Predicting a customer's next touch point from customer journey data

Habets, S.

Award date:
2020

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science
Architecture of Information Systems Research Group

Predicting a customer's next touch point from customer journey data

Master Thesis

Stefan Habets

Committee Members:
Dr. Ing. Marwan Hassani
Dr. Maarten Wolf (KPN)
Dr. Decebal C. Mocanu

Version 1.0

Eindhoven, January 2020

Abstract

Customer journeys are rapidly increasing in popularity, as it is very important for companies to analyze how their customers think and behave. This means that more and more research is being done to investigate the customer journey. These researches investigate how customers traverse their journeys and how they can be improved for the future. However, those researches only focus on improving the process for future customers by analyzing the historical data. This research focuses on helping the current customer immediately, by analyzing if it is possible to predict what the customer will do next. When we can predict what contact a customer will have with the company next, the company can take preventive actions to help the customer beforehand. We propose a model to predict the customer's next contact type, which is called the touch point throughout the research. At first we will analyze the customer journey data by applying process mining techniques. These insights and the historical data of a number of accumulated customer journeys will then be used to train a classifier. This model can be used to perform a prediction on a customer's journey while the journey is still active. Normally machine learning classifiers are trained with data from a static point of time, we however want to use the flow of the customer journey and will include the previous steps a customer has taken into the model. Finally a business scenario will be provided to see that prevention can be useful in practice.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

This Master Thesis is the result of a graduation project conducted at KPN in cooperation with the Eindhoven University of Technology. This graduation research is done within the Architecture of Information Systems (AIS) research group of the Computer Science and Mathematics department.

This chapter is an introduction into the research. In Section 1.1 the context with regards to the problem is described. Then in Section 1.2 the research problem is stated, the goal to achieve is explained and which problems need to be solved to reach the goal. In Section 1.3 the outline of this thesis is listed.

1.1 Context

Nowadays, companies collect all sorts of data from their services and customers. Altogether, data-driven analysis has become more interesting for companies and the collected data is used by companies to analyze all of their products and services. In organizations, such as a telecommunications company, the data is used to analyze the journey of a customer.

Investigating and analyzing often reveals ways to potentially optimize services provided. This investigation is both in favor of the customer and the company, as the service is improved the customer will get a more tailored approach. Thus increasing the customers quality of life, as well as the company whom needs to spend less resources because their services have become more streamlined. For example, a customer that can install his newly received modem on his own or with the help of the website and does not have to call the service desk or worse, need a mechanic. Those customers save the company money because the service desk has less work or no mechanic has to be send, decreasing the overall expenses. Also the customer is happier since he does not have to wait in the phone queue of the service desk or wait at home to receive a mechanic, increasing the customer satisfaction.

One kind of data is the customer journey, the customer journey represents the steps a customer takes with the company. Each step is called a touch point and is defined to be an interaction from the customer with the companies' products or services [6]. These customer journeys are mapped into a customer journey map (CJM) to perform analysis on. The customer journeys are very interesting to analyze for a company since the company can see how customers actually behave. The company has an idea of how a certain process should work, but in practice this might not be the case at all. Reviewing the customer journeys gives insight in how customers follow certain flows. When an important step is missing from the process flow in a majority of the customer

journeys than the company can investigate this matter and see why the customers do not perform those steps. All in all companies such as KPN are continuously trying to improve their services and products to keep their customers satisfied.

1.2 Research Problem

Up until now customer journey analysis is performed to improve processes in hindsight [24]. Historical data is checked and used to iteratively improve the customer experience when interacting with a certain process of the company. Therefore the company is always late to provide immediate support to the customer. When something went wrong, the mistake can later be found in the data to help improve the error for the future. Unfortunately the customer's journey will never behave like the perfectly designed customer journey. You can also argue that this is for the better, as the imperfect journeys make it possible to improve upon.

Already taking a look at what statements we can make about future steps is very interesting to KPN. Therefore to be able to predict a customer's next contact step using the customer journey data, would be fantastic! Because knowing what a customer will do next, gives the company the ability to proactively provide support to the customer. Helping the customer proactively saves time and therefore also costs, as well as increasing the customer satisfaction because they are helped faster. This leads to the following research question defined for this thesis:

> *Is it possible to predict a customer's next touch point by using the historical data from customer journeys and a prediction technique?*

To provide a better structure to this research, the main research question will be divided into five sub-questions. These sub-questions will provide a structured approach to reach the answer of the main research question. These five sub-questions are:

1. What should the output predictions regarding the customer journey look like considering the business objectives of KPN?
2. Which data regarding the customer journey is currently available at KPN?
3. What data is directly usable in the context of predicting the next touch point?
4. How good are the predictions made on the data of KPN?
5. What can the business achieve with the acquired results?

1.3 Thesis structure

This thesis is structured following the steps in the CRISP-DM framework. First however a literature analysis and explanation of the techniques used in this research including the CRISP-DM framework are provided in Chapter 2. Afterwards the preliminaries and a formal problem formulation are described in Chapter 3. In Chapter 4 the business understanding is investigated. After the business side is investigated, the understanding of the data is written in Chapter 5. Following the data understanding, the data preparation is handled in Chapter 6. Now we have an understanding of the business and the data, also all data is prepared thus the modeling is done in Chapter 7. Last step is to evaluate the methods used which is described in Chapter 8. Finally we will conclude with an overall conclusion in Chapter 9.

Chapter 2

Literature analysis

In this chapter the consulted literature is explained. First the methodology of cross-industry standard process for data mining (CRISP-DM) is explained in Section 2.1. Then the customer journey within process mining is described in Section 2.2. To actually perform the prediction, the use of an algorithm is required. Different types of techniques will be taking into account. First the literature will be consulted in Section 2.3. Then we will discuss the used models in Section 2.4 which are logistic regression, random forest and boosted trees. Lastly a neural network will be explored.

2.1 CRISP-DM

The CRISP-DM (cross-industry standard process for data mining) framework [45] is a methodology to structurally approach and solve data mining problems. CRISP-DM consists of the standard steps which should be taken in each data mining project to reach a solution without forgetting any necessary considerations. This also means that the steps taken can be retraced and reproduced. Reproducibility is important when testing because it can be unclear how a certain output is reached. If the steps can be reproduced then the mistake can be pinpointed exactly. This is one of the advantages of using a clear, standardized framework. Another one is that the process is more agile and can thus be iteratively improved.

The CRISP-DM framework consist of six steps as depicted in Figure 2.1. The first five steps are contained within a loop, giving the possibility to iteratively improve your understanding of the problem and the created solution. Although the figure provides arrows which guide the direction of the process. By no means is the sequence of the steps strict and only the important and most frequent dependencies are shown with arrows [45]. Now each explicit step in the process will be explained, a summary can be seen in the table in Figure 2.2.

Business understanding

The first step in the cycle, business understanding, is about comprehending the company. Before a product of good quality can be delivered, it is important to know the background for which the product will be made. When there is an understanding of why the product is needed, it is also much clearer in what way the product should be build. Therefore the project objectives and requirements are investigated from a business perspective. In Chapter 1 the context surrounding the problem and the problem itself are described. In Chapter 4 a more in-depth look will be taken on the business side of the problem. With this information the first sub-question will be answered.

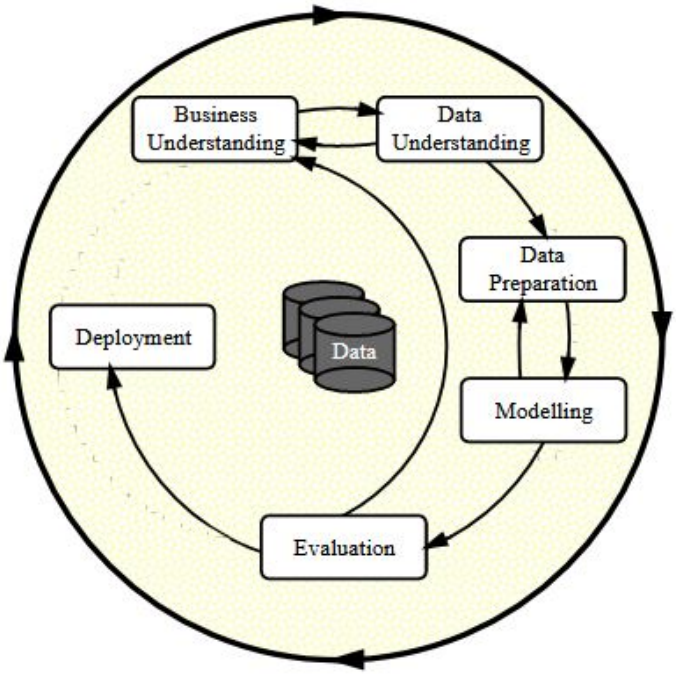


Figure 2.1: CRISP-DM framework [45]

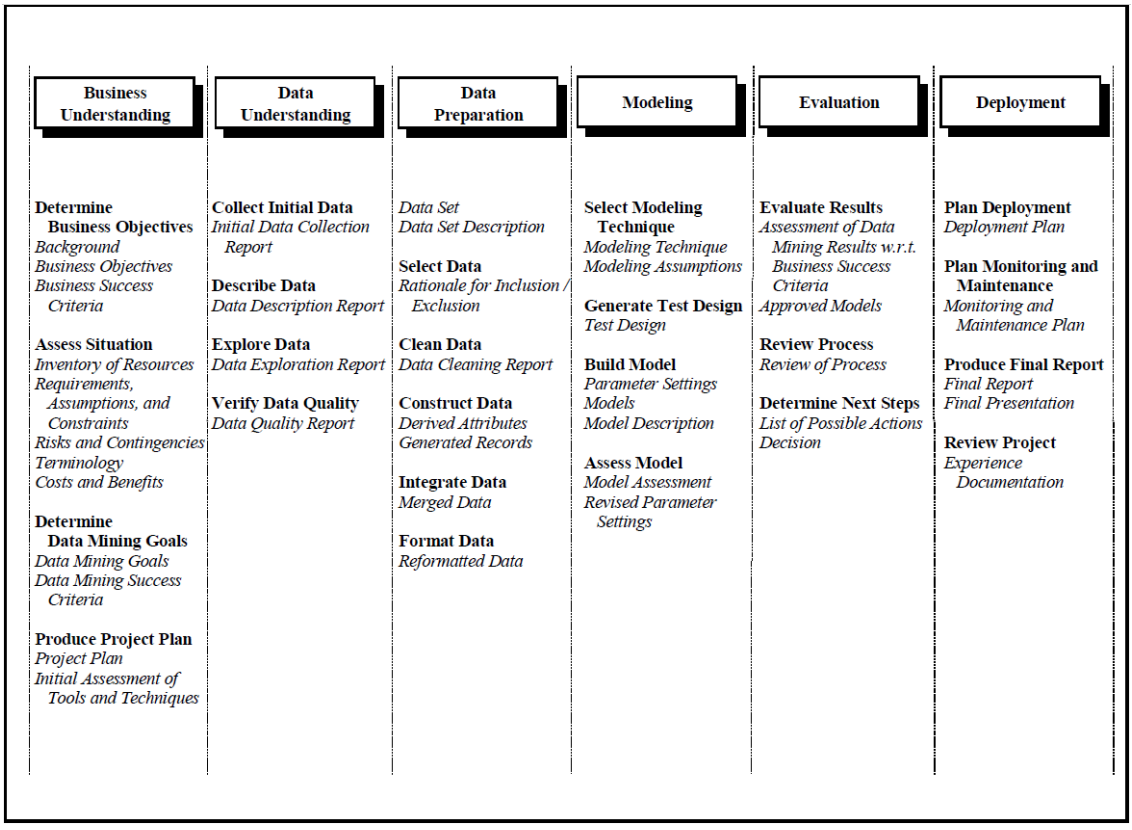


Figure 2.2: CRISP-DM framework details [45]

Data understanding

This step is about exploring data. Get all the data available, then explore how relevant this data is for the goal. Every possible data source should be taken into account, because no data should be missed as it can be very important data to consider when working towards a solution. When all data is collected, the data should be investigated and described. Describing is useful for traceability and also enables more effective teamwork. As other team members can understand the data much quicker when it is described for them already. In Chapter 5 the data will be explored and described, answering the second sub-question of this research.

Data preparation

Data preparation is a very big and important step in each data-driven project. As data is the foundation of every answer obtained, the data has to be a reliable factor. Even if you use the best model in existence if the data inputted into the model is of bad quality, the model's output will also be of bad quality. Although data cleaning is very important it is also very time-consuming and according to Forbes the least enjoyable task for a data scientist [13]. Another part of the data preparation is selecting which data to actually use and which data to exclude from the model. Lastly as this research works with customer journeys, all data needs to be formatted and integrated into the corresponding customer journey. This is all discussed in Chapter 6, meaning also the third sub-question will be discussed there.

Modeling

After all the preparation, the model itself will be created in this phase of the CRISP-DM cycle. A number of different machine learning techniques will be used to see what kind of technique is best suited for this problem. The techniques will be evaluated using metrics and also be compared with each other. This answers the fourth sub-question in the research.

Evaluation

In the evaluation phase, the model is already of high quality from a data analysis perspective. Therefore the results are assessed in line with the business criteria to see if the model fulfills the needs of all set business objectives. Furthermore the overall steps in the process are reviewed. These points address the fifth and final sub-question.

2.2 Process Mining

Process mining is described as the missing link between data mining and process science [41]. Meaning process mining resides between data mining and machine learning on one side and process modeling and analysis on the other. Process mining tries to discover processes based on real event data, to then monitor these processes and lastly improve them. These three aspects of process mining are called the discovery, conformance and enhancement of models [41]. In Figure 2.3 the link between real data and the process models is shown with the three types of process mining.

The discovery technique uses miner algorithms to transform event logs into process models that visualize the data in a clear way for humans. As we cannot see the trend in a big data source but we can comprehend a graph-like model. There is a variety of miner algorithms available, an incomplete list of miners available in one of the popular process mining tools called ProM [42]:

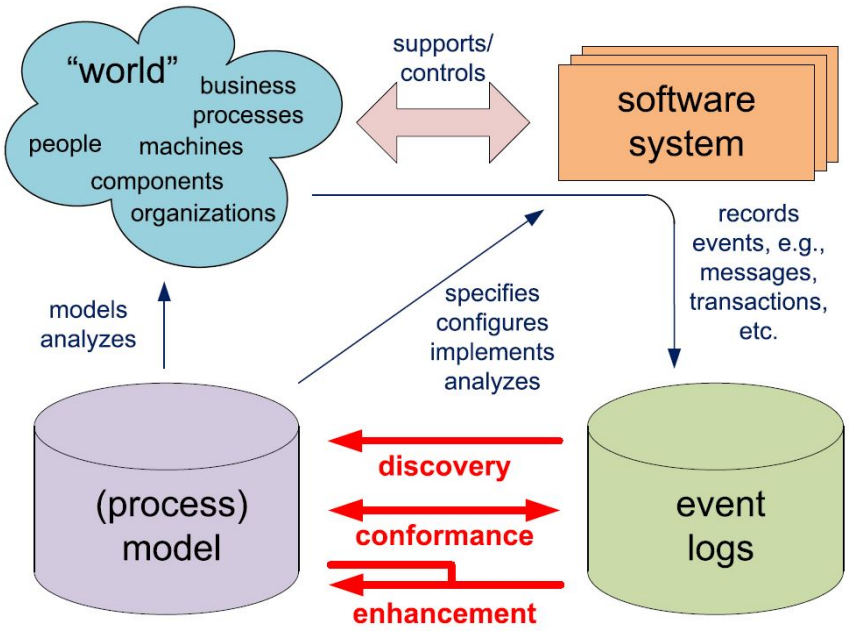


Figure 2.3: Process mining overview [41]

- Heuristic miner [44]
- Fuzzy miner [17]
- Inductive miner [23]

Process mining requires event logs to discover these models. Event logs are specified by a format called eXtensible Event Stream (XES). The XES format is an official IEEE standard as published by the IEEE Standards Association on November 11th, 2016 [2]. The XES standard contains three mandatory aspects. These are the traces, the events within each trace and a timestamp on each event. Additionally each event can have more information linked to itself such as the person executing the event or the category of the event.

Conformance compares an existing process model with the event log data of the same process. Conformance checking works both ways, we can check if the event log conforms to the invented process model but also if the process model conforms to the real data in the event log. Conformance is a good way to check if rules implied in the model are also enforced in the real data. In a process model there is always the trade-off between generalization and precision. The model should be general enough to also be able to handle future behaviour while at the same time be precise enough to not allow unwanted behaviour.

Lastly the enhancement of process models, this entails the improvement or extension of a created process model with data from event logs. There are two types of enhancement, the first being repair. The repair modifies a model to better reflect the actual real life process. Whereas the second type of enhancement is the extension which enriches the model with extra data to provide another perspective on the data.

2.2.1 Customer Journey Maps

A customer journey map (CJM) is a collection of customer journeys, which is the kind of data we use in this research. Customer journeys are the way companies capture the customer experience over a period of time across multiple touch points [24]. Touch points are the different kinds of

contacts a customer can have with the company, as there are ever increasing ways to contact companies with the norm going towards omnichannel management. The sequence of these touch points from a single customer, is the uniquely identified customer journey. A customer can have multiple customer journeys, as a single journey is tied to one experience the customer has with the company. The journey starts when a customer wants to acquire a service, report a malfunction, simply ask a question or anything else for which the customer contacts the company. The journey ends when the customer is helped and satisfied with the reason for which the company was contacted.

The reason why companies track the customer journey is to see if they can be improved to further satisfy the customer. The whole journey is tracked as this provides insight into the thinking and decision making of the customer. Which choice a customer makes at a certain point is of course influenced by the information the customer receives at that moment. Bernard and Andritsos [6] describe the main components of a customer journey map:

- **Customer**: The customer is the stakeholder experiencing the journey and making use of the services from the company. The type of person is dependent on the market of the company.
- **Journey**: The path a customer follows is his journey. There are two types of journeys, the first being the actual journey a customer experienced. This journey documents all steps a customer has taken at which point in time. The second journey type is the expected journey, this means the journey a customer is supposed to follow. This journey is often referred to as the ideal journey and is defined by stakeholders from the company.
- **Mapping**: Mapping is describing the customers' experiences and actions. So the company tracking their customers' actions.
- **Goal**: The customer begins his journey with a certain goal in mind. The customer is trying to reach this goal using the services from the company. The goal is important to know as it can be used to check if a customer took a detour or the optimal route to the goal.
- **Touch point**: A touch point is an interaction between the customer and the company. The same touch point can be used multiple times in a row, meaning there can be cycles in the customer journeys. Also not all available touch points have to be used in a single customer journey.
- **Timeline**: The timeline defines the duration of the customer journey. Starting from the first touch point and ending at the last one. The individual touch points also have timestamps making it possible to order the touch points in the sequence in which they occurred.

Bernard and Andritsos [6] also describe how customer journey maps are linked with process mining. The three main parts needed in an event log are also present in the customer journey map. Therefore the customer journey map is very similar to an event log. A customer journey map consist of multiple journeys which are comparable to the traces in an event log. The touch points in a journey are the events in a trace and each touch points has a timestamp just as an event has. For this reason customer journeys can be used to mine process models and analyze the data.

2.3 Literature on Customer Journey prediction

When searching for literature on customer journeys, most literature also includes a product. This makes them fundamentally different from the problem faced in this research. As in our case we want to study the customers and predict their journey, in the literature they want to predict which product a customer is likely to buy or what service is preferred by the user. For example the literature on OARA [14], this is based on customer journey prediction but also includes a recommendation afterwards, which deviates from the goal in our research.

One of the commonly used techniques in the literature are recommender systems, which are shortly discussed in Section 2.3.1. They are used by Terragni & Hassani [39] in an article on analyzing customer journeys for recommendations [38]. This research is useful to investigate as the basis is similar, again however we want to predict what a customer will actually do and not what a customer will like. Therefore we can investigate their sources and analyze how they tackled the problem and learn from that.

Another area found in literature is called sequence prediction. Specifically a technique called Compact Prediction Tree (CPT) [16] and the improved version CPT+ [15]. This technique will be discussed in Section 2.3.2. Moreover other techniques were investigated but were found unfit for this research. Among them are Markov Chains and pattern mining, they will however not be discussed.

2.3.1 Recommender Systems

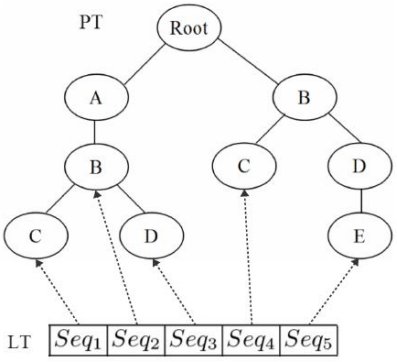
Recommender systems use a customer's past preference for a product or service to build a model [32]. There are two types of collecting the data to build a recommender systems, these are explicit and implicit [21]. In explicit feedback a customer rates the product or service himself. Then products or services similar to each other are computed and inputted into the model. The model is then used to predict a customer's preference in similar products by the rating. For implicit feedback the customer does not have to perform extra work. The preference of the customer is implicitly stated as we assume that the customer likes products or services on which the customer clicked and viewed. The lack of the customer clicking certain products or services also implicitly indicates the lack of preference for those products or services. We will now argue why recommender systems are not used for this research.

As stated before, a major difference is the fact that in a recommender system a product is needed for which the customer can express their rating or even just items which the customer has bought. In the customer journeys in this research there are no products or similar items. This problem can be avoided by seeing the touch points of the customer journey as the products the customer has bought. But even then the problem remains that the recommender system needs other similar products or in this case touch points. However as there is a set, finite number of touch points, we cannot link them well regarding similarity. Meaning the predicting aspect of the recommender system cannot be utilized well, as we do not want to predict what touch point an user will give a high rating but which touch point a customer will actually use next. This means that we deem recommender systems unfit for the problem opposed in this research.

2.3.2 Sequence Prediction

As a customer journey is a sequence, the area of sequence prediction sounds promising. As such a technique called Compact Prediction Tree (CPT) was found [16]. First a short introduction into CPT, then arguments will be given why this technique is unsuited for the research.

CPT uses the customer journey sequences to build a prediction tree (PT). Starting at the root, going down the tree with each step in the sequence as a node and using the end of a sequence as leaves. Then a pointer is added to the leaf to facilitate a lookup table which remembers the sequence name or id. These steps are illustrated on the left side of Figure 2.4. Also an inverted index (II) is kept to quickly check all entries in each sequence, the keys of the table are all the different sequences. Whereas the rows represent the nodes in the tree, i.e. the events in the customer journey. This is shown on the right side of Figure 2.4. Each key leads to a bitset which provides the entries of that sequence. If the bit is set to 1 for a certain key it means that the event is part of the sequence corresponding to that key, while a 0 means that the event is not part of that sequence.



