incrementally, as the data set was too large to add to the file all at once. When all chunks had been added to the file, this file contained the complete pre-processed data set. Furthermore, the data was standardized by each column in the DataFrame. The standardization was performed with a mean of zero and a standard deviation of one to ensure that all attributes were equally as important within the data set. This was done to avoid some attributes having greater impact on the clustering than others.

Outlier Detection

In addition to performing feature selection, we also kept in mind that outlier data points could potentially affect the clustering configurations negatively [24]. Furthermore, outlier detection is also identified as an important procedure in the pre-processing step data cleaning [11] and can lead to a higher level of accuracy in the data [24]. Therefore we wanted to find a way to detect these outliers and potentially remove them from the data set. We decided to analyze the number of actions that had been triggered during each session to see what the most common number of triggered actions in one session was, as well as detect sessions where untypically large frequencies of actions had been triggered. This was done by calculating the number of actions triggered for each session and creating a bar chart where the total sum of triggered actions during each session was plotted against the frequency of each sum. For example, if five actions had been triggered in ten unique sessions, the plotted values would be five on the x-axis and ten on the y-axis.

From the bar chart we gained knowledge of how the number of session clicks were distributed and thus, we were able to decide what threshold to use as the maximum number of session clicks that should be allowed in the final data set. The data points which had a higher frequency of session clicks than the threshold were removed from the data set before further modelling was made. The result of the outlier removal is presented in section 8.2.

7.2.4 Sample a Subset

After performing feature selection and outlier removal, our data set was still of a large size which implied that the resulting plot after clustering millions of data points would be difficult to understand. In order to solve this problem we chose to work with smaller subsets from our data set when plotting the results. Different sample sizes were used for different purposes, further explained in section 7.2.5 to 7.2.8.

Small percentages of the data set were sampled at random with every run where a sample was needed. For example, samples of the pre-processed data set were used when calculating the best value for the number of clusters to use by using the Elbow Method and Silhouette Scores, which are further explained in section 7.2.6. Moreover, a small sample of the total data set was also used for plotting the result of the PCA, the K-means clustering algorithm and the HDBSCAN algorithm, explained in sections 7.2.5 to 7.2.7.

Smaller samples of the total data were primarily used to avoid the long run time when plotting and analyzing a large amount of data such as the complete data set. Small percentages of the total data still represent the original data set well since for example one percent of the data translates to hundreds of thousands of data points. Using a small sample of the data set was especially important when running HDBSCAN and calculating the occurrences of the attributes in each cluster, since these computations had much longer run times than other methods in this study.

Another reason to use a smaller percentage of the data set is because a visualization of a data set containing millions of data points most likely does not result in a valuable plot on which any analysis can be made. As the number of data points increase, it will be more and more difficult to see the individual data points, as everything blurs together.

Due to different samples being used with every run of the different algorithms, the resulting plots of the data points might look slightly different depending on which data points were drawn at random to be included in the sample.

7.2.5 Principal Component Analysis

In order to reduce the dimensionality further and visualize the data points in a graph, we decided to perform PCA, part of the data reduction pre-processing step [11]. PCA helped us find three principal components, linear combinations of the attributes. Together, these explained a substantial amount of the variation in the data set. Because of our principal components being linear combinations of multiple attributes, we could keep information about the characteristics of the data points but reduce the dimensionality of the data. However, when using Scikit-learn, PCA has the limitation of only being able to handle data sets which can fit into main memory. Our pre-processed data set was larger than what the PCA algorithm could handle and therefore we used *Incremental PCA*, or IPCA, which processes the data incrementally. As IPCA gives similar projections to PCA and works to reduce dimensionality [55], it was considered an appropriate tool to use in this case.

Before performing PCA, the data was standardized by columns. Furthermore, after analyzing the results of performing PCA using standardized data, it was decided to also conduct PCA on non-standardized data in hopes of improving the results.

When conducting PCA, we also generated scree plots for the standardized and non-standardized data sets. The scree plots were used to provide a visual representation to help understand the amount of variation which each of the principal components explained. Since the principal components were used to visualize the clustered data in three dimensions, only the three first principal components were used for further analysis. For each of these three components we extracted the loading vector. This gave us an indication of which attributes had the largest loading coefficients and therefore had the biggest impact on each of the principal components. The attributes were sorted in descending order to facilitate comprehension regarding which attributes were the most important ones.

After analyzing the result of PCA for the standardized and non-standardized data sets respectively, we plotted a standardized subset of one percent of the data set onto the first, second and third principal component in order to get a visualization of the distribution of the data along these three axes.

7.2.6 Performing K-means Clustering

Before performing K-means, we investigated the number of clusters, K, which we should use. This was done in two ways, using the Elbow Method and Silhouette Scores. We implemented a function which used the Elbow Method and resulted in a plot showing the elbow criterion, and thus giving us the best value for K. The Elbow Method was implemented by calculating the within-cluster sum of squares for each K in the chosen range of two to ten, using one percent of non-standardized data. The best number of clusters to use was the K for which adding another cluster did not give much lower within-cluster variation. By plotting the number of clusters against the within-cluster sum of squares and finding the point in the graph where the within-cluster sum of squares rapidly drops, we were able to find the elbow criterion.

To verify the choice of K generated from using the Elbow Method, the Silhouette Score method was also implemented. The results obtained from the Elbow Method gave us an indication of how many clusters would be of interest for this study. To perform the Silhouette Scores method we had to pre-define a range for the number of clusters to be used. We used the same span, two to ten clusters, as when conducting the Elbow Method. Furthermore, we used a method in the Scikit-learn library to calculate the average of the Silhouette Score for the chosen values of K. Finally, we plotted the average score against the clusters, once again using a sample of one percent of the non-standardized data, which resulted in a line graph. By looking at the graph's maximum value, we could gain understanding of what the best number of clusters to use was. The final choice of K for K-means clustering was based on the results generated by the two methods.

After retrieving the best number of K, we used this value when running the K-means algorithm. The K-means algorithm was run using the final selected data set, after standardizing the data to ensure that all attributes were of equal importance. The K-means algorithm was run on the entire data set using the K-means function from Scikit-learn. However, memory error issues occurred when running the K-means algorithm, indicating that there was not enough RAM on the computer to execute the code. This caused us to use Mini Batch K-means as an alternative approach. The chosen batch size was set to 10 000. Scikit-learn's MiniBatchKmeans function also has a parameter for how many random initializations should be tested in each run when performing the algorithm, and this parameter was set to ten. The best of the initializations in terms of smallest within-cluster variation was then chosen by the algorithm for further use. For each initialization, the initialization centroids were chosen using the kmeans++ algorithm. Henceforth, the Mini Batch K-means algorithm is referred to simply as K-means.

Due to the number of attributes in the data set, we could not plot the data on all dimensions. However, K-means can be combined with PCA in order to plot a data set containing more than three dimensions. This was done by choosing the first, second and third principal component vectors as the x-, y- and z-axis in a scatter plot, and the K-means cluster labels as the data which was to be visualized.

7.2.7 HDBSCAN

In addition to K-means, the density-based clustering method HDBSCAN was also used to cluster the data points. Since HDBSCAN does not require any pre-defined number of clusters and is not as sensitive to outliers as K-means [37, 40], it was considered a good alternative to K-means. Furthermore, as HDBSCAN can handle clusters of varying densities [40] and provides parameters which were considered easier to manage, it was decided to use this method instead of DBSCAN. By clustering the data points using HDBSCAN, we were able to gain insight as to how another clustering method would cluster the data points and get a nuanced understanding of how the data points could be formed into different clusters.

The HDBSCAN clustering method was implemented using the HDBSCAN method in Python, downloaded from PyPI. When using this method, the two parameters minimum cluster size and minimum samples had to be defined. Using these parameters, we were able to define approximately how many data points which were needed in one cluster in order for it to be classified as a cluster as well as how sensitive to noise the clustering method should be. Several different values for these parameters were tested in order to find a good threshold for where most of the data points were clustered but the method still resulted in reasonable cluster formations according to what was to be expected from analyzing the data points plotted against the first three principal components. Finding the optimum values for these parameters required many trials with values for the minimum cluster size ranging from 5000 to 10 000 and the values for minimum samples ranging from 1 to 100. We did not want the clustering to be very strict and we wanted the minimum cluster size set to a value which would approximately result in a number of clusters which would make the results of HDBSCAN comparable to the results of K-means. Furthermore, both Euclidean distance and Manhattan distance were used in order to see if the choice of distance metric would change the clustering outcome. Because of the long run time of HDBSCAN, only one per mille of the pre-processed data set was used.

7.2.8 Analysis of Cluster Formations

The visualizations of the clustering methods were used to gain understanding of whether the algorithms could produce distinct clusters. However, the cluster labels assigned to each data point works as the main result used for further analysis. After clustering and visualizing the clusters, the sum of all occurrences of each attribute were calculated for each cluster using one per mille of the clustered data set. The cluster formations of the primary method K-means were used to analyze the occurrence of attributes in each cluster. By calculating the sum of each column for each cluster formed using K-means, it was possible to analyze the frequency of the different attributes in the different clusters. This enabled us to better understand what functionalities were most frequently used in each cluster.

Furthermore, a sample of ten data points from each cluster was saved in a new DataFrame. Values for all attributes which were used for clustering were stored together with the corresponding cluster label for each data point. Extracting a sample of data points together with their corresponding cluster labels made it possible to further analyze the result of the K-means clustering method and discover patterns both within and between the different clusters. However, as 10 points from each cluster is small sample of the complete data set and thus cannot represent the entire data set, the extraction of this sample primarily worked to complement the methods mentioned above.

7.2.9 Additional Tests

In addition to the previously described methods, we ran some additional tests in hope of improving the results. The changes explained below were implemented one at a time to make sure that a potential change in the result could be credited to one specific implementation.

Firstly, as the attributes Android and iPhone were one-hot-encoded, we had suspicions that these could potentially affect the K-means clustering negatively. They both only contained binary values, making us concerned that the clustering algorithm would interpret them as labels for the data points instead of numerical values, and thus only cluster according to these attributes. Therefore, we ran K-means clustering without including these two attributes.

In addition to using the joint result of the Elbow Method and Silhouette Scores when choosing a value for the number of clusters to use, we also ran the K-means algorithm for additional values of K which either the Elbow Method or Silhouette Scores result showed was a good choice but the other method did not.

When running K-means, one can use different methods to initialize the K centroids needed before clustering is possible. Scikit-learn offers three different methods: kmeans++, randomized centroids and a pre-specified matrix of the centroids' positions. In addition to using kmeans++, we decided to run K-means by randomizing initial centroids in order to see if the choice of method for initializing the centroids would affect the clustering.

When using the K-means algorithm in Scikit-learn, one can also define a number of maximum iterations over the complete data set that should be allowed before the algorithm stops automatically, regardless of whether convergence has been reached or not. This number is set to 100 iterations by default and in order to assure that the algorithm was not stopping before convergence, we ran additional tests where we changed this to 10 000 iterations.

7.3 Sources of Error

In this section we highlight the possible errors made when conducting this study and the potential results of these errors. Since our method consists of two processes, one qualitative and one quantitative, there is a risk of error occurrence in both processes.

7.3.1 Sources of Error in Qualitative Methods

When using contextual inquiry, Holtzblatt et al. [47, p.11] state that "by capturing issues and modeling each individual user's experience the team records the data that will later be consolidated to build a coherent view of the practices and experiences of the whole user population." This implies that the findings made from a contextual inquiry may be applied to the entire user population in order to gain a better understanding of the usage of the application. In our case, it is important to note that because of the developer being an expert in using and understanding the bank application, the findings of the contextual inquiry are not representative of the whole user population. The developer can be considered more knowledgeable compared to the general population that uses the application as he has been involved in building and testing the application. However, gathering information of how the application works and how it is used by an expert was still considered important as it gave us an insight into how the application can be used and what the available functionalities are. Conducting a contextual inquiry with an expert also improved our understanding regarding the limitations of the application. Furthermore, the knowledge of the application that the developer had can be considered similar to the knowledge of other developers working on the same project in the company, as they all work on the same application on a daily basis and share knowledge between them. The result of the contextual inquiry is therefore considered to be representative of the group of developers working on the bank project at Bontouch.

7.3.2 Sources of Error in Quantitative Methods

As mentioned in section 7.2.3, the quantitative process included data pre-processing, which implies modifying the data set. Since our data initially had such a large amount of attributes and rows, we wanted to remove the data set's unnecessary and irrelevant attributes. In order to do this, we have carefully gone through the data and discussed the attributes with people from Bontouch who have more experience with the data. However, it is possible that some attributes which were removed could have lead to a more distinctive formation of clusters generated from using K-means and that some of the attributes which were kept do not contribute to the cluster formations.

Furthermore, only four months worth of data from the year of 2018 was used in this study. This choice was made as it was considered outside the scope of this study to use all data available, since this would have resulted in the thesis being even more dependent on using appropriate tools for handling big data. There are many new issues which arise when using large data sets, such as extremely long run times. Because of this, the data was reduced to a manageable size. Still, reducing the data set most likely affected the result of this study as a larger data set would have resulted in a more conclusive result that could have been applied to a larger population. However, due to the scope of this study, we believed that data from four months was enough since these files consisted of 800 Gigabytes of data in total.

Another source of error in this study is using progressive loading for handling large sets of data.

As we had access to a computer with 64 Gigabytes of RAM, it was initially thought that this would be enough to handle or data in an efficient way. However, in reality some computations were far too heavy to run using all data and therefore we had to settle for only using a sample of the data. A sample can act as a good representation of the large data set, but it would have been preferable to be able to take advantage of the large data set that was available. Furthermore, the usage of the large data set also lead to complications when running HDBSCAN. The time complexity of HDBSCAN together with the large data set which was used lead to unreasonably long run times. This resulted in the decision of only using one per mille of the data set when running HDBSCAN. However, only using a small fraction of the available data set is assumed to decrease the performance of the clustering method compared to the results which would have been obtained if all data had been used.

Furthermore, working with large sets of data can lead to a lower level of control regarding the pre-processing of the data. Because of the large size of the data set, it has not been possible to open and inspect the file containing the pre-processed data. This means that it is not possible to go over the pre-processed file manually to check whether there were any issues in the pre-processing. It is possible that some errors were made both when running the script to convert the JSON files to DataFrames and when populating the columns in the DataFrames with the number of occurrences of the attributes in each session. We can only, to the best of our abilities, verify that our implementation works in the way it should by printing a few selected data rows.

Two one-hot-encoded attributes were used as we wanted to store information regarding which device the user triggered the session from. These attributes could potentially have a negative impact on the clustering algorithms as K-means sensitive to binary data and one-hot encoded attributes can be considered to act as labels by the clustering method. In addition to this, there could be other attributes which are most often triggered either zero times or one time during a session. This would result in more attributes only consisting of zeros or ones as well, further exacerbating the result. As we cannot manually go over the data set, it is difficult to know how much binary data the data set contains, other than the two attributes we have modified to contain binary data. Regardless, K-means cannot handle data of mixed types [56], that is when a data set contains both categorical and continuous data, making one-hot encoding necessary in order to use information about the device model when clustering.

