Applying data mining in telecommunications

Radosavljevik, D.

Chapter 6

Service Revenue Forecasting in Telecommunications: A Data Science Approach

Radosavljevik, D., van der Putten, P.

Based on an extended abstract published in Benelearn 2017: Proceedings of the Twenty-Sixth Benelux Conference on Machine Learning. pp. 187–189 (2017)

In many industries revenue forecasts can drive important business decisions, such as when and where to intervene in order to accomplish business targets. For service based businesses such as telecommunications, timely and precise service revenue forecasts are of great importance. Furthermore, having a simulation platform for service revenue forecasting can provide the business with an idea of how these measures could play out in practice under multiple scenarios. This chapter discusses a real-world case of revenue forecasting in telecommunications. Apart from the method we developed, we will describe the implementation, which is the key factor of success for data science solutions in a business environment. We will also describe some of the challenges that occurred and our solutions to them. Last, but not least we will present the results of this process and explain our problem specific choice for error measure.

6.1 Introduction

In this chapter we focus on the postpaid segment of mobile telecommunications, where service revenues can be split into fixed (subscription fee which already contains certain services) and variable revenues (based on additional usage of services not included in the subscription). Furthermore, operators charge differently for using services while abroad (roaming) and for making international calls. Operators also charge other operators interconnect fees for incoming calls (a customer from operator B calls a customer from operator A), which results in costs and revenues not visible to the customer.

T-Mobile Netherlands, the operator where we developed and deployed this research, has about 2 million postpaid customers with more than 4000 combinations of differently priced rate plans and voice, SMS and Internet bundles. In order to forecast the revenue figure for one month, one has to account not only for the different usage patterns throughout the year, but also for the inflow of new customers, changes in contract of the existing ones and the loss of customers to competition. The systems that are used for actual customer billing are not built for simulation of revenues, as these are too embedded into operational processes. This makes the task of forecasting revenue across different scenarios far from trivial.

The standard approach for forecasting service revenues developed by the operator where we conducted this research is a very high level approach, comprising of many weeks of manual labor. It is executed via complex Excel (Microsoft Corporation, 2010) sheets and it delivers a non-transparent view of the forecasted service revenues, with no possibility of a breakdown per product (i.e. rate plan). Furthermore, due to the Excel based tooling, the process requires a lot of attention to detail in order to avoid errors and does not allow for efficient calculation of multiple scenarios, because it takes too long to generate the results.

Therefore, it was necessary to develop a new approach for service revenue forecasting in order to overcome the weaknesses of the existing one. The end objective

of this research, i.e. the task presented to us by the operator is to predict the service revenues for one year ahead on a monthly basis. As stated in the previous chapter, we wanted to adapt the work described there to a completely different domain. Hence the research question in this chapter is:

How can data mining be used to predict and simulate service revenues in telecommunications? In other words, does the approach developed in chapter 5 generalize to a different domain such as finance?

This chapter could be relevant to researchers interested in forecasting revenues in other industries, especially service based businesses. Furthermore, the simulation framework we have developed for testing multiple scenarios might be of interest to researchers in other domains as well, for example micro-simulation (Li and O'Donoghue, 2013).

The rest of this chapter is structured as follows. In Section 6.2 we present the related work. A short overview of our approach is given in Section 6.3. We address the process of data collection and understanding in Section 6.4. A detailed description of our end-to-end process is provided in Section 6.5. The results are presented in Section 6.6, while the discussion and the lesson learned are in Section 6.7. We address the limitations our research and future work in Section 6.8 and provide the conclusion in Section 6.9.

6.2 Related Work

Given the importance of the process of service revenue forecasting, it is surprising there is not a lot of research on this topic. There is literature available regarding revenue forecasting in the government sector: Fullerton (1989) compared time series with econometric models and composite models, while Feenberg, Gentry, Gilroy and Rosen (1989) tested the rationality of the methods used for this purpose. Weatherford and Belobaba (2002) addressed revenue management in the airline industry, while a comparison of forecasting methods for hotel revenue management is the focus of Weatherford and Kimes (2003). Time series were used by Trueman, Wong and Zhang (2001) in order to forecast revenues of online firms.

On the other hand, a good overview of applications of machine learning for supply chain forecasting was given by Carbonneau, Laframboise and Vahidov (2008), who compared the performance of various algorithms on the sales data of the Canadian foundry industry. According to this research, even though complex models such as neural networks and support vector machines performed somewhat better than a linear regression model in terms of Mean Absolute Error, the difference was not statistically significant. Heikkilä (2002) focused on supply chain forecasting in telecommunications, but this paper is mostly addressing the problem from a perspective of the relationship between suppliers of telecom network equipment and telecom operators.