II

Item	Seq_1	Seq_2	Seq_3	Seq_4	Seq_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	0	0	1	0
D	0	0	1	0	1
E	0	0	0	0	1

Figure 2.4: A Prediction Tree (PT), Inverted Index (II) and Lookup Table (LT) [16]

When the prediction tree, lookup table and inverted index are constructed, we can start to predict sequences. We want to find sequences similar to the input sequence s, this can be efficiently done by using a bitwise intersection of the entries in s and the inverted index. The intersection is a logical AND and yields all the keys in which s is a subset, meaning the sequences are similar to the input sequence s. Then using the pointers in the lookup table for the keys, we can traverse all the sequences similar to s. When traversing the CPT algorithm will store all entries starting after the last entry in common with s, this part of the sequence is defined as the consequent. The entries of all consequents are then counted and the entry with the highest score is the prediction's outcome.

There are several disadvantages to this technique, which are the reasons the technique is not used in this research. One of the disadvantages is the fact that this algorithm does not deal well with sequences who have multiple entries of the same kind. As in the counting process all entries are counted by keeping track of the number of occurrences. This is very different then having an entry appear twice in the same sequence. Possibly a renaming or adding suffixes solution can be found for this problem, but it will slow down the algorithm. The main disadvantage we found, is that the algorithm does not take order into account in two hindering ways. Firstly when doing the bitwise intersection the order is ignored, the same entries have to be present in the sequences but not in the specific order as the input sequence. This is necessary for us, as the order in which a sequence appears can be very impactful for the rest of the sequence. Secondly the order of the consequent is not taken into account and all entries are counted with equal weight. Meaning the predicted item does not have to be the directly next item but can be an item which is very common in every string at a certain point, for example the closing item. Due to these reasons among others, we will not use this technique for our research.

2.4 Machine Learning Techniques

Machine learning is a part of artificial intelligence (AI) [29]. We used to provide the computer with an algorithm to solve a certain problem. However sometimes we do not have this algorithm, but we do have a huge amount of data [1]. For example we cannot define a clear algorithm which classifies emails as spam but we can collect millions of emails for which we know that they are spam. Then we give this information plus millions more emails that are not spam, to a computer and we let the computer learn on what features an email can be classified as spam. We want the computer to learn the algorithm for us based on the data we provide. This is the core idea of machine learning.

In our email example, the data consist of the emails and the corresponding labels either spam or

not spam. The emails are the input data, while the labels are the output data. The input data consist of the features on which the machine learns and the output depends on these features. In statistics the output feature is therefore also known as the dependent variable, while the input features are known as the independent variables.

Machine learning is based upon mathematical and statistical theories. These techniques were priorly used for pattern recognition and they are described in detail by Bishop among others [7]. We can divide machine learning into supervised learning and unsupervised learning. For unsupervised learning we solely provide the data without the corresponding output labels, meaning in our example we would only provide the emails and not the labels whether the email is spam or not spam. Unsupervised learning is used for recognizing patterns and to find structure in the data by means of grouping or clustering it.

Supervised learning contains the input data as well as the corresponding output labels. Supervised learning is used to predict the label of the input data, thus to categorize the output of the data. In our example of the spam emails, supervised learning would tell us whether an email is spam or not spam. Within supervised learning there are two categories, regression and classification. The main difference between regression and classification is that regression predicts the output in continuous values, while classification predicts the output in categorical values such as class labels. For this research we will be using classification.

Scikit-learn

Scikit-learn is a Python module in which a wide range of state of the art machine learning techniques are incorporated [31]. This module will be used in this research and as such the algorithms used for the models do not have to be created from scratch. In machine learning the data is split between the independent and the dependent variables. The data is split into two sets, one being the dataset used in training the model and the other is the dataset for testing and validating the model. A standard splitting proportion is 70% train data and 30% test data. The data is split because we do not want to train the data and then use the same data to test the model, as this could skew the results of the model. Even though the algorithms are included in the scikit-learn library, we still need to tune the input parameters and therefore a short basis of the models used in this research will be described next.

Logistic Regression

Logistic regression is named after the function it uses, the logistic function is also known as the sigmoid function [20]. The sigmoid function is a S-shaped function and is illustrated in Figure 2.5. As can be seen the function values are within the range zero to one, this is very desirable in statistic modeling as the output can be used as probabilities and no normalization has to be applied. The sigmoid function is defined as:

$$f(x) = \frac{1}{1 + e^{-z}}$$

To now create the logistic model used in the logistic regression we have to fill in z with:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

Giving us the function:

$$f(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k)}}$$

The X_k are the independent variables used, where k is the number of features. The β are called the coefficients and they are weight estimates of the corresponding X parameters. β_0 is the intercept,

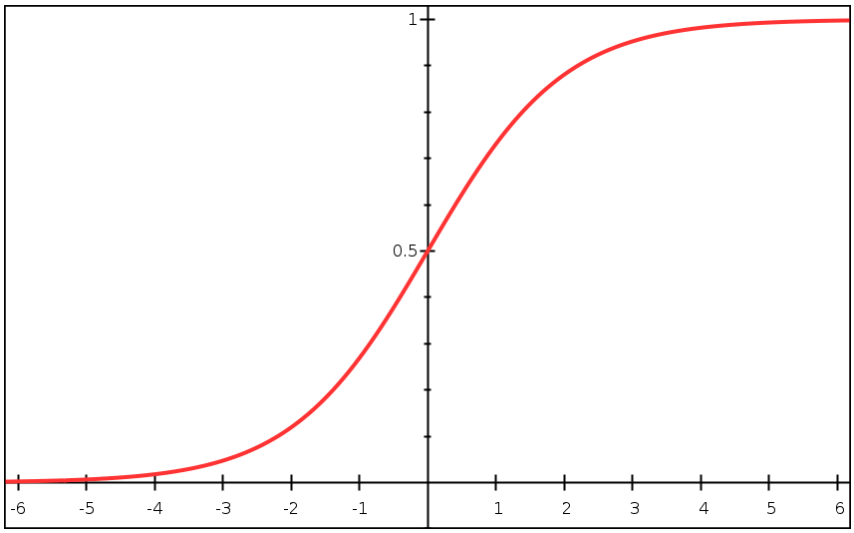


Figure 2.5: Sigmoid function

i.e. the base constant when all features are zero. The other β_i for $1 \leq i \leq k$ define the slope, i.e. they shape the line.

In standard logistic regression model the output is a dichotomous variable, meaning it has two possible values. This can be represented by the Boolean values 0 and 1. When training the model, the β values are tuned to maximize the likelihood of predicting a high probability when the output variable is a 1 and a low probability when the output variable is of class 0.

Random Forest

A random forest is a classifier build out of many tree predictors, with each tree independently sampled from values of a random vector [8]. These tree predictors are called decision trees and they are the building blocks of the random forest model. Decision trees create a tree structure by splitting the dataset into smaller subsets using rules [9], an example is shown in Figure 2.6. The decision tree creates rules for the internal nodes to split the dataset into subsets which contain different classes. Therefore the most impactful features are used at the top of the tree, like in the figure if you have work to do then you have to stay in. While otherwise it also depends on the weather if you want to go outside. The downside of decision trees is that they tend to overfit their problem because near the bottom of the tree the ruling gets overly specific and is not suited for other datasets anymore. This can be prevented by tuning the hyperparameters of the tree.

Whereas overfitting is a downside of decision trees, random forests are actually very robust against overfitting [25]. As said the random forest consist of many trees, however these trees should not be too correlated with each other. Because when they are it does not matter if you have one tree or hundreds of them, as the outcome will be the same for all. The random forest strength comes from the fact that the majority of models will be correct and the trees protect each other from individual errors. Therefore the random forest uses bagging to train the decision trees on different samples of the dataset. Bagging is taking random samples with replacement and it reduces variance and overfitting.

Another way to have less correlation is to introduce feature randomness. In a standard decision tree, all features are considered when splitting a node. Using feature randomness we only consider a random subset of features to make a split on. Using these two techniques the trees have lower correlation and more diversification. A random forest is illustrated in Figure 2.7. As we can see

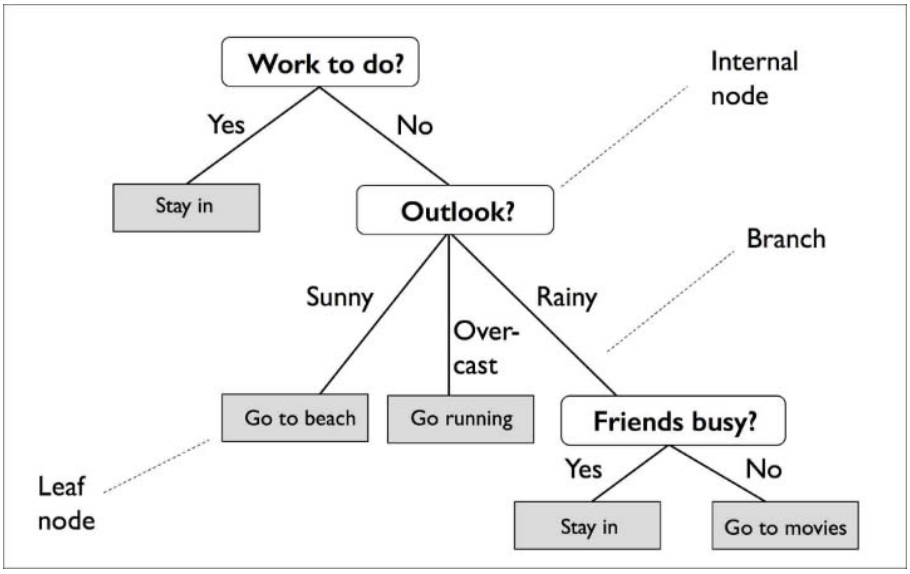


Figure 2.6: Decision tree structure [26]

there are n trees in the forest and each tree predicts a class, then all votes are counted and the majority vote determines the final outcome class.

Gradient boosted trees

Gradient boosted trees are like random forest made from multiple trees. However whereas random forest are based on bagging, boosted trees are based on the concept of boosting. Boosting uses multiple weak learners, meaning the trees used are very small and can even be decision tree stumps, i.e. only have one split. Boosting uses the fact that a set of weak learners create a single strong learner [34]. We will explain boosted trees using the example in Figure 2.8.

The algorithm starts off with a weak learner in this case a decision tree stump. Then it will add another weak learner to improve the result, this new weak learner is focused on the samples which the previous learner got wrong as can be seen in iteration 2. The algorithm will keep adding learners which improve the faults of their predecessors. Not every weak learner is added as they could also decrease the quality of the model. In gradient boosted trees this is done by gradient descent optimization, meaning there is a loss function which is minimized to create the best overall model [11].

Neural Networks

Neural networks are complex networks based on the human brain. Just like the human brain a neural network consist of a bunch of neurons connected with each other. Using the neurons the neural network learns from the data in a similar way as the brain does. The neurons are divided into three layers, the first is the input layer. This layer receives the data input from the user. The input layer then passes the information onto the hidden layer. The hidden layer performs all the mathematics and is therefore sometimes seen as a black box. A black box is a system of which you can view the inputs and outputs but know nothing about the internal workings. Lastly the hidden layer provides the data to the output layer, which provides the output.

In Figure 2.9 a feedforward neural network is presented, this figure is a basic neural network with only one hidden layer. A neural network can have multiple hidden layers and this will increase

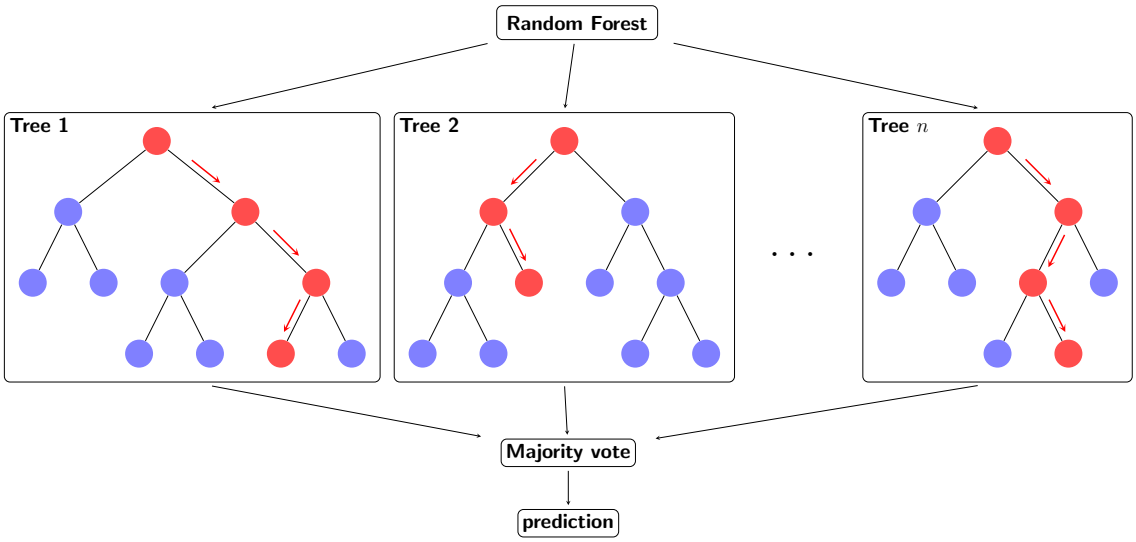


Figure 2.7: Random forest

the complexity the network can handle. A single neuron in the hidden layer can be seen in Figure 2.10. The neuron has several inputs x_i and their corresponding weight matrices w_i. The weight determines the strength of the input variable or in other words the weight determines how important the feature is [18]. The inputs are added while also including a bias b to offset the value, changing when the activation function triggers. The bias can be compared to the intercept in logistic regression, so a base constant. Lastly the summed value is passed through an activation function, this can vary but a commonly used basic function is the sigmoid activation function [28]. This is the same function as described in the logistic regression. The activation maps the value to the range between 0 and 1 in case of sigmoid or sometimes between -1 and 1 with other activation functions like tanh.

Even though neural networks are a great tool to explore many problems, they do also have some disadvantages. One of the disadvantages is especially problematic for this research. Standard neural networks do not have a short-term memory, when they process a sequence of data they have forgotten the first entry when they are at the next entry. As in this research we are working with customer journeys which are sequences, we need the neural network to have a short-term memory. For this we need a recurrent neural network (RNN), which is a class of artificial neural networks. In contrast to feedforward neural networks where signals only travel one way, from the input to the output. In a RNN the signals can be fed back into previous layers creating loops, as can be seen in Figure 2.11. This gives the network the ability to have memory. We will focus on a particular type of RNN named Long Short-Term Memory (LSTM), which was first introduced by Hochreiter [19].

Long Short-Term Memory

RNNs have memory but in practice they are unable to learn long term dependencies, this problem was documented by Bengio et al. [3]. To solve this problem the LSTM was invented, the inside of a LSTM neuron is shown in Figure 2.12. We will describe the basics of the elements in the neuron to see how LSTMs retain their memory longer [30].

The first layer consists of a sigmoid layer and a pointwise multiplication operation, this is called the forget gate layer. The sigmoid can be between 0 and 1 for each feature, meaning if a feature gets a 0 it will be completely forgotten and if it has a 1 it will be remembered in total.

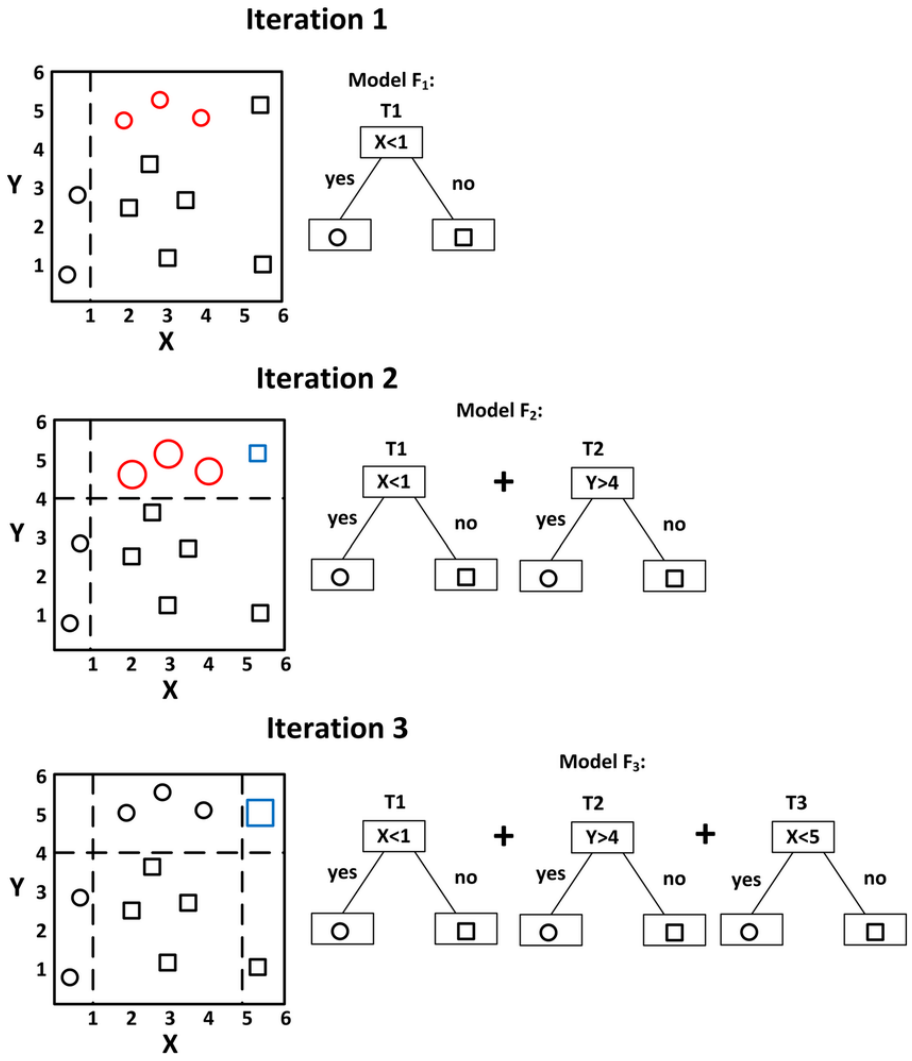


Figure 2.8: Boosted trees [49]

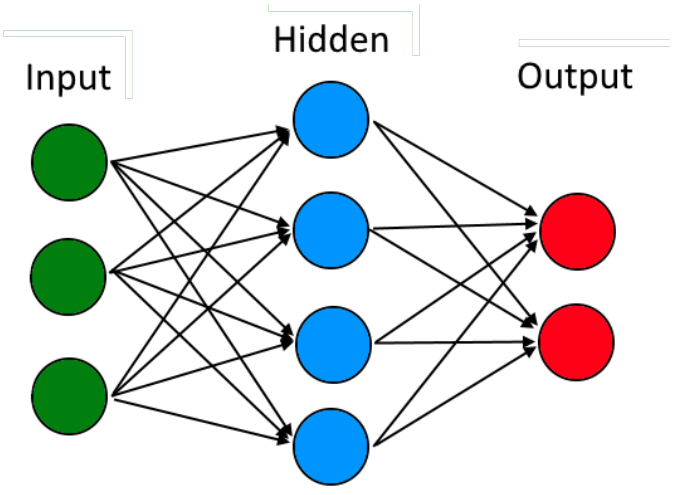


Figure 2.9: Feedforward Neural Network

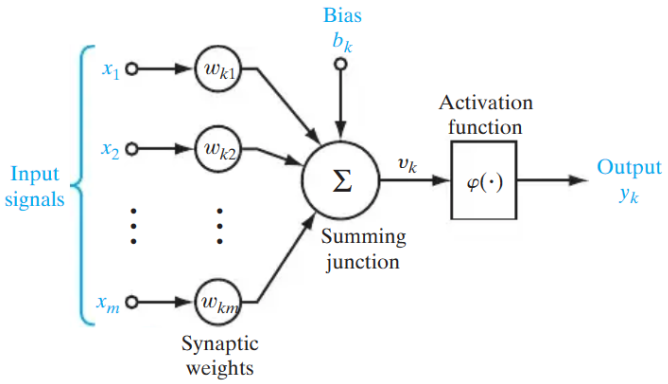


Figure 2.10: Single basic neuron [18]

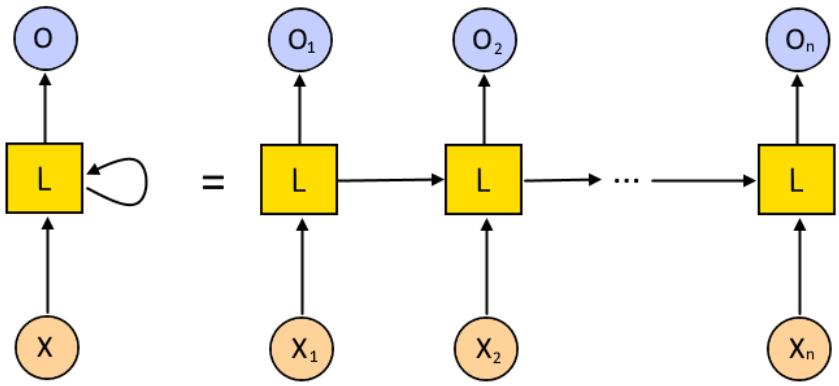


Figure 2.11: Recurrent Neural Network

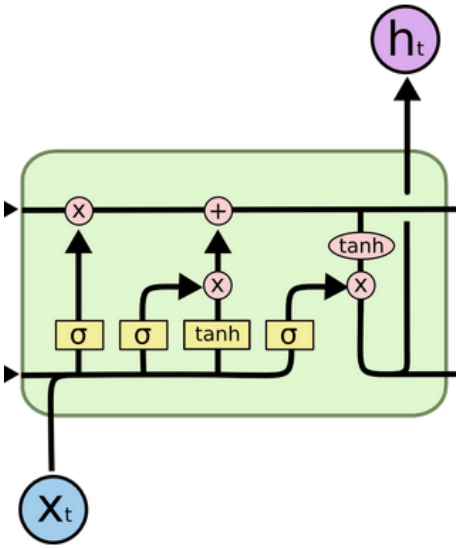


Figure 2.12: Single LSTM neuron [30]

The next part consist of two layers, a sigmoid and a tanh layer. This part is called the input gate layer and it provides the values we want to update. Again the sigmoid gives the importance of each value with the values between 0 and 1. The tanh layer creates a vector of values that will be added to the current memory state.

