MEASURING AND COMPARING PREDICTIVE POWER OF CHURN ESTIMATION METHODS

By

Gilian Ponte

Pre-MSc Marketing

June 2018

Supervisor: Dr. K. Dehmamy

University of Groningen

Faculty of Economics and Business

Nettelbosje 2

9747 AE Groningen

g.r.ponte@student.rug.nl

student number: 25916321

Abstract

This article presents a comparative study of churn estimation methods. Telecommunication providers can no longer rely on a steady customer base. Machine learning methods are applied to the problem of customer churn in the telecommunications industry. In the first section, relevant variables for explaining churn behaviour are evaluated. Followed by a description of the methodology of a logistic regression, decision trees, bagging, boosting and a neural network to model churn behaviour. These models are estimated and evaluated by comparative performance measures. The results show that bagging performs better than decision trees, boosting, neural network and a logistic regression.

Keywords: Churn, machine learning, logistic regression, neural network, bagging, boosting, decision tree, Gini coefficient, top decile lift and hit rate.

Table of contents

1. Introduction

Customer relationship management is a strategy based on of building, managing and strengthen the relationship with customers (Vafeiadis, et al., 2015). Predicting churn and preventing customers from churning can save companies hundreds of thousands of dollars (Neslin, et al., 2006). The cost of attracting new customers is twelve times the cost of retaining your existing customers. Moreover, service providers can no longer rely on a steady customer base. In telecom service industries churn rates range from 23.4% to 46% (Neslin, et al., 2006). This makes customer churn a significant problem for companies. Hence, getting a clear view of which customers are likely to churn enables companies to focus on customers that are likely to churn and reactivate them (Neslin, et al, 2006). This study uses a dataset from an anonymous telecom provider.

2. Problem statement

Numerous studies have described methods for predicting customer churn (Kim & Yoon, 2004: Ahn, Han & Lee, 2006: Neslin, et al., 2006: Risselada, Verhoef & Bijmolt, 2010: Nie, et al., 2011.: Lu, Lin, Lu & Zhang, 2014.: Vafeiadis, et al., 2015). Predicting customer churn contains a decision in terms of a nominal variable, where zero describes *no* cancellation of the contract and one means the customer cancelled the contract. Methods that become relevant when the dependent variable, churn or no churn, is a binary decision are decision trees, logistic regression, neural networks, random forest, support vector machines, bagging and boosting (Neslin, et al., 2006: Günther, et al., 2011: Vafeiadis, et al., 2015). Respectively, all of these techniques when estimated have proved to have predictive power for predicting customer churn. However, there is little knowledge of modern techniques for predictive customer churn. Modern techniques need to be explored to find methodologies that might possess more predictive power (Neslin, et al., 2006: Risselada, Verhoef & Bijmolt, 2010).

Au et al. (2003) propose that neural networks outperform decision trees on large datasets. Also, experimental results showed that a neural network outperformed logistic regression and decision trees for churn prediction (Mozer et al., 2000). Ha, Cho & Maclachlan (2005) find that a neural network outperforms boosting and bagging. Therefore, this study compares the predictive power of a neural network with a decision tree, a logistic regression, bagging and boosting to predict churn using the dataset of an anonymous telecom provider.

3. Conceptual model

This chapter defines all the relevant variables that are strong predictors of customer churn and are used in estimating the models for a comparison between methodologies.

3.1 Churn

Throughout fields, the definitions of churn differ. Churn could be defined as the customers that cease doing business with a company in a given time period (Neslin et al., 2006). Or the termination of the contractual or noncontractual relationship between the customer and a company (Bijmolt, Leeflang, Block, Eisenbeiss, Hardie, et al., 2010).

Another definition by Risselada, Verhoef & Bijmolt (2010) is the termination of a contract. More specific, in the telecommunications industry, the definition of churn is defined as; the action that a customer’s telecommunications service is cancelled by a customer-initiated action (Nie, et al., 2011).

3.2 Heavy users

Usage is described by Kim & Yoon (2004) by the total calls a user makes in the subscription service. The calls a user makes has a significant effect on the probability to churn of customers (Kim & Yoon, 2004: Ahn, Han, & Lee, 2006). In the dataset, these variables are expressed as the total minutes and total calls on an interval scale.