The final quantitative source of error lies in the formation of principal components. Our data set contained more attributes than could be visualized on three dimensions. Compressing the attributes to only three principal components meant that some information about the data points would be lost. However, this was done in order to visualize our results but the visualization cannot fully represent the actual cluster formations as we have projected all attributes in the final data set onto three dimensions. Projecting the data points also leads to difficulties in analyzing the characteristics of the data points by reading the plot. If the data set had contained three dimensions originally, the frequency of each attribute for one data point (which equals one session) would have been apparent directly from reading the graph as each attribute would have corresponded to one of the three axes. However, as we are using more than three dimensions, we are not able to use the visualization directly to analyze the characteristics of the data points. Instead, we proceeded as explained in section 7.2.8.

8 Result

In this chapter we present and discuss the findings from conducting feature selection and PCA, finding good values for the number of clusters, K, to use and finally running the K-means algorithm and HDBSCAN algorithm. By using the relevance and redundancy analysis framework by Yu et al. [22], we were able to remove irrelevant and redundant features and the final chosen attributes are presented in section 8.1. The bar chart used for outlier detection is presented in section 8.2. Furthermore, the results of the PCA using both standardized and non-standardized data are presented in section 8.3. Following this, in section 8.4 we present the findings from using the Elbow Method and Silhouette Scores to find good values of K. Furthermore, the clusters generated from running the K-means algorithm are presented in section 8.5. In table 7 in section 8.6, the frequency of each triggered attribute is displayed for each cluster created using K-means. In the same section in table 9, we present an extraction of data points which are analyzed in order to find interesting patterns both within and between the clusters. Lastly, the results of HDBSCAN are presented in section 8.7.

The additional tests in section 7.2.9 did not improve the results obtained using the primary methods explained in sections 7.2.3 to 7.2.6. Therefore, these results are not discussed further.

8.1 Feature selection

Feature selection resulted in a data set of 23 attributes and 64 174 510 rows. The attributes can be seen in table 1 below, together with an explanation of the meaning of each attribute. The functionalities explained by these attributes are also discussed in section 5. By using feature selection as explained in 7.2.3, we were able to reduce the data set from approximately 800 Gigabytes to seven Gigabytes. The 23 attributes which were kept in the final data set all describe important features within the application, except for sessionTimestamp which works as the ID for each row but was not used in computations. Henceforth, we therefore refer to the total number of attributes as 22 attributes.

Table 1: Description of attributes.

Letter Representation	Feature	Meaning
	SessionTimestamp	Timestamp of when session was launched. Unique for every session and therefore acts as an ID.
A	BANK_QUICK_TRANSFER_SUCCESS ’transfer-amount-category’: ’predefined-amount’	The user has made a quick transfer to one of its own accounts, using one of the pre-defined amounts.
B	BANK_QUICK_TRANSFER_SUCCESS transfer-amount-category’: ’free-select-amount’	The user has made a quick transfer to one of its own accounts, using a freely selected amount.
C	PAGEVIEW ’page’: ’accounts’	The user has visited the start page, the page for viewing the balances of all accounts.
D	PAGEVIEW ’page’: ’account-upcoming-events’	The user has visited the page for viewing all upcoming transactions for a specific account.
E	PAGEVIEW ’page’: ’bundle’	The user has visited the page for viewing the bundle.
F	PAGEVIEW ’page’: ’tools’	The user has visited the tools page.
G	PAGEVIEW ’page’: ’expenses-category-list’	The user has visited the page for viewing the user’s expenses sorted into different categories.
H	PAGEVIEW ’page’: ’new-task’	The user clicked the plus sign to create a new transaction.
I	BANK_TASK_OCR_USED ’scanned-fields’: ’true’	The user has used the camera to scan an invoice.
J	DASHBOARD_CHANGE ’result’: ’expenses’	The user has chosen to see the chart for its expenses.
K	DASHBOARD_CHANGE ’result’: ’none’	The user has chosen to see a custom image.
L	SHOW-E-INVOICE	The user has clicked the e-invoice button to view its invoice.
M	ACCOUNTS_DETAILS_SEARCH	The user has used the search bar to search among all transaction for a specified account.
N	SHARE_ACCOUNT_NUMBER	The user has clicked the button to share an account number with someone and the message has been sent successfully.
O	HAS_NO_ACTIVE_SAVINGS_TARGETS	The user has not set a savings target.
P	BANK_TASK_SMART_DATE_PICKER ’date’: ’28’	The user has used the smart date picker to set the date of when an invoice should be payed to the 28th day of the month.
Q	BANK_TASK_SMART_DATE_PICKER ’date’: ’earliest’	The user has used the smart date picker to set the date of when an invoice should be payed to as soon as possible.
R	BANK_TASK_SEND_BUNDLE	Logged when user presses the ”send” button to send ”bundle” directly from ”bundle” preview, after creating a transaction.
S	BANK_TASK_SUCCESS ’receiver-type’: ’other-account’	Logged when a bank task to another account was deemed successful. A bank task is when the user pays a bill or transfers money to an account.
T	BANK_TASK_SUCCESS ’possible-quick-transfer’: ’true’, ’receiver-type’: ’own-account’	Logged when a bank task to one of the user’s own accounts was successful. A bank task is when the user pays a bill or transfers money to an account.
U	Android	Logged when the session was triggered from an Android device.
V	iPhone	Logged when the session was triggered from an iPhone.

8.2 Outlier Detection

As mentioned in section 2.6.1, the primary clustering method K-means is sensitive to outliers. As a result of this, it was important to conduct outlier detection and potentially remove data points which might have negative impact on the cluster formations due to their deviating characteristics. Moreover, removing outliers reduced the size of the data set further. By calculating the sum of each row in the data set, we knew how many actions were triggered in each session. The sum of actions triggered were plotted against the occurrence of each sum for ten percent of the data set. By looking at the distribution of row sums, we can see how many actions are most often triggered in one session. In figure 12 we see that the majority of all sessions contain 2-20 clicks. However, there are still many sessions which contain more than 20 clicks but this is difficult to see from figure 12 as the y-axis covers a large span.

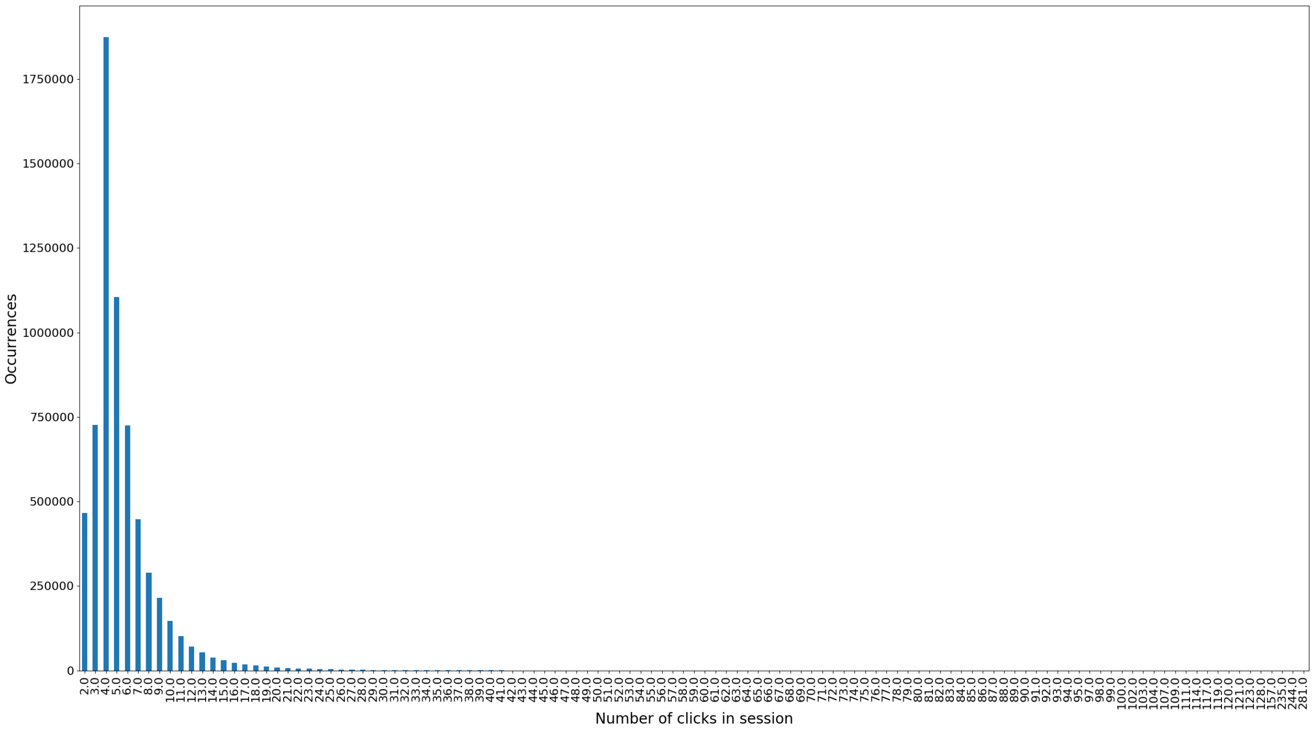


Figure 12: Bar chart for outlier detection using ten percent of the data set.

The x-axis in figure 12 consists of approximately 120 different values, all which are actual sums of clicks for one or more sessions in the data set. After examining the bar chart further it was decided to use approximately half of these values and setting the threshold to 60 clicks in one session. This would allow us to keep a big part of the original data but remove the data points which had very large totals of session clicks and could potentially have negative impact on the cluster groupings. After removing the outliers according to the threshold, the value of removed outliers totaled to 4067 data points. Out of the total number of data points, 64 174 510, removing 4067 data points is not considered a large enough quantity to affect the credibility of the clustering or impact the other computations in a negative way.

8.3 Principal Component Analysis

The PCA was conducted on both standardized non-standardized data. As apparent from figures 13 and 15, using the standardized data resulted in the principal components explaining less of the variation compared to the non-standardized data. However, when looking over the tables 2, 3, 4, 5 and 6, it is apparent that the attributes in the standardized data set have a more even distribution of the loadings compared to the attributes in the non-standardized data set. The complete results of these two analyses are presented below.

8.3.1 PCA on Standardized Data Set

A scree plot of the principal components' explained variance using standardized data is presented in figure 13. The PCA using a standardized data set shows that the first three principal components explain 13.0 %, 10.4 % and 7.1 % of the total variation in the data set. As we mainly used the PCA for visualization of our data points on three dimensions, we were not able to use additional principal components even though this would have lead to a higher total explained variation.

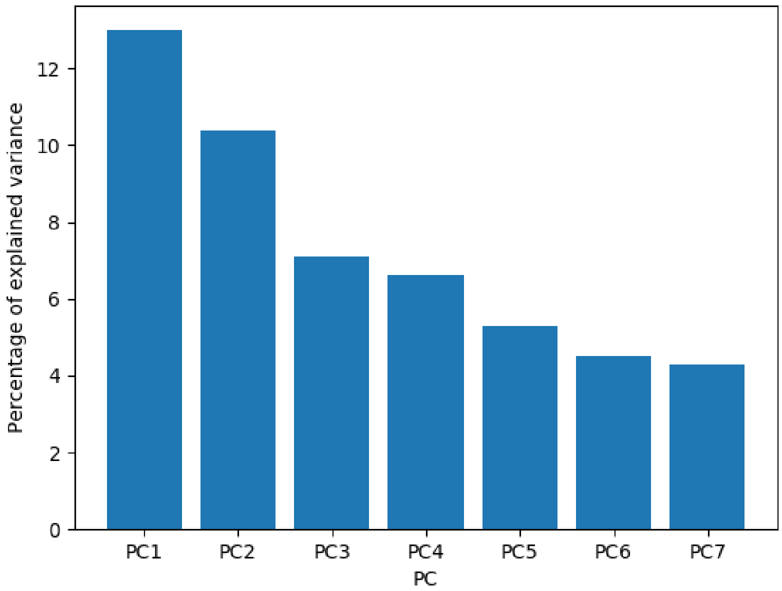


Figure 13: Scree plot over principal components using standardized data.

When conducting the PCA, the names and loadings of each attribute were extracted. This enabled us to see which attributes were of the highest importance in each principal components. In tables 2, 3 and 4, the names of the attributes together with their loadings are displayed for principal component one, two and three after conducting PCA on standardized data. A 0.2 threshold was set, meaning we only display attributes which have higher loadings than 0.2, as attributes with lower loadings were not considered to be of significant importance to the principal component.

It is important to mention the low variance explained by the different principal components, as seen in figure 13. Ideally, the variance should have been higher for the first three principal components in order to be more certain that the three principal components actually explained most of the variation in the data set. As seen in figure 13, the cumulative sum of variance explained by the first three principal components is approximately 30 %. This is considered quite low and is therefore an indication of the fact that the PCA did not result in what we consider ideal projections of the 22 dimensions onto a three dimensional space.

Moreover, table 2 shows that BANK_TASK_OCR_USED{'scanned-fields': 'true'} has the highest importance in the first principal component using standardized data. This attribute explains the that the user has scanned an invoice during the session. Furthermore, the completion of a transaction to another account, called BANK_TASK_SUCCESS{'receiver-type': 'other-account'} has the second highest importance followed by clicking the plus sign to create a new transaction, called PAGEVIEW{'page': 'new-task'}. Nevertheless, all attributes in the first principal component using standardized data have quite similar loadings.

Table 2: Attributes with corresponding loadings using standardized data for first principal component.

Attribute	Loadings
BANK_TASK_OCR_USED 'scanned-fields': 'true'	0.426003
BANK_TASK_SUCCESS 'receiver-type': 'other-account'	0.382732
PAGEVIEW 'page': 'new-task'	0.364724
BANK_TASK_SEND_BUNDLE	0.297440
BANK_TASK_SMART_DATE_PICKER 'date': 'earliest'	0.284284
PAGEVIEW 'page': 'bundle'	0.263267
DASHBOARD_CHANGE 'result': 'expenses'	0.256869
Android	0.240732
iPhone	0.240732
BANK_TASK_SUCCESS 'possible-quick-transfer': 'true', 'receiver-type': 'own-account'	0.219047

In the second principal component using standardized data, presented in table 3, whether the session was started from an Android device or an iPhone turned out to be most important. This is followed by the attribute DASHBOARD_CHANGE{'result': 'expenses'} which explains whether the user is using the feature of viewing one's expenses in a bar chart, where the expenses are divided into different categories, or not.

Table 3: Attributes with corresponding loadings using standardized data for second principal component.