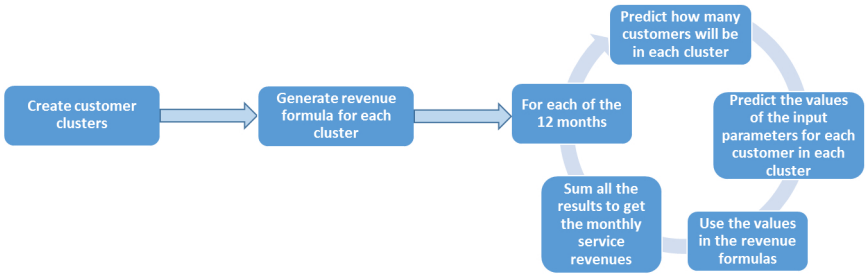

Figure 6.1: Overview of the Service Revenue Forecasting Process

However, none of these papers describe the process of service revenue forecasting in mobile telecommunications.

6.3 Overview of Our Approach

As stated above, the company where we developed this research required a much more timely, detailed and flexible approach for service revenue forecasting. In this section we provide an overview of the method we designed for this purpose. The ultimate goal of this approach was to forecast service revenues for one year on a monthly basis. A more detailed description of each step follows in Section 6.5. Nevertheless, the simplest description of our method is given on Figure 6.1: First, create clusters of customers; second, create revenue formula for each cluster based on certain inputs; third, predict how many customers will be in each cluster in each month; fourth, predict the values of the input parameters for each customer in each cluster for each month; last, use these inputs in the service revenue formulas generated and sum up to generate the monthly revenues.

As suggested in the previous chapter, we based our solution for service revenue forecasting on the method we developed for forecasting network load (Radosavljevik and van der Putten, 2014). Similar to the concept of load pockets (Feinberg and Genethliou, 2010) introduced in the previous chapter (i.e. network cells), we here used the concept of revenue pockets. Namely, we treated clusters of customer rate plans as revenue pockets.

The rationale behind clustering the rate plans is the following. It was not feasible to come up with a single service revenue model for all customers due to the many different rate plans that the operator offered (i.e. more than 4000 combinations of voice minutes, SMS and megabytes in a bundle). Furthermore, the usage habits and billing were quite different for customers in various rate plans. Despite the fact that all customers in a single rate plan could already be treated as a cluster due to their

similarity in usage and the way they are billed for services, a unique formula for each price plan was not feasible either, as some rate plans only contained a few customers. Therefore, some sort of rate plan based clustering for customers was necessary. We treated these clusters of rate plans as revenue pockets because we expected each of these clusters to contain customers with similar usage and similar billing methods.

After creating these clusters, we created service revenue models for each of them. These models were predicting monthly service revenues based on a set of input values (i.e. monthly values for usage parameters such as used voice minutes, SMS and megabytes) for each customer in a given cluster, and they were built using two months of historic data.

However, the clusters we created will not remain the same for the entire year and neither will the values for the input parameters. Therefore, we needed to make assumptions on both the quantity of customers per cluster per month (customer base changes) for the prediction period and how will the input values develop in the same period (input values changes).

In order to establish the monthly state of each cluster, we began with a snapshot of the customer base split onto rate plan clusters from the month before the prediction period. Next, we accounted for the monthly inflow and outflow of customers from each cluster, as well as customers renewing contracts and subsequently moving to a different cluster. This was performed via generating new customers for the inflow, removing certain customers for the outflow and migrating certain customers for the contract renewals, resulting in a new monthly cluster state.

After creating the monthly customer base snapshots for the full year ahead, we also needed to make assumptions for the input values for the rest of the year. This was done by extrapolating the development pattern for all of the inputs using a two year history per rate plan aggregated monthly.

Finally, after having calculated the monthly values for each input parameter for each customer in each cluster (including the incoming, outgoing and migrating customers), we used them in the revenue formulas we generated for each cluster. Summing up these values returned the monthly revenues.

It is worthwhile mentioning that we used generic off-the-shelf hardware for this purpose: a server with 128GB of RAM and 16 processing cores.

6.4 Data Collection and Understanding

Unlike in an academic research setting where seeking the best algorithm is the key to solving the problem and getting data is as easy as downloading a data set, in a business setting data collection and preparation represent a large chunk of the total work. Typically, the data is not available from a single data source, let alone structured in a format suitable for machine learning. In our case the data was spread across various Oracle (Oracle, 2011) and Teradata (Teradata Corporation, 2017) databases. The first thing we needed to do is unify the data (invoice data, usage data, interconnect