3.3 Account length

The account length is defined as the time a customer has a subscription to the service providers, measured on an interval scale. Kim & Yoon (2004) suggest that the duration of a subscription has an effect on the user behaviour and therefore also on the probability to churn. This effect is explained by the lock-in of operators since consumers are likely to stick with the same provider to avoid switching costs (Kim & Yoon, 2004).

3.4 Costs

According to Nie et al. (2011), revenue from customers influences the likelihood to churn. High costs for customers might be a cause for consideration to switch or cancel the contract with the current provider (Kim & Yoon, 2004). Especially, if these costs exceed the switching costs. The revenue is expressed in a ratio variable in the dataset.

4. Methodology

For estimation of the models, five methodologies are evaluated in terms of their relevance and predictive power. A logistic regression, decision tree, bagging, boosting and neural networks are described.

4.1 Logistic regression

A logistic regression is one of the most used approaches to estimate the probability to churn (Neslin, et al., 2006: Risselada, Verhoef & Bijmolt, 2010). A logistic regression also referred to as a logit, is similar to a linear regression. Similar in a way that, a logistic regression is also able to test for a hypothesis. This implies that a logistic regression is able to test for significant variables, whereas a decision tree, bagging and boosting, neural networks are only able to present variable importance (Bischop, 2006). Moreover, a regression is able to output continuous variables, whereas a logistic regression only estimates the probability belonging to a group between 0 and 1 due to its sigmoid nature (Malhotra, 2010: Vafeiadis, et al., 2015). According to Bischop (2006), a logistic regression connects the latent utility to the probability of churning by a cumulative distribution function (CDF). This process is displayed in figure 1.

$$\begin{tikzcd}
\text{Utility} = \beta 0 + \beta 1Xi \arrow[rr] \arrow[dr, "\text{CDF}"'] & & \text{Choice} \\
& \text{Probability} \arrow[ur] &
\end{tikzcd}$$

Figure 1, cumulative distribution function

From the calculation of the probability through the CDF, the probability is converted to a choice, to churn or not. Consider that churn has two outcomes: the customer churned and the customer did not churn. When the probability is greater than 0.5, then the predicted value of churning is set to 1, when the value is below 0.5 the predicted value is set to 0. The probability of a customer churning is modelled using the logit model (Bishop, 2006: Malhotra, 2010).

4.2 Decision trees

Decision trees have been proved to be a popular methodology to predict churn (Nie, et al., 2011), especially due to their human readability and resistance to errors in the data (Mitchell, 1997). Decision trees classify instances by organizing information extracted from a training dataset in a hierarchical top-down structure composed of nodes and ramifications. Each node in the tree specifies an attribute of the instances (Mitchell, 1997: Nie, et al., 2011).

The most common algorithm for creating a decision tree is ID3 (Mitchell, 1997). The first question this algorithm needs to answer is which attribute is most useful for classifying instances. Recent studies have developed an algorithm CD4.5, which performs better in terms of predictive power than the original ID3 algorithm (Hssina, Merbouha, Ezzikouri, & Erritali, 2014). Both ID3 and CD4.5 are based on the information theory that uses an information gain measure called *entropy* to determine which variables separate the instances in each step of the decision tree. Entropy is a measure of the (im)purity of an arbitrary collection of examples, or the amount of unpredictability in a set of events (Mitchell, 1997). Bishop (2006) defines entropy as: “The average amount of information needed to specify the state of a random variable”.

With the definition of entropy, the next step is to define information gain. Mitchell (1997) defines information gain as: “the measure of the effectiveness of an attribute in classifying the training data”. Information gain is based on the decrease in entropy, to find the attribute that returns the highest information gain. The information gain for all attributes is calculated, which forms the hierarchy of the attributes in the decision tree (Mitchell, 1997: Bishop, 2006).