Lastly the layer which tells what data to output. This consist of another sigmoid layer which determines the importance of the values we will output. The tanh is there to transform the values into the range -1 to 1.

2.5 Theoretical framework

The data we are handling consist of customer journeys, as described in Section 2.2.1 customer journeys are very suited for process mining. The journeys have a clearly defined flow and they therefore can be seen as processes. We will explore the options of using process mining in this research. On the other hand we want to predict the next data point. This will be done with the data mining field and in particular with machine learning (Section 2.4). Data mining is more focused on finding patterns in the data. In this research we are exploring how we can combine the two fields, we explore to enrich one field with parts of the other field. We use the flow from the customer journeys as well as machine learning techniques from data mining.

As we need to predict, we will have to use the data mining techniques for prediction. However in data mining the data used is from a static viewpoint, for this the process mining aspect is used to retain the flow of the journey in the data. Another purpose of process mining is to better understand the data on which we perform the predictions.

Chapter 3

Preliminaries and problem formulation

In this chapter the problem is described in a formal way and an overview of the process is shown in Figure 3.1.

The data consists of customer journeys in the form of event logs. An event log $L = (Tr_1, Tr_2, \ldots, Tr_n)$ consists of a collection of traces. A trace is a sequence of events $Tr_i \in E^*$, where E is a collection of events. An event $e_i = (c_j, a_k, t)$ needs to include an uniquely identifiable customer journey c_j, an activity a_k from the set of possible touch points $A = \{a_1, a_2, \ldots, a_l\}$ and lastly a timestamp t when the event took place [41]. Thus an event log has a collection of traces, which contain events that denote the activity that happened on what time and referring to what customer journey.

In machine learning we use a dependent variable Y and independent variables X_i. These variables relate to each other in the following way if the problem is linear, $Y = w_1X_1 + w_2X_2 + w_3X_3 + \cdots + w_nX_n$. Where w_i is a weight belonging to the independent variable X_i. The X_i are the features which are present in the data, these features are the events $e \in E$. Thus the dependent variable is our output variable which relies on the independent variables. The independent variables provide information to compute the outcome of the dependent variable.

We are researching to predict a touch point in the customer journey. Formally this means that we want to predict the activity a of event e_{i+1} given that we are at event e_i belonging to the same customer journey c_j and having the timestamp $t_i < t_{i+1}$. We also have the information of previous events which will help determine the next touch point. Meaning that e_{i+1} is the dependent variable on our independent variables of the previous steps. Thus $e_{i+1} = \sum_{k=1}^{i} w_k e_k$, but we not only use touch points there is also additional information in the customer journey. Meaning if we would have two other features p, q in our dataset, we add them as independent variables as well in the following way:

$$e_{i+1} = \sum_{k=1}^{i} w_k e_k + \sum_{l=1}^{i} w_l p_l + \sum_{m=1}^{i} w_m q_m$$

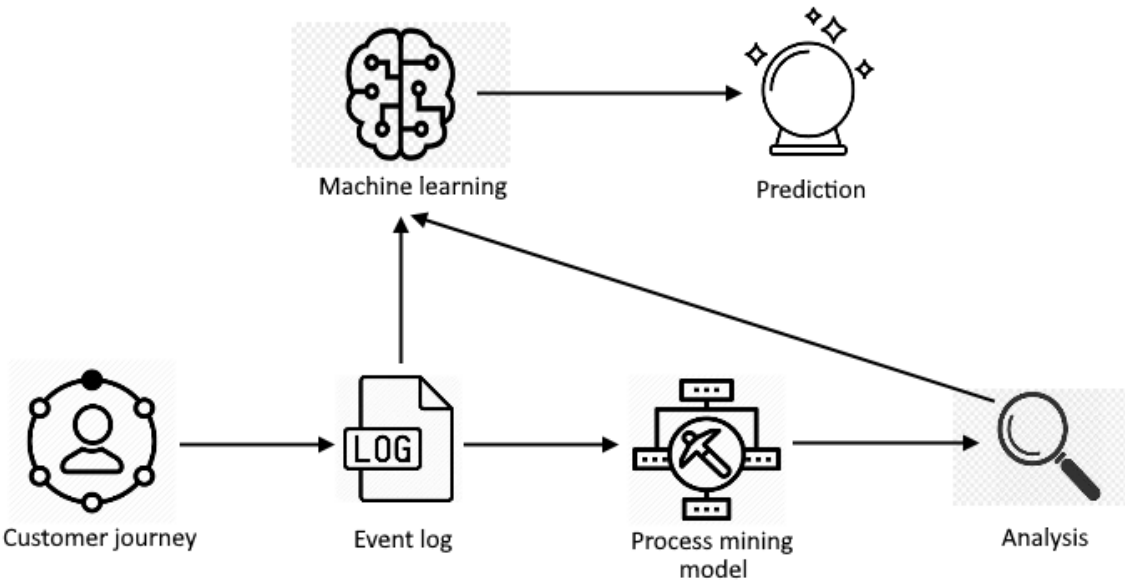


Figure 3.1: Methodology overview

In Figure 3.1 we can see the steps that can be used to perform predictions on the customer journey. This methodology includes these steps:

1. Identifying the data for the customer journeys (Chapter 4).
2. Understanding the data in the customer journeys (Section 5.2).
3. Preprocessing the data as event logs (Chapter 6).
4. Discovering process models of the customer journeys (Section 5.3).
5. Analysing the process models and data (Section 5.3).
6. Applying machine learning to the customer journey data (Chapter 7).
7. Using the prediction in a business case (Chapter 8).

The main goal is to try and predict what touch point a customer will use next. This prediction is required if the company wants to try and prevent this next step from happening. This can be done by helping a customer proactively or even better by making sure that the next step is never needed. This does not only save the company resources as the customer needs less attention, the customer will also be more satisfied as the goal of the journey is reached faster.

Chapter 4

Business understanding

First the business situation will be explained in Section 4.1. In Chapter 1 the context surrounding the business problem is introduced. In this chapter the underlying business problem is expanded upon in Section 4.2. From the business problem the business objectives and goals are extracted in Section 4.3.

4.1 Business Situation

KPN is a market leader for telecommunications in the Netherlands. They provide many different types of services for people living in the Netherlands, as well as different services for companies in the business market. The main services provided for customers are mobile telephony, internet for households and interactive television. The different types of services can be combined to get extra benefits on top of the packages.

As KPN has many customers and services, KPN provides various ways to get in contact with them. Why a customer wants to get into contact with KPN can be for many different reasons such as having a question about a service, an invoice or the installation of a new piece of hardware. Customers do not only come into contact with KPN for questions, but also for reporting malfunctions or acquiring a new subscription with KPN and a dozen more possible reasons. To facilitate these contacting customers, KPN has a number of channels to reach them.

Probably the most well-known channel is the telephone. Customers can contact the service helpdesk of KPN over the phone and ask for the information they seek. When they are not at the right department, their call will be put through to the right one. For customers calling is a reliable and easy way to come into contact with KPN. To reduce the number of phone calls received by the callcenter, an AI called conversational is put in place. Conversational comes into play before the customer gets into contact with a real employee. The customer is asked to state his question, then the AI will automatically classify the question. If this succeeds, the AI will send a link to the customer where the relevant information can be found on the KPN website. So for example if the customer says: “I do not know how to install my modem.” The conversational AI will send a link to the web page on which a guide to install modems can be found. If the classification does not succeed or the customer ignores the link, he will be put through to an employee of the callcenter.

Another way to contact KPN which is more self service, is the KPN website. On the website a customer can go to the Service & Contact section. In this section, first a broad division is made in the type of information required. The options consist of mobile phones, internet, television, landline and administration. When a customer selects a topic, he gets a list of much more specific

items relevant to the chosen topic. With this flow, a customer is guided to find the information he seeks from a top level view distilled down to a very specific issue. Next to this self service part of the website, there is also the option to start a live chat with a KPN employee. In the chat the customer can ask his or her question and the employee can respond with relevant information or ask followup questions. So a chat basically acts the same as a phone call to the callcenter. The last way on the website to get information is the KPN forum. On the forum a customer can find questions asked by fellow customers to see if they are relevant and already answered. If this is not the case, the customer can ask their own question. The question can then be answered by fellow customers or the moderators that are active on the forum.

Other ways to contact KPN are via social media platforms, like Facebook and Twitter. On these platforms customers can also ask for the information they want. An offline way to get into contact with KPN are the KPN stores. They are located all throughout the Netherlands, mostly in cities.

Customer journey

The customer journey at KPN is defined to end after a customer has not been in contact with KPN for at least seven days. Meaning that a customer journey can be of arbitrary time duration but the time duration between touch points is no more than seven days. Thus if we have a touch point tp_1 at time 0 days, another touch point tp_2 at time 6 days and a third touch point tp_3 at time 8 days from the same customer. Then this is all still a single customer journey even though the entire journey is longer than seven days as shown in Figure 4.1. This is the case because tp_2 is within seven days of tp_1 and tp_3 is within seven days of tp_2, which was the definition to be a customer journey. If tp_2 would not exist than the duration between tp_1 and tp_3 would be eight days which is more than the set seven day limit. Meaning that their would be two journeys, one including tp_1 and the other journey including tp_3 as shown in Figure 4.2.

The customer journey is a customer-driven process as the customer journey always starts with a touch point which is a contact made from the customer to KPN. Most of the touch points are also customer-driven, meaning that the contact comes from the customer and not from KPN. There are some static touch points like logistics and service ticket, these touch points are the followup from a journey started by the customer. As of now there are no touch points in the data of KPN initiating contact, like sending emails or starting phone calls. In the future these could be added to make the customer journey a more complete, overall process.

Most of the customer journeys are quite short, meaning of length one, two and three. The short length is actually a good sign for KPN because it means that the customer is helped within a few steps.

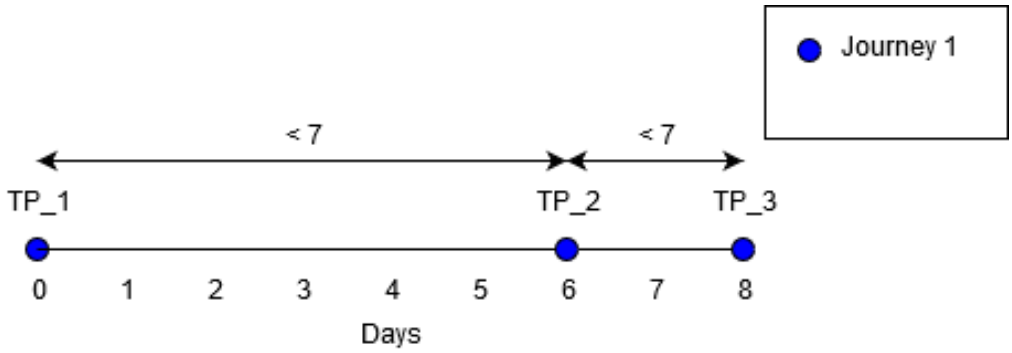


Figure 4.1: One customer journey

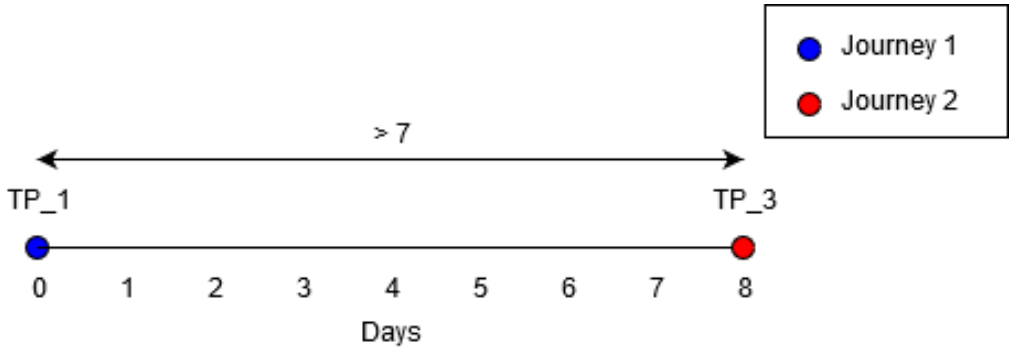


Figure 4.2: Two customer journeys

4.2 Business Problem

As many modern companies also KPN has a department focused on data-driven development. In this department most of the collected data is analyzed and scrutinized for improvements. Part of this is inventing new and innovative ways to inspect the data or create a new way of measuring the data. Another big role is linking different kinds of data to get insights into new perspectives, as this can be very helpful.

Regarding customer support, there are three high cost factors within KPN. As such those three are interesting to investigate and in context of this research, they are interesting to predict. Therefore let's elaborate upon the three main cost factors regarding customer support. In a perfect world a customer would never have to call KPN. All changes, sales, terminations and troubleshooting problems are being resolved using the online portal of KPN. As we do not live in a perfect world, KPN offers customer support through their telephone service helpdesk. The service helpdesk can be used in case of a question which could not be answered by the KPN website and there are always people who prefer to call instead of using self-service options. The latter group consist largely of elderly people who are not proficient with modern day technology and instead choose to use the already known to them technology, the telephone. However keeping the call centre operational is quite expensive. KPN provides around the clock support for malfunctions, technical support and questions about the theft or lost of a mobile phone. Contact support for all questions is opened from 08:00-22:00 [22]. Meaning every call made costs KPN money, this has already led to the creation of self-service portals online and robotic chat services.

Another big impact factor regarding cost is the mechanic. Sending a mechanic to a customer is costly for both KPN as for the customer. Mechanics need proper training and planning the mechanics in a way that travel time is minimized is hard and thus costly. The mechanic is unfortunately partly unavoidable, as he needs to fix issues on the customer side. Though there is

also a part which is unnecessary, it is hard to distinguish the latter from the former. Improving the quality of manuals might reduce the number of mechanics but they will always be needed. Even though, predicting that a mechanic will be needed can certainly help. As preemptive steps can be taken to offer a customer to receive a mechanic. Doing so will save the cost of a customer having to make a repeat call to KPN.

Lastly another defect which is costly, namely the swapping of hardware for the customer. When a modem or tv-box has a defect and needs to be replaced, it creates a lot of administration and logistics. Especially keeping track of all aspects of the swap in the logistics is not a trivial task. Even though a large part of the modems returned are not actually broken. The customer will send the malfunctioning modem back to KPN and KPN has to send the same model modem from their warehouse to the customer.

These three costs are part of the customer journey and loads of different kinds of research is being done to better understand and improve them. The main costs for these three touch points is overhead, they are often unnecessarily repeated. The costs of sending two mechanics is substantially higher than one mechanic who spends a bit more time at a customer. Not only those three items are investigated, the whole customer journey is under the loop and is being streamlined more and more.

4.3 Business Objectives

As KPN is a commercial company, the main objective is to retain customers which are subscribed to the services of KPN. To facilitate this KPN wants to maximize the customer satisfaction at the same time as minimizing the costs where this is possible, while still maintaining the highest quality support, products and services. The three cost factors described in Section 4.2 fall into the group which costs can be further reduced. This has led to an interest in being able to predict them, making it easier to handle them or even more ideal to prevent them. Especially the touch points which are repeated as they are most times deemed unnecessary. If a customer has called KPN with a question and then has to call again, the customer possibly could have been helped during the first phone call. This cuts out the service helpdesk employee which is the main contributor to the cost of a call. However to be able to do this, prediction about this phone call is required.

The objective for KPN is to resolve the issue of the customer within one contact, as all repeating touch points are considered excessive. To achieve this goal and besides the answer to the first sub-question is to get a prediction on what type of contact (i.e. touch point) a customer will use next and if possible also the subject for which the customer comes into contact with KPN. As it can make quite a difference if the customer calls for explanation on his bill or to cancel their subscription.

As this is their first research into predicting steps in the customer journey, KPN is interested in all possible insights they can get from looking at the data in this new way. The steps on the way to the final goal of predicting the next step are quite exploratory. Nonetheless these exploratory steps are useful for KPN to get new ideas and see the data from a different angle. As these ideas can be relayed to the correct department which can act upon these initial hunches.

Chapter 5

Data understanding

"It is a capital mistake to theorize before one has data." – Sherlock Holmes

Data is everywhere and a lot of our communication and actions are done online. This also entails that most of our activities are stored and can be accessed to analyze. Developing using data and the flow it provides is the current trend of data-driven technology. The start of each research based on a large amount of data consist of collecting the data and then inspecting the data to get an understanding. These actions are described in the coming sections.

5.1 Initial Data Collection

This research is in cooperation with the data analytics department of KPN, therefore they know how important data is. This is very helpful and causes the data to already be collected. At KPN all the data is stored in multiple online databases. The data can be accessed by the employees of the analytics department with the right permissions. Within this department there are also people working on customer journey, as predicted by Gartner that by 2018 60% of large organizations will develop in-house customer journey mapping capabilities [12]. KPN is busy with investigating the retrospective side of customer journeys, this means that the data concerning customer journeys is already ordered and linked from different sources into a format for the customer journeys. As the data is stored in databases, the data has to be extracted from there. Again this is an existing KPN framework, which we can use for this research. As such the SQL queries used to extract the data, will not be described here. These SQL queries are run and output multiple weeks of data. For this research we have two datasets with eight weeks of data, which both already include over a million rows of data and almost half a million customer journeys.

5.2 Data Description

In this section the initial investigating of the data is described. In Figure 5.1 a snippet from the initial dataset is shown. Later into the research we got the second dataset, which included a different column than the initial dataset. At first we want to make a division between important and less important columns. Inspecting the headers of the columns gives an idea of the data inside of the column. The data is also discussed with domain experts of KPN, who know what the column headers represent and what kind of data is in each column.

First we have identified the less interesting rows and will discuss these, the less interesting columns are *row_number*, *customer*, *start_time* and *end_time*. The column *row_number* is just an incre-

	A	B	C	D	E	F	G	H	I	J	K	L
1	row_number	journey_id	CUSTOMER	CONTACT_TYPE	BUCKET_NAME	CALLREASON	LOG1	LOG2	LOG3	START_TIME	END_TIME	CONTACT_TYPE_next
2	1	1	0	logistiek	Levering	Diversen - Doosje	Diversen - Doosje r		nvt	06-03-19 18:08	06-03-19 18:08	eind
3	2	2	0	logistiek	Levering	Retourdoos Glas	Retourdoos Glas		nvt	19-03-19 16:57	19-03-19 16:57	eind
4	3	3	0	monteur - levering	Levering		DUO-mon	nvt	nvt	17-04-19 15:53	17-04-19 18:19	eind
5	4	4	0	monteur - levering	Levering		DUO-mon	nvt	nvt	05-04-19 16:59	05-04-19 18:20	eind
6	5	5	0	monteur - levering	Levering		DUO-mon	nvt	nvt	17-04-19 14:35	17-04-19 17:04	eind
7	6	6	0	monteur - levering	Levering		DUO-mon	nvt	nvt	19-04-19 16:02	19-04-19 16:36	eind
8	7	7	0	conversational	TV	Digitenne	nvt	nvt	nvt	04-03-19 10:12	04-03-19 10:15	eind
9	8	8	0	conversational	Internet & bellen	Storing internet	nvt	nvt	nvt	02-04-19 10:35	02-04-19 10:43	call
10	9	8	0	call	Internet & bellen	Melden storing - I	Melden st	Internet	Internet b	02-04-19 10:36	02-04-19 10:43	logistiek
11	10	8	0	logistiek	SWAP	Garantiepakket m	Garantiep	Arcadya	nvt	03-04-19 10:37	03-04-19 10:37	eind
12	11	9	0	order	Termination	Termination: Con	Unknown	nvt	nvt	27-02-19 12:18	29-03-19 01:55	eind
13	12	10	0	conversational	Billing & collections	Factuur	nvt	nvt	nvt	28-03-19 16:44	28-03-19 16:59	eind
14	13	11	0	online	TV	Beleef: Films-Tv	Beleef Filn	nvt	nvt	20-04-19 22:05	20-04-19 22:05	online
15	14	11	0	online	TV	Beleef: Films-Tv	Beleef Filn	nvt	nvt	21-04-19 13:49	21-04-19 13:49	eind
16	15	12	0	online	TV	Beleef: Films-Tv	Beleef Filn	nvt	nvt	11-03-19 16:05	11-03-19 16:05	online
17	16	12	0	online	TV	Beleef: Films-Tv	Beleef Filn	nvt	nvt	12-03-19 18:28	12-03-19 18:28	eind

Figure 5.1: Initial data snippet

mental value to have the number of a row. This is not useful as input data for a model, thus this column will be dropped. The *customer* column is only important when personal customer information needs to be linked but for now all the data is set to 0. As we do not want to link customers yet before we have to. This is also in mind with the recent GDPR (General Data Protection Regulation) [40]. Lastly the *start_time* and *end_time* columns, these are useful because they are used to sort the steps in a customer journey into the right order. But they are very clear in their meaning and do not need explanation. They are not used as input for our model as the journeys are already sorted in the right order.

The columns that are interesting to use in the prediction model later will now be investigated and explained. The column with *journey_id* is very important as this is the primary key, i.e. uniquely identifies each journey. This id makes it possible to link different touch points together and see which steps a customer has taken in his journey. Without this information, there is no possible way to link the rows together to form a customer journey.

The column called *contact_type* is the type of contact a customer has with KPN, this attribute is what we have defined as a touch point in this thesis. Meaning this is the attribute on which we want to do the prediction, we want to predict which type of contact the customer will use next. Therefore all the distinct touch points will be looked at and briefly explained in the list below.

Touch Points

- **call**: This touch point means that a customer has called with the service helpdesk of KPN. The reason behind the phone call can vary a lot, from a malfunction to the theft of a mobile phone.
- **call - dvb**: This is when a call has to be forwarded to another department of the service helpdesk where employees are trained in other skills. For example if a customer wants to buy a product or service, he will be put through to the sales department of the helpdesk.
- **chat**: The chat occurs when a customer is on the KPN website and uses the online portal to ask a question to the chatbox. First a bot will respond but later a real life employee may continue the conversation if necessary.
- **conversational**: Conversational is when a customer calls the service helpdesk of KPN but before a real employee is on the phone. A bot will ask the customer to state his question, the bot then tries to classify the question. If the question is general, the bot will send a link

to the webpage on which an answer for the question can be found. Conversational helps to reduce the number of calls that have to go through to actual employees.