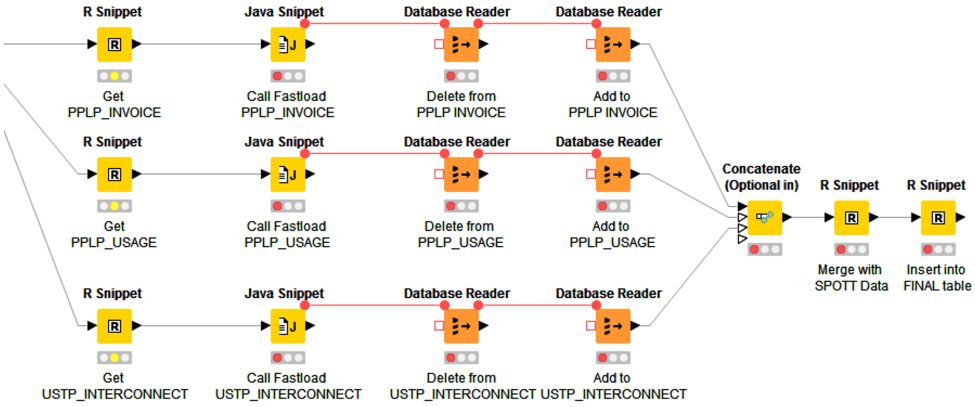

Figure 6.2: The ETL Process in KNIME using RJDBC

data, rate plans/bundles and inflow and outflow of customers) into a single sandbox which we can use to construct our flat table. We chose to use Teradata as a destination for our data set because we had the most administrative flexibility there. Any other database system would do. Also, we used the open source tools R (R Development Core Team, 2008) for moving data between data sources and KNIME (Berthold et al., 2007) for automating the process. The process is shown in figure 6.2.

Even though R is not typically used as an ETL tool, its RJDBC library (Urbanek, 2014) is very useful when necessary to export data from various database systems. For importing the data into our Teradata database we used its fastload functionality. KNIME is an analytics platform that enables using data residing in different databases and code written in different languages (Java, R, Python etc.) to be executed in a work flow fashion. It also contains a large implementation of various machine learning algorithms. During the ETL stage, we used KNIME as a kind of an orchestrator between R, Oracle and Teradata. Its visual interface makes debugging easier, as it is very clear to see which node in the diagram has failed.

After migrating the data to a single database we needed to restructure it, as it was recorded in a way corresponding to how it is presented to customers on an invoice. However, not all invoice items are service revenues (e.g. the handset fee and application purchases are not service revenues). Non service revenues are out of scope. Isolating the service revenues required a lot of expert help as each revenue type has a different code. Importing and fully aligning the data with what is officially billed and reported took almost 2 months of work. Next to this, distinction between usage that is billed vs. usage that is already part of the subscription was necessary.

The usage abroad was treated separately, because it is billed in a different manner than the usage in the home country. Last, but not least, the interconnect usage, which is not part of the customer invoice was added. Operators bill each other for the service of connecting calls between customers from a different operator. Even though these revenues are not directly billed to customers, they are part of the service revenues.

6.4.1 Data Set Description

In this subsection we describe the outcome variable and the predictors we used.

As the goal of the model is to forecast **Service revenues**, this is the obvious choice for the outcome variable. Service revenues can be further broken down into:

- **Monthly Recurring Cost**- Subscription cost paid by the customers regardless of usage (this is a constant)
- **Out of Bundle Revenues**- charged when customers use services after using up their subscription
- **Roaming Revenues**- charged when customers use services while abroad
- **Interconnect Revenues**- charged between operators when two customers of different operators call each other

Initially, the task given was to predict total service revenues without the breakdown listed above. Therefore, we used the Monthly Recurring Cost as an input.

In table 6.1 we list the inputs we used to predict the service revenues. It is worthwhile mentioning that usage abroad (roaming) is charged differently than usage in the home country (national usage). This is changing under EU regulations: all usage within EU countries is treated like national usage. Similarly, international calls made from the home country are charged differently than national calls.

We collected two months of monthly aggregated anonymized data for the outcome variable and the inputs in order to make our predictions. T-Mobile Netherlands, the operator where the research was performed, has two million customers with consumer postpaid contracts, so there were about 4 million records in the data set. Given the size of each cluster, ranging from minimum 500 customers (so 1000 data points) to more than 30,000 customers, this should be sufficient amount of data to model the dependencies between the inputs and the outcome per cluster. We only used two months of history for service revenue modeling because the pricing of services can change substantially over time, so the most recent data was the most relevant. The operator is under legal obligation to keep this data in order to be able to reproduce the invoices if necessary, for a period of three years. For trending the future values of the input parameters as they are crucial for predicting the future service revenues, we used data aggregated per cluster per month with history of 24 months.