Attribute	Loadings
Android	0.517806
iPhone	0.517806
DASHBOARD_CHANGE 'result': 'expenses'	0.436295
BANK_TASK_SEND_BUNDLE	0.224939
PAGEVIEW 'page': 'accounts'	0.208163

In table 4, the most important attributes of the third principal component using standardized data are presented. The attribute with the highest loading, BANK_TASK_SUCCESS{'possible-quick-transfer': 'true', 'receiver-type': 'own-account'}, explains that a transaction has been made to one of the user's own accounts and that the user has used the standard way of transferring money instead of making a quick transaction. The second most important attribute, BANK_TASK_SEND_BUNDLE, explains that a user has clicked the button to sign all transactions in the "bundle". The third attribute describes a user viewing the "bundle" page.

Table 4: Attributes with corresponding loadings using standardized data for third principal component.

Attribute	**Loadings**
BANK_TASK_SUCCESS 'possible-quick-transfer': 'true', 'receiver-type': 'own-account'	0.571478
BANK_TASK_SEND_BUNDLE	0.512343
PAGEVIEW 'page': 'bundle'	0.334938
SHOW-E-INVOICE	0.334460
BANK_TASK_SUCCES 'receiver-type': 'other-account'	0.230875
BANK_TASK_SMART_DATE_PICKER 'date': 'earliest'	0.205784

Finally, the standardized data set was used for plotting the data points against the first three principal components, PC1, PC2 and PC3. Figure 14 shows the distribution of the data points on the three dimensional space. From this figure it is apparent that almost all data points are close to each other with only one small division in the middle. There are also a few outliers. The explanation behind why figure 14 is formed the way it is, lies in the loadings of each principal component. The attributes with the biggest impact on each principal component are BANK_TASK_OCR_USED{'scanned-fields': 'true'} for the first principal component, Android for the second principal component and BANK_TASK_SUCCESS{'possible-quick-transfer': 'true', 'receiver-type': 'own-account'} for the third principal component. These three attributes affect the appearance of the figure the most.

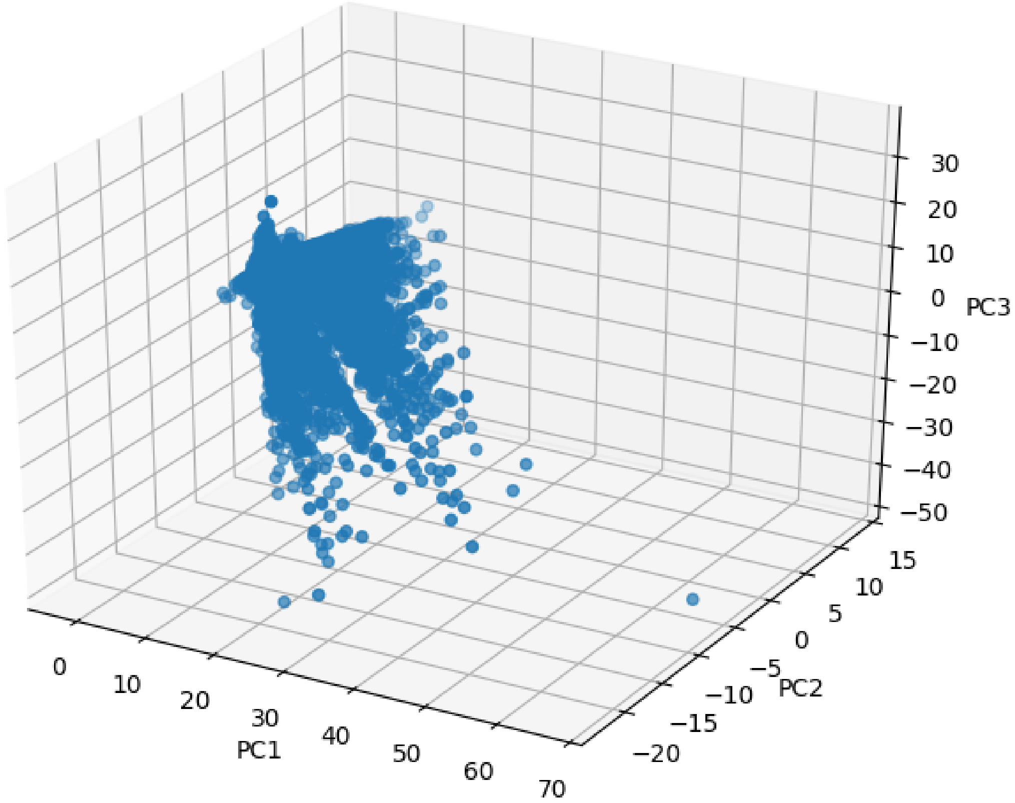


Figure 14: PCA transformation of one percent of data points.

8.3.2 PCA on the Non-standardized Data Set

Using non-standardized data, the PCA resulted in the first three principal components explaining 42.8 %, 18.3 % and 9.7 % of the total variation, as presented in figure 15. These three principal components explain more of the variation in the data set than the principal components obtained when using standardized data did. This is a consequence from using non-standardized data, since this will cause some attributes to have bigger impact on the outcome than others.

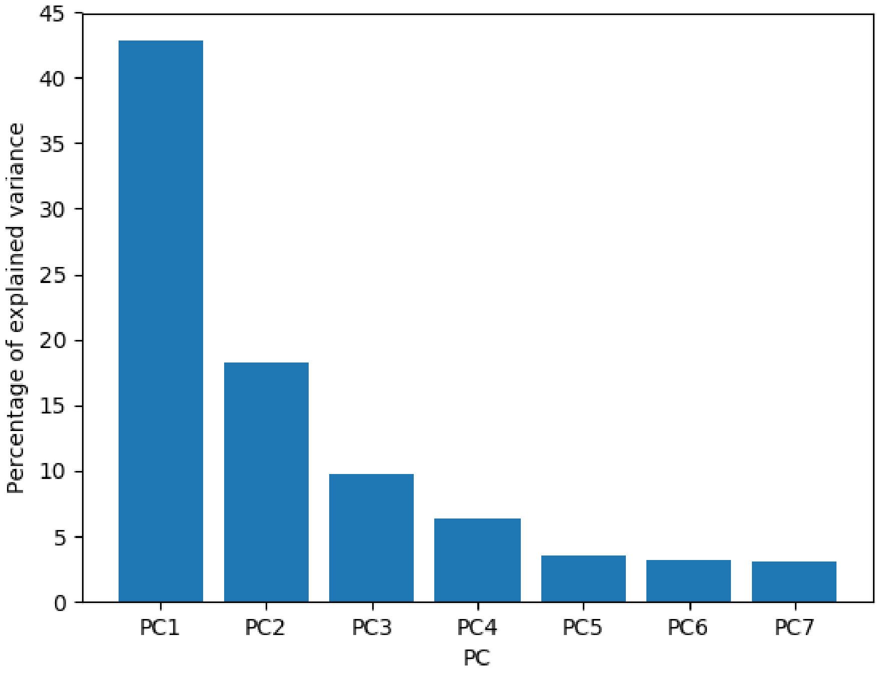


Figure 15: Scree plot over principal components using non-standardized data.

In tables 5 and 6, principal component one and two using non-standardized data are displayed. The third principal component of the non-standardized data set is not presented as all attributes in the third principal component had lower loadings than the set threshold of 0.2. The results in tables 5 and 6 show that in the first principal component, the attribute PAGEVIEW{'page': 'accounts'}, counts for 97 % of the importance amongst the attributes in the first principal component. In the second principal component, DASHBOARD_CHANGE{'result': 'expenses'} is the most important attribute, followed by Android and iPhone.

Table 5: Attribute in the first principal component with corresponding loading, using non-standardized data.

Attribute	**Loadings**
PAGEVIEW 'page': 'accounts'	0.971322

Table 6: Attributes in the second principal component with corresponding loadings, using non-standardized data.

Attribute	**Loadings**
DASHBOARD_CHANGE 'result': 'expenses'	0.860278
Android	0.328174
iPhone	0.328174

In the non-standardized data set, most of the variation can be explained by one single attribute in both the first and second principal component. This is because the loading vector places much weight on PAGEVIEW{'page': 'accounts'}, causing the importance of the other attributes in the first principle component to decrease. Performing PCA on non-standardized data resulted in substantially larger loadings for the attributes with the highest variances and these would have dominated the clustering if no standardizing would have been done before clustering. Therefore, if non-standardized data had been used for further analysis, it would have been more difficult to see any interesting patterns between different attributes. Because of this, no further visualization of the data points similar to figure 14 will be provided.

8.4 Finding Good Values of K

The Elbow Method and Silhouette Scores were used to the find best values of clusters, K, to use when running K-means. The Elbow Method and Silhouette Scores in figures 16 and 17 give somewhat similar indications to the best value of K to use. The complete results of these two methods are presented below.

8.4.1 Elbow Method

Figure 16 shows the plot from running the Elbow Method algorithm in order to find the best number of clusters to use. Figure 16 shows that *four*, *five* or *six* clusters are the best values for the numbers of clusters to use on the chosen range of two to ten clusters.

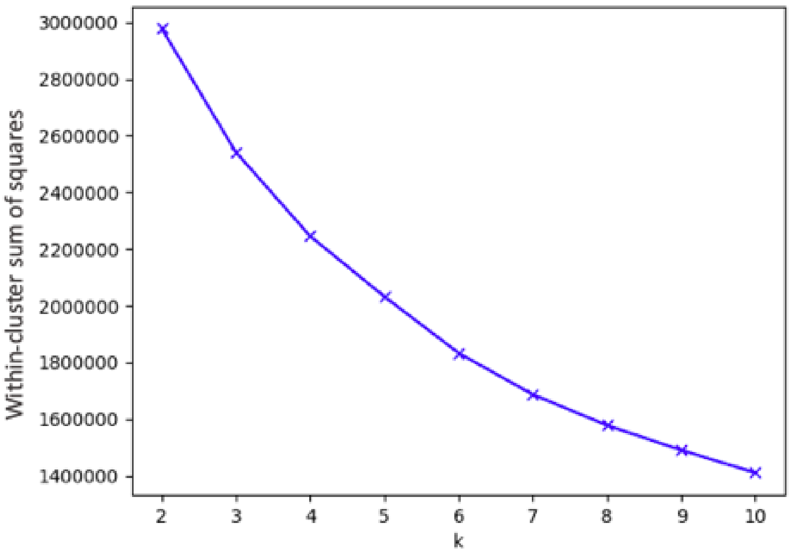


Figure 16: Elbow method plot showing the within-cluster variation against the number of clusters.

8.4.2 Silhouette Scores

In order to verify the result from the Elbow Method, we also used Silhouette Scores to calculate the best number of clusters to use. Figure 17 shows the result of the average Silhouette Scores plotted against the number of clusters, K, on the range of 2-10 clusters, the same range used in the Elbow Method. According to figure 17, the best number for K was *four* or *six* clusters. K=5 is the worst option according to the Silhouette Score method which is also why no further analysis is conducted using this number of clusters.

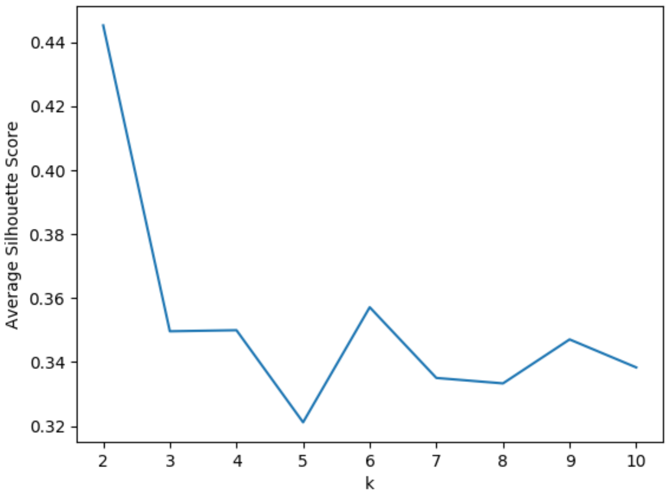


Figure 17: Silhouette Score plot for the cluster range 2-10.

Furthermore, the average Silhouette Scores in figure 17 are all relatively low. This means that although the scores are positive, which can be interpreted as them being correctly assigned to a cluster, they are also all quite close to zero, meaning that the data points could potentially be assigned to another cluster without making the cluster cohesion worse. The best accuracy is achieved for K=2 where the accuracy is around 45 percent. This is followed by K=4 and K=6, where the average Silhouette Scores are around 35 percent.

8.5 K-means Clustering

The results of running K-means are divided into three sections, in the first two sections, the results from performing K-means using two different values of K, four and six, are presented. Several runs were made using standardized data and a sample of one percent of the total data was used for visualization. Running additional tests for K=2, the best number of clusters to use according to the results of the Silhouette Scores, confirmed the impact of the device model as K-means, in this case, clustered all sessions triggered from an Android device into one cluster and all sessions triggered from an iPhone into the other.

8.5.1 K-means Using Four Clusters

Figure 18 visualizes the result from running K-means using four clusters. The four clusters are all represented by different colors where the purple cluster is the largest. Some of the clusters, especially the purple and the pink clusters, are somewhat overlapping whereas the blue and grey clusters are more separated from the others.

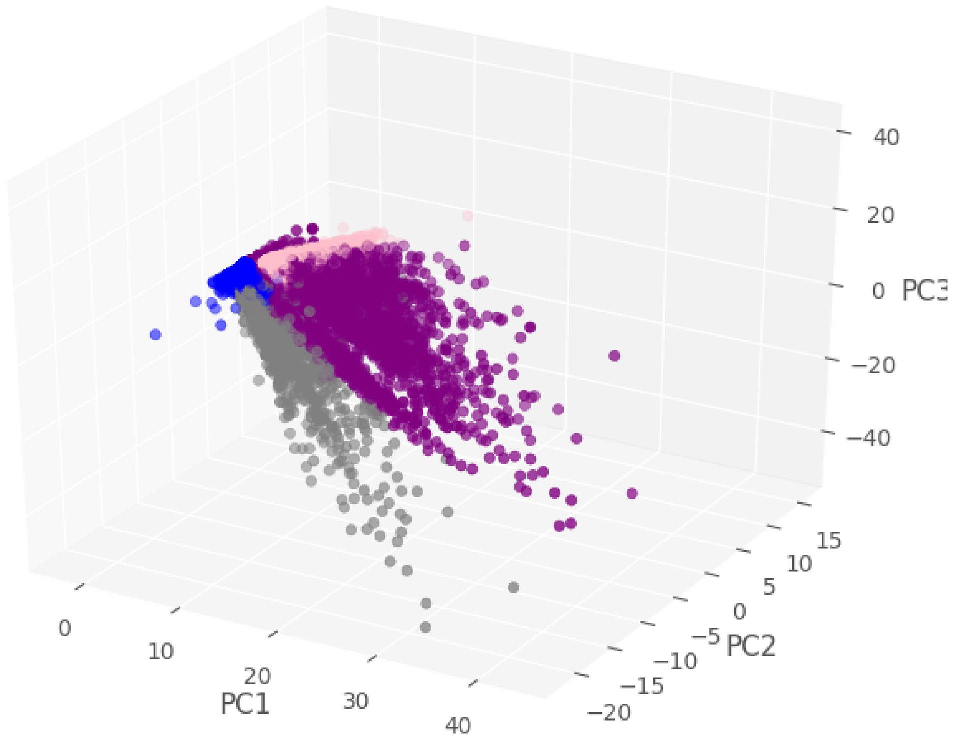


Figure 18: K-means with K = 4.