4.3 Bagging

Bagging has been proven extremely effective in improving the predictive accuracy of neural networks and decision trees (Ha, et al., 2005: Risselada, Verhoef, Bijmolt, 2010). Bagging consists of bootstrapping and aggregating. Bootstrapping in the form of training multiple tree models, also called a committee, that has each been estimated on an nth part of the original dataset. The results of these multiple models are aggregated to calculate a probability of churning for each customer (Breiman, 1996: Bishop, 2006).

4.4 Boosting

Boosting has been proven effective in increasing the performance of customer churn in retail and telecommunications companies (Vafeiadis, et al., 2015). Lu, Lin, Lu & Zhang

(2014) present boosting as a method to “boost” the predictive accuracy of a learning algorithm. One of the best known boosting techniques is AdaBoost, short for ‘adaptive boosting’ (Bishop, 2006). Ricci, Rokach Shapria and Kantor (2010) describe boosting as: “an iterative procedure to adaptively change the distribution of training data by focusing more on previously misclassified records”. Initially, all records are assigned equal weights. But, unlike bagging, weights may change at the end of each boosting round: Records that are wrongly classified will have their weights increased while records that are classified correctly will have their weights decreased.

4.5 Neural networks

Artificial neural networks are among the most effective learning methods currently known (Mitchell, 1997). Lecun, Bengio & Hinton (2015) describe a neural network as a methodology that discovers intricate structure in large datasets by using the feedforward backpropagation algorithm. Where feedforward stands for the type of neural network architecture where the connections are “fed forward”. Nie, et al (2011) describes a neural network as a supervised feed-forward neural network and usually consists of input, hidden and output layers, as visible in figure 6. Neural networks employ stochastic gradient descent to attempt to minimize the squared error between the network output values and the target values for these outputs (Mitchell, 1997: Bishop, 2006).

5. Performance measures

The top decile lift and Gini coefficient measure the accuracy of the churn model predictions. These measures are used to compare the predictive power of multiple estimation methods (Risselada, Verhoef & Bijmolt, 2010: Neslin, et al., 2006: Nie, et al., 2010: Holtrop, Wieringa, Gijsenberg & Verhoef, 2017).

5.1 Top decile lift

The top decile lift measures to what extent the model is to identify the customers with a high churn rate from other customers (Holtop, Wieringa, Gijsenberg & Verhoef, 2017). The customers are divided into ten separate groups, each 10% of the customers, according to their churn probability. The percentage of churn is calculated in the highest group and divided by the average churn probability. The result indicates to what extent the model was able to identify the top churners in comparison to the random selection of customers (Verhoef & Wieringa, 2011).

5.2 Gini coefficient

The Gini coefficient is an extension of the top decile lift measure. Compared to the top decile lift the Gini coefficient describes an overall performance of the model. The Gini coefficient is a number between zero and one, where zero is a bad performance and one is a perfect performance. It is calculated by dividing the area between the cumulative lift curve and the 45-degree line by the total area under the 45-degree line (Risselada, Verhoef, & Bijmolt, 2010).

6. Evaluation

This chapter presents the estimation of all the described methods. These methods are evaluated accordingly to the performance measures.

6.1 Decision tree

All the variables available in the dataset are used for the estimation of the decision tree presented in figure 3 is estimated. As in the following formula:

$$\Pr(Churn) \sim f(AccountLength,\ TotalCharge,\ TotalCalls,\ TotalMinutes,\ IntlPlan,\ VoiceMailPlan,\ VoiceMailMessage,\ DayMinutes,\ DayCharge,\ DayCalls,\ EveMinutes,\ EveCharge,\ EveCalls,\ NightMinutes,\ NightCharge,\ NightCalls,\ \ CustServCalls)$$

The probability of churning is a function of all the variables available in the dataset. All the variables in the dataset were added, as decision trees are very robust and are not able to identify significant variables, but only judges them on importance (Bishop, 2006).