Table 6.1: Inputs used for creating service revenue models

variable	Description
Monthly Recurring Cost	Subscription cost paid by the customers regardless of usage (this is a constant)
MBs within bundle	Amount of MB(megabytes) used within subscription
MBs outside bundle	Amount of MB(megabytes) used outside subscription
MBs in roaming within bundle	Amount of MB(megabytes) used abroad within subscription
MBs in roaming outside bundle	Amount of MB(megabytes) used abroad outside subscription
Voice Minutes within bundle	Duration of voice calls used within subscription
Voice Minutes outside bundle	Duration of voice calls used outside subscription
Voice Minutes International within bundle	Duration of international voice calls used within subscription
Voice Minutes International outside bundle	Duration of international voice calls used outside subscription
Voice Minutes in roaming within bundle	Duration of voice calls used abroad within subscription
Voice Minutes in roaming outside bundle	Duration of voice calls used abroad outside subscription
SMS within bundle	Count of SMS used within subscription
SMS outside bundle	Count of SMS used outside subscription
SMS in roaming within bundle	Count of SMS used abroad within subscription
SMS in roaming outside bundle	Count of SMS used abroad outside subscription
Incoming MMS count	Count of MMS messages received by the customer (this is service that is not used by many customers so the most frequent value is 0)
Incoming SMS count	Count of SMS messages received by the customer
Incoming Voice Calls Count	Count of voice calls received by the customer
Incoming Voice Calls Duration	Duration of voice calls received by the customer
Other usage	Count of usage of other services offered by the operators (e.g. Fax, MMS; services that are not used by many customers so the most frequent value is 0)

6.5 Clustering, Modeling and Deployment

In this section we will discuss our end to end approach to modeling service revenues. As stated in the introduction of this chapter, we have a five step approach to forecasting service revenues for a year. Therefore, this section is divided into five subsections. The first subsection describes how we clustered the customers. The second describes how we created revenue formulas for each cluster based on the inputs we listed in the previous subsection. The third subsection explains how we accounted for the monthly changes in the clusters with respect to how many and which customers remain, join or leave the cluster. In the fourth subsection we explain the method we used to extrapolate the values of the input parameters for each customer in each cluster for each month. Last, in the fifth subsection we explain how we combined the products of the first four steps in order to forecast the monthly revenues.

6.5.1 Clustering the Data

As stated in the introduction section, it was not feasible to come up with a single service revenue model for all customers due to the many different rate plans that the operator offered. Furthermore, the usage habits of customers are quite different for the various rate plans, as well as the way they are billed for the usage. However, a unique formula for each price plan was not feasible either, as some of these only contained a few customers. Therefore, we divided the entire customer base into 156 clusters, manually combining rate plans and bundles of similar characteristics (similar number of minutes, SMS and MB in the subscription), with a lower limit of 500 customers per cluster. Some clusters consisted of only one rate plan, in case when this rate plan already contained many customers (in certain rate plans there are more than 30,000 customers). Clustering algorithms were considered, however we had enough of clear structure in the rate plans to avoid this.

6.5.2 Generating the Service Revenue Formulas per Cluster

The approach we used is similar to the network capacity model described in chapter 5, where we discussed load pockets to simulate network service quality. Here, we treated each cluster as a revenue pocket, because each cluster consisted of customers who are billed in a similar manner for using services. Therefore, we created a model for each cluster of customers, taking various types of usage as input and service revenues as output, and saved it for scoring later. This process is automated using KNIME. A screen shot of the process is shown on Figure 6.3. For modeling revenues per cluster we used a data set containing two months history, which guaranteed a minimum of one thousand data points per cluster. As previously stated, we limited the data to two months of history because pricing of services can change substantially over time, so the most recent data was the most relevant in order to express the relationship between usage and service revenues.

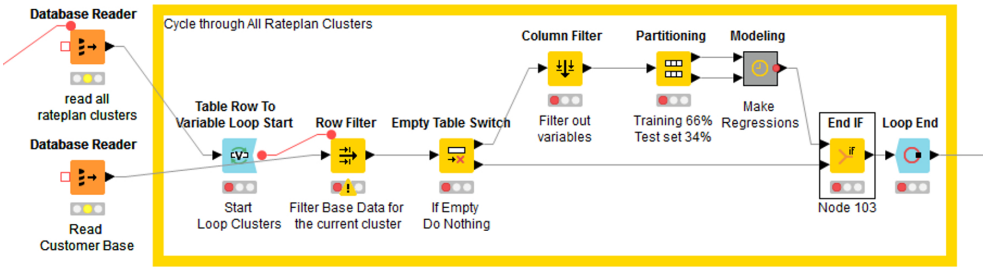

Figure 6.3: Modeling Workflow in KNIME

A KNIME workflow consists of nodes which perform certain tasks and pass on the computed result to another node. This process begins by firstly reading out the 156 clusters (node 'read all rate plan clusters' on Figure 6.3) and reading out all of the customer data (node 'Read Customer Base') including the cluster they belong in, the input variables (see Subsection 6.4.1) and the outcome (service revenues). At that moment, the data set consists of around four million data points. Next, we loop through the clusters and for each cluster the following is performed. First, only the data for customers belonging to the current cluster is isolated. The size of the data set per cluster ranges between 1000 and over 60,000 data points. Just for safety, we check if the result is empty, so that the rest of the code does not execute on an empty data set. If the resulting data set is not empty, we leave out certain variables that we do not use as inputs (e.g. name of the rate plan etc.). Next, we split the data for that cluster onto training and testing set with ratio 66%:34%, respectively. In the end, we run the modeling node. In the modeling node, a service revenue model is generated and saved using the cluster name, so we can use it for forecasting the service revenues.