As clearly visible in figure 18, the K-means algorithm is incapable of partitioning the purple cluster into two different clusters. As previously explained in section 7.3.2, the data points in figure 18 are projected onto three dimensions. It is therefore not possible to say for certain that the distribution seen in the figure is the actual distribution of the data points in 22 dimensions. Therefore, we cannot come to the conclusion based on the positions of the data points seen in the visualization, that the cluster formations are not as good as they can or should be.

However, when analyzing the obtained results we conclude that it would have been preferable if the purple cluster had been divided into two clusters. The data points in the pink cluster which are located in the neighborhood of the purple cluster should have been clustered into the purple cluster, and the data points in the purple cluster which form a long and narrow row of data points close to the grey cluster should have formed a new separate cluster. Alternatively, it is our opinion that the blue cluster should have been stretched out to include the data points in the purple cluster lying closest to the grey cluster.

Finally, if the actual distribution is similar to the distribution in figure 18, the data set might not be completely suitable for K-means clustering, indicating that other clustering methods might be more appropriate to use. This is due to K-means being sensitive to clusters of non-convex shapes, present in figure 18.

8.5.2 K-means Using Six Clusters

The results of the Elbow Method and Silhouette Scores also indicated that K=6 would be a good choice for the number of clusters to use when running K-means. The result of running the K-means algorithm with K=6 is seen in figure 19. Using K=6, the K-means algorithm has put most of the data points which for K=4 made up the grey and the purple clusters together into one cluster, the pink cluster. What was the pink cluster for K=4 is now a larger purple cluster and the blue cluster is more spread out than it was for K=4. Furthermore, three new clusters, the grey, green and yellow clusters, have been formed. These are all placed closely together and are greatly overlapping with each other. It is our opinion that instead of having these three overlapping clusters, the pink cluster should have been separated into three different clusters where each narrow and long row of data points should have been placed into a separate cluster.

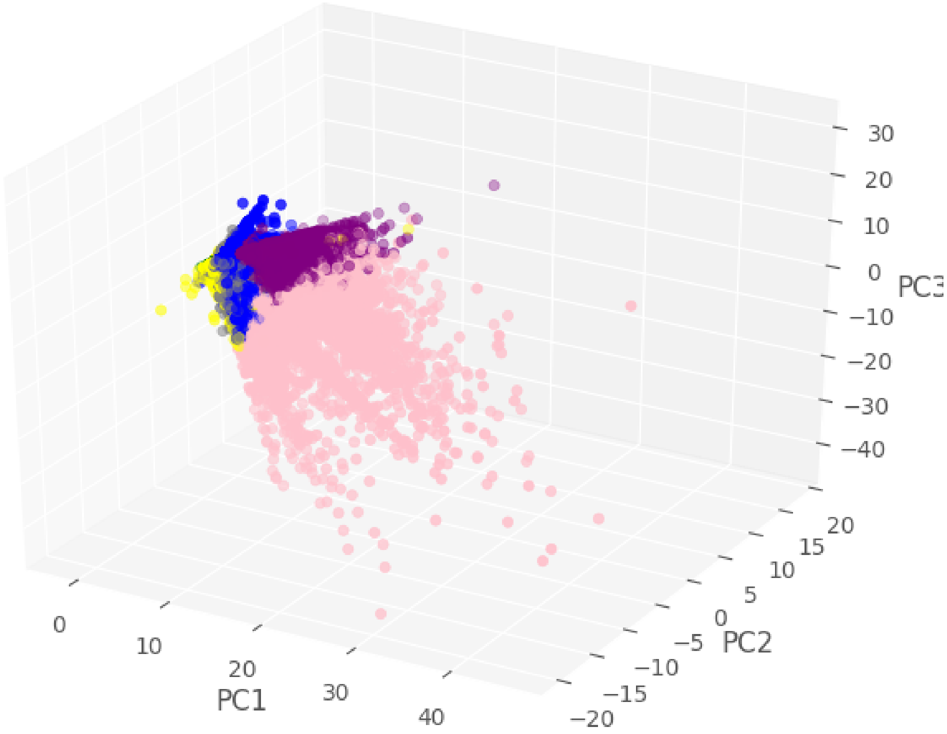


Figure 19: K-means with K = 6.

8.6 Attribute Occurrence in Clusters

In this section we present the result from calculating the attribute occurrences in each cluster for one per mille of the clustered data set that was formed using K-means with K=4. After viewing the results of the K-means algorithm run using K=4 and K=6, it was decided to use the results of K=4 for further analysis. This decision was made due to K-means producing more distinct clusters when K=4. The attribute occurrences were calculated by computing the sum of the occurrences of each attribute in all clusters using K=4. After these calculations, we were able to see which functionalities were triggered most frequently in each cluster. In table 7, the occurrence of how many times each functionality was triggered as well as the occurrence of each attribute per session are presented for each of the four clusters. Moreover, the three most frequently triggered attributes for the four clusters are presented in table 8.

Table 7: Sum of how many times each functionality was triggered for each cluster and, in parenthesis, how many times each functionality was triggered per session in each cluster.

Attribute	**Sum in cluster 1 (occurrences/session)**	**Sum in cluster 2 (occurrences/session)**	**Sum in cluster 3 (occurrences/session)**	**Sum in cluster 5 (occurrences/session)**
BANK_QUICK_TRANSFER_SUCCESS 'transfer-amount-category': 'predefined-amount'	15 (0.03)	1779 (0.04)	54 (0.02)	1265 (0.07)
BANK_QUICK_TRANSFER_SUCCESS transfer-amount-category': 'free-select-amount'	58 (0.10)	3051 (0.07)	124 (0.05)	1237 (0.07)
PAGEVIEW 'page': 'accounts'	3067 (5.09)	117064 (2.78)	8728 (3.27)	31679 (1.69)
PAGEVIEW 'page': 'account-upcoming-events'	189 (0.31)	5758 (0.14)	661 (0.25)	2633 (0.14)
PAGEVIEW 'page': 'bundle'	1839 (3.05)	2591 (0.06)	1135 (0.43)	1291 (0.07)
PAGEVIEW 'page': 'tools'	15 (0.02)	1131 (0.03)	10 (0.003)	32 (0.002)
PAGEVIEW 'page': 'expenses-category-list'	112 (0.19)	2560 (0.06)	176 (0.07)	1714 (0.09)
PAGEVIEW 'page': 'new-task'	20 (0.03)	128 (0.003)	4105 (1.54)	894 (0.05)
BANK_TASK_OCR_USED 'scanned-fields': 'true'	122 (0.20)	831 (0.02)	4202 (1.57)	61 (0.003)
DASHBOARD_CHANGE 'result': 'expenses'	61 (0.10)	505 (0.01)	3060 (1.15)	22119 (1.18)
DASHBOARD_CHANGE 'result': 'none'	24 (0.04)	508 (0.01)	25 (0.01)	178 (0.01)
SHOW-E-INVOICE	924 (1.53)	42 (0.001)	3 (0.01)	1 (0.00005)
ACCOUNTS_DETAILS_SEARCH	8 (0.01)	508 (0.01)	25 (0.01)	178 (0.01)
SHARE_ACCOUNT_NUMBER	0 (0)	0 (0)	26 (0.01)	0 (0)
HAS_NO_ACTIVE_SAVINGS_TARGETS	495 (0.82)	30874 (0.73)	2016 (0.76)	14891 (0.80)
BANK_TASK_SMART_DATE_PICKER 'date': '28'	0 (0)	0 (0)	231 (0.08)	10 (0.001)
BANK_TASK_SMART_DATE_PICKER 'date': 'earliest'	216 (0.36)	44 (0.001)	1003 (0.38)	43 (0.002)
BANK_TASK_SEND_BUNDLE	78 (0.13)	3567 (0.08)	2378 (0.89)	66 (0.004)
BANK_TASK_SUCCESS 'receiver-type': 'other-account'	225 (0.37)	2686 (0.06)	3717 (1.39)	99 (0.005)
BANK_TASK_SUCCESS 'possible-quick-transfer': 'true', 'receiver-type': 'own-account'	55 (0.09)	1880 (0.04)	1357 (0.51)	29 (0.02)
Android	22 (0.04)	6 (0.0001)	2144 (0.80)	18736 (0.9999)
iPhone	581 (0.96)	42156 (0.9999)	522 (0.20)	3 (0.0001)
Total number of sessions	603	42162	2666	18739

Table 8: Top three most frequently triggered attributes in each cluster.

Frequency Ranking	**Cluster 1**	**Cluster 2**	**Cluster 3**	**Cluster 4**
1	PAGEVIEW 'page': 'accounts'	PAGEVIEW 'page': 'accounts'	PAGEVIEW 'page': 'accounts'	PAGEVIEW 'page': 'accounts'
2	PAGEVIEW 'page': 'bundle'	iPhone	BANK_TASK_OCR_USED- 'scanned-fields': 'true'	DASHBOARD_CHANGE 'result': 'expenses'
3	SHOW-E-INVOICE	HAS_NO_ACTIVE_SAVINGS_TARGETS	PAGEVIEW 'page': 'new-task'	Android

In tables 7 and 8, it is evident that K-means is clustering quite strict on the device attributes: iPhone and Android. In the first cluster, there are 22 Android devices and 581 iPhones. Moreover, iPhone is the second most frequently triggered attribute in the second cluster, as seen in table 8. In the sample shown in table 7, there are 42156 iPhone users and six Android users in the second cluster which means that the device model is an important attribute in this cluster. In the third cluster the distribution is more even, but there is still a larger quantity of Android users than iPhone users. In the fourth cluster, the majority of all users used an Android device, with only three sessions being triggered from an iPhone. This shows that after pre-processing with one-hot encoding, which converted the device model in to binary data, the device model can be perceived as a label for each data point. This was one of our suspicions but the decision to keep these attributes was made due to us finding interest in understanding how the use of other attributes could depend on the choice of device.

It is worth mentioning that PAGEVIEW{'page': 'accounts'} is the most triggered action in all four clusters, as seen in table 8, and that cluster 1 is smaller in size than the other clusters. Furthermore, when looking at the number of triggers of each attribute in each cluster, it is noticeable that cluster 1 is the only cluster in which the attribute PAGEVIEW{'page': 'bundle'} and SHOW-E-INVOICE have on average more than one occurrence per session. In cluster 2, no other attribute other than PAGEVIEW{'page': 'accounts'} has an average of more than one occurrence per session. Furthermore, this attribute has on average more than one trigger per session in all clusters. In cluster 3 several attributes, in addition to PAGEVIEW{'page': 'accounts'}, have on average more than one occurrence per session. Examples of these attributes are BANK_TASK_SUCCESS{'receiver-type': 'other-account}, DASHBOARD_CHANGE{'result': 'expenses'}, PAGEVIEW{'page': 'new-task'} and BANK_TASK_OCR_USED-{scanned-fields': 'true'}. Another significant characteristic of cluster 3 is that it is the only cluster in which the attribute SHARE_ACCOUNT_NUMBER has been triggered. In cluster 4, DASHBOARD_CHANGE-{'result': 'expenses'} is the only attribute, other than PAGEVIEW{'page': 'accounts'}, with more than one occurrence per session on average. Explanations of the meanings of these attributes can be found in table 1 in section 8.1.

Furthermore, we extracted a sample of ten data points per cluster. This resulted in 40 data points being used for analysis regarding what functionalities are used in each session. The result of this extraction is presented in table 9 where the attributes are represented by a letter from A to V. Which letter corresponds to which attribute is presented in table 1.

Table 9: Sample of extraction of data points from each cluster.

Data point	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	label
1	0	0	7	0	4	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1
2	0	0	3	0	2	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1
3	0	2	6	2	4	0	0	0	0	0	0	4	0	0	1	0	4	0	0	0	0	1	1
4	0	0	10	0	9	0	0	0	0	0	0	5	0	0	1	0	5	2	2	0	0	1	1
5	0	0	4	0	3	0	0	0	1	0	0	2	0	0	0	0	0	0	1	0	0	1	1
6	0	0	4	0	2	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1
7	0	0	3	0	2	0	0	0	4	0	0	1	0	0	1	0	1	0	4	0	0	1	1
8	0	0	5	3	2	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	1
9	0	0	2	0	2	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1
10	0	0	4	1	3	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	1	1
11	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
12	0	0	5	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
13	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
14	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
15	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
16	0	0	3	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
17	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
18	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2
19	0	0	4	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1	2
20	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	2
21	0	0	8	0	2	0	0	0	0	0	0	0	0	0	1	0	0	1	0	3	0	1	3
22	0	0	7	0	1	0	0	5	5	0	0	0	0	0	2	0	2	2	4	0	1	0	3
23	0	0	3	0	0	0	0	2	1	1	0	0	0	0	1	0	1	1	1	0	1	0	3
24	0	0	2	0	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	1	1	0	3
25	0	0	3	0	1	0	0	1	2	2	0	0	0	0	1	0	0	0	2	0	1	0	3
26	0	0	2	0	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	1	1	0	3
27	0	0	1	0	0	0	0	4	1	0	0	0	0	0	0	0	0	1	1	0	1	0	3
28	0	0	4	0	2	0	0	0	3	0	0	0	0	0	1	0	0	0	3	0	0	1	3
29	0	0	2	0	0	0	0	2	1	1	0	0	0	0	1	0	2	1	1	0	1	0	3
30	1	0	3	0	0	0	0	2	1	2	0	0	0	0	1	0	0	1	1	0	1	0	3
31	2	0	2	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	1	0	4
32	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	4
33	1	0	4	3	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	4
34	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4
35	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4
36	0	0	5	0	0	0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	4
37	0	0	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	4
38	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	4
39	0	0	2	1	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	1	0	4
40	0	0	2	0	0	0	7	0	0	3	0	0	0	0	1	0	0	0	0	0	1	0	4

In table 9, several patterns regarding the triggered attributes in each cluster are revealed. For example, it is once again noticeable that attribute PAGEVIEW{'page': 'accounts'} (column C) is triggered in all sessions. Furthermore, the distribution of triggers of attribute HAS_NO_ACTIVE_SAVINGS_TARGETS (column O) is quite even in all clusters. Attribute PAGEVIEW{'page': 'bundle'} (column E) and SHOW-E-INVOICE (column L) are frequently triggered in cluster 1 but are not as frequently triggered in the other clusters. Cluster 2 does not have any other frequently triggered attributes than the ones which are triggered in all clusters. Attribute PAGEVIEW{'page': 'new-task'} (column H) is triggered frequently in cluster 3 but not in the other clusters. Attribute PAGEVIEW{'page': 'expenses-category-list'} (column G) is only triggered in cluster 4. Furthermore, the distribution of which type of device the sessions are triggered from is uneven in all clusters. The majority of the sessions in cluster 1 are triggered from an iPhone (column V) and the majority of the sessions in cluster 3 are triggered from an Android device (column U). In cluster 2 and 4, all sessions are either triggered from an iPhone or an Android device.