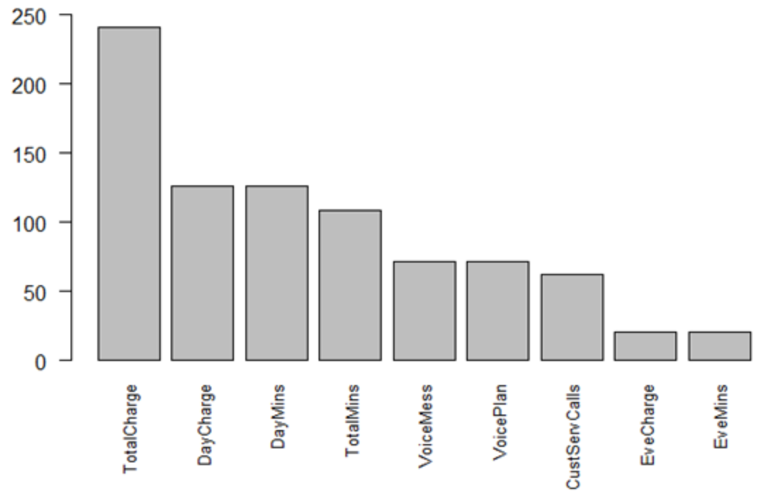

Figure 2, the variable importance of the decision tree

Based on information gain, splitting the dataset based on the total a customer was charged lead to the highest information gain and therefore is the most important variable, as presented in figure 2. Moreover, it is used as the first variable to split the decision tree as visible in figure 3.

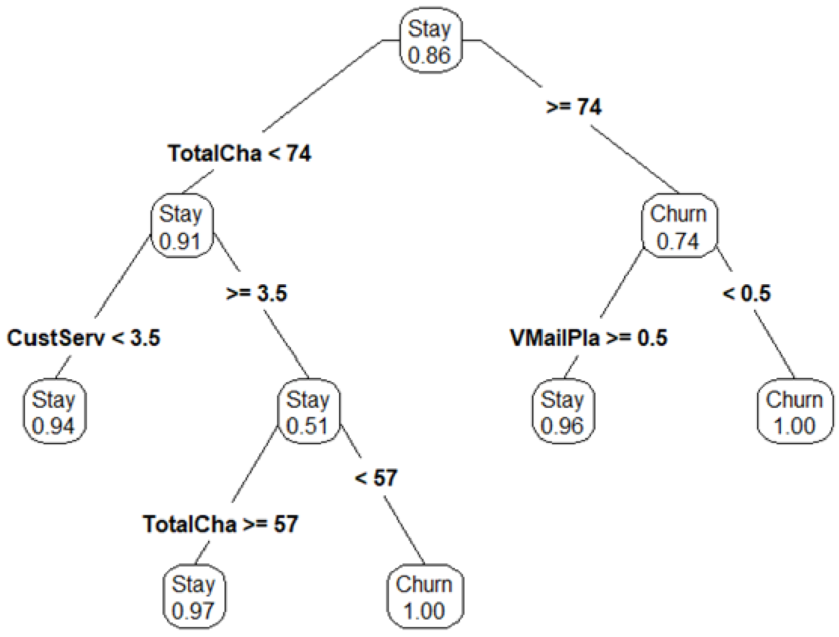

Figure 3, estimation of the decision tree.

Figure 3 could be interpreted as, when a customer has a total charge of 35 euros and called the customer service more than 4 times, the customer is .94 likely to stay. For example, when a customer has called for 70 minutes in one month and was charged 120 cents per minute, the total charge would be 84 dollars. Also, the customer has a voicemail plan. This means that customer has a probability of 96 percent to stay with the company. This shows how easy it is to interpret a decision tree.

The decision tree was able to predict 94.84% of all the customers' status correctly. The decision tree is better at identifying the customers with a high churn rate from other customers than the logistic regression, as the top decile lift is 6.47. However, the overall performance of the decision tree is weaker compared to the logistic regression, as the Gini coefficient is .53.

6.2 Logistic regression

In table 1, two logistic models are presented. The first logistic estimation includes all the available variables in the telecom dataset, which results in model one. A check for highly correlated variables are included in Appendix B. This resulted in a high correlation between minutes called and charge. Therefore, these two variables are not added to the estimation at the same time in all the models.