We initially setup the process using linear regression as implemented in KNIME (Berthold et al., 2007) for modeling. However, we also experimented with other algorithms, such as support vector regression, random forests, regression trees, polynomial regression etc. KNIME allows for integration with WEKA (Hall et al., 2009), so we tested WEKA's implementation of linear regression combined with a wrapper as well. This is similar to what we used in the previous chapter for modeling network load. Last but not least, motivated by the No Free Lunch theorem (Wolpert and Macready, 1997) we made a so called 'winner' model, where each cluster was modeled by the algorithm that was the best performing on that particular cluster on the test set. From a methodological perspective, the results of the 'winner' model should not be compared to the rest, because we used the test set to select the winning model.

As stated in Subsection 6.4.1, service revenues can be further broken down into: Monthly Recurring Cost, Out of Bundle Revenues, Roaming Revenues and Interconnect Revenues. We had data for each of these variables, therefore we decided to look into modeling them separately. This was motivated by two factors: the business required a breakdown of revenues into components and we expected error reduction when modeling them separately, even when compounding the error.

We expected the model error to reduce due to the following. The Monthly Recurring Cost is independent of usage and can be treated as a constant. Therefore, it requires no modeling, which allows for error reduction. The other three components are much more linear in nature and completely dependent on usage. Therefore, they can be modeled using linear regression. Furthermore, the variable selection process is much more straightforward when modeling the components of service revenues. For example, when modeling the roaming revenues, we only use the input variables from Table 6.1 containing the word roaming in their name (Voice Minutes, SMS and MB in roaming, both within and outside bundle), as these are the only variables that can influence the roaming revenues. Similarly, all the variables from Table 6.1 with the word Incoming in their name (Incoming MMS, SMS, Voice call count and duration) are used for modeling the Interconnect revenues. All other variables from Table 6.1 are used for modeling the Out of Bundle Revenues.

The results of both modeling methods (modeling service revenues as a whole and modeling components of service revenues separately) are shown in Subsection 6.6.1.

6.5.3 Generating the Monthly State of the Clusters- Inflow, Changes in Contract and Outflow of Customers

Customer base growth or reduction substantially affects the total service revenues generated. Furthermore, customers changing their rate plans can also have influence on the revenues they generate. For the purpose of our service revenue forecasting process, we have received the absolute amount of incoming and outgoing customers, as well as the amount of contract renewals per month from the Marketing and Sales department.

For the incoming and renewing customers we also received the breakdown per rate plan cluster to be used in the forecasting process. It is worthwhile mentioning that these figures were based on the signing moment and not on the subscription activation moment. The difference between signing and activation can range up to four months. Revenue generation only occurs after the subscription has been activated. In order to bridge this, we created a translation between activation and signing using historic percentages of customers that activate their subscriptions within zero, one, two, three and four months.

For the outflow of customers, we created the breakdown per rate plan cluster using recent churn history per cluster and the totals acquired from the business. This part of our approach could be improved, but given the time limitation for the project

it was a good enough solution for forecasting purposes. Please note that modeling churn for budgeting purposes and modeling churn for campaigning purposes are processes that are very different. Modeling churn for budgeting purposes requires only the total expected amount of churners, while modeling churn for campaigning purposes requires churn scores for each customer so that they can be approached with an extension offer. Literature mostly addresses churn modeling for campaigning purposes (see chapters 2 and 3). One potential approach to using campaigning churn models for budgeting purposes would be to create the churn scores as true probabilities (not just scores) and then sum these up in order to get the absolute amount of churners. However, this was out of scope of this research for the moment.

We accepted the business plans for inflow, contract renewal and outflow of customers for two different reasons. First, these figures were also used in the standard budgeting process, so they would be useful for benchmarking purposes. Second, the inflow, renewals and outflow of customers are highly dependent on the business plans of the operator (e.g. planned discounts and marketing campaigns), so relying only on historic data without input from the business may not lead to best prediction results.

The monthly state of the clusters for each consecutive month is generated by taking the state of the cluster in the previous month, then adding the planned amount of new "imaginary" customers to the cluster (the customer inflow of that cluster for that month), removing randomly selected customers from that cluster for the case of outflow (the planned amount of churners for that cluster for that month) and also migrating randomly selected customers to different clusters (the planned migrations for that cluster for that month). For the new and the migrating customers, the business also provides the Monthly Recurring Cost, as it is part of their planning. It is worthwhile mentioning that the customers that are selected for churn are still generating revenues in the month they churn, so they were only removed once the revenues were calculated. In the month the churn occurs, they were just marked as churners to be removed from the cluster next month.

6.5.4 Generating the Values of Input Parameters- Scaling Factors

In order to predict the service revenues for one year, we also needed all the inputs listed in Table 6.1 to be extrapolated on a monthly basis for the same period, except for the Monthly Recurring Cost, which is a constant and does not change on a monthly level.