8.7 HDBSCAN Clustering

The additional clustering method HDBSCAN was used with several different values for parameters minimum cluster size and minimum samples as well as using both Euclidean and Manhattan distance. The results presented in figure 20 show the results of running the HDBSCAN algorithm on one per mille of the pre-processed data set with parameter values minimum cluster size=7300 and minimum samples=3 using Euclidean distance. This resulted in four clusters where the green, yellow and purple clusters are visible and the fourth one is not from the angle in the figure. The grey data points in figure 20 are noise.

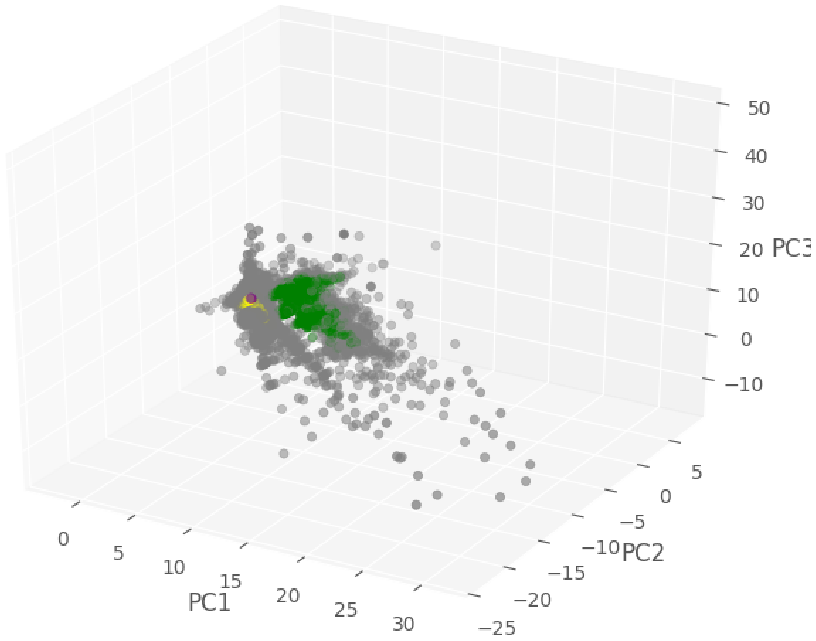


Figure 20: HDBSCAN on one per mille of the data points with parameters minimum cluster size set to 73000 and minimum samples set to 3.

As seen in figure 20, most data points are classified as noise and the model has not completely separated the clusters from each other. The poor results can partly be explained by the difficulty in choosing appropriate values for the parameters but could also be a result of the decision to only use one per mille of the pre-processed data set. K-means was run on the entire data set and this could be the reason for why the K-means algorithm resulted in more distinct clusters compared to HDBSCAN. Because of the different sample sizes that were used, the two algorithms can therefore not be compared in a fair way. Another contributing factor to the difficulty in achieving good visual results is that the 22 data points are projected onto three dimensions. As previously mentioned, this means that the distribution of data points visible in figure 20 does not have to equal the actual distribution and cluster formations in the data set.

9 Discussion

In the following discussion, we analyze the user patterns which were presented in section 8. Firstly, in section 9.1, we discuss the mutual patterns which apply to all clusters and the possible reason for the existence of these patterns. Furthermore, in section 9.2 to 9.5 we deliberate on what patterns are unique for each cluster and what the patterns say about the use of the functionalities in the application. In section 9.6, we compare the different clusters and discuss similarities and differences between the identified user patterns. We will also discuss the impact of the device model on the revealed user patterns. Lastly, in section 9.7, we discuss what alterations can be made to ensure that the application is suited for the identified user patterns. The analysis of what alterations can be made, is based on the fundamental principles of design and the discoverability category of the first principles of interaction design, which were presented in section 3.

Even if there are some indications of K-means not being completely suitable for the problem presented in this study, it provided superior results over HDBSCAN. Therefore, the following discussion will only be based on the best result of running K-means, achieved using K=4.

9.1 Mutual Patterns

The most frequently triggered attribute in all clusters is PAGEVIEW{'page': 'accounts'}. As explained in section 5, this attribute is triggered when a user visits the page where he or she can view his or her bank accounts. However, it is also the start page which the user lands on after logging into the bank application, meaning this attribute is triggered at least once in every session. It is also the only page from which the user can access several important features such as viewing the user's account balances and making transactions, meaning that the user may have to go back and forth from and to this page several times during a session.

Attribute HAS_NO_ACTIVE_SAVINGS_TARGETS is also triggered frequently in all clusters, which indicates that it is quite uncommon for users to create savings targets, no matter the type of device being used. This might be a result of the button to create a savings target being perceived as difficult to find. It might also be a result of the user not understanding the purpose of this functionality or because users generally do not think that this is something which they find valuable to use. As explained in section 5, in order to create a savings target in the iOS version of the application, the user can either click on a specific account and then click on the overflow menu in the right upper corner (see Appendix A: Account details) followed by clicking "Create Savings Target" (see Appendix A: Creating savings target and send account details), or the user can click on the "tools" page in the menu bar followed by clicking "Create Savings Target" (see Appendix A: View of tools page). The Android version of the application contains a similar navigation path to this functionality, as you click on a specific account followed by clicking on the overflow menu in the upper right corner and finally clicking on "Create Savings Target". As explained in section 5, the Android version of the application does not contain a menu bar at the bottom of the screen and because of this, there is no visible path to the "tools" page. Instead, the user can click on the overflow menu in the upper right corner of the "accounts" page, followed by clicking "tools" and then clicking "Create Savings Target" to create a savings target. The button to create a savings target can be found in two different ways in both the iOS and the Android version of the application, meaning that efforts have been made to enable users finding this functionality quickly. Because of this, the reason for why this attribute is frequently triggered is most likely due to the fact that the user does not find it valuable to use or does not understand the purpose of the functionality. Therefore we propose re-evaluating how this functionality is presented in the application and providing further guidance on *how* to create a savings target and *what* it can be used for. Furthermore, the reason for most Android users not using this functionality could be a result of it being more difficult to find the ”tools” page in the Android version of the application, due to there not being a menu bar at the bottom of the screen. This can also be the reason for why PAGEVIEW{’page’: ’tools’} is more frequently triggered among iPhone users than Android users.

BANK_TASK_SUCCESS{’possible-quick-transfer’: ’true’, ’receiver-type’: ’own-account’} is triggered in every cluster, as seen in table 7. This attribute explains the fact that users have used the standard way of transferring money in order to make a transaction to their own accounts, instead of using the short-cut. Furthermore, the short-cut to transfer money to the user’s own accounts is explained by the attribute BANK_QUICK_TRANSFER_SUCCESS{’transfer-amount-category’: ’predefined-amount’} and BANK_QUICK_TRANSFER_SUCCESS{’transfer-amount-category’:’free-select-amount’}. This short-cut simplifies the process and should result in BANK-_TASK_SUCCESS{’possible-quick-transfer’: ’true’, ’receiver-type’: ’own-account’} never being triggered, as every user should opt for the short-cut solution. However, the results in table 7 point to users still using the standard way of transferring money when transferring to their own accounts, indicating that users are having issues with the short-cut functionality. This could depend on the user not finding the short-cut functionality or not understanding how to use it properly. It could also depend on the users not appreciating the short-cut functionality.

Furthermore, attribute BANK_TASK_SMART_DATE_PICKER{’date’: ’earliest’} is also triggered in all clusters. This attribute explains that the user has clicked the short-cut button to enable the transaction being processed as soon as possible. The other options are to fill in the date manually or to choose from the other short-cut button which when clicked, changes the transaction date to the 28th of the month. If the user clicks the button to change the transaction date to the 28th, attribute BANK_TASK_SMART_DATE_PICKER{’date’: ’28’} is logged. Looking at the result in table 7 for each cluster, it is clear that BANK_TASK_SMART_DATE_PICKER{’date’: ’earliest’} is triggered often and that BANK-_TASK_SMART_DATE-_PICKER{’date’: ’28’} is seldom triggered, which could mean that most users prefer the transactions being processed as soon as possible. It is also possible that the user has long-pressed on the ”28th” button in order to choose another date as the short-cut button. In this case, another attribute which is not used in this study would be triggered. Nevertheless, BANK_TASK_SMART_DATE_PICKER{’date’: ’28’} is rarely triggered in each cluster.

9.2 Cluster 1

In cluster 1, there are more sessions triggered from an Android device than from an iOS device. Furthermore, in in table 9, cluster 1 is the only cluster in which SHOW-E-INVOICE is triggered and as apparent from table 7, this attribute is also triggered on average more than once in every session. This points to the attribute being of significant importance in cluster 1. In the other clusters, this attribute is not triggered at all and thus, it does not have as much importance in these clusters.

Before signing an invoice, one can choose to scan an invoice or add it manually. After this, the user can choose to add the transaction to the ”bundle” or to sign it directly. According to the presented results, attribute BANK_TASK_SUCCESS{'receiver-type': 'other-account'} is triggered almost twice as often as attribute BANK_TASK_OCR_USED{'scanned-fields': 'true'}, indicating that the scanning functionality is only used 50 % of the times when an transaction is signed directly. As 50 % of the users are using this functionality and the button to scan an invoice is clearly visible, we believe that the navigation to this functionality is decent, and thus not in need of improvement. The reason could instead either be a result of the user not preferring to use the scanning functionality over filling in the fields manually or vice versa. It could also depend on the user having difficulties understanding the scanning functionality.

Attribute PAGEVIEW{'page': 'bundle'}, triggered when a user clicks on the page to view the "bundle" is also frequently triggered in cluster 1. Similarly to SHOW-E-INVOICE, PAGEVIEW{'page': 'bundle'} is also triggered more than once per session on average. As these two attributes are frequently triggered in this cluster, one can assume that the two attributes are also often used together in one session. This conclusion is reasonable since the option to view an e-invoice is only available for the transactions present in the "bundle". The fact that PAGEVIEW{'page': 'bundle'} is triggered very frequently but BANK_TASK_SEND_BUNDLE is seldom triggered points to the fact that most people in this cluster seem to add transactions to the "bundle" and view the "bundle" but do not actually sign the transactions which are in the "bundle" very often. The reason for this could be that users are choosing to add multiple invoices to the "bundle" during a longer period of time and then when all invoices have been added, they sign everything in the "bundle" at once.

In conclusion, cluster 1 is the smallest cluster but the users in this cluster seem to be interacting with the application in several different ways and in many cases, multiple functionalities are triggered in the same session. This points to cluster 1 being an active cluster.

9.3 Cluster 2

In cluster 2, the most common attributes are HAS_NO_ACTIVE_SAVINGS_TARGETS, iPhone and PAGEVIEW{'page': 'accounts'}. All these attributes are logged without the user interacting with the application. This is due to both HAS_NO_ACTIVE_SAVINGS_TARGETS and iPhone being triggered upon starting a session and PAGEVIEW{'page': 'accounts'} always being triggered because of it being the start page. However, the device model is an important attribute in this cluster and it seems that cluster 2 puts major importance on the iPhone attribute. This is the largest cluster and the low occurrences of attributes per session for most attributes in cluster 2, point to the fact that most people do not interact with the application. Instead, most users in this cluster seem to be checking their account balances without doing much else, indicating that cluster 2 is a passive cluster.

9.4 Cluster 3

Cluster 3 is the cluster with the highest frequency of occurrences per sessions for BANK_TASK_SUCCESS{'possible-quick-transfer': 'true', 'receiver-type': 'own-account'}, as seen in both tables 7 and 9. Attributes PAGEVIEW{'page': 'new-task'}, BANK_TASK_SEND_BUNDLE, BANK_TASK_OCR_USED{'scanned-fields': 'true'} and BANK_TASK_SUCCESS{'receiver-type': 'other-account'} are often triggered in the sessions in this cluster. Attribute PAGEVIEW{'page': 'new-task'} explains that the user has initialized a new transaction using the standard way of transferring money to other accounts. BANK_TASK_SEND_BUNDLE means that the user has signed the "bundle" and attributes BANK_TASK_OCR_USED{'scanned-fields': 'true'} and BANK_TASK_SUCCESS{'receiver-type': 'other-account'} indicate that the user has scanned fields of an invoice when creating a transaction and then signed the transaction. The general conclusion that one can draw from analyzing this cluster is that most users in this cluster are using the application to transfer money. Attributes BANK_TASK_SMART_DATE_PICKER{'date': '28'} and BANK_TASK_SMART_DATE_PICKER{'date': 'earliest'} are also triggered more often than in several other clusters, which further proves the statement that many people are making money transactions in this cluster.

Similarly to cluster 1, many sessions in cluster 3 contain just as many, or more, triggers as the number of devices in this cluster. This points to the fact that in many sessions, attributes are being triggered more than once. This indicates that cluster 3 should be considered an active cluster.

The results of table 7 show that attribute SHARE_ACCOUNT_NUMBER is only triggered in cluster 3. Furthermore, it is only triggered 26 times in 2666 sessions meaning that it is not considered an important attribute in this cluster or in any other clusters. Because of the low frequency of triggers, it can be concluded that users do not use the functionality of sending an account number by text message very often. This functionality, as explained in section 5, is triggered by clicking on an account and then clicking the overflow menu in the upper right corner, followed by clicking "Share Account Number". Although this functionality could be considered quite difficult to find, our analysis is that this is not the reason for why this functionality is not used very often. We believe that because of new technology in the field of mobile payment solutions and peer-to-peer payment, where the account number is not needed in order to transfer money to other people, fewer people choose to share their account numbers for the purpose of money transferring. However, the results cannot verify this explanation of why this functionality is not being used and therefore, the impact that new technologies such as peer-to-peer payment has on existing functionalities in bank applications would be interesting to research further.

9.5 Cluster 4

Cluster 4 is the second largest cluster overall and the cluster which contains the largest set of Android devices. Moreover, few bank task actions are being triggered in this cluster. For example, out of 18739 sessions, BANK_TASK_SUCCESS{'receiver-type': 'other-account'} is only triggered 99 times and BANK_TASK_SEND_BUNDLE is only triggered 66 times. This points to the fact that this cluster is a passive cluster. Furthermore, in this cluster, attribute DASHBOARD_CHANGE{'result': 'expenses'} is triggered on average more than once per session. Together with attribute PAGEVIEW{'page': 'accounts'}, this is one of the most used functionalities in cluster 4. This indicates that the functionality of looking over one's expenses in the bar chart is either easy to find among users or seems to be appreciated by the users in cluster 4; it could also be a combination of the two.