- **logistiek**: The logistics part is for the swap and distribution of hardware. This is a more static step in the process, as a type of hardware is requested and the warehouse has to perform the logistics to get it to the customer.
- **monteur - levering**: This means a mechanic for delivery. The delivery to a customer who has gotten a new subscription or an upgrade and wants a mechanic to perform the installation of the new modem or box.
- **monteur- ondergrond**: This stands for mechanic - underground. Meaning that there has to be done actual digging or crawling in the crawlspace of the house. The mechanic will then check and replace the cable(s) if necessary. A mechanic - underground is not often needed and mostly only after a regular mechanic - service has come by the house already.
- **monteur - service**: This is the regular mechanic for service. The mechanic is send when a customer calls with a malfunction which cannot be solved by himself or over the phone with the service helpdesk employee.
- **online**: Online is when the customer checks the website of KPN. This can be for everything on there, even if it is just browsing. Online can be hard to link to the actual customer as most people are not always logged in into their account.
- **order**: In this dataset an order has two categories, namely move and termination. Termination is when a customer cancels his subscription, so the provided services have to be stopped. Move is when a customer changes address and therefore the services, like internet and TV, have to be changed to the new address as well.
- **service ticket**: The service ticket is also like the logistics an intermediate, more static step. The ticket is created by a service helpdesk employee who has a customer with a problem on the phone. The service ticket states all the information needed for later reference if the customer calls again or for the mechanic to be informed about the problem.
- **winkel**: Winkel is dutch for store. So the store is when a customer walks into the store and speaks with an employee. The reason can also vary, it can be to buy a new product or to ask for information or even to report a malfunction.

For the next column we have a difference in the two datasets. The key feature of the initial dataset is an important column named *bucket_name*. In this column the bucket in which the touch point is categorized is shown. Not all touch points use the same buckets to be categorized into, which is interesting. We will check which touch points are connected to which bucket.

A few touch points do not have a lot of buckets. Like **monteur - service** has only a single bucket called *Service*. The same holds for **monteur - levering**, it only has a single bucket called *Levering*. The **monteur - ondergrond** has two types of customer complaints as buckets. Whereas the **service ticket** only has one bucket being a customer complaint.

As already described at the touch point paragraph, the **order** touch point has two categories: *termination* and *move*. The **logistics** touch point also has two categories but these being *delivery* and *swap*. Meaning it is either a delivery of hardware or it is a return of (malfunctioning) hardware.

The other touch points have quite some more buckets. The following touch points all have at least ten buckets in common where **conversational** has exactly only these ten. Here is a list of these ten buckets:

- *Bestellen*: Ordering
- *Billing & collections*: Monthly bills and payment

- *Internet & bellen*: Internet and telephone subscriptions for households
- *Loyalty*: Customer loyalty
- *Mobielinternet & bellen*: Internet and telephony subscriptions for mobile telephones
- *Product & propositie*: Product and value proposition
- *Sales*: Sales
- *Service*: Service with problems
- *TV*: TV packages and subscriptions
- *Undefined*: Missing information

Besides these ten buckets, **call**, **call - dvb**, **chat** and **winkel** have another four additional buckets. These being *Customer relationship management*, *Nog niet toebedeeld* (not yet categorised), *Ontvangen & installeren* (received & installation) and *Unknown*. Lastly there is the touch point **online** which also has the ten buckets and additionally *Nog niet toebedeeld* (not yet categorised), *Ontvangen & installeren* (received & installation) and *zm* (business market). This concludes all buckets and how they are linked to their respective touch points.

In the newer dataset we have a different key feature column. This dataset contains a column with the category of the customer journey. This new segmentation of customer journeys was developed by KPN in parallel with this research and when it was released we wanted to compare this new dataset with the old dataset. The category can be one of ten and is meant to give a clearer view on which track the customer is. The idea is that primary indicator touch point define the category and surrounding secondary touch point inherit the category. This creates a smoother journey defined by a category and it is then clearer to see if a customer is performing two journeys in parallel. Like for example when a customer is renewing his mobile subscriptions but at the same time he has a disruption in his internet. This is logged as one journey but with these categories we can see that it is exactly about two different types of journeys running parallel. The ten different categories are:

1. *New customer*: Becoming a new customer at KPN.
2. *Buy more*: Buying more products or services.
3. *Renew mobile*: Prolonging your mobile phone subscription
4. *I change*: Changing your order or subscription or service
5. *Move*: Moving house
6. *Disruption*: A problem with your service or product
7. *Payment*: Information regarding invoices and payments
8. *Use*: The general use of products and services
9. *Churn*: A customer who is churning, meaning leaving KPN as a customer.
10. *KPN changes*: When KPN changes a product or service relating a customer.

The columns left to inspect are *callreason*, *log_1*, *log_2* and *log_3*. These columns are related as most of the time *callreason* is the concatenation of the three *log* columns. However sometimes the *callreason* is not filled in or only the *callreason* is filled in but the *log* columns are not. The idea behind the *log* columns is that the information in *log_1* is more general than the information in the other two *log* columns. So the most specific items are mentioned in *log_3*. However in general *log_3* is empty and *log_2* is as well.

The *callreason*, *log_1*, *log_2* and *log_3* columns are all manual, human input. This means that the data can be a complete sentence written by a human, this leads to a large number of slightly

different entries. However for touch points involving callcenter employees the *callreason* is a concatenation of the three *log* columns. These *log* columns are logged by the employee using a standard list of options, out of this list the employee can select the subject of the call/chat/service ticket. Meaning that the *log* and *callreason* columns of these touch points are much more structured.

In this section the second sub-question has been answered. The question being: Which data regarding the customer journey is currently available at KPN? The answer: The data described in this section. Which are the current touch point, this lets us know on which step of the journey the customer currently is. The previous touch points which are indirectly provided because the journeys are linked by unique id and each step is registered, so we can induce the previous touch points in the journey. The buckets or categories that provide us with a clearer indication of the type of customer journey and lastly the callreason, which provide more detailed information on the touch point.

5.3 Data Exploration

Now that we have the data available, we will explore the data further and get some insights. We have reviewed and shortly explained all touch points. Now we will look at the relations between the touch points.

5.3.1 Touch point occurrence

At first we inspect the number of occurrences for the different touch points. In Table 5.1, the frequency of each touch point in the dataset is shown. All touch points were counted and then normalized, which is the result visible in Table 5.1. At the top we see the expected, namely the **eind** touch point which involves 48% of all entries. This makes sense because each customer journey ends with **eind**. In general our customer journeys are not long meaning that **eind** occurs very often.

Next we see a large number of **calls** and **conversational**. As this is the primary way for people to contact KPN, this also is no surprise to see. A good observation to see is the **online** touch point comes fourth in our frequency list. This could either be people browsing a lot but hopefully also solving their problems using the online guides, as this would reduce the number of phone calls. Near the bottom we see the **monteur - levering** and **monteur - ondergrond**, each less than 0.5% of the total entries. This is certainly expected from **monteur - ondergrond** and very good to see reflected in the data.

5.3.2 Variants and process overview

The journeys are very different for each customer but it could be the case that multiple customers have the same journey as there is only a limited number of touch points. To get insight in this we used a process mining tool to load all journeys from the data and show us the variants, this is shown in Figure 5.2. First of all we observe that the most common variants are all of short length. This is a good sign because it means that the customer was helped quickly. However it makes it harder for the prediction. As we see that the number of journeys that are not of length one, are all of very small sample size and there are in total 16567 different variants. This means that there are no standard journeys in which we can categorize. For prediction this would be very helpful as we could try to deduce the patterns of these standard journeys.

Because there are so many variants in the customer journey, a clear process overview is hard to find. With the standard frequency at 0.1 there are almost no connections between the touch points

Baseline of touch points	
eind	48%
call	21%
conversational	13%
online	7%
logistiek	3%
service ticket	2%
monteur - service	2%
order	1%
winkel	1%
chat	1%
monteur - levering	0%
monteur - ondergrond	0%

Table 5.1: Baseline touch points in the data

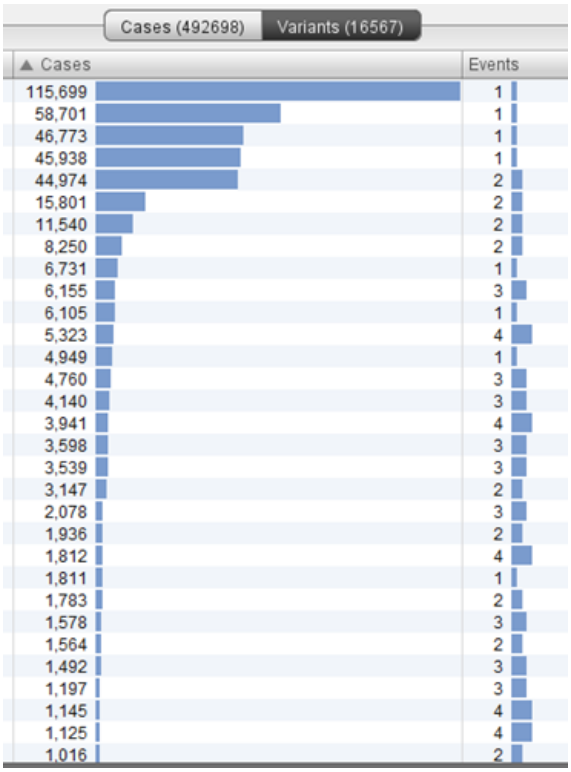


Figure 5.2: Variants of customer journey

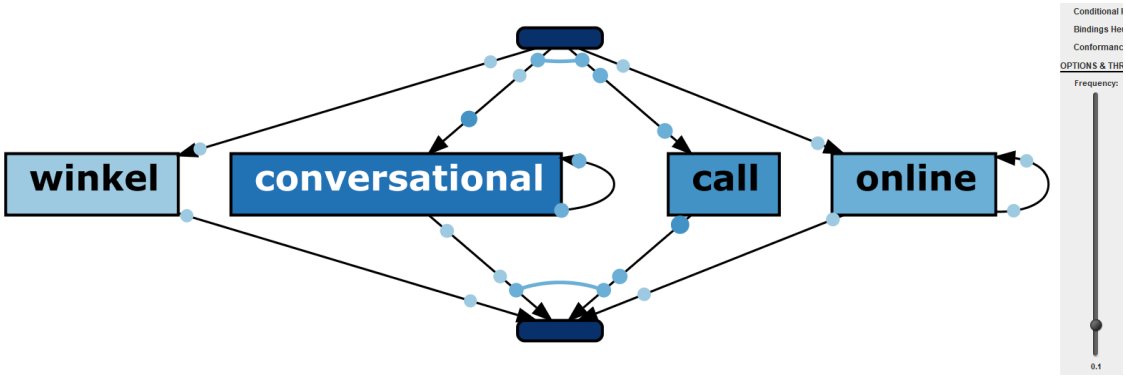


Figure 5.3: Process overview using heuristic miner

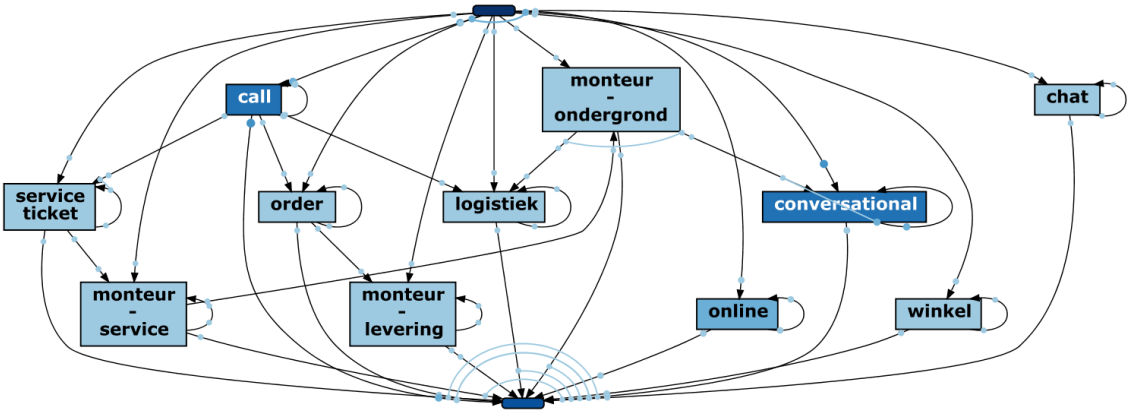


Figure 5.4: Process overview using heuristic miner

as can be seen in Figure 5.3. This is expected because of the many variants. In Figure 5.4 the process overview, made by the heuristic miner, can be seen with the frequency set to 0. In this figure we do see some connections.

We observe on the left of the figure that a **call** often leads to an **order**, **service ticket** or **logistiek** (logistics). These are interesting observations and when thought about make sense. When a customer calls KPN it can be about a malfunction or other question, this leads to a **service ticket**. The call can also be about acquiring a service, thus **order**. Moreover it could be a defect piece of hardware, leading to **logistiek** (logistics).

From the **service ticket** a followup is the **monteur - service** (mechanic - service), which is expected. The same holds for an **order** inducing the **monteur - levering** (mechanic - delivery). When a **monteur - service** is not enough to fix the problem a **monteur - ondergrond** (mechanic - underground) is sent, as already stated in the explanation of **monteur - ondergrond** in Section 5.2. So this dependency is also no surprise to see.

The link between **monteur - ondergrond** → **logistiek** and **monteur - ondergrond** → **conversational** is not immediately clear. Possibly for **monteur - ondergrond** → **conversational** the customer calls to ask for an update while the **monteur - ondergrond** is still working on the problem.

5.3.3 Journey inspection

The process overview above does not give an indication of the numbers behind the links in the customer journeys. Therefore a simple overview from the start of the customer journeys is created per touch point. In the overview, Figure 5.5, all journeys starting with **call** are displayed. The journeys are inspected up to a depth of four, where the remainder of the journeys is cut off. The leaves consist of the number of journeys following the touch points in the nodes. The leaves are shown per mille of the total number of journeys starting with **call**. Lastly we pruned all leaves which are less than 2‰ for readability. We observe the most frequent journey is a single **call** and then ending. This happens in 60,7% of the customer journeys, which is an enormous part. This proves for **call** our statement that most of the journeys are of very short length.

To further understand the data, another graph was created. This graph looks at the next touch point in a journey of consecutive identical touch points. Meaning the graph displays the percentages of the touch points which are next, following the touch point currently highlighted in the graph. More formal, on the x-axis the number of times the currently highlighted touch point is already repeated is shown and on the y-axis the percentage of next touch points is shown based on the total number at that depth. For significance, the number of journeys at each depth has to be at least one hundred.

Two of these graph are displayed in Figure 5.6 and Figure 5.7 for **call** and **conversational** respectively. These graphs are worth noting and help with getting insight in the data. For the first graph about the **call** touch point, we observe for the lower repeating touch points that most journeys end. But when a customer consecutively calls KPN the chance that he will call KPN again grows larger. This is an interesting insight and this might point to a subject worth investigating for KPN. Checking what these customers that call six times in row have in common, can help to improve a certain aspect that is the cause of this many number of calls. But for this research it is good to know these kind of ground truths, so that if we make a prediction on a sequence of **calls** the prediction most likely will also be a **call**. For the **conversational**, Figure 5.7, we see a similar trend as with **call**, the more often **conversational** is consecutively used the higher chance that the next touch point will also be **conversational**. However at the start we see that **conversational** leads to a **call**, which makes sense. Because either the **conversational** helps the customer or the customer will be put through to an employee of the callcenter.

5.3.4 Distribution of journey types

For the new dataset we looked at the distribution of the journeys regarding how many types one journey includes, which can be seen in Tables 5.2 and 5.3. Meaning in this dataset each customer journey has a category which is one out of the ten types of journeys. These tables show how many different journey types are included in one customer journey. We see that when ignoring customer journeys of length one, the biggest value is when there are two journey types in one customer journey. This is closely followed by just one type per journey. More than two types per journey has a much smaller percentage. Another interesting observation is the fact that there are a lot of customer journeys of length one, there are almost a 100k of them.

To check more in depth, we looked which types are often together in a journey in Table 5.4. This table is symmetrical on the diagonal as we only look at correlation. The biggest correlations we observe is between 8) Use and 6) Disruption, this could be explained by the fact that people are trying different ways to solve their disruption and some usages are logged in the 8) Use type wrongly. Another big correlation is 8) Use and 4) I change, as customers will probably use the services after or before changing them. Also again this could be explained by the fact that a customer is using all kinds of services and some are logged as 8) Use instead of the bigger overall journey 4) I change. We see that 8) Use is in correlation with a few other types as well and all of these could maybe be explained by the same reason. 6) Disruption is in correlation with 2) Buy

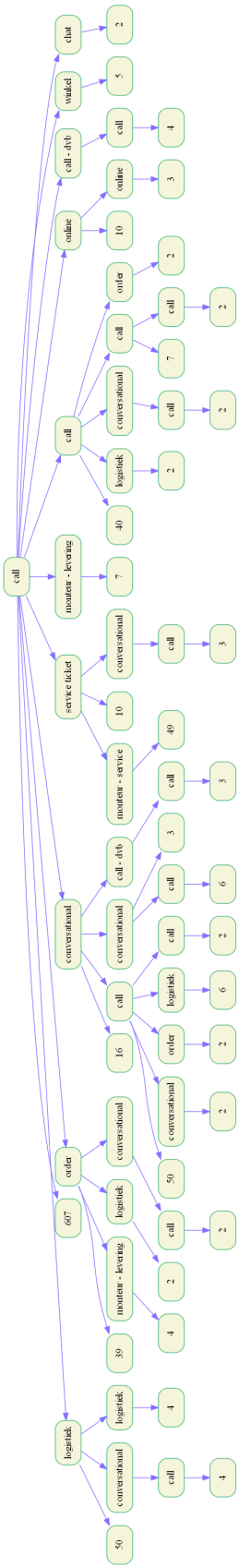


Figure 5.5: Customer journey mapping starting with call per mille

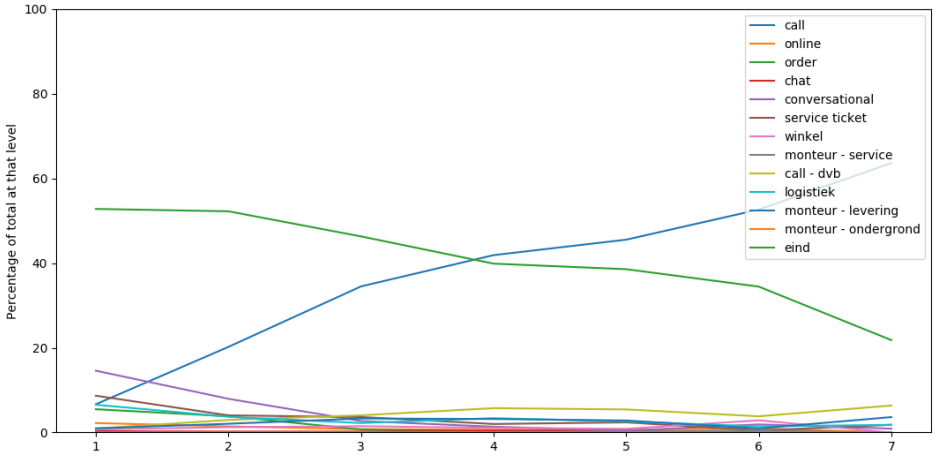


Figure 5.6: Consecutive call as touch point

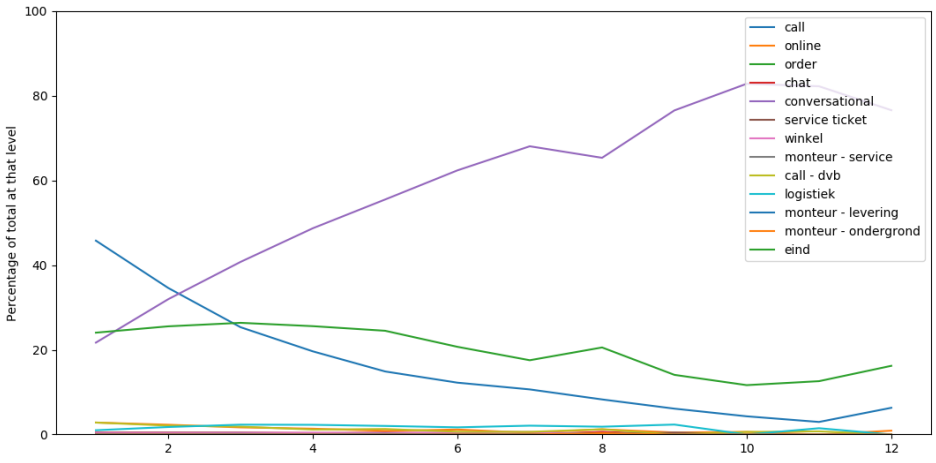


Figure 5.7: Consecutive conversational as touch point

Number of types	Percentage	Absolute
1	39%	55914
2	40%	57532
3	14%	20731
4	5%	6752
5	1%	2078
6	0%	539
7	0%	91
8	0%	15
9	0%	1

Table 5.2: Journey distribution without length 1

Number of types	Percentage	Absolute
1	63%	149606
2	24%	57532
3	9%	20731
4	3%	6752
5	1%	2078
6	0%	539
7	0%	91
8	0%	15
9	0%	1

Table 5.3: Journey distribution with length 1

	1)	2)	3)	4)	5)	6)	7	8)	9)	10)
1) New Customer	0	1276	1230	4046	194	2255	1359	2381	732	53
2) Buy More	1276	0	5563	7442	659	10233	4788	10593	2702	179
3) Renew Mobile	1230	5563	0	9972	1260	7334	5894	10849	3532	126
4) I change	4046	7442	9972	0	1914	11522	9107	16325	5047	490
5) Move	194	659	1260	1914	0	1626	1179	2223	1504	26
6) Disruption	2255	10233	7334	11522	1626	0	6102	17532	3357	229
7) Payment	1359	4788	5894	9107	1179	6102	0	10237	2906	113
8) Use	2381	10593	10849	16325	2223	17532	10237	0	5046	255
9) Churn	732	2702	3532	5047	1504	3357	2906	5046	0	77
10) KPN Changes	53	179	126	490	26	229	113	255	77	0

Table 5.4: Correlation between journey types

More and 4) I change. The disruption may be caused by the fact that the customer acquired new services or changed a service and the service does not work as intended or as imagined by the customer and therefore a disruption is logged.

Chapter 6

Data preparation

"By failing to prepare, you are preparing to fail." - Benjamin Franklin

For the third step in the CRISP-DM cycle, we will prepare the data. Using the business understanding from Chapter 4 and the data understanding from Chapter 5, we select the data features which are relevant to use for this research. Afterwards we will clean the data as necessary, to be of optimal use for the models. Lastly the data is formatted into the correct shape to use as input for the models.