This part of the process was developed by the Mobile Network department of the operator. Forecasting the total usage of voice minutes, SMS and Internet services (total amount of megabytes) on network level is a standard part of their work. They use these forecasts to predict the network load, so that they can intervene before an overload occurs. However, we needed values per rate plan cluster, so they applied the same approach on the cluster data and all the input parameters listed in Table 6.1 (except for the Monthly Recurring Cost). For this purpose, a time series approach

(Hamilton, 1994) implemented in R using the packages tseries (Trapletti and Hornik, 2017) and forecast (Hyndman and Khandakar, 2008) was used. The results of these functions would return average value per month for each parameter and every cluster. We corrected these general monthly trends using the number of working days, weekends and holidays per month because we observed that the total daily traffic for the operator substantially differs for weekdays, weekends and holidays. For example, the total voice minutes per day is much higher on weekdays compared to weekends or holidays. Naturally, the number of days per month also plays a role in the monthly averages. We also addressed the seasonality in the roaming parameters (usage abroad): these parameters have much higher values during holiday periods, especially in the summer months. In the end of this step we created so-called scaling factors which give the ratios between the values of the input parameters of this month and their values in the previous month. These scaling factors were then used to generate the monthly usage of each customer and consequentially for forecasting the service revenues.

6.5.5 Putting it All Together

After generating the monthly values for the inputs and the monthly states of the customer base, they are used for forecasting the total service revenues for one year. This is achieved using the following process:

1. *Get the customer base and values of input parameters for the current month*
2. *Repeat 12 times*
   - *2.1 For each cluster*
     - *2.1.1 Create the customer base for next month by*
       - *2.1.1.1 Removing customers marked for discontinuation of contract in the current month*
       - *2.1.1.2 Adding customers that signed a contract ("imaginary customers") including their Monthly Recurring Cost*
       - *2.1.1.3 Changing the rate plan for customers who renewed their contract including their Monthly Recurring Cost*
       - *2.1.1.4 Marking customers to discontinue in the next month*
     - *2.1.2 Use the scaling factors to create the values for the inputs*
     - *2.1.3 For each input parameter the new customers are given the average value for their cluster*
     - *2.1.4 The input values are plugged into the revenue models corresponding to the cluster to predict the revenues for each customer in the cluster*
     - *2.1.5 Sum the values of all customers in the cluster*
   - *2.2 Sum the revenues of all clusters in that month and move to the next month*

Table 6.2: Algorithm performance

	Amount of Winning Models	Correlation Coefficient	MAE
Linear Regression KNIME	44	0.847	2.210
Linear Regression WEKA	66	0.855	2.105
Polynomial Regression	22	0.818	2.137
Random Forest	19	0.816	2.393
Simple Regression Tree	5	0.727	3.020
Winner		0.858	2.063

Please note that all customers added in step 2.1.1.2 of the procedure above are treated as average customers for that rate plan. The input parameters used for calculating their revenues are averages of the values for the input parameters of all other customers in that cluster in that month, as these are unknown customers and there is no information about their usage.

In the case of forecasting the service revenues as a whole (without a breakdown onto components), step 2.1.4 of this algorithm is performed only once for each cluster, using the service revenues model generated for that cluster. In the case of forecasting the components separately, we run the values of the input parameters through three models (the corresponding models for that cluster for Interconnect Revenues, Out of Bundle Revenues and Roaming Revenues, respectively) and add them to the Monthly Recurring Cost in order to calculate the service revenues:

$$\begin{aligned} TotalServiceRevenues(predicted) = MonthlyRecurringCost \\ +InterconnectRevenues(predicted) \\ +OutOfBundleRevenues(predicted) \\ +RoamingRevenues(predicted) \end{aligned} \tag{6.1}$$

6.6 Results

In this section we will present the results of the service revenue modeling process as well as the results of the forecasting process for one year.

6.6.1 Results of Service Revenues Modeling

As stated in Subsection 6.5.2 of this chapter, we used two different approaches to model the service revenues. The results of the process when modeling service revenues as a whole (not using the breakdown into components) are shown in Table 6.2. The values for the correlation coefficient between the actual values and predicted values, and the mean absolute error (in euros) are shown in the columns Correlation

Table 6.3: Modeling Service Revenue Components

	Correlation Coefficient	MAE
Interconnect Revenues	0.987	0.292
Out Of Bundle Revenues	0.871	1.384
Roaming Revenues	0.935	0.242
Total Service Revenues	0.931	1.918

Coefficient and MAE, respectively. Both were calculated using the actual and predicted value of the total service revenues on the test set using the same algorithm to model all clusters. From all the algorithms we used, the WEKA implementation of linear regression (using a wrapper for preselecting variables for modeling) performed the best in terms of Correlation Coefficient and Mean Absolute Error. It is interesting to note that using more complex algorithms did not improve performance in general, which is similar to the results of Carbonneau et al. (2008).