The Akaike information criterion (AIC), which indicates a goodness of fit of the logistic regression meanwhile penalizing for adding variables, has a value of 1628. While the Bayesian information criteria (BIC) which accounts for the sample size, resulted in a value of 1713. Model one was able to predict 84.65% of all the customers correctly (Akaike, 1974).

Table 1, estimation of the logistic regression

	Model 1			**Model 2**			**Model 3**		
	B	σ		B	σ		B	σ	
(Intercept)	-8.68	.83	***	-8.48	.82	***	-7.40	0.55	***
Account Length	.0014	.00		-			-	-	
Total Charge	18.41	11.12		.33	.10	**	.36	0.09	***
Total Calls	-.08	.03	**	-.08	.03	**	-	-	
Total Minutes	-4.89	2.99		-.01	.00	*	-.01	0.01	
International Plan	2.05	.17	***	2.05	.17	***	-	-	
Voicemail Plan	-1.81	.66	**	-.92	.17	***	-1.69	0.65	**
Voicemail Message	.03	.02		-	-		.03	0.02	
Daytime Minutes	1.78	1.11		-.03	.01	*	-.03	0.01	
Daytime Charge	-	-		-	-		-	-	
Daytime Calls	.08	.03	**	.08	.03	**	-	-	
Evening Minutes	3.34	2.06		-.01	.00	*	-.01	0.00	
Evening Charge	-	-		-	-		-	-	
Evening Calls	.08	.03	**	.08	.03	**	-	-	
Night Minutes	4.07	2.49		-	-		-	-	
Night Charge	-	-		-	-		-	-	
Night Calls	.08	.03	**	.08	.03	**	-	-	
Customer Service	.51	.05	***	.51	.04	***	.45	0.04	***
Gini Coefficient	**.66**			**.67**			**.54**		
Top Decile Lift	**3.19**			**3.35**			**3.12**		
Akaike Inf. Cr.	1628			**1627**			1768		
Bayesian Inf. Cr.	1713			**1692**			1832		
Hit score	0.85			0.85			0.85		

Sign. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’

Removing all the insignificant variables resulted in the estimation presented in model two. This estimation has an AIC value of 1627, a lower BIC compared to model one of 1692 and is able to predict 84.89% of all customers correctly.

It is not possible to interpret the coefficients of the logistic regression, in the same way as a linear regression. Only the signs of the logistic regression give a direction to the variable on the probability of a customer churning. The total charge, having an international plan, the number of day, evening and night calls, and the number of calls to customer service have a positive effect on the probability to churn. While the total number of calls, the total number of minutes called, having a voicemail plan, number of daytime, evening minutes reduce the probability of a customer churning.

Marginal effects are calculated to describe the impact of the variables on churn. Marginal effects describe the change in churn probability, as x increases with one unit, holding all other variables in the model constant or at average observation (Torres-Reyna, 2014). Table two shows the marginal effects of model two from the logistic regression. This table shows the same directional effect as described before, however now it is possible to describe the impact of the variables on churn.

Table 2, marginal effects of logistic regression model two.

	B	σ	
Total Charge	1.95	.61	**
Total Calls	-1.42	.51	**
Total Minutes	-.57	.24	*
International Plan	.31	.03	***
Voicemail Plan	-.06	.01	***
Daytime Minutes	-.89	.37	*
Daytime Calls	1.04	.38	**
Evening Minutes	-.27	.13	*
Evening Calls	1.13	.39	**
Night Calls	.91	.33	**
Customer Service Calls	.37	.03	***

Sign. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

For example, as the total charge of a customer increases with one dollar, the probability of a customer churning increases with 1.95 percent. Moreover, when a customer calls more in total, the probability of churning decreases with 1.42 percent.

Model three in table one is a result of the fit on the important variables from the decision tree, see table 1. Compared to the other models, the AIC and BIC are significantly higher. Also, this model was less good in identifying top churners and the performance of the overall model was weaker. Surprisingly, the hit rate of the model was comparable to the other models. Therefore, the estimation of model two is used to compare the logistic regression with the decision tree, bagging, boosting and a neural network.