Users who have triggered attribute DASHBOARD_CHANGE{'result': 'expenses'} can also click on the bar chart to view details about the users' expenses, such as viewing expenses in different categories over time. This action corresponds to attribute PAGEVIEW{'page': 'expenses-category-list'}. Although attribute DASHBOARD_CHANGE{'result': 'expenses'} is frequently triggered, there are few sessions in which attribute PAGEVIEW{'page': 'expenses-category-list'} is triggered. This can either mean that users do not understand that they can click on the bar chart or that they do not find it useful to see details about the expenses, but whether the reason for not triggering this attribute has to do with lack of understanding or not finding the attribute useful is difficult to verify.

9.6 Comparisons Between Clusters

Clusters 2 and 4 are the largest clusters and as seen in table 8, the top three attributes in these clusters are mostly attributes which are logged upon logging in to the bank application without the user performing specific actions. Therefore, these two clusters are characterized as passive. Furthermore, the device model has high importance in both cluster 2 and cluster 4. In cluster 2, nearly all sessions are triggered from an iOS device, and in cluster 4 almost every session is triggered from an Android device. Clusters 1 and 3 also contain a majority of either Android or iPhone users. Because of the uneven distribution of device models in the clusters, it is interesting to analyze differences between these clusters in order to gain insight as to whether the device model affects the other actions being triggered in the sessions.

When conducting the semi-structured interviews with both the UX-designer, the project manager and the Design Lead, they all mentioned that performing a quick transaction was considered to be a quite difficult task. All interviewees explained that this functionality has been perceived as both difficult to find and to understand how it should be used amongst users. Therefore, we had suspicions that the attributes BANK_QUICK_TRANSFER_SUCCESS{'transfer-amount-category': 'predefined-amount'} and BANK_QUICK_TRANSFER_SUCCESS{'transfer-amount-category': 'free-select-amount'} would not be triggered very often and that most people would use the standard way of transferring money, even when it came to making transactions to the user's own accounts. After analyzing table 7 and comparing cluster 2 and cluster 4, one can conclude that the attribute occurrences per session for both BANK_QUICK_TRANSFER_SUCCESS{'transfer-amount-category': 'predefined-amount'} and BANK_QUICK_TRANSFER_SUCCESS{'transfer-amount-category': 'free-select-a-mount'} are quite equal in both clusters. Furthermore, as the occurrence per session of BANK_TASK_SUCCESS{'possible-quick-transfer': 'true', 'receiver-type': 'own-account'} also is close to equal in clusters 2 and 4, one can assume that the choice of device model does not affect the ability to perform a quick transaction to the user's own account.

The attribute PAGEVIEW{'page': 'tools'} is more frequently triggered in clusters 1 and 2 than in clusters 3 and 4, meaning that it is more common among iPhone users to visit the "tools" page. This could be a result of users finding it more difficult to navigate to the "tools" page in the Android version of the application than in the iOS version. This is a reasonable assumption to make, since the Android version does not contain a menu bar from which the "tools" page can be easily accessed. Instead, the user finds it by clicking the overflow menu in the upper right corner, a button which does not give any indication of the "tools" page being accessible from there. The reason for why the attribute is more frequently triggered among iPhone users could also depend on the fact that it is easier to press the "tools" page icon by accident on the iOS version since it is more easily accessible.

From analyzing table 7, it is evident that DASHBOARD_CHANGE{'result': 'expenses'} is triggered more frequently in cluster 4 than in cluster 2, meaning that it is more common for this attribute to be triggered from an Android device. In fact, this attribute is triggered on average more than once per session in clusters 3 and 4, both of which contain more Android users than iPhone users. In the two clusters which contain mostly iPhone users, clusters 1 and 2, only a small fraction of the users are using this functionality. This is interesting as there is no evident difference between how this functionality is accessed in the two different versions of the application.

Another difference worth mentioning when comparing clusters 2 and 4 is that the percentage of sessions in which the user has used the functionality of signing the "bundle" is higher in cluster 2 than in cluster 4. This points to this functionality being more frequently used among iPhone users than Android users. On the other hand, in cluster 3, which contains mostly Android devices, the frequency of occurrences per session of this attribute is quite high. This indicates that some Android users are frequent users of this functionality as well. However, as cluster 2 is much larger than cluster 3, we conclude that iPhone users are using this feature more frequently than Android users. There is no apparent difference between signing the "bundle" from an Android device or from an iPhone, meaning the user patterns cannot be explained by differences in design or by users of one device having more difficulties in understanding this functionality than users of the other.

Finally, when analyzing the characteristics of the different clusters, it is noticeable that cluster 1 and 3 are the active clusters. When comparing the clusters, it can be concluded that the frequencies of attribute occurrences per session in each cluster generally are higher for clusters 1 and 3 than for cluster 2 and 4. Clusters 1 and 3 are also the smallest clusters, meaning it is quite uncommon overall for sessions to contain multiple actions. Furthermore, actions such as transferring money to another account and scanning an invoice are considered to be more complex interactions than changing the dashboard view to show one's expenses in a bar chart; the latter can be done with only one swipe. The fact that clusters 1 and 3 contain higher percentages of these types of complex interactions and that the top three attributes for these clusters, as seen in table 8, are attributes in which interaction is needed for them to be triggered, further proves that these clusters are more active than clusters 2 and 4.

9.7 Alteration of the Application To Better Suit the User Needs

The purpose of this study is not to redesign the mobile bank application which has been used as the subject of study in this thesis. Instead, the goal is to make recommendations of smaller changes which will result in the application being better suited for user needs that were discovered in this study. The following discussion includes understanding and discoverability, the most important characteristics of good design according to Norman [44]. Furthermore, the recommendations of smaller changes are made based on Norman's fundamental principles of design as well as the first principles of interaction design presented by Tognazzini [46].

The first functionality which, according to the results, can be modified is setting a savings target, represented by the attribute HAS_NO_ACTIVE_SAVINGS_TARGETS. Since this attribute is frequently triggered, one can conclude that the users generally do not discover this feature, do not find it valuable to create a savings target, or do not understand how to use it. If a user is having trouble finding the feature, then the design contradicts one of the most important characteristics of good design: discoverability. Therefore, we suggest that this feature can either be removed or that a re-evaluation of how this functionality is presented in the application is necessary. According to Tognazzini, if a user cannot find the functionality, it will be perceived by the user as the functionality not existing. If the reason for the users not using this feature is that they do not understand it, the design contradicts the other most important characteristic of good design:

understanding. To solve this issue, one can add signifiers, one of Norman's fundamental principles of design, in order to guide the user in how to create a savings target and explain what it can be used for.

Another attribute that is not being triggered frequently is SHARE_ACCOUNT_NUMBER. As preciously mentioned, the reason for why this feature is used rarely could depend on the influence of peer-to-peer mobile payment. Our analysis is that the increase of usage of peer-to-peer mobile payment solutions can lead to the need of this feature decreasing. Since this attribute is seldom triggered, one can argue that the same improvements proposed for the attribute HAS_NO_ACTIVE-_SAVINGS_TARGETS could be applied. Furthermore, removing this feature will most likely not affect the majority of users in a negative way.

Furthermore, since BANK_TASK_SUCCESS{'possible-quick-transfer': 'true', 'receiver-type: 'own-account'} is triggered quite often, our results point to the fact that many users transfer money through the standard way instead of using the quick transaction method. The quick transaction works as a short-cut and is according to our results quite difficult to find, not appreciated or difficult to understand how to use. The lack of understanding or discoverability regarding the quick transaction method among users is especially noticeable in cluster 3 where the standard way of transferring money is used more for transferring money to the user's own account than the quick transaction method. Therefore, a re-evaluation regarding how this feature is displayed and explained to the user should be made. As mentioned in section 5, the quick transaction method is revealed by swiping left on an account on the iOS version of the application or tapping on the account on the Android version. These implemented solutions are marked by arrows and icons respectively in order to signify to the user that there is a functionality hidden behind the account. Swiping or tapping on an account reveals three circles with different money amounts on them, which can be dragged to another account in order to transfer money to that account. There are implemented signifiers to help the user understand that he or she should drag the circle instead of just clicking on it, and because of this we believe that the feature affords the drag and drop functionality. This also proves that efforts have been made to ensure that users discover this functionality, as well as understand how they should use this functionality. However, as the results reveal that the standard way of transferring money is still used in many sessions to transfer money to the user's own accounts, we suggest further investigation regarding how this functionality is perceived among users. We propose that user-tests, as mentioned in the first principles of interaction design regarding discoverability, should be conducted to gain understanding whether the users are having difficulties discovering this functionality or understanding how to use it, as this can help in deciding which course of action should be taken to improve this functionality.

Furthermore, we consider the quick transaction feature to be a hidden treasure because it is a short-cut of an often-used function. We believe that when a user finds it, he or she will experience the Wow-factor, as described in section 3.2. However, further analysis of the quick transaction functionality brings up a discussion of whether this functionality is a part of a core task of the bank application and therefore should not be hidden. As deliberated on in section 3.2, the design should never hide core tasks since this could appear irritating to the user. Moreover, this is also pointed out by Tognazzini who explains that controls necessary for successful use of software always should be visibly accessible. In our case the core task, transferring money to your own account, can be reached in two ways, either by using the standard way or using the quick transaction. Since there are two alternative ways of performing this core task, we argue that this core task is not hidden because of the option to perform the task using the standard way not being hidden. The quick transaction can therefore be considered a hidden treasure by providing a smoother way of performing the task.

As mentioned earlier, one can conclude that most sessions in clusters 2 and 4 contain few triggered actions, hence they are passive clusters. The functionality that was most frequently triggered in these two clusters, as well as in clusters 1 and 3, is PAGEVIEW{'page': 'accounts'}. As this functionality is triggered often, one can come to the conclusion that the understanding and discoverability among users regarding this functionality is good. This should come as no surprise since this page is the start page of the mobile bank application. Furthermore, due to this being the most frequently triggered action in every cluster, we would like to suggest another way for users to view their account balances. From the results, we identify a latent user need regarding viewing the balances of the users' accounts in a more convenient way. This need has not been expressed to us by users but is instead captured by us through analyzing the results presented in tables 7 and 9.

Another insight from the result is that BANK_TASK_SMART_DATE_PICKER{'date': 'earliest'} is more frequently triggered in all clusters than BANK_TASK_SMART_DATE_PICKER{'date': '28'}. This goes to show that users in most sessions are using the "earliest" alternative over "28th". Why this is, can depend on several different factors such as the users not knowing they can change "28th" to a date which they prefer by using long-press on that button, or that they simply prefer "earliest". However, the most likely reason is that the "earliest" button is pre-filled, and that most users do not bother to change it. Due to this primarily being a quantitative study, where we have not asked any users about their motives when using these functionalities, we cannot get an understanding of which of these explanations is the most accurate one. To highlight the functionalities of the "28th" button, we make the suggestion that the design for this button could be made more intuitive with signifiers such as a describing text or a drop-down list where it is clear that the user can choose from different date options. In addition to being hidden, the long-press feature can also be considered a quite complex feature and therefore this functionality falls victim to "any attempt to hide complexity will serve to increase it", one of the first principles of interaction design regarding discoverability. This means that it might become more difficult for the user to use the functionality because of the attempt to hide it in order to ensure simplicity in the application. Conducting these changes can enable more users using the functionality. However, if the reason for users triggering the "earliest" button more frequently than "28th" is that the users prefer the "earliest" alternative, the proposed implementations would enable users to understand the different possibilities of this functionality. Moreover, we have only used the "28th" as an attribute but in the raw data, all dates available when using the long-press button were logged as attributes. Therefore, one reason for why most sessions included BANK_TASK_SMART_DATE_PICKER{'date': 'earliest'} and not BANK_TASK_SMART_DATE_PICKER{'date': '28'} might be that the 28th of the month is simply not the most used date. Using another date could have changed our results and for example revealed that users prefer to transfer money on the 26th of the month since this is the day after people in Sweden receive their salaries. One way to grasp why BANK_TASK_SMART_DATE_PICKER-{'date': 'earliest'} is frequently triggered is to, as previously mentioned, conduct user-tests, the fifth principle of discoverability. Performing user-tests makes it easier for the observer to understand the motives of the users. User-tests are also mentioned by Tognazzini as one of the first principles of interaction in the discoverability category, which further emphasizes the importance of this action.

For some attributes, there are noticeable differences in the number of occurrences per session between the clusters without there being any evident reason for why this is. For example, this is evident when analyzing the attribute DASHBOARD_CHANGE{'result': 'expenses'}, which is more frequently triggered in clusters 3 and 4 than in clusters 1 and 2. The majority of the sessions in clusters 3 and 4 are triggered from Android devices meaning that it is more common for Android users to trigger DASHBOARD_CHANGE{'result': 'expenses'}. However, as the design of this functionality is the same in both the Android version and the iPhone version of the application, the reason for why this is cannot be concluded from the quantitative results. Therefore, using a qualitative approach such as conducting user-tests with both Android and iPhone users could help reveal a potential underlying reason for why this is. Furthermore, there are few sessions in which PAGEVIEW{'page': 'expenses-category-list'} is triggered, which indicates that users are either not understanding that they can click on the bar chart, or that they do not find it useful to use. To enable discoverability, we recommend adding signifiers but to fully understand the reason why users are not using this functionality, user-tests are needed.

Moreover, most functionalities seem useful and well-placed in the application. For example, when performing a transaction, all attributes related to that task are triggered in the same session, indicating that the functionalities regarding this are appropriately located. Furthermore, we conclude that users are using the "bundle" in the intended way, as users seem to understand that they are supposed to add multiple invoices to the "bundle" and sign everything at once. However, to improve discoverability regarding some functionalities which the results indicate might be difficult to find, a designer can always add message hints or pop ups in order to make it easier for the user to find a certain feature. Message hints and pop ups are examples of what Tognazzini calls active discovery and these can be implemented anywhere in the mobile bank application where they are needed. However, the designer must be very careful so it does not create pain points for the user.

After using the mobile bank application, our opinion is that the feedback in the application is generally good. For instance, when a quick transaction has successfully been made, the balance of the account to which money has been transferred turns green and shows the new balance. Furthermore, after adding a new transaction, one is immediately presented with the option of signing it directly or placing it in the "bundle", meaning that the transaction request has been acknowledged by the application. Lastly, after creating a savings target, a pop up is presented explaining that the savings target has been acknowledged and following this, a pie chart with an overview of how much you have left to save before reaching the goal is displayed both next to the balance on the account on the start page as well as on the page for that specific account. All these examples help prove that the user is given feedback regarding his or her actions. Because of this we do not believe that the reason for the results sometimes indicating that some functionalities are not being used at all or in the correct way has to do with poor feedback. Instead we believe that the above mentioned improvements such as adding signifiers and using active discovery will help the users discover and understand the functionalities.