This is why we created the Winner model, using the best performing algorithm per cluster. In most cases, the WEKA implementation of linear regression is the winner of this contest (column Amount of Winning Models in table 6.2), followed by the KNIME implementation of linear regression, then polynomial regression, random forest regression and simple regression tree. From a methodological perspective, the results of the Winner model should not be compared to the rest, because we used the test set to select the winning model. Strictly speaking, in order to be able to compare the Winner approach with each of the individual models, one should use an intermediate test set (i.e. validation set) to select the best model per cluster and then check the performance of the Winner model on a test set which was not used for selecting it. However, the results of the Winner model we report in Table 6.2 are a good indication that this approach could bring an improvement.

We also tried to test with Tree Ensemble and Support Vector Regression. However, these algorithms did not contribute significantly in reducing the total error while substantially increasing the computation time (to hours per cluster in some cases).

Next, as explained in Subsection 6.5.2, we modeled the total service revenues by modeling each of components of service revenues separately.

The results of this experiment are shown in table 6.3. It is worthwhile mentioning that we calculate the total service revenues according to equation 6.1: by adding together the revenue components and the Monthly Recurring Cost (which is a constant for each customer).

Therefore, the Mean Absolute Error and the Correlation Coefficient are related to the three predictions added together, so it is not a sum of the errors separately. Treating the errors separately results in a Mean Absolute Error of 2.199. However, our objective is to predict the Total Service Revenues better. In order to make a fair comparison to the results in table 6.2, we should be comparing the error on Total Service Revenues from the first experiment with the total error of the second experiment

as shown in table 6.3, which shows an improvement over the first approach (MAE of 1.918 vs. MAE of 2.063).

Even though it is possible that deploying a "winner" approach similar to the one we refer to in table 6.2 onto the separate revenue components could further decrease the overall error, we opted to use linear regression. This was based on both the performance shown in table 6.3 as well as the readability of the linear models, which contributes to the transparency of the model and acceptance by the business.

6.6.2 Results of the Forecasting Process for a Full Year

The goal of the model was to forecast service revenues for a full year. As already stated in the first section of this chapter, in order to predict the service revenues for a full year, it was necessary to first forecast the monthly values of the input parameters as well as the changes in the customer base via the inflow, changes in contract and outflow of customers for the same period.

To validate our approach we used MAE, which is a typical prediction level measure. However, the key measure which was the only relevant measure for the end users of the model and the business altogether was how close the sum of our predictions is to the actual revenues. When predicting the revenues for the first four months of 2017 our model was only 0.3% off from the actual revenues, while the error of the standard budgeting process was 8 times higher. Due to confidentiality, we cannot include detailed numbers. This error measurement makes sense from a business perspective: the objective of the model was to predict total service revenues for one year. It was also the only feasible measure to compare the performance of the old and new process, as the old process could not measure MAE. As stated in Section 1 of this chapter, the old approach was very high level and it only reported a monthly total, without breaking the revenues onto clusters or customers.

We display the forecasting results in Qlikview (QlikTech International AB, 2014), providing drill down opportunities per rate plan, sales channel, voice minutes or Internet bundle etc. The business has gained better insights into the forecast than the actuals, resulting in a request to update operational reporting on the actuals in the same way.

The model is currently under review by the operator's financial controlling group. On top of the achieved performance in terms of accuracy it also substantially reduced manual labor in the revenue forecasting process. This model also provided an opportunity for testing multiple scenarios, which was previously not possible.

6.7 Discussion and Lessons Learned

One of the most difficult challenges we were facing during this research was to gain acceptance of the end-customer of the model. Several factors influenced this positively, but the key factor was the transparency of the model. Therefore, we

opened the source data, the forecast for the input parameters and the regression models to the end-users. Using linear regression was a key factor here, due to readability. If we were using more complex models, it would be almost impossible for the users to understand them in detail and get past the black-box phenomenon. Therefore, even though in some cases the more complex models lowered the MAE, we opted for a more uniform and transparent model. Furthermore, when comparing the performance between the old and new approach we used a simple error measure, which was the difference between the outcome of either of the methods and the actual results. Even though we agree this is not the most precise way of estimating the error, it was pivotal for the success of our method, because we were tasked to forecast the revenues for a full year. Of course, the monthly errors were also shown, so the users could see that there were no large fluctuations.

Most importantly, we used the same data sources and definitions of revenues as the financial department (the end user). However, we did create our own data repository where all the data was stored together. Therefore, we had to verify with the end users that the historic totals per month were matching. Transforming the data from the source systems into a format that was suitable for our analysis required a long time and a lot of domain expert knowledge, but it also helped us get the end users on board in the process. A data repository similar to ours can also be very useful for monthly financial reporting, hence a similar repository is now being built for operational purposes.