6.3 Bagging and boosting

Bagging creates multiple decision trees over subsets of the telecom dataset and aggregates the outcomes of these trees to estimate the probability of churning (Breiman, 1996). All the variables that are in the dataset are used to estimate the churn probability. With these variables, the model was able to predict the customer status 94.84% of all customers correctly. The estimation method resulted in a comparable top decile lift of 6.47. However, the overall performance of the bagging method is higher than the decision tree and logistic regression, as the Gini coefficient is .74. The relative variable importance in relative percentages is visible in figure 4. It shows that, as in the decision tree, the total charge, voicemail plan and customer service calls are very important variable to estimate customer churn.

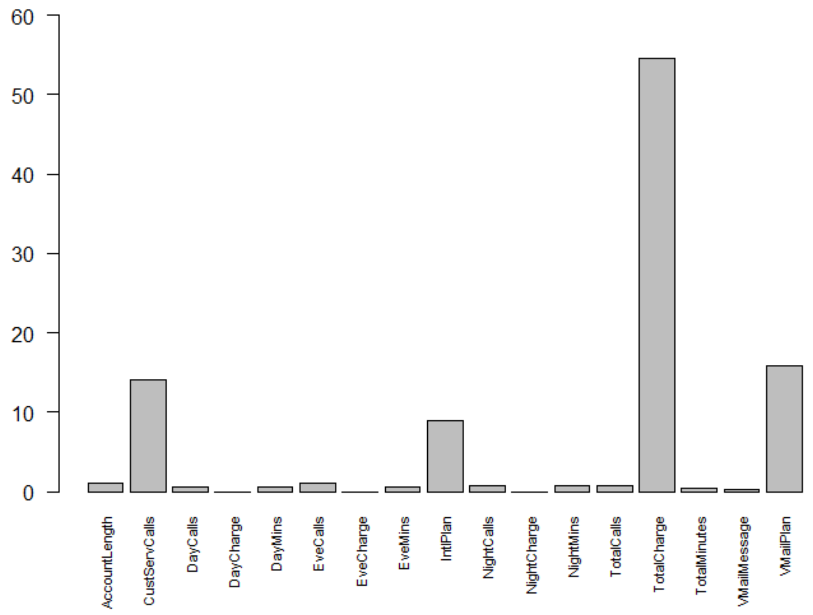

Figure 4, the relative importance of variables for bagging in percentage.

Using the boosting estimation method, 94.96% of all customer statuses are predicted correctly. The ability to separate the high probability churners from the other customers of the model is equal to the bagging model. However, the overall performance of the model is better compared to all previous described estimation methods (Gini coefficient is .77). The variable importance of the boosting estimation is presented in figure 5. Where it also shows that the total amount of money charged was most important. However less important than in the bagging estimation. Here seems to be more of a distribution among the importance of the variables.

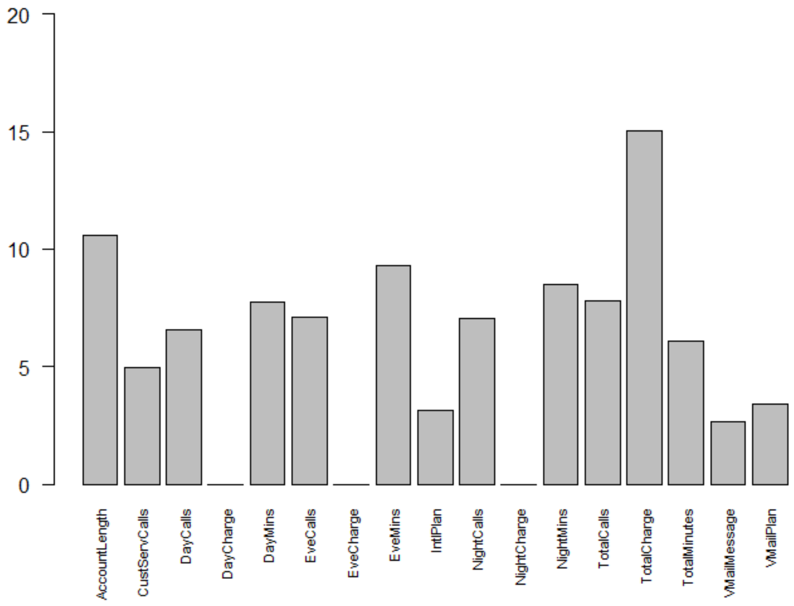

Figure 5, the relative importance of variables for boosting in percentage.