In summary, according to our analysis most of the functionalities we have chosen to further investigate in the application are frequently used. The ones that are more rarely used are the hidden features such as the quick transaction and the long-press on the date button, both which therefore can be improved through signifiers and active discovery in order enable more frequent usage. Furthermore, attributes SHARE_ACCOUNT_NUMBER and HAS_NO_ACTIVE_SAVINGS_TARGETS are also not used frequently. This results in the application providing more features than used by the users. However, if the designer chooses to make changes in the application to improve discoverability and understanding, he or she has to be careful not to introduce anything that could be perceived as a pain point for the users. This is because all smaller changes might not be appreciated by every user. In order to ensure that changes are being appreciated by the users, it is important to conduct user-tests on a regular basis, as explained by Tognazzini. Conducting user-tests can also help improve the understanding of why certain functionalities are not being used by users.

10 Conclusion

As proven in this thesis, it is possible to conduct a data-driven study in order to understand how functionalities in a mobile bank application are used by the users. By using the K-means algorithm, we have uncovered underlying user patterns in the mobile bank application and gained a deep insight into how the application is used among users.

The identified patterns indicate that most sessions contain few triggered actions, meaning they can be classified as passive. The most frequently triggered action is viewing the balances on the users' accounts. Furthermore, in each passive cluster, the device model is of higher importance than in the two active clusters. It is more common for bank tasks such as paying an invoice to be triggered in the active clusters than in the passive clusters. As most sessions are passive, one can conclude that users are generally not taking full advantage of all functionalities which are offered in the application. However, in all four clusters, there are some actions which are seldom triggered such as creating a savings target, sharing the user's account number, using the quick transaction and the option to long-press in order to change the pre-set date when creating a transaction. Our results therefore indicate that there are some pain points in the application which cause the user to not use certain functionalities or take full advantage of the provided short-cuts which are supposed to simplify the process of using the mobile bank application.

The reason for why these functionalities are not used by the users is either that the user does not discover the functionality, that the user does not understand how to use the functionality or that the user do not benefit from using the functionality. In order to increase the usage of the seldom triggered functionalities, eliminate the pain points and provide an application which better suits the needs of the user, one can add signifiers such as describing texts which lets the user know how to use the functionality and what the purpose of the functionality is. Another option is to use active discovery in order to guide the user and give hints as to how certain functionalities should be used. However, whether these pain points exist because of a lack of discoverability, understanding, or not finding a need for the functionalities cannot be established from the results presented in this study. Therefore, additional qualitative methods such as user-tests are needed to further establish the reasons for the existence of these user patterns.

Despite the fact that we were not able to define the underlying motives of the users, this study presents a extensive statistical overview of user patterns within the mobile bank application. By using quantitative research methods and applying a large data set of user actions, we were able to reveal concealed user patterns which can represent a larger group of users compared to what had been possible if a qualitative research method such as interviews with a select number of users had been used. Furthermore, we believe that the applied quantitative methods in this study can be applied to other mobile applications with available data of user interactions, but verifying this requires more research. It is possible that using the presented unsupervised models together with other types of data of user interactions will not result in findings which are similar to the findings made in this study. As we in this study purely focus on unsupervised learning, we are not able to measure the accuracy of our models and therefore it is the opinions of the observer which set the limits of what patterns are good enough for the study.

11 Future Research

There are several areas of this study which could be investigated further in order to gain additional knowledge about the usage of the bank application and to improve the models which were used. In this final section we present relevant future work and some modifications which could be made to further improve the results.

11.1 Handling the Data Set Differently

In this study, progressive loading was used to handle our large data set. However, it could have been better to use a cloud solution as this would have enabled us to divide the work on different nodes and thus reduce the run time of heavy computations. This would in turn have made it possible to use a larger subset of our data when running the algorithms. Therefore, future research is suggested in the field of using cloud solutions for the type of problem presented in this thesis.

The functionalities that are interpreted as most important in the mobile bank application were used for clustering. However, analyzing other attributes which are less frequently used would also be interesting as this could reveal additional patterns regarding how these functionalities are used together with the main functionalities. On the other hand, this could also lead to more sparsity in the data, as these functionalities are not triggered often, which could potentially exacerbate the clustering. Therefore, although it might be interesting for future work, using additional attributes should be done with caution.

Another possible future improvement would be to use a data set that includes data from a longer period of time in order to gain a better insight if functionalities are used differently during vacations and holidays. Due to the limited scope of this thesis, four months of data was used for Android devices and iOS devices respectively.

11.2 Including the iPad Application

This study did not take iPad users into account as these users use the iPad version of the application. The iPad version looks different to the mobile versions. Therefore, it would be interesting to see how iPad users use the application designed for them and if there are any similarities or differences between mobile application users and iPad users in how the core functionalities are used together.

11.3 Further Analyzing the Results of HDBSCAN

Due to time constraints and the poor performance of HDBSCAN in finding good clusters based on the data point positions, HDBSCAN was not used in further analyses of the data points to gain understanding of which functionalities are often used together within the bank application. Therefore, future research could be done focusing on improving the HDBSCAN model and extracting data points to use for analysis based on the cluster formations found by HDBSCAN. As HDBSCAN can handle clusters of varying density and find noisy data it is still considered a good model to use for the problem presented in this study, but more time spent on implementing the algorithm in the appropriate way is needed to ensure better results. Furthermore, as the time complexity of HDBSCAN was too large for HDBSCAN to be implemented using all data, we were not able to run the algorithm on all data that we had available. It would therefore be interesting to run HDBSCAN using another solution for handling large data sets. This could enable the usage of all data points in the clustering algorithm and it would interesting to see if this could improve the results.

11.4 Implementing and Verifying the Results

In this study, we have used data of user behavior in a mobile bank application in order to understand how the application is used. The results were later on used to provide recommendations of how the application could be altered to better fit the revealed needs of the user. Moreover, it was considered outside of the scope of this study to implement the findings in the actual bank application and to verify our findings with real users of the application. However, it would be interesting to see how the recommendations would work when implemented in the application and more importantly how they would be perceived by the users.

References

[1] Staykova1 KS, Damsgaard J. Adoption of Mobile Payment Platforms: Managing Reach and Range. Journal of theoretical and applied electronic commerce research. 2016;11(3):65–85. Available from: `https://scielo.conicyt.cl/scielo.php?script=sci_arttext&pid=S0718-18762016000300006`.

[2] Singh N, Jain A, Raw RS. Comparison analysis of web usage mining using pattern recognition techniques. International Journal of Data Mining & Knowledge Management Process (IJDKP). 2013;3(4):137–147. Available from: `https://pdfs.semanticscholar.org/5d11/788da68680dd2d2a524ecf1c3194be1e9855.pdf`.

[3] Kraft C. User Experience Innovation: User Centered Design That Works. 1st ed. Apress; 2012.

[4] 5 secrets to improve user experience and increase conversions; 2017. [Online]. Available from: `https://uxplanet.org/5-secrets-to-improve-user-experience-and-increase-conversions-e3b0fb2e4500`.

[5] Slob A, Verbeek PP. User Behavior and Technology Development. In: Verbeek PP, Slob A, editors. Technology and User Behavior. Dordrecht: Springer; 2006. p. 3–12.

[6] Do you personally use a smartphone?* - by age;. Accessed: 2019-03-26. [Online]. Available from: `https://www.statista.com/statistics/300402/smartphone-usage-in-the-uk-by-age/`.

[7] Szczepański M. Briefing - European app economy State of play, challenges and EU policy. European Parliamentary Research Service;. Accessed 2019-03-19. [Online]. Available from: `http://www.europarl.europa.eu/RegData/etudes/BRIE/2018/621894/EPRS_BRI(2018)621894_EN.pdf`.

[8] Ucros M. 10 Sneaky Ways Companies Are Collecting Data to Understand Customers;. Accessed: 2019-03-28. [Online]. Available from: `https://medium.com/@melodyucros/10-sneaky-ways-companies-are-collecting-data-to-understand-customers-be0b9089d54a`.

[9] Pazzani MJ, Billsus D. Content-Based Recommendation Systems. In: Brusilovsky P, Kobsa A, Nejdl W, editors. The Adaptive Web. vol. 4321. Springer, Berlin, Heidelberg; 2007. p. 325–41. Available from: `https://link.springer.com/book/10.1007/978-3-540-72079-9`.

[10] Cheng X, Fang L, Yang L, Cui S. Mobile Big Data. In: Mobile Big Data. 1st ed. Springer; 2018. p. 1–11.

[11] Han J, Kamber M, Pei J. Data Mining Concepts and Techniques. 3rd ed. Morgan Kaufmann Publishers; 2012. Available from: `https://learning.oreilly.com/library/view/data-mining-concepts/9780123814791/?ar`.

[12] Burkov A. Introduction. In: The Hundred-Page Machine Learning Book. Andriy Burkov; 2019. Available from: `http://themlbook.com`.

[13] James G, Witten D, Hastie T, Tibshirani R. An Introduction to Statistical Learning with Applications in R. Springer; 2013. Available from: `http://www-bcf.usc.edu/~gareth/ISL/`.

[14] Kuhn M, Johnson K. Introduction. In: Applied Predicitve Modeling. New York: Springer; 2016. p. 1–16.

[15] Brownlee J. A Gentle Introduction to Sparse Matrices for Machine Learning; 2018. [Online]. Available from: `https://machinelearningmastery.com/sparse-matrices-for-machine-learning/`.

[16] Theodoridis S, Koutroumbas K. Introduction. In: Pattern Recognition. Elsevier Inc; 2009. p. 1–12.

[17] Eirinaki M, Vazirgiannis M. Web Mining for Web Personalization. ACM Transactions on Internet Technology. 2003;p. 1–27. Available from: `https://dl-acm-org.ezproxy.its.uu.se/ft_gateway.cfm?id=643478&ftid=148718&dwn=1&CFID=131087892&CFTOKEN=73b7c878fbc46d3c-5BCF989C-FA71-FDF1-F38547445967631F`.

[18] Brownlee J. 7 Ways to Handle Large Data Files for Machine Learning. Machine Learning Mastery;. Accessed: 2019-02-19. [Online]. Available from: `https://machinelearningmastery.com/large-data-files-machine-learning/`.

[19] Yao J, Mao Q, Goodison S, Mai V, Sun Y. Feature selection for unsupervised learning through local learning. Pattern Recognition Letters. 2015;53:100–107. Available from: `https://www.sciencedirect.com/science/article/abs/pii/S0167865514003559`.

[20] Dash M, Liu H. Feature Selection for Clustering. School of Computing, National University of Singapore. n d;Available from: `http://www.public.asu.edu/~huanliu/papers/pakdd00clu.pdf`.

[21] Cai D, Zhang C, He X. Unsupervised Feature Selection for Multi-Cluster Data. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining; 2010. p. 333–342. Available from: `http://www.cad.zju.edu.cn/home/dengcai/Publication/Conference/2010_KDD-MCFS.pdf`.

[22] Yu L, Liu H. Efficient Feature Selection via Analysis of Relevance and Redundancy. Journal of Machine Learning Research. 2004;5(Oct):1205–1224. Available from: `http://www.jmlr.org/papers/volume5/yu04a/yu04a.pdf`.

[23] Anwesha, Dey L. Outlier Detection and Removal Algorithm in K-Means and Hierarchical Clustering. World Journal of Computer Application and Technology. 2017;5(2):24–29.

[24] Chawla S, Gionis A. k-means–: A unified approach to clustering and outlier detection. In: Proceedings of the 2013 SIAM International Conference on Data Mining; 2013. p. 189–197. Available from: `https://epubs.siam.org/doi/pdf/10.1137/1.9781611972832.21`.

[25] Tan PN, Steinbach M, Kumar V. Data. In: Introduction to Data Mining. 1st ed. Pearson Education; 2006. p. 19–95.

[26] One Hot Encoder. Scikit learn;. Accessed 2019-03-04. [Online]. Available from: `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html`.

[27] Keogh E, Mueen A. Curse of Dimensionality. In: Sammut C, Webb GI, editors. Encyclopedia of Machine Learning and Data Mining. Boston, MA: Springer US; 2017. p. 314–315. Available from: `https://doi.org/10.1007/978-1-4899-7687-1_192`.

[28] Powell V, Lehe L. Principal Component Analysis. Setosa;. Accessed 2019-03-19. [Online]. Available from: http://setosa.io/ev/principal-component-analysis/.

[29] Anton H, Rorres C. Euclidean Vector Spaces. In: Elementary linear algebra. 11th ed. Wiley; 2014. p. 131–82.

[30] Ringnér M. What is principal component analysis? Nature Biotechnology. 2008;26(3):303–304.

[31] Wang XD, Chen RC, Zeng ZQ, Hong CQ, Yan F. Robust Dimension Reduction for Clustering With Local Adaptive Learning. IEEE Transactions on Neural Networks and Learning Systems. 2019;30(3):657 – 669.

[32] Steinbach M, Ertöz L, Kumar V. The Challenges of Clustering High Dimensional Data. In: New Directions in Statistical Physics. Springer, Berlin, Heidelberg; 2004. p. 273–309. Available from: https://www-users.cs.umn.edu/~ertoz/papers/clustering_chapter.pdf.

[33] Hastie T, Tibshirani R, Friedman J. Unsupervised Learning. In: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. Springer; 2009. p. 485–585. Available from: https://web.stanford.edu/~hastie/Papers/ESLII.pdf.

[34] Arthur D, Vassilvitskii S. k-means++: The Advantages of Careful Seeding. 2007;p. 1027–1035. Available from: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf.

[35] Sculley D. Web-scale K-means Clustering. In: Proceedings of the 19th International Conference on World Wide Web. WWW '10. New York, NY, USA: ACM; 2010. p. 1177–1178. Available from: http://doi.acm.org.ezproxy.its.uu.se/10.1145/1772690.1772862.

[36] Béjar J. K-means vs Mini Batch K-means: A comparison;Available from: https://upcommons.upc.edu/bitstream/handle/2117/23414/R13-8.pdf.

[37] Chakraborty S, Nagwani NK, Dey L. Performance Comparison of Incremental K-means and Incremental DBSCAN Algorithms. International Journal of Computer Applications (0975 – 8887). 2011;27(11):14–18. Available from: https://arxiv.org/pdf/1406.4751.pdf.

[38] Ali T, Asghar S, Naseer Ahmed Sajid. Critical analysis of DBSCAN variations. In: 2010 International Conference on Information and Emerging Technologies; 2010. p. 1–6. Available from: https://ieeexplore.ieee.org/document/5625720.

[39] Ester M, Kriegel HP, Sander J, Xu X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. 1996;p. 226–231. Available from: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf.

[40] Comparing Python Clustering Algorithms;. Accessed: 2019-04-16. [Online]. Available from: https://hdbscan.readthedocs.io/en/latest/comparing_clustering_algorithms.html.

[41] Oh Y, Kim Y. A Hybrid Cloud Resource Clustering Method Using Analysis of Application Characteristics. In: 2017 IEEE 2nd International Workshops on Foundations and Applications of Self* Systems (FAS*W); 2017. p. 295 – 300.