6.1 Data selection

In Chapter 5 data understanding, the relevant data is already shortly discussed but in this section the rationale behind the decisions will be further explained.

Dependent variable

The most important data is the output variable called the dependent variable. In general when making predictions the dependent variable will be the prediction data. So as Y select the column that contains the data about the prediction made. Then check if this column needs to have noisy data removed or some reformatting.

In this research the predicted variable is the data in the column *contact_type_next*. In this data we want to change the options available for prediction. Discussing with KPN has lead to the conclusion that the **call - dvb** (call - forwarded) touch point needs to be excluded from the data. From a prediction perspective this touch point does not make sense, as we want to predict which contact type a customer will use next. In the case of **call - dvb** the contact type is a **call**, but the **call** is forwarded between teams internally. For the customer this does not make a difference, as it is still a single **call** for them. The reason we exclude this touch point from our data is because it causes extra noise for the next touch point after **call**. Also predicting **call - dvb** has no added business value for KPN. The **call - dvb** happens naturally after a **call** is made by a customer and he has to be put through, so predicting this has to be done immediately or else the call is already forwarded. How the data is cleaned from **call - dvb** is discussed in the next Section 6.2 on data cleaning.

Independent variables

For the independent variables X_i the columns with information have to be selected. For the customer journey this will always be the column containing the touch points, as the touch points are the most important feature in a customer journey. Besides the touch points some additional information regarding the reason of the contact can be added, like a callreason or category. If it would be helpful then also customer information can be added to provide more personal information to the model.

In our research the most important feature is the touch point, i.e. the column *contact_type*. This column contains the current touch point in the journey and has the same values as the outcome variable minus **eind**. Meaning that also for this column the touch point **call - dvb** needs to be removed. Furthermore since the contact type is of great importance for predicting the next contact type, we also want to include previous touch points belonging to the current journey. The steps taken in a journey can be of utmost importance to predict the right outcome. Since a customer can already be following a certain track for which there are clear next steps, in this case knowing these previous touch point will greatly improve the accuracy of the prediction.

To provide meaning to the touch points, the column with *bucket_name* is used. The bucket provides a sort of category for the current touch point. Since this category provides extra information for the touch point, it is important to know and will help to improve the predicted result. As we also include previous touch point, we will likewise include the previous buckets in our data. This for the same reason as with the touch points, if the buckets indicate the customer is taking a clear path. Then we will certainly improve the result by adding the previous buckets in the journey.

Another data feature which supports the information surrounding the touch point is the *callreason*. As described previously the *callreason* consist of a written or selected reason made by a KPN employee. As such there occur many different variations in *callreasons* of which some are very similar to each other. They can even be as similar as using different punctuation or capital letters. This makes them less suited for the use in prediction, however they contain valuable information. This is why we will perform a basic cleaning performance on the *callreason* data, so it is usable for our models. Because there are still many options available for *callreason*, we will only use the directly previous *callreason* and the current *callreason*.

6.2 Data cleaning

Cleaning the data is a very important step in data preparation. The real world is impure which leads to noisy, incomplete and inconsistent data as said by Zhang et al. [48]. In every project you need to check the data for noise and decide how to remove the noise, this can be done by simply removing the entire row or by filling in gaps in the data.

To reduce the noise in our data we have removed the touch point **call - dvb**. We simply do this by removing the entire row in which **call - dvb** is the *contact_type*, because we will run a script later that reshapes the data in the right way.

For the *callreason* we had to reduce the number of variations in similar entries as there is no similarity measure in our prediction techniques. Optimally this would be done with the help of natural language processing (NLP) [27], but this can be a whole research on its own and thus deemed out of scope for this research. Therefore we made the decision to limit the *callreason* to a maximum of ten characters. We chose ten because it cuts off all too specific parts of the reason while still providing enough room within the first ten characters to be different.

Lastly we had the buckets *undefined*, *nog niet toebedeeld* and *unknown*. These are all non-informative buckets and therefore we aggregated them into one bucket instead of three separate

buckets. We looped over the data and whenever we encountered one of the three buckets, we replaced the entry with *missing*.

6.3 Data formatting

Lastly the data needs to be formatted in the right way. When the data needs to be in the shape of an event log, there needs to be an uniquely identifiable journey, an activity and a timestamp. For the use in machine learning most data has to be converted to numbers so label encoding. With categorical data the data has to be one-hot encoded to prevent metrics like mean and average from being applied to them.

In the initial dataset of this research each row contains a step in a customer journey, as seen in Figure 5.1. For the prediction we want our models to also have the data of the previous steps. Therefore we will add columns to our dataset which contain the previous data steps in the journey. We will do this for touch point, bucket/category and callreason. For callreason we will only include the previous data but for touch point and bucket/category we will add seven columns, one for each previous step up till seven.

We will perform this task with a Python script for which a pseudocode algorithm is shown in Listing 6.1 for the touch points and callreasons. The same method is applied for the bucket/category. First we create variables which will hold our previous touch point and previous callreason. We also create empty lists for both touch point and callreason to which we will append the data. We loop over all rows in our dataset and for each we check if the journey id is the same as the previous journey id. If this is the case we append the previous touch point and callreason to our lists. Else we add an empty string to our lists. Lastly we update the journey id, contact type and callreason. The callreason is limited to the first ten characters as explained in the section above. After this script we add the lists as columns in our dataset. Now we have our transformed dataset that we will use for our models.

```
previous_journey_id = 0
list_previous_touch_points = []
list_previous_callreasons = []
previous_touch_point = ""
previous_callreason = ""

for row in dataset:
    if previous_journey_id == current_journey_id:
        list_previous_touch_points.append(previous_touch_point)
        list_previous_callreasons.append(previous_callreason)
    else:
        list_previous_touch_points.append("")
        list_previous_callreasons.append("")

    previous_journey_id = row['journey_id']
    previous_touch_point = row['contact_type']
    previous_callreason = row['callreason'][:10]
```

Listing 6.1: Transform dataset

Now we have all our data that we want to input into our models. There is one last step we need to perform before the data can be used by the models. As the models cannot handle nominal values as input. Therefore we will have to use one-hot encoding of the features. As all our features are nominal features we have to apply one-hot encoding to each column. Nominal values are also called categorical values and each value describes a different category. This means that no ordering or other measures can be applied to them. As most machine learning models only accept numbers we have to encode the input. If we just label encode the input as shown in Table 6.1 then some machine learning techniques still perform measures on the data. For example if we would have

Table 6.1: Label encoding

Car brand	Label encoded
Opel	1
BMW	2
Volvo	3
BMW	2

Table 6.2: One-hot encoding

Opel	BMW	Volvo
1	0	0
0	1	0
0	0	1
0	1	0

an Opel which is represented by a 1 and a Volvo which is a 3 then the average would be (1+3)/2 = 2. Thus the average of an Opel and Volvo would be a BMW, this does not make sense at all. Therefore we need one-hot encoding, this creates a new column for each value and only that value is a 1 and all other columns have a 0. An example of this can be seen in Table 6.2.

6.4 Conclusion

In this chapter we discussed the data. We first selected the relevant data namely the touch points, the bucket/category and the callreason. Then we cleaned the data of noise by amongst other things removing the **call - dvb** touch point. Lastly we formatted the data in the right way to be used as input for the training of our models by one-hot encoding it. This last part also answers our third sub-question which was: What data is directly usable in the context of predicting the next touch point? The answer is that no data is directly usable but after some cleaning and most important, after one-hot encoding the data it is usable to predict the next touch point. If new categorical data would have to be added, this can be done easily after one-hot encoding it.

Chapter 7

Modeling

There is no such thing as a free lunch, this saying is also true when choosing the perfect model for a problem. Therefore we have to test multiple different models in order to find the one best suitable for our problem. This is named the 'No free Lunch theorem' and is derived in 1996 by Wolpert for machine learning [46]. A year later the paper released for optimization with all the mathematics behind their theorem [47]. As already described in Chapter 2.4, we will use four different kind of modeling techniques. We will use a logistic regression, random forest, boosted trees and a LSTM neural network. Then we will measure the performance of the models with a metric and assess them.

7.1 Models

We have four different models, here we will describe how we tuned the parameters for each of them. As the training of some models takes quite some time, we use a tool which can save and load the models. This tool is called Joblib [43] and is very straightforward to use. We will use the commands *dump* and *load* on the models, to either save or load a trained model. The only requirement is that you will have to use the exact same features in your test data as you did when training the model.

Logistic regression

The logistic regression is the most basic technique we use. We only have to set a few parameters for this model. As our problem consist of predicting one of multiple outcomes, we need a multi-class classification model. Therefore we have to set the *multi_class* parameter in our logistic regression to *multinomial*, then it will use cross-entropy loss to find the best model. The other parameter we set is the solver, we cannot use liblinear for our multi-class problem as this is only suited for binary problems. We choose for the SAGA solver [10]. This solver performs well in practice and is faster on large datasets than other solvers. We will use these settings to train our logistic regression model.

Random forest

Decision trees are always an option in machine learning, we will use the random forest here. For the random forest we set two parameters while leaving the rest on their defaults. We set the number of estimators, the *n_estimators* parameter. This dictates the number of trees used in the

forest, we put this value at a thousand trees. If we go higher the forest causes memory errors on the machine training the models. However a thousand trees are enough to get an average and reduce overfitting. To further control overfitting we set the *max_depth* parameter to 25. This makes sure that each tree stops at a depth of 25, this will improve the computation time and memory management whilst also reducing overfitting and increasing generalization of the model.

XGBoost

XGBoost is also based on decision trees. For XGBoost we have to choose the objective function to perform the gradient boosting on, in our case we choose for the $multi{:}softprob$ option. This indicates that we are dealing with a multi-class output and the softprob refers to the softmax function. Instead of returning the label, it returns the probability for each output. Similar to the random forest, here we also choose a $n\_estimators$ of a thousand and a *max_depth* of 25 to help with memory management and overfitting.

LSTM

Hyperparameter tuning is very important for neural networks. There are a few ways to do the hyperparameter tuning. We can manually tune the hyperparameters but this is very time consuming and you need an expert or else the tuning will not be much of an improvement. The most standard option in parameter tuning is gridsearch. Gridsearch evaluates all the different possible combinations of parameters in a grid-like manner, therefore it is called a gridsearch. However testing all different possible combinations of parameters and finding the best combination takes a lot of computational power and time. This is why a randomized gridsearch was introduced, the randomized gridsearch does not compute all possibly combinations but it randomly chooses a subset of them. By Bergstra and Bengio [4] it is empirically and theoretically shown that randomly chosen trials are more efficient for hyperparameter tuning than trials on a grid.

However there is also a disadvantage in the randomized gridsearch. Randomized gridsearch does not adapt its behavior based on the previous outcomes. This means that a poorly chosen parameter can prevent the model from learning effectively. For example if the dropout rate should be between 0 and 0.5 but we test for values between 0 and 1 then 50% of the tests will return bad results. This is an unnecessary waste of time and therefore the range in which the hyperparameters lie, needs to be chosen carefully. To avoid the problem of not being able to learn from your training history another hyperparameter tuning method was invented. The Bayesian optimization methods by Snoek et al. [36] are capable of learning from the previous trials. Bayesian optimization creates a surrogate objective function to approximate the best hyperparameters for the real model. A study by Bergstra et al. [5] shows that Bayesian optimization methods produce significantly better results whilst also limiting the computation time. Therefore we will use Bayesian optimization in this research. We will now shortly discuss the different hyperparameters we will tune.

- **Gradient descent optimization algorithm**: This optimization technique tries to minimize the loss function after each iteration by tweaking the weights. Some of these optimization algorithms are Momentum, Adagrad, RMSprop, Adam and Adamax [33]. Adam is in general the best performing optimizer [33].
- **Number of neurons in hidden layer**: The number of neurons in the hidden layer determines how well the model learns without underfitting or overfitting.
- **Dropout**: Dropout is a regularization method which drops out random nodes to reduce overfitting and improve overall model performance.
- **Batch size**: The batch size defines the number of samples that will be used every iteration. This hyperparameter is also a balance between not overfitting the model and unable to

escape a local minimum. In general it is advised to use a power of two as batch size since this would increase efficiency.

- **Epochs**: The number of epochs is the number of times your model trains on the entire dataset. If this number is too high it will cause overfitting on the opposite side if it is too low then there will be underfitting.

7.2 Experimental evaluation

7.2.1 Metrics

To answer the fourth sub-question: How good are the predictions made on the data of KPN? We first have to define what 'good' means. Here in the experimental evaluation we will measure the performance with metrics. We have a multi-class classification problem and therefore we have to choose suitable metrics for this kind of problem [37]. As metrics we will use precision, recall and F_1-score. To visualize them better we will make confusion matrices and bar plots.

Precision is defined as

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

The precision are the true positives divided by the total of predicted results. This can be interpreted as measuring how many of the predicted results are actually correct. So what percentage of the results we predicted to be positive is also truly positive.

Recall, also called sensitivity, is defined as

$$\text{Recall} = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

The recall are the true positives divided by the total of positive elements. This can be interpreted as measuring how many of the positive results are actually predicted. So what percentage of the total positive elements is predicted as positive.

Lastly the F_1-score is defined as $2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$. The F_1-score is the harmonic mean of the precision and recall. In practice this means that the outcome is not sensitive to outliers bigger than the mean but outliers lower than the mean will have an influence on the outcome. Meaning if we have a precision of 0.9 but a recall of 0.1, the F_1-score will be 0.2 while a standard average would be 0.5. This means that both the precision and recall have to score high to get a good F_1-score. However if we get a low F_1-score we cannot tell if the score is caused by the recall or precision and we will have to a look at the confusion matrix to find the problematic cases.

7.2.2 Results

We have trained the four different models on the two different datasets which were split into in total three different options. The datasets are split into a training set on which the models are trained and a testing set on which the trained models are tested.

Model comparison

First we will compare the four different kinds of models with each other. For this we show a bar plot depicting the F_1-score for the different models in Figure 7.1. We also created these bar plots for the precision and recall and these can be viewed in Appendix A. On the x-axis we plotted the

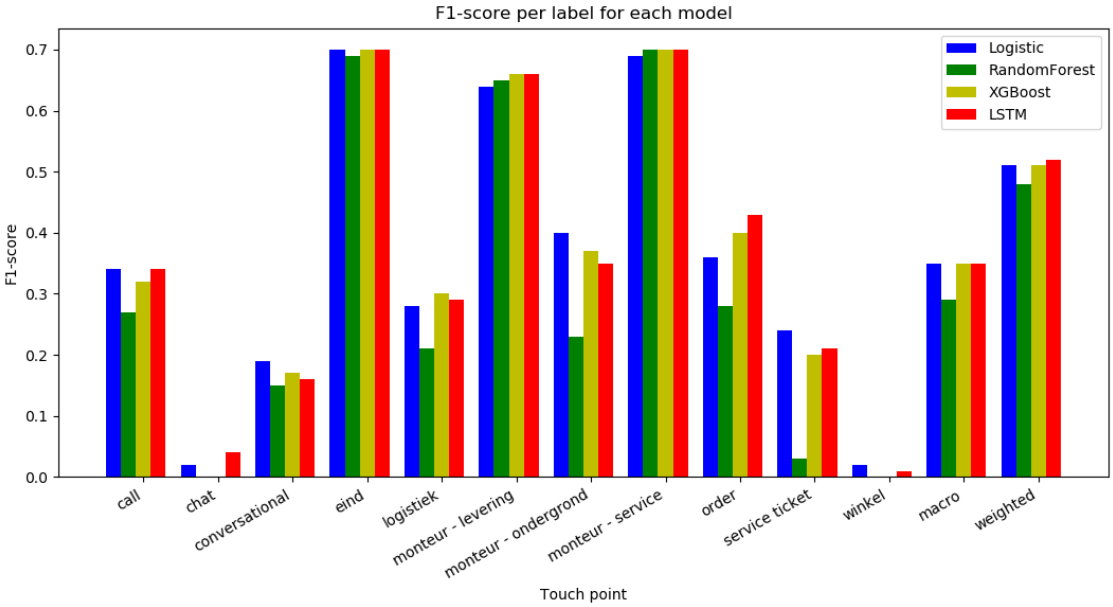


Figure 7.1: Bar plot based on the ten journey dataset comparing models

predicted labels, i.e. the touch points. Also a macro average and a weighted average is shown. On the y-axis we plotted the metric, in the case of Figure 7.1 this is the F_1-score. We plotted four different colored bars which are the four different model types, which color relates to what model can be seen in the legend. To compare the different models, it is best to look at the macro average and weighted average. Overall we see that the macro average is equal for the logistic, XGBoost and LSTM models with only the random forest (RF) underperforming. The same observation holds for the weighted average. The models perform similar with only RF underperforming. The reason that RF is worse than the other three could be caused by the fact that we did not tune the parameters of the RF, while we did tune the LSTM and XGBoost boosts itself.

Logistic regression variants

For logistic regression we tried three different 'thresholds', in binary logistic regression one can set the threshold of one of the two classes to defer from 50%. But in multinomial logistic regression this is not possible because you cannot set thresholds as there can be multiple classes who then fulfill the threshold. Thus in multinomial logistic regression the highest probability is chosen. Meaning we tried three different ways of applying a function to the probabilities after which the highest is chosen. We used the standard, a version in which the touch point **eind** is ignored and one where we subtract the baseline probability of that touch point from the standard probability which we call the delta version. Meaning in our dataset we have a baseline of 48% **eind** so if the standard logistic regression gives us 90%, we then subtract 48 and get a 42% probability for **eind**. We do this for all touch points and the baseline percentages of all touch points can be found in Table 5.1. We tried these three different options to see if they would make an improvement in the outcome, as our dataset is unbalanced ignoring **eind** or subtracting the baseline might be a way to normalize.

The normalized confusion matrices of these three options are displayed in Figures 7.2, 7.3 and 7.4. The confusion matrices are normalized vertically and therefore we can observe the precision on the diagonal of each touch point. However more interesting, we can also observe what percentage

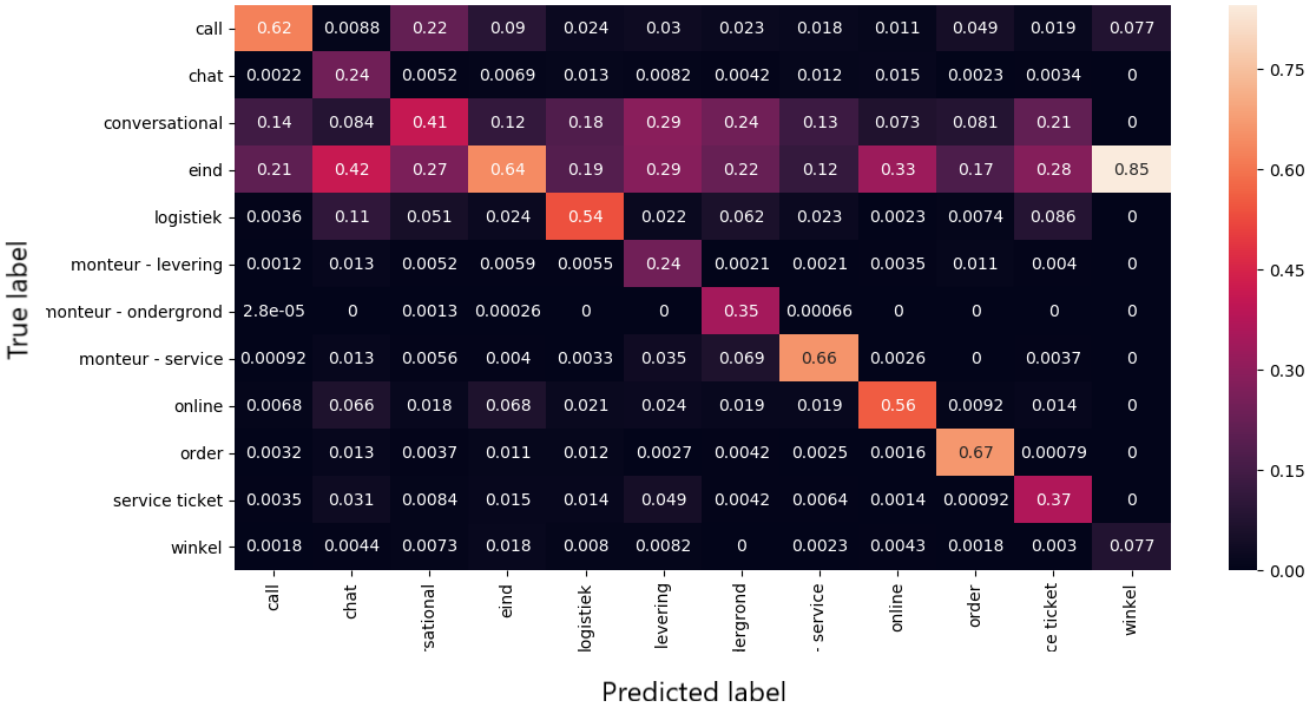


Figure 7.2: Normalized confusion matrix of standard logistic regression

of the total we predicted wrong per column. In all three figures we see that most of the wrongly predicted touch points fall into the **eind** touch point, this does make sense as **eind** is the biggest touch point. This makes predicting **eind** a safe bet and is therefore the fallback choice when there is much uncertainty.

We will now compare the first two figures, the standard (Figure 7.2) and ignore **eind** (Figure 7.3) variants. The standard variant scores better on almost all touch points. This is explained by the fact that we ignore **eind** and this creates extra wrong predictions in the other columns reducing their precision. So the variant in which we ignore **eind** is not a success.

Next we will compare the first and third figure, the standard (Figure 7.2) and delta (Figure 7.4) variants. Checking the diagonal we see that **eind** performs better for the delta variant but the other touch points perform worse. We see that in the delta even when subtracting the baseline percentage of each touch point, that **eind** is still very big and therefore a lot of predictions are still placed into **eind** which is wrong. Meaning comparing between the standard and delta we observe that in the delta variant more predictions are wrongly predicted as **eind**. This decreases the precision for each touch point except **eind**. In the end after comparing the three different options we decided to remain with the standard logistic regression for further all results of logistic regression.