6.4 Neural network

The neural network is estimated on data scaled by a min-max scale to a fixed range from zero to one. This improves the predictive performance and time of the neural network to estimate (Sola & Sevilla, 1997). In figure 6, the variables are displayed as the input of the neural network, by one hidden layer, 5 neurons and biases (B1 and B2) the values of the inputs are transformed into an output in the form of probability of a customer to churn.

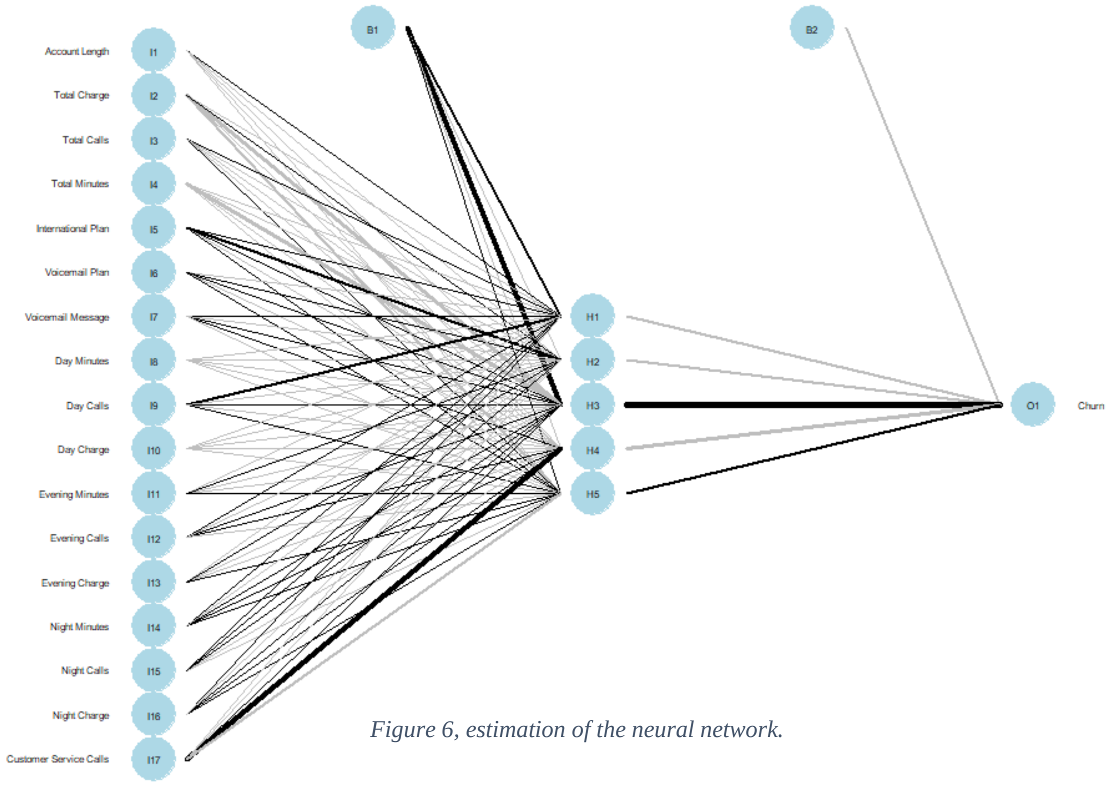

Figure 6, estimation of the neural network.

All variables are included in the estimation of the neural network. By estimating the model, the model adjusts the neurons and biases until the squared error between the estimated churn probability and actual churn value 0 or 1 (Mitchell, 1997: Bishop, 2006). On the test data, the estimation resulted in a top decile lift of 5.69, a Gini coefficient of 0.77 and a hit score of 92%.