From a technical perspective, one of the lessons learned was to keep all tables in memory, instead of on disk. We used a server machine with 128GB of RAM memory (standard off the shelf hardware), which was sufficient for this purpose. This reduced the run time of the total modeling process from one hour to 10 minutes when using linear regression (when modeling the total service revenues by components we generate 468 models- three for each of the cluster). When using the more complex algorithms, the training for some clusters lasted longer than a few hours to calculate, especially when the sample size was large (more than 30,000 samples).

Trying to forecast the service revenues for a full year created a challenge from a run time perspective: our first run was too slow- 52 hours for predicting only four months ahead (so just one third of the task). We were able to reduce this to about 15 minutes per month of prediction (or approximately 3 hours of total runtime) by using R instead of KNIME for some data transformations, keeping everything in memory and only writing to disk once the full month was calculated, as well as optimizing the process flow of generating "new" random average customers for the inflow and contract changes and removing randomly selected customers from the base for the outflow. The main advantage of R versus KNIME was that table joins and aggregations using the dplyr package (Wickham and Francois, 2016) performed much faster. Ultimately, the whole process would run fastest if it was entirely performed in R. However, we are at the moment using the combination of R and KNIME for two reasons. First, there is no business requirement to further optimize performance, as

this process already replaces months of manual work with approximately 3 hours execution time. Second, this combination of tooling is much easier to debug and detect errors if they occur, given that KNIME displays the modeling pipeline in a graphical manner.

The short runtime and the flexibility of changing the inputs created opportunities for testing of multiple scenarios. For example, we used the inflow of customers, renewing contracts and outflow as they are provided by the business, using their default planning as a baseline. Furthermore, we used the historic data for the revenue models' inputs (the usage of services) to predict their values in the future. Our approach offers testing different scenarios for both the inflow, outflow and renewing contracts, as well as different levels of changes in the inputs for the revenue formulas. For example, instead of only relying on the historic data for making assumptions on how the inputs for the revenue models will develop in the future, the operator could now test the effect of increased or decreased usage onto service revenues. This particular functionality of our approach enabled the operator to test their assumptions stemming from the European regulations on abolishing roaming charges within the EU. Furthermore, one can also simulate different pricing levels (mostly via the Monthly Recurring Cost for new customers or customers migrating to a different cluster), which was not possible with the standard forecasting process. Being able to simulate on micro level across three different dimensions (changes in customer base, usage and pricing) created a versatile simulation platform that could be of interest to researchers or data mining practitioners in many different fields.

One interpretation of Pareto's rule (Pareto, 1964), especially popular in business, is that 80 percent of the result can be accomplished with 20 percent of the effort (Koch, 2011). In business circumstances, 80 percent is very often enough. Although the runtime of approximately 3 hours can be optimized further, this will only be done once the model is in full production use and if requested by the end users, keeping in mind that the runtime of the approach previously used for this purpose was measured in weeks.

Our approach can generalize to revenue simulation of any subscription based service with usage based pricing. Furthermore, it could be applicable to supply chain management, or the demand side of it, if applied to industries that have multiple products and many different customers purchasing them, for example fast moving consumer goods. One can imagine creating clusters of various products and forecasting their consumption and consequentially the revenues this would generate. Another potential application of this method in telecommunications would be adapting it to the supply chain management of mobile phone purchases, as each phone comes in multiple configurations (even colors). Storing these devices for a long time is expensive, as they lose their value over time when they are not sold, so creating a low level granularity model per device would be valuable.

6.8 Limitations and Future work

One of the limitations of this research is that we randomly selected customers from each cluster to exclude when modeling the outflow, using the quantities received from the business. This could be improved by calculating actual churn probabilities for each customer in a given cluster and excluding the ones with the highest rank, even if we were to accept the quantities provided by the business. The same scenario could apply for customers that are migrated to a different cluster (renewing customers).

An additional limitation of this research is accepting the total figures for inflow, outflow and migrations of customers per month as created by the business. We do not have detailed knowledge on how these are created. As part of our future work, we envision improving some of the business processes we took as business input in this research. For example, forecasting the amount of outgoing customers and the breakdown per rate plan can be improved by using survival analyses (Miller Jr, 2011) or fitting churn curves. The same applies for renewing customers. We already started an initial investigation into these processes.

Using these two approaches, we could improve the churn and renewal forecasting both from the perspective of total customers that will churn or migrate, as well as which customers would churn or migrate from each cluster.

As a part of our future work, we could also envision applying an approach similar to this to the field of supply chain management in mobile telecommunications, more specifically the mobile phone demand mentioned in the discussion section.

6.9 Conclusion

Addressing the research question from section 6.1, we have managed to generalize the approach we developed in chapter 5 to the financial domain, and use data mining to predict and simulate service revenues in telecommunications. We used generic hardware, open source tools for the majority of the process, and simple algorithms in a complex deployment flow to optimize a key business problem. The added value of this model is not only in enabling the business to have much better and much more timely insights in future revenues, assuming that the standard business plans for customer base growth are implemented. This model provides a scenario simulation platform giving the business an opportunity to test the potential measures designed to increase revenues at a much higher pace.