Models per step

We have tried another method to see whether that would work well. Each customer journey consists of steps, it could be possible that depending on the step of the journey that customers would behave differently. Up till now we are training our models on all the data including all steps. We will now try to train models only on a specific step and a few steps in order to compare the differences. The result of certain touch points is plotted in Figure 7.5 and Figure 7.6. In these

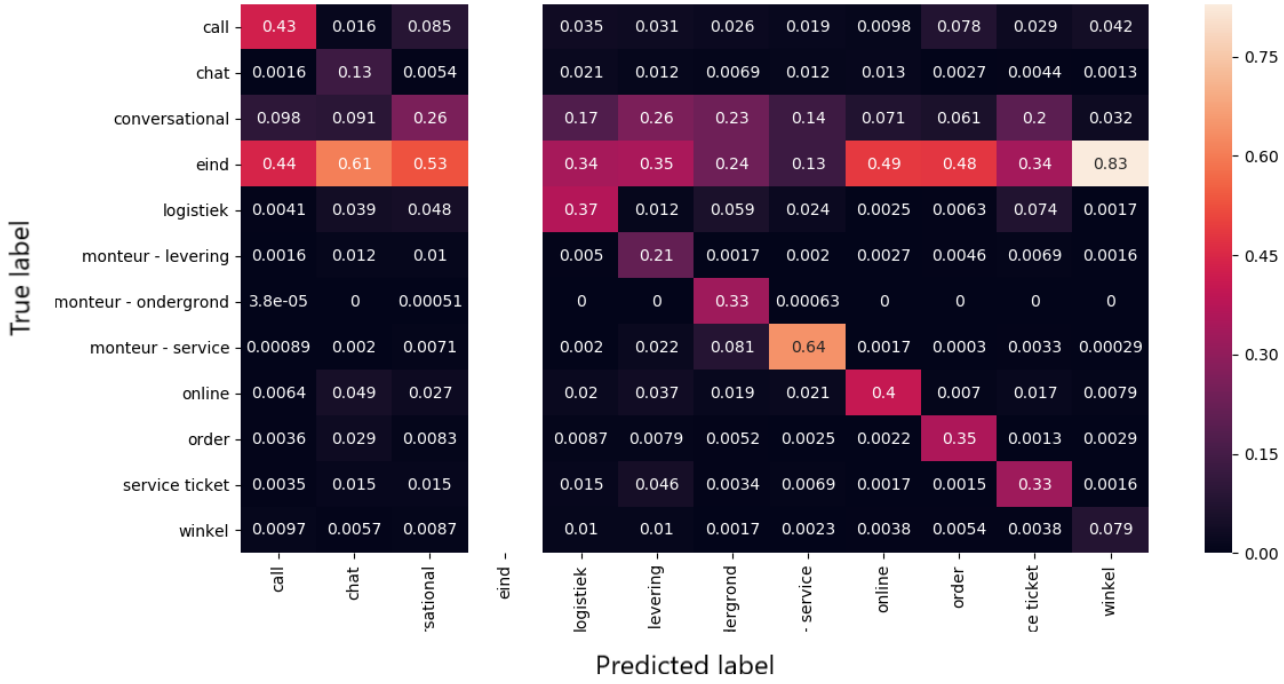


Figure 7.3: Normalized confusion matrix of no **eind** logistic regression

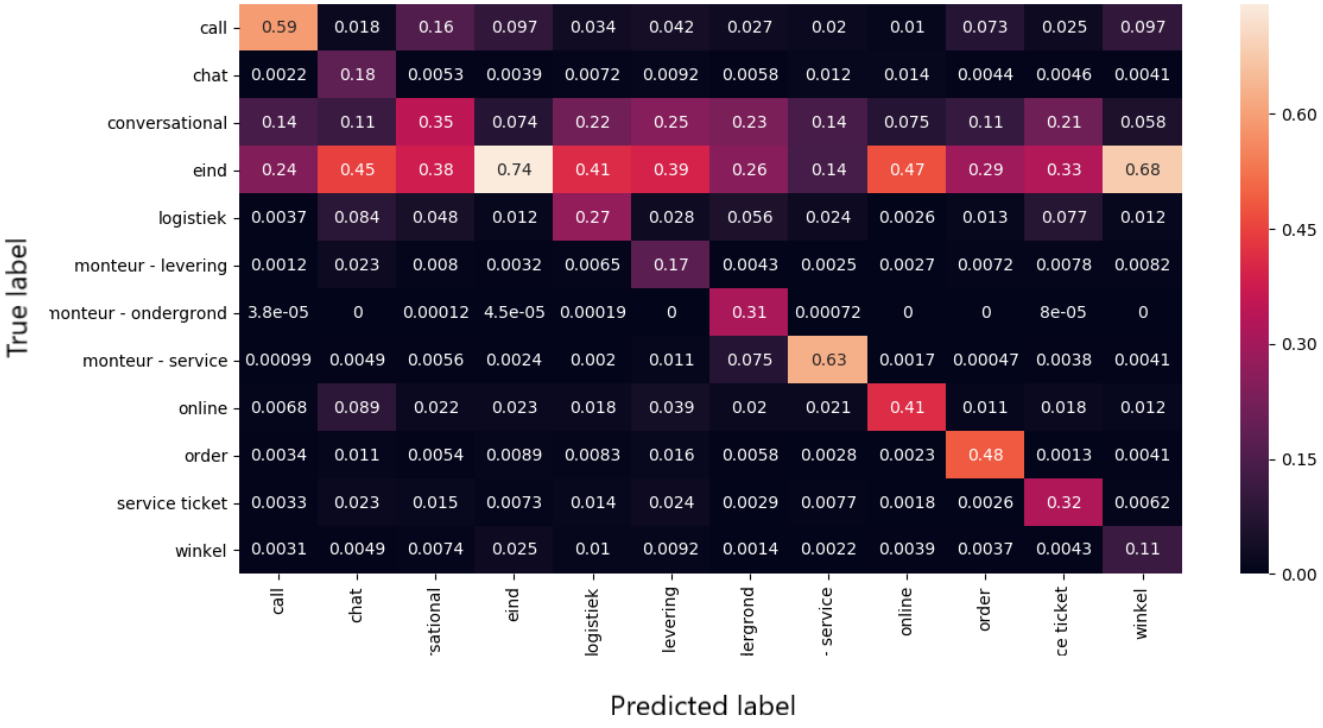


Figure 7.4: Normalized confusion matrix of delta logistic regression

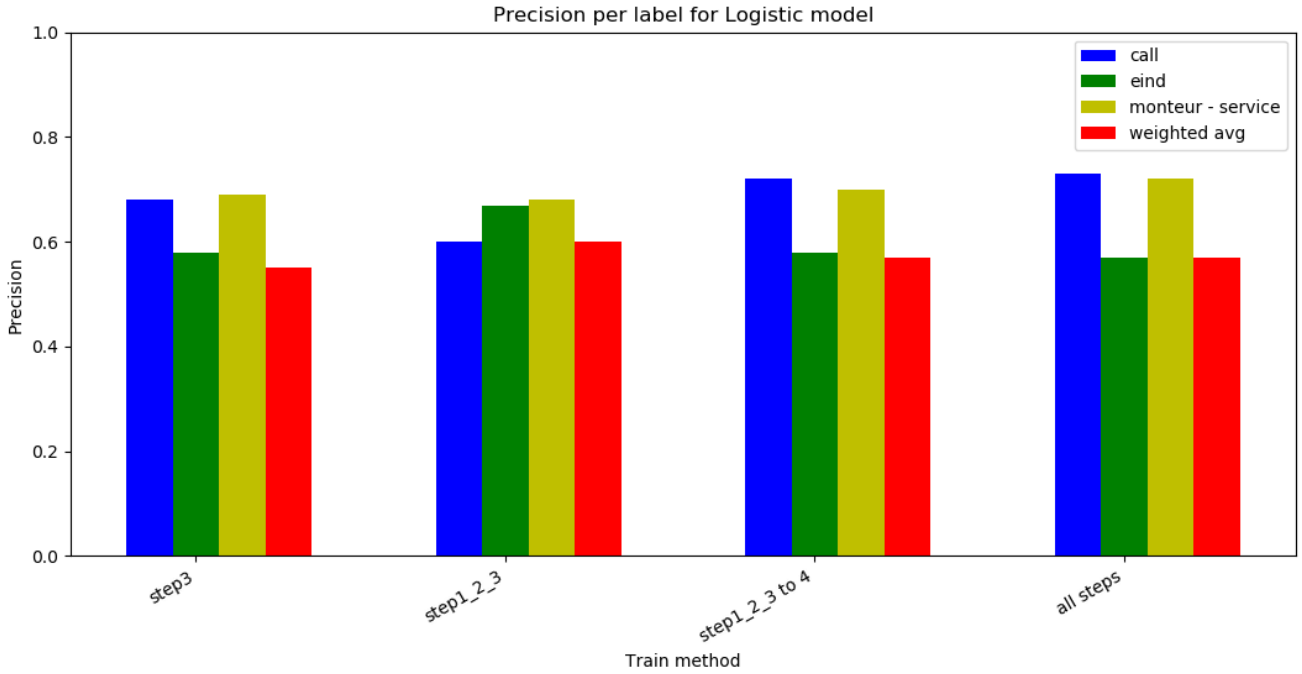


Figure 7.5: Precision for labels trained with different data steps

figures we can see on the y-axis the precision or F_1-score for each label. The steps of data used is displayed on the x-axis, whereas the bars depict a certain touch point which can be found in the legend. We used four types of data to train the model. We have the data from only a specific step, meaning only the features from journeys which are at a certain step of their journey. This is labeled by *step3* and it uses data only from the respective step. Then we used data from some steps, such as the data from the first, second and third step. These are shown with the labels *step1_2_3* and *step1_2_3 to 4*, where the *step1_2_3* also includes journeys of length one and two and also predictions on them. Whereas *step1_2_3 to 4* only includes journeys of length three and thus only predicts the fourth step in a journey. Lastly we have the trained model using all of the data, labeled with *all steps*.

We observe that *step3* and *step1_2_3 to 4* are very similar. *step1_2_3 to 4* performs slightly better by 1% or 2% on all labels. If we compare these to the *step1_2_3* model then we see that **call** performs worse and **eind** is higher. This is caused by the fact that in the model of *step1_2_3* there are also short journeys of length one and two which means that there are more journeys ending thus increasing the number of **eind**. Lastly we compare the results to the model trained on all steps, *all steps*, again for the touch points there is a slight increase of 1% or 2%.

We can conclude that there is an improvement by adding previous steps to the training data. Even if this improvement is only little, this can also be caused by the fact that there are only two steps of previous data. When the length of the journey increases and there are more steps remembered this might increase the performance. Lastly as we saw that the difference between training on only the first three steps in model *step1_2_3 to 4* and training on all steps in the model *all steps* is not that much. We will only train the models using all steps as training data, since training models for each individual step creates more overhead and costs a lot more time.

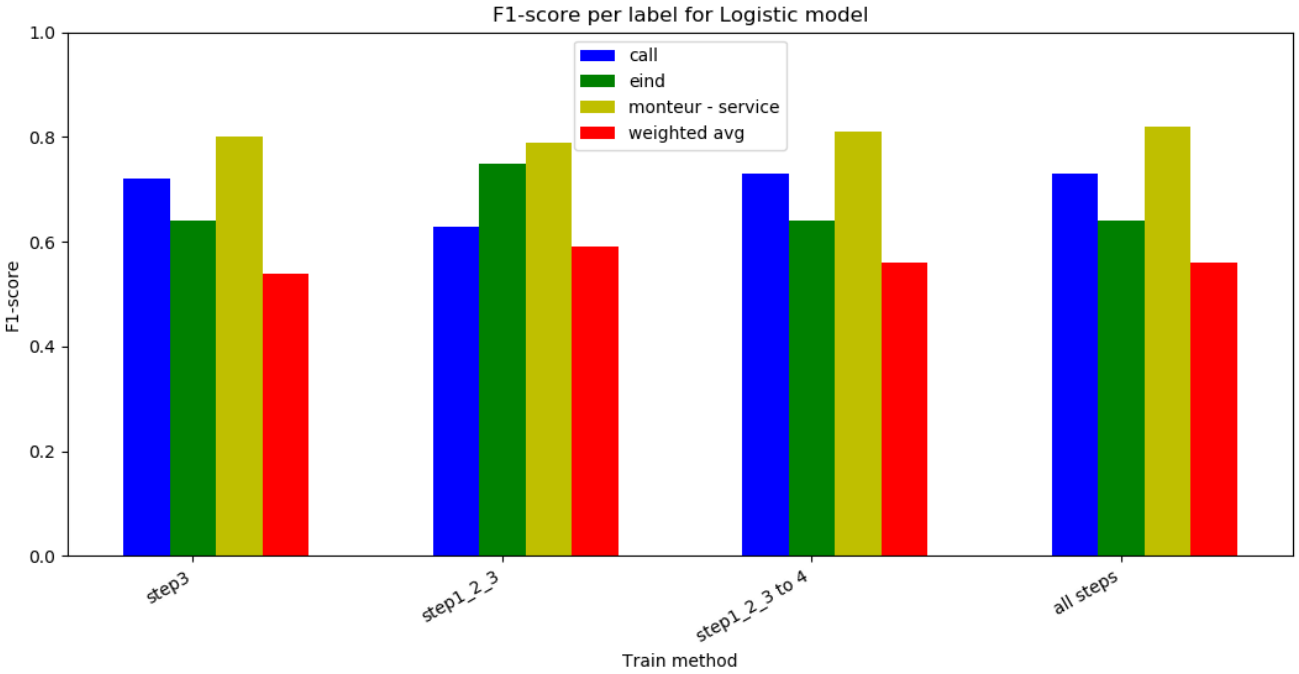


Figure 7.6: F_1-score for labels trained with different data steps

Comparison of datasets

Lastly we will compare the three different types of datasets. To recap we have the initial dataset with buckets and then we have the dataset with the journey types. We used the journey type dataset in two ways. The first way is by just using the dataset trained on all data in a journey, so only sorting by journey id. In the other way, we group the journeys on journey id and also on journey type. This creates more journeys and therefore also smaller journeys. However these journeys should all be related to the same subject as they share the same journey type. The precision, recall and F1-score are shown in the Figures 7.7, 7.8 and 7.9 respectively. In the figures the metric is on the y-axis and the different touch points on the x-axis. The bars are the three different datasets as well as a dummy baseline for reference. For some datasets certain touch points are not available and this is displayed by a small negative value, for example for the ordered dataset there is no **conversational** label.

First we will inspect the overall score with the F1-score measure shown in Figure 7.9. Comparing the macro average and weighted average, we see that all three datasets outperform the dummy baseline which is good. For the macro average the bucket dataset performs the best and the ordered dataset the worst. While the ordered dataset performs the best in the weighted average, the bucket dataset also performs well. To explain this we investigate the individual labels. We see that the bucket dataset performs well compared to the others for many labels except for **monteur - levering**. This explains why the bucket dataset scores high for both the macro average and weighted average. Whereas the ordered dataset performs very well on the **eind** touch point and also good on some others but also very bad on half of the labels. This explains why the ordered dataset does not have a high macro average. The ordered dataset scores well on the weighted average because the touch point **eind** contains a very large part of the labels.

We observe two touch points that observe very poorly. These touch points are **chat** and **winkel**

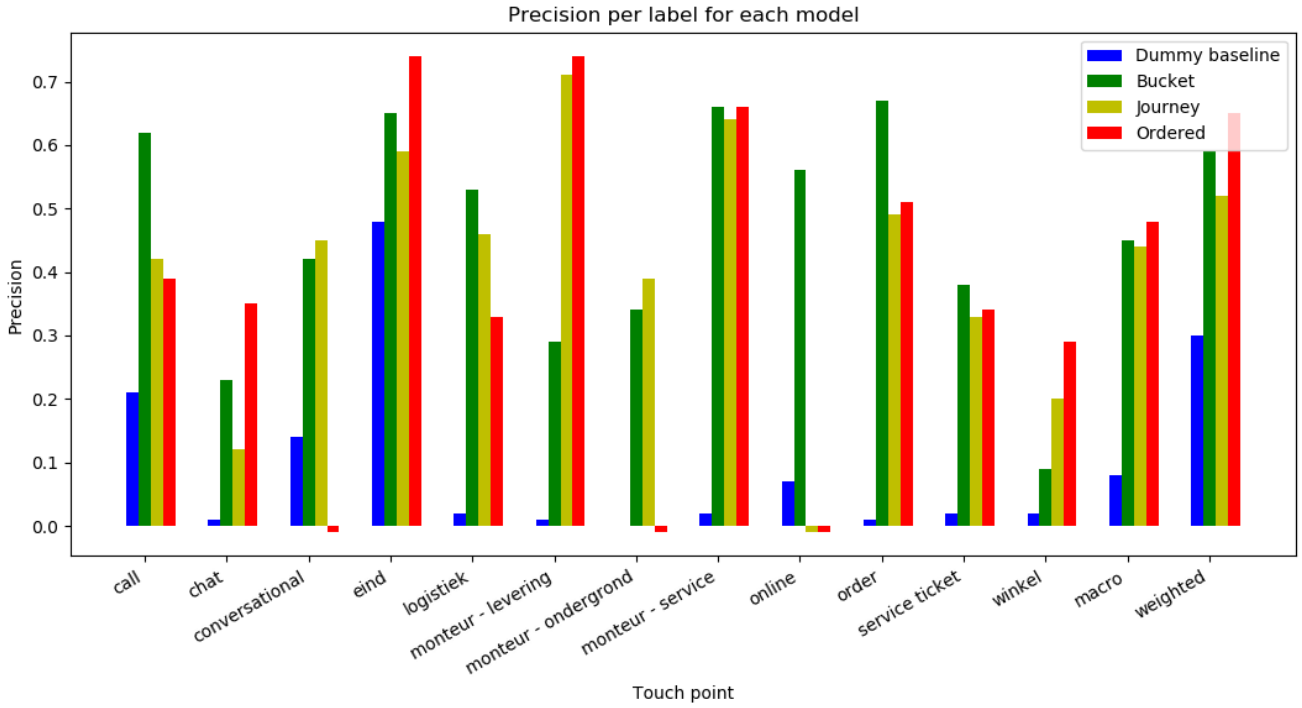


Figure 7.7: Precision comparing dummy and three datasets

(store). This could be explained by the fact that these are two of the smaller labels and therefore less tuned on by the models. However thinking about these touch points, they both do not belong to clear processes. A customer can go to the store whenever he wants but he is never expected to go to a store. This makes the store a very unpredictable touch point. The chat has the same issue only when we see a customer online, we could predict that he is going to chat. However there is never a clear indication that the customer will chat with a KPN employee.

Now we will look at the touch points most interesting to us, namely **call** and **monteur - service**. We observe that **monteur - service** is very good predictable in all datasets. This is likely caused by the fact that a mechanic for service is always send in a reaction to something and it is not sent out of the blue. The indications are used by the model to predict when a mechanic will be sent. Looking at the **call** touch point only the bucket dataset performs well and the ordered dataset performs very poorly. The poor performance of the ordered dataset could be caused by the fact that there is no **conversational** data in this dataset and **conversational** is one of the biggest indicators that a **call** might follow.

We have looked at the F_1-score which is build up from the precision and recall. Therefore we will inspect those as well to get a better insight into the specifics. For precision the bucket dataset performs well on most touch points. The ordered dataset also performs pretty well and especially on the **call** touch points it performs not too bad. This indicates that the recall of **call** has to be very low as the F_1-score was low as well. Indeed when looking at the recall Bar Plot 7.8, we see that the ordered dataset performs very bad on **call**. We see in general that the recall is lower than the precision for the same touch points. Apparently it is hard to find most of the correct classifications, this might be caused by the fact that we are working with a multiclass problem and this causes there to be more options in labels to predict.