[42] Bholowalia P, Kumar A. EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN. International Journal of Computer Applica-

tions. 2014;105(9):17–24. Available from: https://pdfs.semanticscholar.org/5771/aa21b2e151f3d93ba0a5f12d023a0bfcf28b.pdf.

[43] de Amorim RC, Hennig C. Recovering the Number of Clusters in Data Sets with Noise Features using Feature Rescaling Factors. Information Sciences. 2015;324:126–145. Available from: https://www-sciencedirect-com.ezproxy.its.uu.se/science/article/pii/S0020025515004715.

[44] Norman D. The Design of Everyday Things: Revised and Expanded Edition. Basic Books; 2013.

[45] Cooper A, Reimann R, Cronin D. Goal-Directed Design, Understanding Users: Qualitative Research. In: About Face 3 : The Essentials of Interaction Design. 3rd ed. Wiley; 2007. p. 1–26, 49–74.

[46] Tognazzini B. First Principles of Interaction Design (Revised & Expanded); 2014. Accessed: 2019-05-08. [Online]. Available from: https://asktog.com/atc/principles-of-interaction-design/.

[47] Holtzblatt K, Beyer H. Introduction, Field Research: Data Collection and Interpretation. In: Contextual Design Evolved. Morgan & Claypool; 2015. p. 1–3, 11–20.

[48] Sathya M, Isakki P. Apriori Algorithm on Web Logs for Mining Frequent Link. In: 2017 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS); 2017. p. 1–5. Available from: https://ieeexplore.ieee.org/document/8303127.

[49] Zhang X, Brown HF, Shankar A. Data-driven Personas: Constructing Archetypal Users with Clickstreams and User Telemetry. In: CHI '16 Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems; 2016. p. 5350–5359. Available from: https://dl.acm.org/citation.cfm?id=2858523.

[50] Lim DL, Bentley PJ, Kanakam N, Ishikawa F, Honideni S. Investigating Country Differences in Mobile App User Behavior and Challenges for Software Engineering. IEEE Transactions on Software Engineering. 2015;41(1):40 – 64. Available from: https://ieeexplore.ieee.org/abstract/document/6913003.

[51] Fu B, Lin J, Li L, Faloutsos C, Hong J, Sadeh N. Why people hate your app: Making sense of user feedback in a mobile app store; 2013. Available from: https://www.cs.cmu.edu/~leili/pubs/fu-kdd2013-wiscom.pdf.

[52] Garrett JJ. The Elements of User Experience: User-Centered Design for the Web. Peachpit Press; 2002.

[53] Bryman A. Sampling, Interviewing in Qualitative Research. In: Social Research Methods. Oxford: Oxford University Press; 2012. p. 183–208, 469–98.

[54] Pandas Data Analysis Library;. Accessed: 2019-04-24. [Online]. Available from: https://pandas.pydata.org.

[55] 2.5. Decomposing signals in components (matrix factorization problems). Scikit learn;. Accessed 2019-02-28. [Online]. Available from: https://scikit-learn.org/stable/modules/decomposition.html.

[56] Huang Z. Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Knowledge Discovery. 1998;2(3):283–304. Accessed: 2019-04-24.

Appendix A

Mockups of the iOS version of the mobile bank application used in this study.

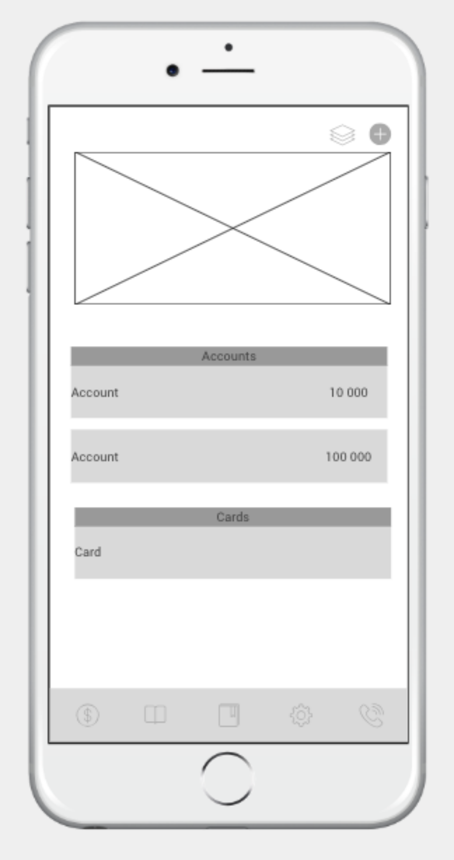


Figure 21: View over accounts

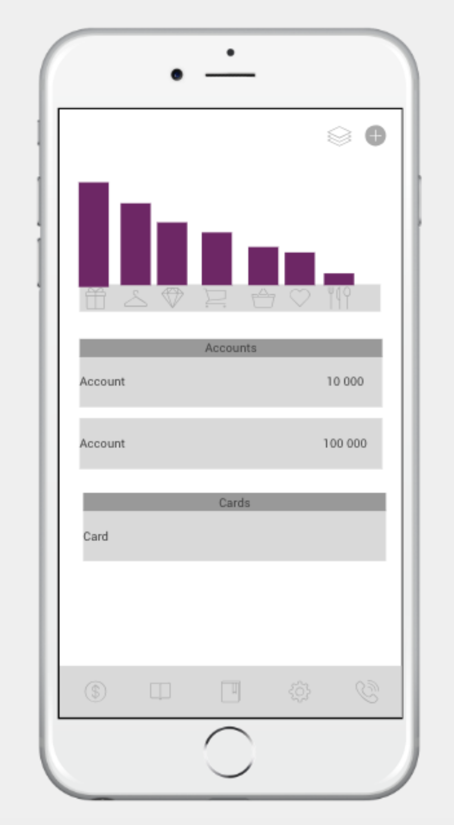


Figure 22: Bar chart of expenses

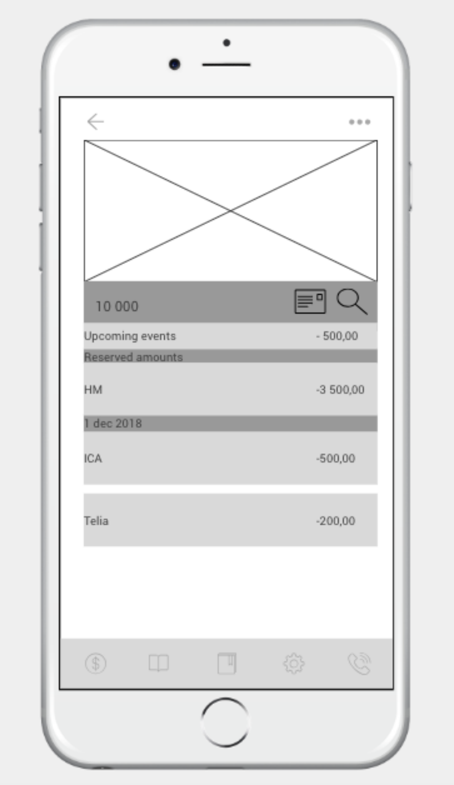


Figure 23: Account details

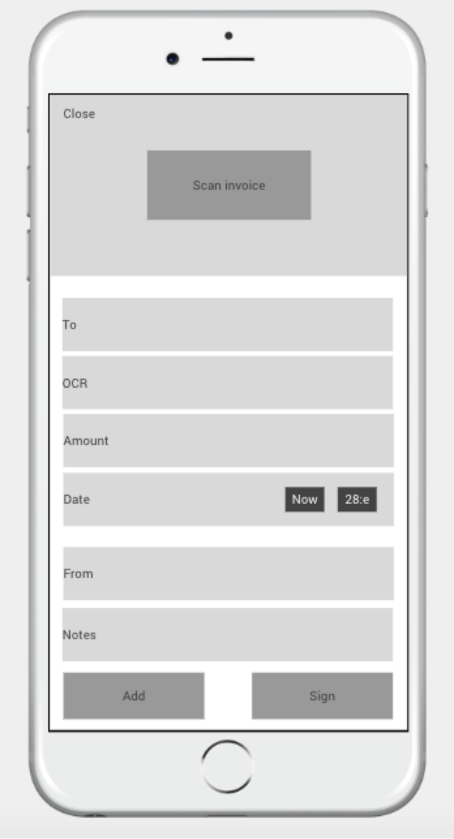


Figure 24: Scan an invoice.

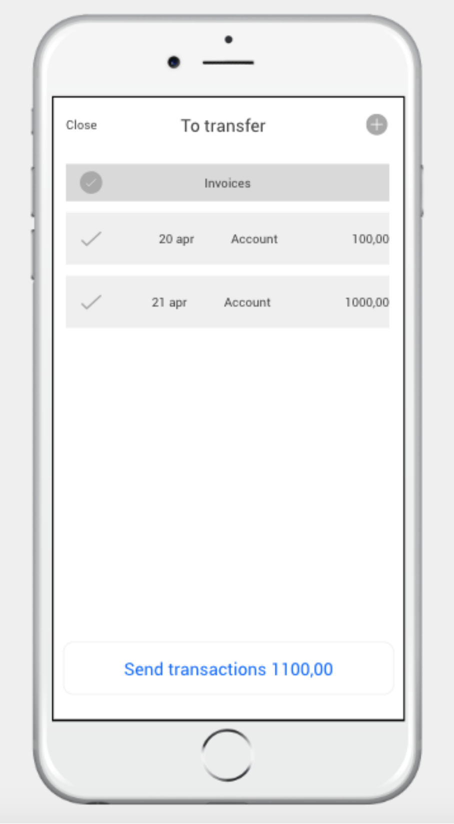


Figure 25: The bundle

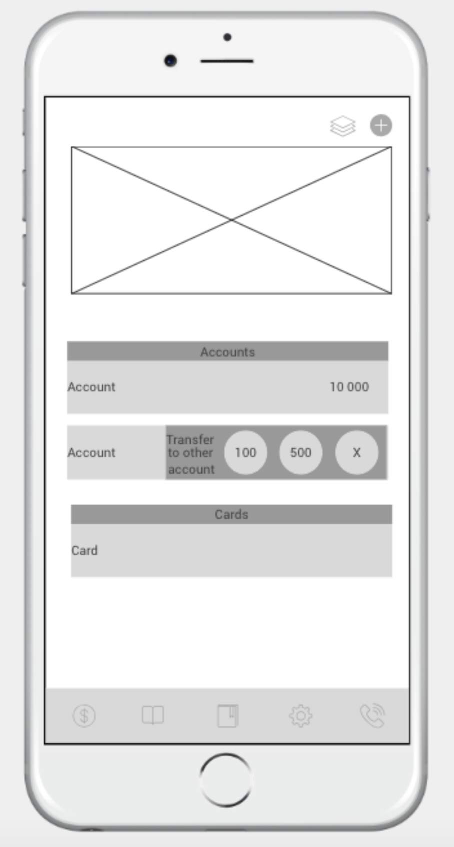


Figure 26: Quick transfer

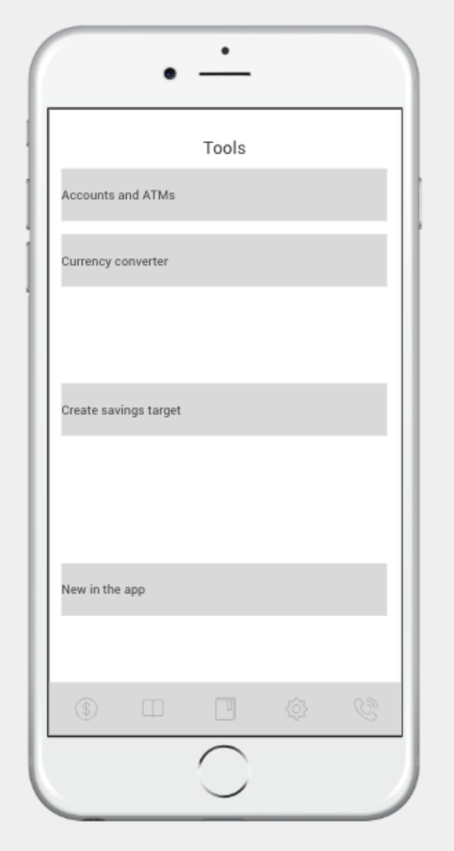
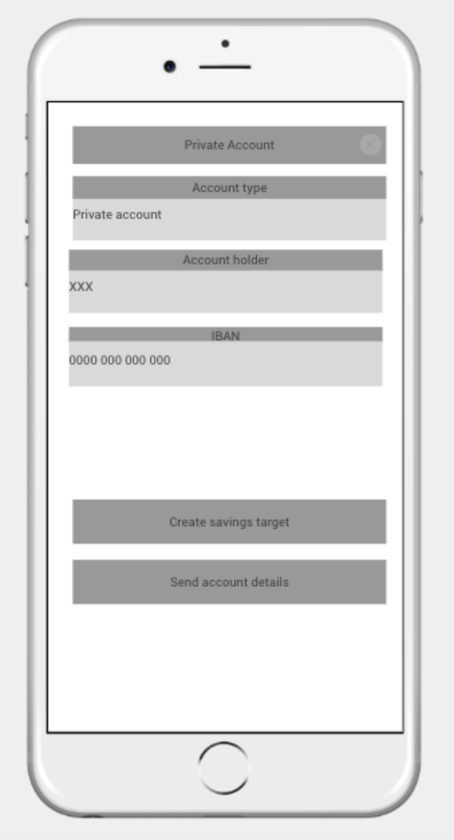


Figure 27: View of tools page



Account type
Private account

Account holder
XXX

IBAN
0000 000 000 000

Figure 28: Creating savings target and send account details

Appendix B

Interview with Filip Eriksson, UX-designer of the mobile bank application and Tilda Dahlgren, Governance Project Manager. 25 January 2019.

- Vad jobbar ni med nu?
- Kan ni berätta om hur designar för applikationens olika användare?
- Hur jobbar ni utefter dessa identifierade användare?
- Finns det något ytterligare ni skulle vilja veta om era användare? Vad?
- Berätta gärna lite om er arbetsprocess när ni ska skapa en ny applikation.
- Kan du berätta om dina ansvarsområden som UX-designer för den mobila bankapplikationen?
- Vilka ansvarsområden har ni som UX-team?
- Hur sker underhållsarbetet för dig som UX-designer?
- Hur får ni feedback från användarna?

Appendix C

Interview with Hannah Mittleman, Design Lead of the mobile bank application, 15 February 2019.

- Berätta om din roll som Design Lead.
- Kan du berätta om hur designar för applikationens olika användare?
- Kan du berätta om varför ni sparar data om användares användning av applikationen och vad ni hoppas få ut av det?
- Hur gick/går tankegången när ni valde/väljer events i Flurry?
- Finns det något ytterligare ni skulle vilja veta om era användare? I så fall vad?
- Finns det något samband som du vet nu att ni är intresserade av att undersöka inom en session? Vad?