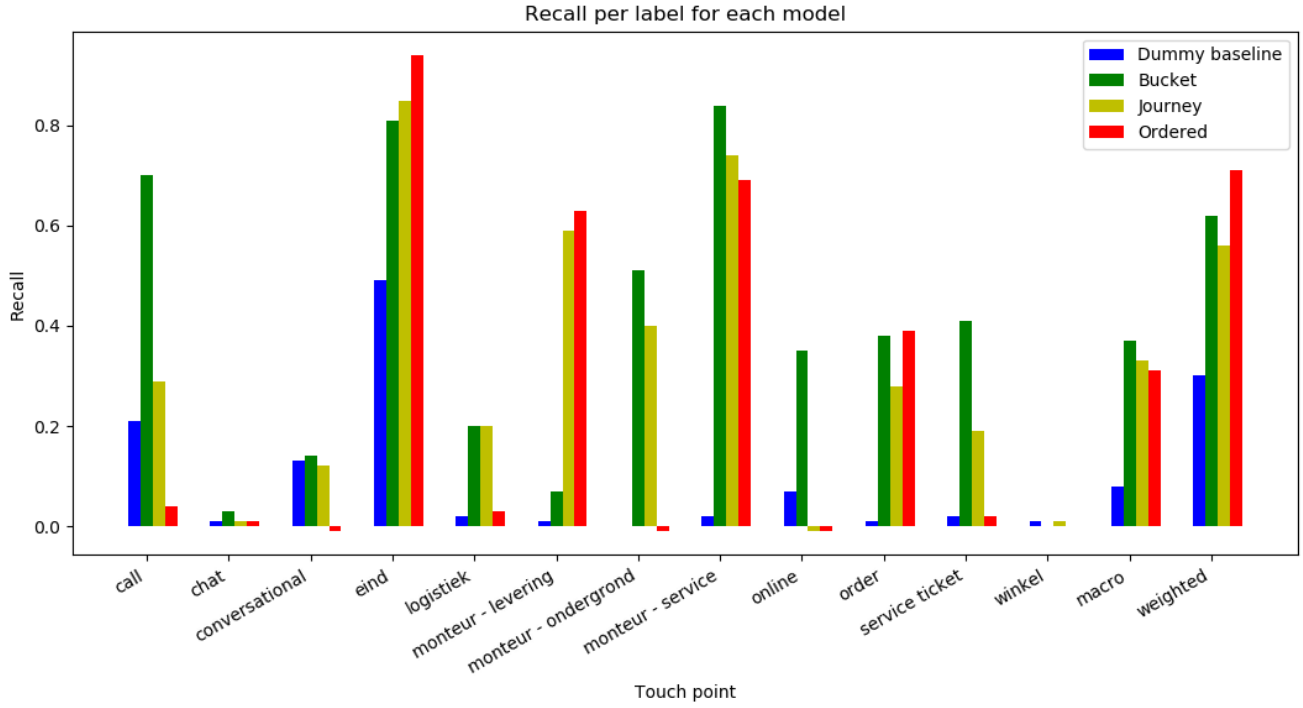


Figure 7.8: Recall comparing dummy and three datasets

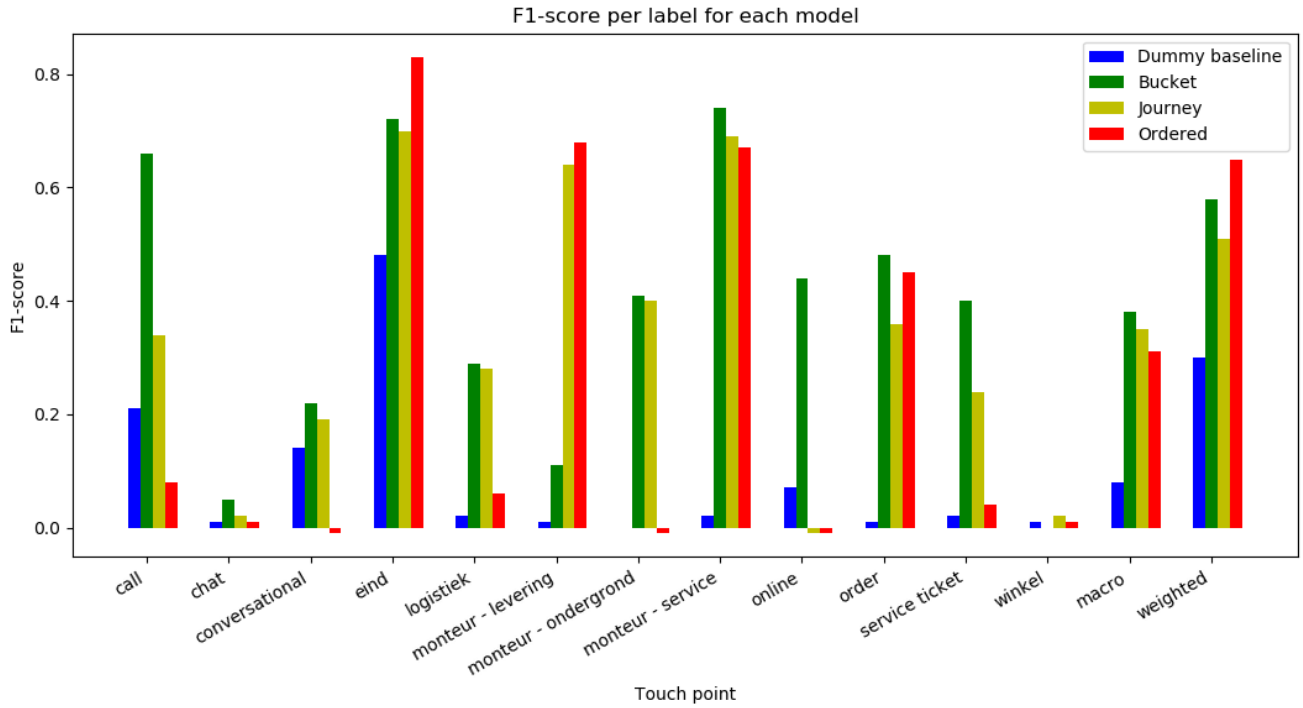


Figure 7.9: F_1-score comparing dummy and three datasets

Chapter 8

Evaluation

8.1 Business evaluation

We compared the results from the different models and dataset with each other in Chapter 7.2. Now we will evaluate how these results effect the business. To do this we will first recap how and in what way the business will use the prediction. After that pros and cons can be formulated on why the business should use this or not.

KPN wants to use the predictions to help prevent future customer contact relating the same issue. For example KPN wants to proactively help the customer at the current contact point with some extra questions and attention, if this would help the customer to resolve his issue and therefore not contact KPN again in the near future. This not only reduces the costs for KPN but improves the quality of life for the customer, as the customer does not have to wait in queue with the helpdesk again or get the feeling that his problems are not being resolved. So overall we have to weigh the tradeoff between putting in extra effort to prevent the customer from contacting again versus helping more customers in general by not inquiring for more information.

8.1.1 Impact and effort of prevention

The touch points we consider for prevention are calls and mechanics. Now we will consider how much impact the prediction can have on the business and the customer but likewise how much effort the preventive method costs. A mechanic has a lot of impact when prevented because a customer does not have to go through the process of staying at home for a mechanic and the customer gets a feeling that KPN is really there to help. However the downside being when the followup mechanic is unavoidable, then the first mechanic spends extra unnecessary time at the customer. Time is a big issue for mechanics as they are on a tight schedule already. Investing extra care for a customer might not fit the schedule or be beneficial. In conclusion the mechanic can be very impactful when he can prevent another mechanic but a mechanic that spends extra care in vain is impactful in a negative way.

Considering the prevention of a call, the helpdesk has to make more effort and ask additional questions to prevent the consecutive call. Spending some extra attention to a customer is certainly less impactful than a mechanic. However there will still be a time loss for the service helpdesk and for the customer. The service helpdesk is already crowded by calls and is trained to help people in the best but also quickest way. A customer might not be interested in these additional questions after his problem is solved, as the customer wants to move on with his day and not spend anymore time on the phone with the helpdesk. On the other hand when a call is predicted correct and this extra effort is put into the first call, this will be significantly less time consuming

than a customer calling again. Because the customer then has to go to the process of stating who he is, in order for the helpdesk employee to lookup his file and then explain his issue again to this new helpdesk employee. These steps are overhead when the call can be prevented by an earlier helpdesk employee. In conclusion the impact of preventing a call is a meaningful time saving and when wrongly predicted is a bit of extra time spent.

Another aspect to consider is the fact that we might be able to predict correctly but we do not know the context of the next touch point. Without knowing the context trying to prevent poses an extra risk that the next call made might be about something entirely different and thus the call was unpreventable to start with, even with extra attention in a prior call.

8.1.2 Cost balance overview

To put the impact and effort into perspective, we create a scenario from the business side. This shows us expected cost values for the business and what results should be achieved to make prevention beneficial.

Call prevention

Looking at the actual numbers in the precision Bar Plot 7.7, we see that the best precision for **call** is 0.62. This means that six out of the ten calls we predicted are indeed correctly predicted. The best recall from the Bar Plot 7.8 is 0.70, which is even higher than the precision and means that we predict seven out of the ten actual calls made. These values are alright but the precision is still on the lower side and KPN may want more certainty that investing the extra time is actually useful. Another issue is the fact that we do not know the context of the call which might mean that the predicted call is just unpreventable. This leads to a new question to research, how many of the calls can actually be prevented? This question is out of scope for this research but we will make a graph to compare different values of call prevention.

Lets put these numbers into a scenario business case with a cost overview. The result can be seen in Figure 8.1. On the y-axis we have the euros saved by trying to prevent calls. On the x-axis we plot different values for the amount of calls we can prevent. The four lines represent different values of precision for the prediction. The rationale and formulas behind the graph will now be discussed.

In our dataset with eight weeks of data we could predict 65k calls, this is the total number of positives. Using the recall we can calculate the number of true positives as follows:

$$\text{Recall} = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Meaning with a recall of 0.7, we find $65000 * 0.7 = 45000$ of the total calls. These 45k calls are our true positives, i.e. correctly predicted calls. With the precision we find the number of calls we predicted wrong as follows:

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

If we then fill this in with a precision of 0.62, we get $0.62 = \frac{45000}{45000+X}$. Resulting in $X = \frac{45000}{0.62} - 45000 = 28000$, this is the 28k calls we predict wrong. So we predict 45k calls correctly and 28k calls incorrectly. Now we need to add time and money to these numbers. According to KPN an average call costs 7 euro and in our dataset an average call takes 10 minutes and 43 seconds. This means that the cost of a call per minute is $\frac{7}{10.72} = 0.65$ euro. If a callcenter employee extends the call by asking extra questions he prolongs the call, in this scenario we assume that asking a few extra questions takes 1.5 minutes additionally. However we believe that some people will

Preventable calls (1)	+888k
Unpreventable calls (2)	-347k
Uninterested customers (3)	-54k
Wrong predicted calls (4)	-303k
Uninterested customers with wrong calls (5)	-32k
Total profit/loss	+152k

Table 8.1: Profit/loss from preventing calls scenario

decline the extra questions and we will assume that this is 50% of the customers, this will only take 10 seconds longer than a standard call. This means that extending a phone call when asking additional questions increases its costs by $0.67 \times 1.5 = 0.98$ euro. But when the customer does not want additional questions it only costs $0.67 \times 1/6 = 0.11$ euro extra.

Now we can calculate the loss and profit of applying call prevention. We have 45k calls predicted correct of which we assume 15% is preventable, 35% unpreventable and for 50% the customer is not interested. This provides us with three subformulas. The first calculates the profit made by successfully preventing calls, which we assume to be 15%. Next we calculate the loss from calls we tried to prevent but were unpreventable, which we assume to be 35%. Lastly we calculate the loss from customers declining the extra care, which we assume to be 50%.

The profit from preventable calls is:

$$Calls \times percentage\ of\ preventable\ calls \times (call\ cost - cost\ of\ extending) \tag{1}$$

The loss from unpreventable calls is:

$$Calls \times percentage\ of\ unpreventable\ calls \times cost\ of\ extending \tag{2}$$

Lastly the loss from customers who are not interested is:

$$Calls \times percentage\ of\ uninterested\ customers \times cost\ of\ extending \tag{3}$$

However we have the 28k incorrectly classified calls as well. For these there are also the unpreventable calls and the customer declining extra care but there are no calls being prevented as the prediction was wrong. The loss of calls which are extended with extra care:

$$Wrong\ calls \times percentage\ of\ unpreventable\ and\ preventable\ calls \times cost\ of\ extending \tag{4}$$

Loss from calls with customers who are uninterested:

$$Wrong\ calls \times percentage\ of\ uninterested\ customers \times cost\ of\ extending \tag{5}$$

Now that we have all subparts we can calculate the total profit and loss as shown in table 8.1. Giving us a 7k euro profit with 15% preventable calls, 62% precision and 70% recall.

Recall only influences the amount of money that is gained or lost, it does not influence the loss or gain positively or negatively. As the recall only tells us the percentage of correct calls we found from all correct calls, meaning that it signifies the magnitude of the total effect that could be achieved. So when the recall is low then the prevention methods will only be applied to a small subset of the total applicable cases. This means that when the recall is very low it might not be worth to set up the prevention techniques as it will rarely be applied.

The above example was based on the total average precision for calls. We checked how our model performed when we predict a call leading to a call. Then the precision dropped to 49% correct, if we look in the Graph 8.1 we see the orange line showing 50% precision. Meaning with our 49%

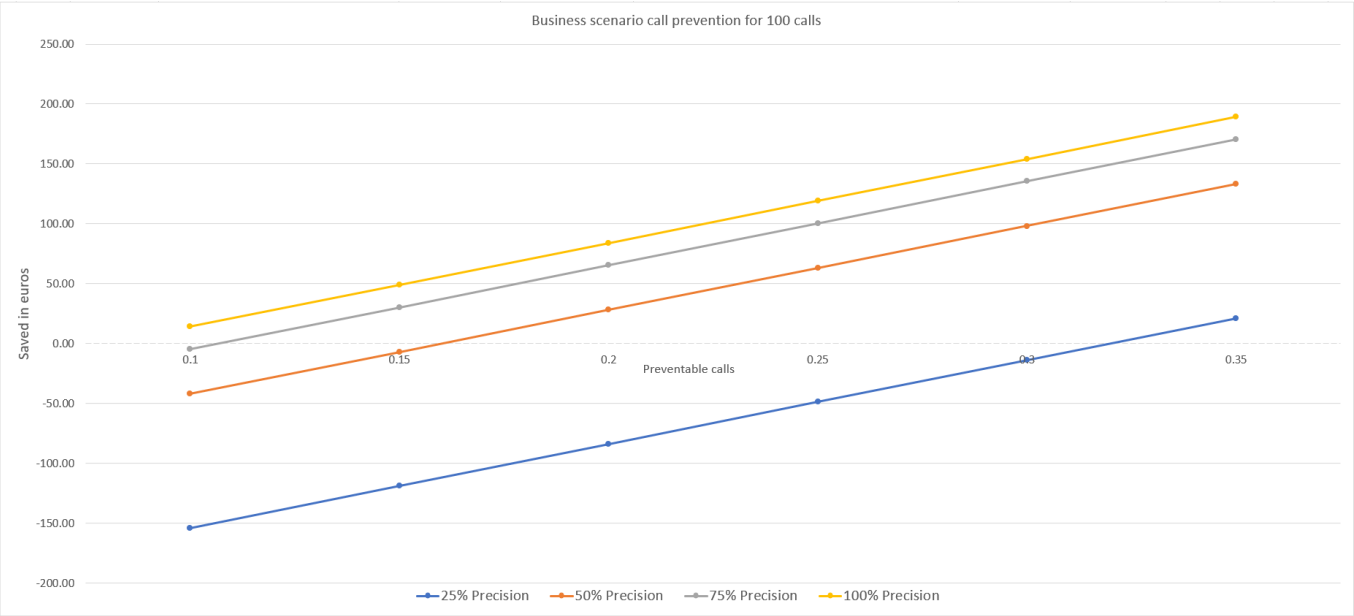


Figure 8.1: Business scenario on call prevention

precision, if the call prevention is 16% we are on a break even point at 0 euro profit and loss. However even then it could still be beneficial for KPN because the customer satisfaction might increase as customers are taken extra care of. Meaning that even if there would be a small loss, KPN might decide that they still want to perform the prevention techniques. Not for the monetary benefits but for the customer satisfaction.

We also observe in the Graph 8.1 that the four different lines are increments of 25%, however the gap between the lines are of different height. There is a huge monetary improvement between 25% and 50% precision, while the gap between 75% and 100% precision is significantly smaller. The big difference from 25% to 50% precision is due to the fact that there are many wrongly predicted calls at 25%. These are very expensive as extra time is committed with no return value. At 25% there are three calls predicted wrong for one call predicted right, this makes a big difference with the one call predicted wrong for every call predicted right at 50%. Also there are a lot less calls we can prevent and consequently gain profit from. Therefore the precision should at least be 50% to counter every wrong prediction with a right prediction or the expenses of wrong predictions have to decrease significantly, which is not possible in this scenario.

Another aspect of the graph is the percentage of preventable calls, displayed on the x-axis. For the scenario described we assumed that 15% of the predicted calls can be prevented, meaning that 35% is unpreventable as we assume that 50% of the customers are not interested. In the graph we plotted different values for the preventable calls and thus changing the percentage of unpreventable calls as well. We see that for 100% and almost for 75% a call prevention of 10% is enough to make a profit on the prevention technique. Whereas for 25% precision the preventable calls need to be 32% before a profit can be made. Both of these observations seem unrealistic and we will be somewhere in between them. This is the orange line of 50% precision in the graph. The break even point for 50% precision is at 16% call prevention, meaning if in practice the call prevention showed to be higher then we know that with a 50% precision there will be a profit. The other way around if the precision seems to be lower in practice then we know that the call prevention has to improve for the prevention technique to be profitable.

Preventable mechanics	+900
Unpreventable mechanics	-400
No extra help possible	-100
Wrong predicted mechanics	-5.7k
No extra help by wrong prediction	-1,1k
Total profit/loss	-6.4k

Table 8.2: Profit/loss from preventing mechanics scenario

Mechanic prevention

For mechanic - service prevention we follow the same steps as with the call prevention, so we will only state the differences. An average mechanic visit costs 60 euro and the average time spend by a mechanic is an hour. We assume that a part of the mechanics costs are already made when he traveled to the location, therefore we assume that 20 euro is travel cost and 40 euro is for the actual time spend. For the mechanic we assume he spends three minutes extra at a location to see if he can prevent an upcoming disruption and when he needs to fix something to prevent, this will take fifteen minutes in total extra than a regular visit.

For the mechanic - service our overall precision is 66%, however we want to look at mechanics immediately followed up by a mechanic. Then the precision of our model is only 9%. With this information we can calculate the table as we did for the calls, this is shown in Table 8.2. We see that there is a loss of 6.4k with these variables, most of which comes from wrongly predicted mechanics which is due to the bad precision. Therefore we will plot a graph with different precision thresholds in order to see which precision value would suffice to make a profit. The graph we made is shown in Figure 8.2.

Comparing the graph of the mechanics to the one for calls, we see that they are very similar. We see however that the lines in the mechanic graph are lower then the same lines in the call graph. This means that it is more expensive to have mechanics use prevention techniques to prevent another mechanic and that the profit from preventing a mechanic is relatively not as much as preventing a call. For mechanic the break even point for 50% precision is at 18% mechanic prevention, so 2% higher than the break even point for calls. We also see that for a 25% precision no profit is made even when there is a 35% mechanic prevention. So for mechanics to be a viable prevention target, the precision needs to be higher than 50% also depending on the percentage of mechanics that can be prevented.

Conclusion

We will conclude by answering the fifth sub-question which was: 'What can the business achieve with the acquired results?' With the current precision for mechanics, the use of a prevention technique is not profitable at all. The precision is way too low to even consider using it in practice. However for the call prevention we have 49% precision which is profitable when more than 16% of the calls can be prevented. Also the customer satisfaction might improve by applying the prevention technique which is something to keep in mind as well. We provided a graph for both scenarios in which we look at different variables for precision and prevention percentage. We saw a big difference between 25% and 50% precision while the gap between 75% and 100% was much smaller. Mostly caused by the expenses that wrong predictions cost. Thus the business can achieve a monetary profit when the precision and prevention percentage are high enough whilst also having customer satisfaction increase when preventing successful.

Figure 8.2: Business scenario on mechanic prevention

Chapter 9

Conclusions

In this thesis we have discussed the customer journey and the touch points within it. We investigated to see if the touch points are predictable. We will first provide a summary using the research sub-questions in Section 9.1 and afterwards the conclusion to the main research question is given in Section 9.2.

9.1 Summary

KPN is performing much research into a customer's journey and how to improve it. The research is all after the fact, meaning that they learn from mistakes in the past and make improvements for the future. The next step is to get ahead of mistakes and prevent them beforehand, which is investigated in this research. First we want to get a better overview of the customer journey. Therefore we started this research by investigating the customer journey. We applied process mining techniques to them but we found no clearly defined journeys. This is caused by the fact that there are too many variants in the customer journey, as the customer journey is not a streamlined process but it is unique for every customer. After this we started thinking about techniques we could use for the prediction. So we searched through the literature and looked at recommender systems and sequence prediction before settling on machine learning from data mining.

To have a structured approach to our research, we followed the framework of CRISP-DM [35]. We started off with understanding the business and the corresponding sub-question: 'What should the output predictions regarding the customer journey look like considering the business objectives of KPN?' We saw that there are multiple ways to contact KPN for a customer. For example the customer can contact KPN using the telephone or the online web portal. For KPN the goal is to help customers in the most satisfying way, for the customer this means the sooner the better. This is a reason why KPN wants to predict the next touch point of a customer, to be able to proactively help them. Thus the type of output of the predictions should be what touch point a customer will use next.

After understanding the business, we now need to understand the data before we can use it. The understanding of the data was accompanied by the sub-question: 'Which data regarding the customer journey is currently available at KPN?' As this research is in cooperation with the data department of KPN, they promptly provided a dataset with the data of the customer journey. In the dataset the journey id, the touch points with their timestamps and some additional information like bucket and callreason are included. This is also the answer to our sub-question on what data is available. A touch point is a contact way of the customer, some example touch points are

call, online, store and mechanic. We further explored the data in Section 5.3, we looked at the distribution of touch points and journey types among other things.

Since we now understand and have explored the data, the next step is to make the data ready for prediction as this is our goal. Data needs to be formatted in the right way for machine learning techniques to be able to train models on them. Therefore the third sub-question: 'What data is directly usable in the context of predicting the next touch point?' We first cleaned the data of some noise, like aggregating data together to reduce variety. Then we transformed the dataset to include additional columns with the previous touch points in them, this so that the model can learn the history of the journey. Lastly we answered the sub-question by stating that we need to encode all of our categorical variables into one-hot encoded variables, as machine learning techniques require this.

Now all data is ready to be used for prediction. It is time for the next step in CRISP-DM, the modeling itself. As for each problem a different model can be better suitable, we created four machine learning models namely logistic regression, random forest, XGBoost and LSTM. After we have created the models we have to evaluate them, therefore the sub-question: 'How good are the predictions made on the data of KPN?' To answer this we first have to define good. We will measure how good a model is by using metrics. We will use precision, recall and F_1-score as metrics. We tried several ways of predicting the next touch point, comparing the models gave similar results. Evaluating different datasets, we saw that the dataset with the column with bucket data performed the best.

The last step is to evaluate the results. We do this by asking the sub-question: 'What can the business achieve with the acquired results?' We answered this by sketching a scenario in which repeated calls are prevented. We showed that with the current precision and a high enough call prevention rate, this scenario would be beneficial for KPN as it would save call costs.

9.2 Conclusion

To conclude this research, we will answer the main research question. The main research question of this research is:

> *Is it possible to predict a customer's next touch point by using the historical data from customer journeys and a prediction technique?*

We showed that we can use machine learning models to predict the next touch point of a customer. The prediction is based upon the customer journey which includes all used touch point in the order they were used. The journey also includes information about the touch point, this is either a bucket or journey type and also a callreason. This research is a first step in the touch point prediction at KPN and therefore we have set a baseline model on which can be improved. The baseline model is a logistic regression and it performs alright for the **call** and **monteur - service** touch points. These two are our main touch points of interest because they are interesting to prevent. The **call** and **monteur - service** are predictable as there are indicators for them to be predicted. The model performs poor for the **chat** and **winkel** touch points, this can be expected as going to the **winkel** is very unpredictable because it is not a step in a process.

For KPN this research is to get insights into prediction in customer journeys and if there is the possibility to continue investigating this area. Concluding this thesis with an answer to the main research question: We showed with this research that there indeed is a possibility to make predictions on touch points based on the customer journey with the help of a prediction technique. We set a baseline model for future research to improve upon.

9.3 Future Work

This research was a first exploratory research into the topic of predicting touch points in a customer journey. Reaching the goal of seeing if we can predict touch points, we found multiple options to perform further research on. We will discuss multiple options to improve the current results.

- A first addition to improve the results is to add more data for the models to train on, not only in quantity but adding more features. A first addition could be to add customer information, like service type, duration of subscription but also information like age, sex and demographics. A hypothesis could be that elderly people are more inclined to call instead of using the online services. With this information, we have a more personalized model and this hypothesis can be proved or disproved.
- For now there were too many variants in customer journeys to create standard journeys or to aggregate them into sections, another research can investigate the possibility of finding similarity between journeys. This could be done by defining a distance measure to score the similarity of two journeys. This score can then be used by a prediction model to better predict the next step, as the model can learn what steps similar journeys have taken.
- We used the dataset of the ten journey types, but this was a first iteration by KPN and possibly the reason that it performed poorly. When the categorization into ten journeys is straightened out more, training the models again could help to improve the results. It might be worth to only use data for a single journey type and train models for all ten journey types instead of training the models on the data of all ten journey types.
- Instead of trying to predict all touch points, focus on predicting a single touch points for example **call**. This makes the problem binary, it is either a call or not. When the problem is binary it reduces the complexity a lot. When focusing on a single touch point, it is a lot easier to optimize for this single touch point.
- In this research we are trying to predict what touch point will occur immediately after the current one. It could be interesting to try and predict if a touch point will eventually happen before the end of the journey. Meaning that it does not have to occur directly next. If a customer is first going online on the website and calling afterwards, it would be great to predict that the customer will call in the near future.
- The research can be broadened to not only predict touch points but try to predict the context in the form of bucket or journey type as well. As knowing about what context the customer will contact KPN, will be very useful in helping to prevent the contact. Predicting touch point and context at the same time is a multi-label (and multi-class) classification problem which brings its own complexity.
- A last option is to improve the data itself. We are not using any techniques to research the callreason, but one could apply some form of natural language processing (NLP) technique to improve the callreason. When the data is of higher quality then the results will likewise be of higher quality.

Bibliography

[1] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2009. 10

[2] IEEE Standards Association. IEEE 1849-2016 - IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams. https://standards.ieee.org/standard/1849-2016.html, 2016. 7

[3] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994. 14

[4] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012. 40

[5] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*, pages 2546–2554, 2011. 40

[6] Gaël Bernard and Periklis Andritsos. A process mining based model for customer journey mapping. In *Forum and Doctoral Consortium Papers Presented at the 29th International Conference on Advanced Information Systems Engineering (CAiSE 2017)*, volume 1848, pages 49–56. CEUR Workshop Proceedings, 2017. 1, 8

[7] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006. 11

[8] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 12

[9] Leo Breiman. *Classification and regression trees*. Routledge, 2017. 12

[10] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014. 39

[11] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001. 13

[12] Shubhangi Vashisth Jim Davies Jason Daigler Gareth Herschel, Brian Manusama. Market guide for customer journey analytics. *Gartner Inc*, 2018. 24

[13] Gil Press. Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says. https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#738105d06f63, 2016. [Online; accessed 26-June-2019]. 6

[14] Joel Goossens, Tiblets Demewez, and Marwan Hassani. Effective steering of customer journey via order-aware recommendation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 828–837. IEEE, 2018. 8

[15] Ted Gueniche, Philippe Fournier-Viger, Rajeev Raman, and Vincent S Tseng. Cpt+: Decreasing the time/space complexity of the compact prediction tree. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 625–636. Springer, 2015. 9

[16] Ted Gueniche, Philippe Fournier-Viger, and Vincent S Tseng. Compact prediction tree: A lossless model for accurate sequence prediction. In *International Conference on Advanced Data Mining and Applications*, pages 177–188. Springer, 2013. 9, 10

[17] Christian W Günther and Wil MP Van Der Aalst. Fuzzy mining–adaptive process simplification based on multi-perspective metrics. In *International conference on business process management*, pages 328–343. Springer, 2007. 7

[18] Simon S Haykin et al. *Neural networks and learning machines*. New York: Prentice Hall,, 2009. 14, 16

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 14

[20] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013. 11

[21] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee, 2008. 9

[22] KPN. KPN Service. `https://www.kpn.com/service.htm`, 2019. [Online; accessed 08-July-2019]. 22

[23] Sander JJ Leemans, Dirk Fahland, and Wil MP van der Aalst. Discovering block-structured process models from event logs containing infrequent behaviour. In *International conference on business process management*, pages 66–78. Springer, 2013. 7

[24] Katherine N Lemon and Peter C Verhoef. Understanding customer experience throughout the customer journey. *Journal of marketing*, 80(6):69–96, 2016. 2, 7

[25] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002. 12

[26] Lorraine. Classification and Regression Analysis with Decision Trees. `https://dev.to/nexttech/classification-and-regression-analysis-with-decision-trees-jgp`, 2019. 13

[27] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999. 36

[28] Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA:, 2015. 14

[29] Nils J Nilsson. *Principles of artificial intelligence*. Morgan Kaufmann, 2014. 10

[30] Christopher Olah. Understanding lstm networks. *Colah's blog*, 2015. 14, 16

[31] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011. 11

[32] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–59, 1997. 9

[33] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 40

[34] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990. 13

[35] Colin Shearer. The crisp-dm model: the new blueprint for data mining. *Journal of data warehousing*, 5(4):13–22, 2000. 55

[36] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012. 40

[37] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437, 2009. 41

[38] Alessandro Terragni and Marwan Hassani. Analyzing customer journey with process mining: From discovery to recommendations. In *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 224–229. IEEE, 2018. 9

[39] Alessandro Terragni and Marwan Hassani. Optimizing customer journey using process mining and sequence-aware recommendation. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 57–65. ACM, 2019. 9

[40] European Union. General data protection regulation. *Official Journal of the European Union (OJ)*, 2016. 25

[41] Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. Springer, April 2016. 6, 7, 18

[42] Boudewijn F Van Dongen, Ana Karla A de Medeiros, HMW Verbeek, AJMM Weijters, and Wil MP Van Der Aalst. The prom framework: A new era in process mining tool support. In *International conference on application and theory of petri nets*, pages 444–454. Springer, 2005. 6

[43] Gaël Varoquaux and O Grisel. Joblib: running python function as pipeline jobs. *packages. python. org/joblib*, 2009. 39

[44] AJMM Weijters, Wil MP van Der Aalst, and AK Alves De Medeiros. Process mining with the heuristics miner-algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP*, 166:1–34, 2006. 7

[45] Rüdiger Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. Citeseer, 2000. 4, 5

[46] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996. 39

[47] David H Wolpert, William G Macready, et al. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997. 39

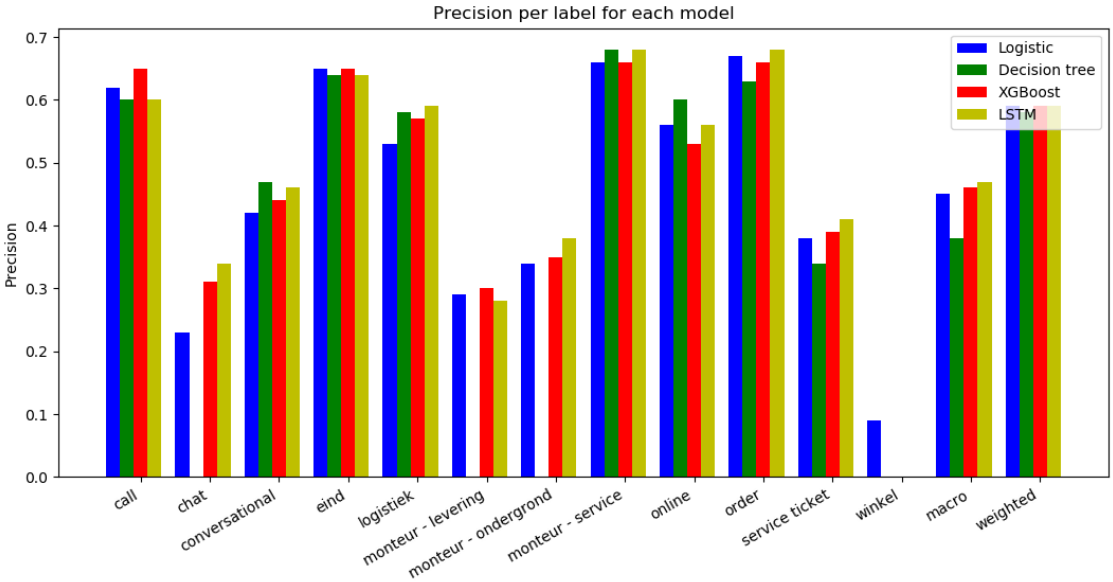
[48] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. *Applied artificial intelligence*, 17(5-6):375–381, 2003. 36

[49] Zhongxing Zhang, Geert Mayer, Yves Dauvilliers, Giuseppe Plazzi, Fabio Pizza, Rolf Fronczek, Joan Santamaria, Markku Partinen, Sebastiaan Overeem, Maria Peraita-Adrados, Antonio Silva, Karel Sonka, Rafael Ro, Raphael Heinzer, Aleksandra Wierzbicka, Peter Young, Birgit Hgl, Claudio Bassetti, Mauro Manconi, and Ramin Khatami. Exploring the clinical features of narcolepsy type 1 versus narcolepsy type 2 from european narcolepsy network database with machine learning. *Scientific Reports*, 8, 12 2018. 15

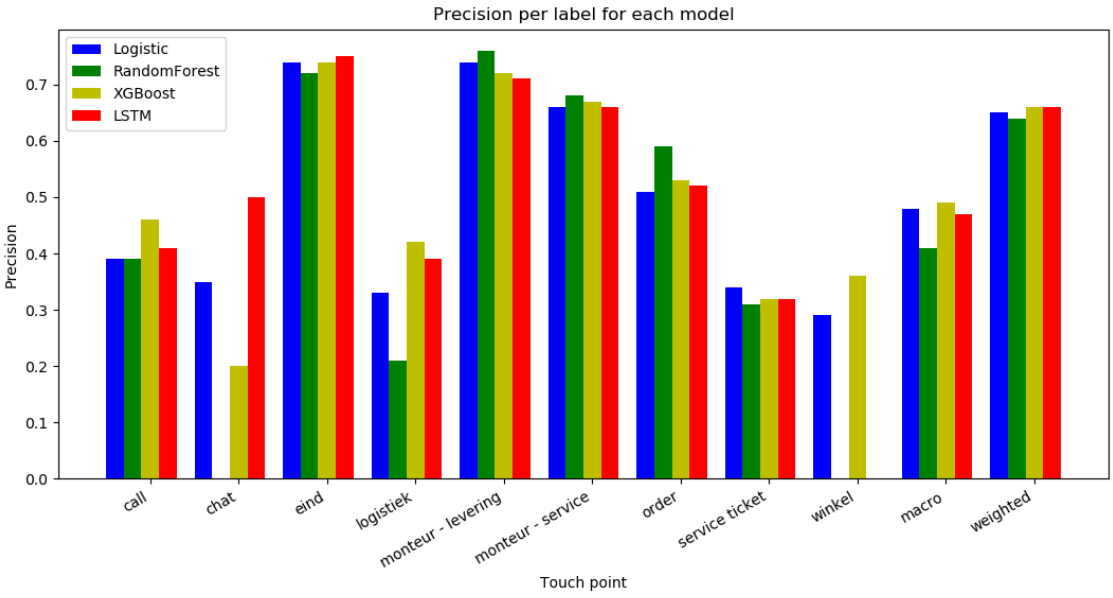
Appendix A

Bar plots

Below are the precision and recall bar plots of the different datasets, as only the F_1-score bar plots are shown in Section 7.2.2.

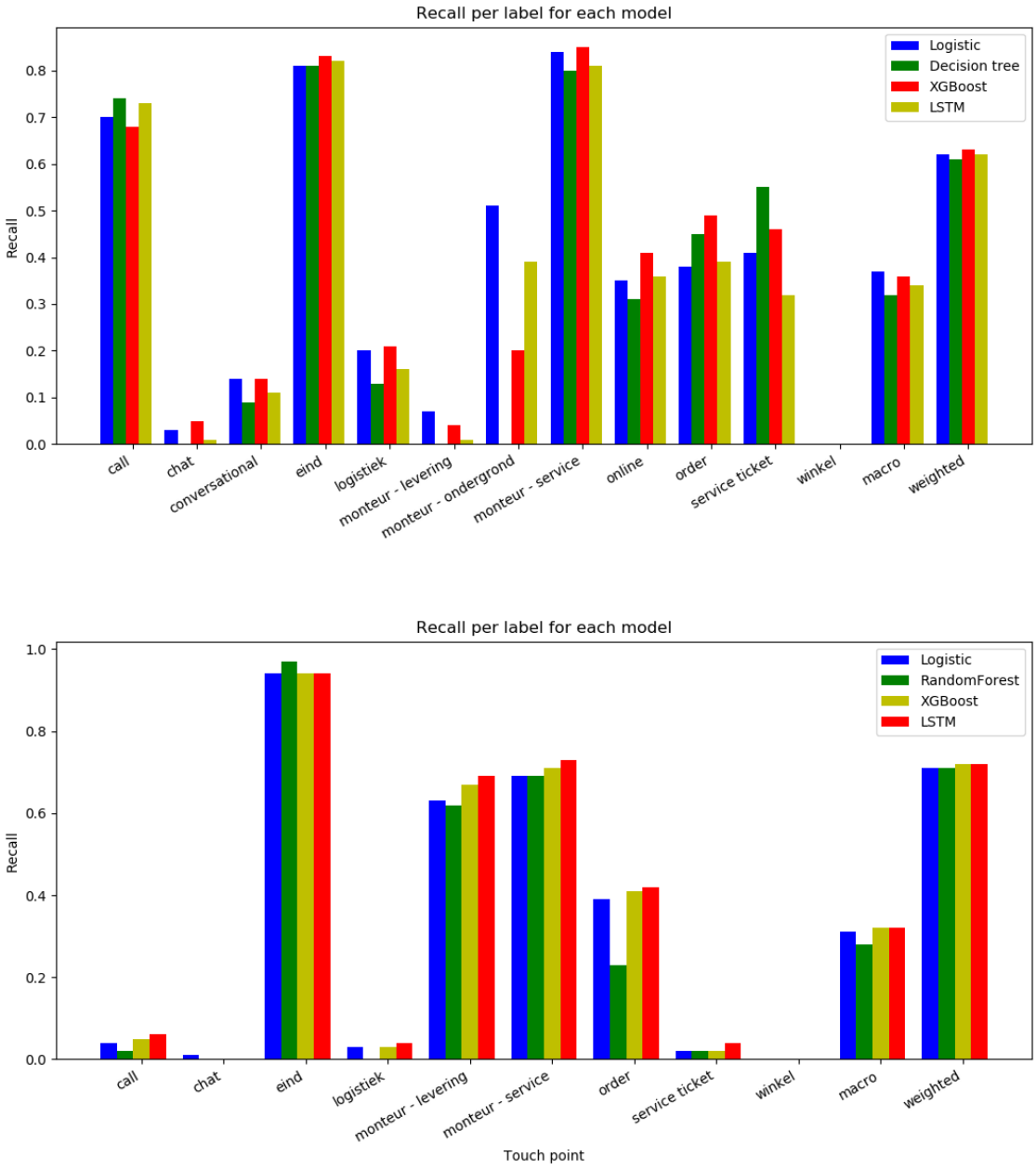


Bucket dataset



Ordered dataset

Precision bar plots

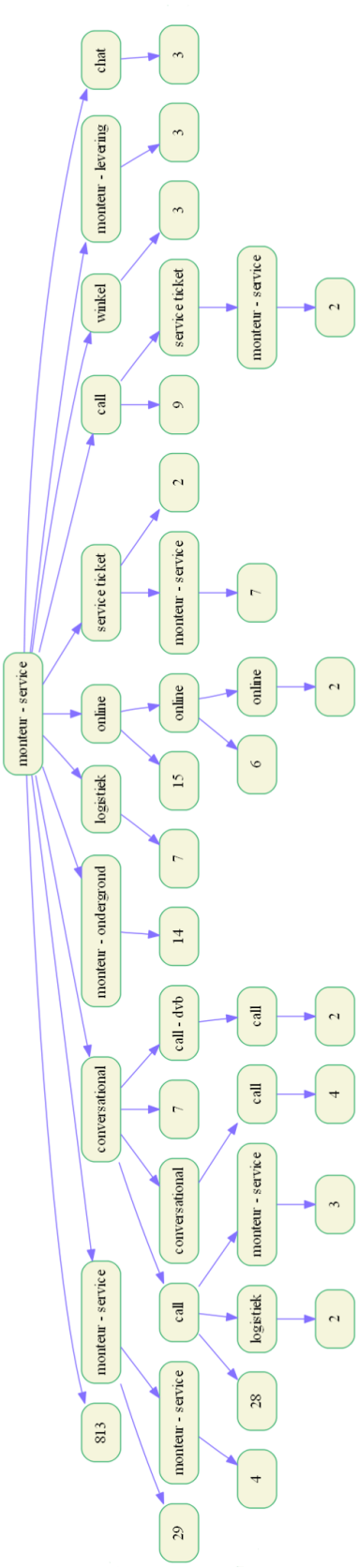


Recall bar plots

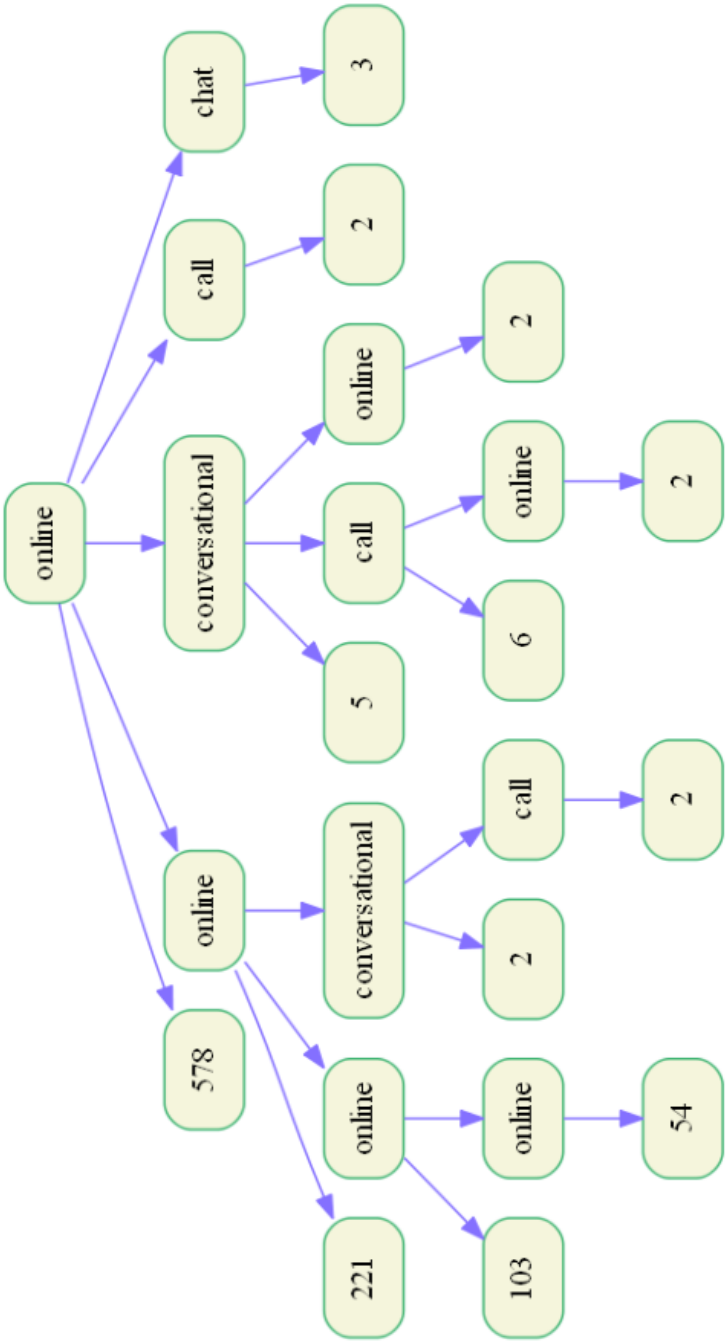
Appendix B

Promille graphs

The promille graph of the customer journeys starting with **monteur - service** and **online** are shown, as in Section 5.3.3 only the promille graph of **call** is shown. These graphs show the first four steps of all customer journeys starting from the respective touch point and how they are distributed. We see that most customer journeys are very short with more than half even being of length one.



Monteur - service promille graph

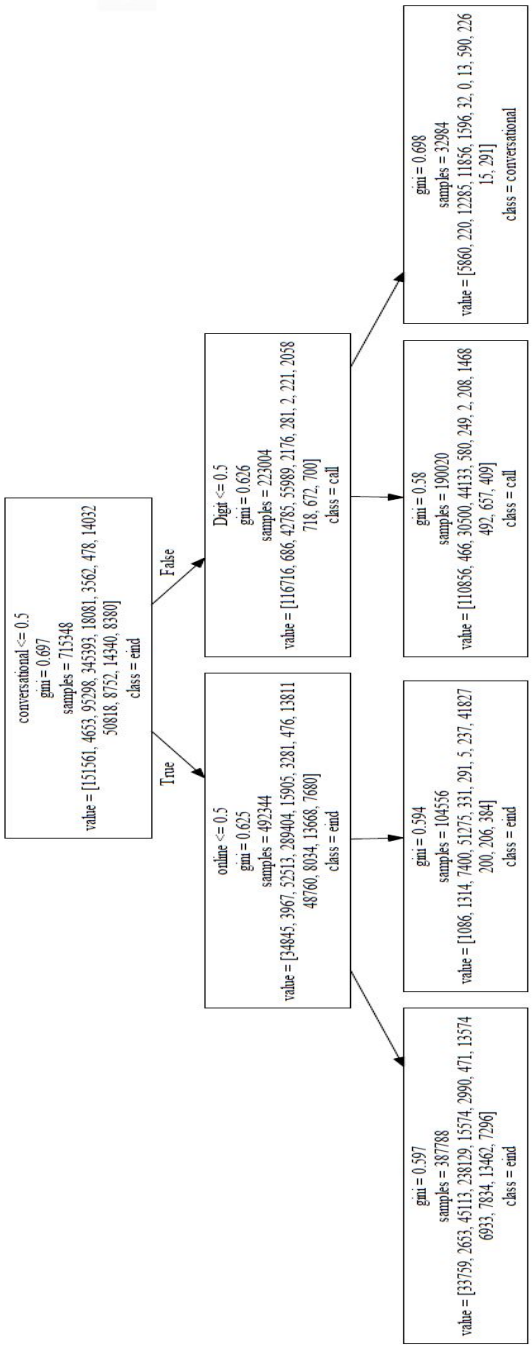


Online promille graph

Appendix C

Decision tree

To investigate interesting features we took a look at decision trees and below is a small example of a decision tree of depth two. Here we see that **conversational** is the most important feature to determine the outcome of a prediction. After that the touch point **online** plays an important role or the bucket *Digit* which is short for Digitenne. Digitenne is a TV product of KPN, used for interactive television. We did look at deeper trees but they are not easily captured in an image.



Decision tree of depth two