7. Conclusion and discussion

7.1 Conclusion

Churn prediction is very important for enterprises in a competitive market to retain valuable customers, as the telecom sector. Therefore, to build an effective churn prediction model, which has a certain level of predictive power is relevant (Tsai & Lu, 2009).

Au et al. (2003) proposed that neural network would overperform decision trees in predicting customer churn. Mozer et al. (2000) presented that neural networks outperform a logistic regression and a decision tree. Ha, Cho & Maclachlan (2005) proposed that also bagging and boosting would be outperformed by a neural network. Therefore, the need to compare modern machine learning techniques in terms of predictive power is evident. As displayed in figure 7, the neural network did perform in terms of overall performance (Gini coefficient).

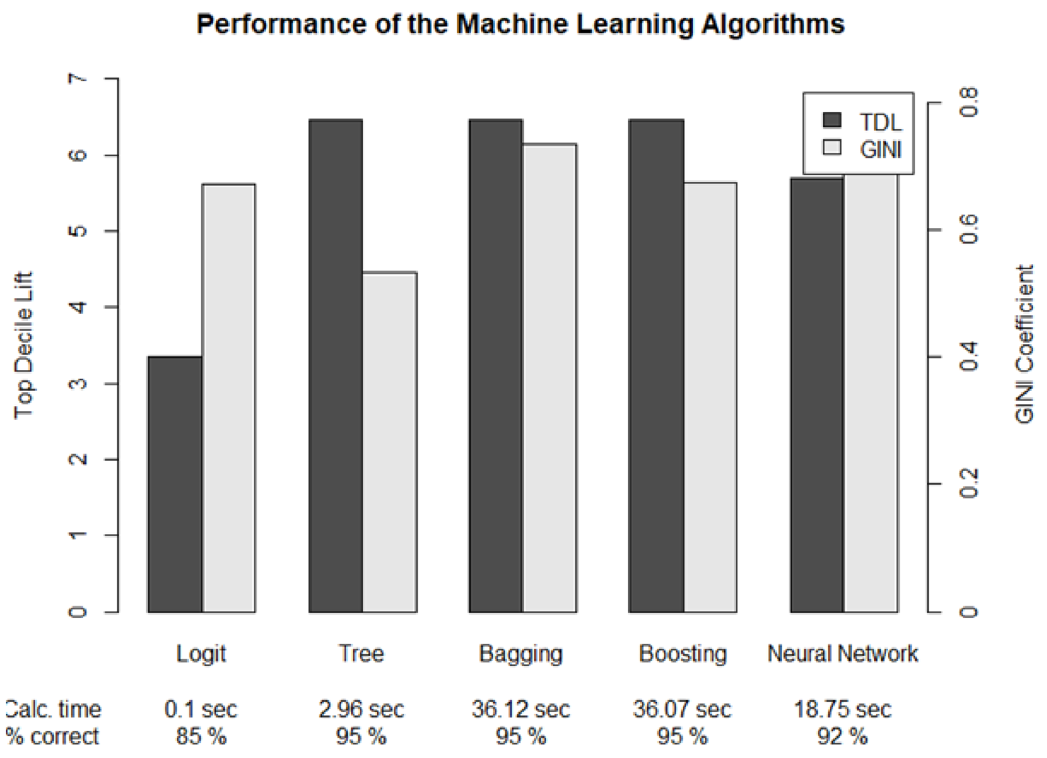

Figure 7, overall performance of the machine learning algorithms.

However, the neural network estimation did not overperform the other estimations in terms of separating the top churners from other customers (Top Decile Lift), the amount of correct predicted churn statuses and overall time to train the model.

The most competitive estimation methods are the bagging and boosting methods. These methods outperform the other estimations, except for the neural network in terms of the Gini coefficient. However, the top decile lift is more important for managers than the Gini coefficient and the hit rate, because a manager wants to find a model that finds those people that are the most likely to churn. These customers that are most likely to churn are targeted by the managers to convince them of becoming a loyal customer. Considering the time it takes to create a bagging or boosting model, the decision tree is competitive in terms of top decile lift. However, choosing a decision tree might imply that the overall prediction is weaker compared to boosting and bagging.

Moreover, hit rate appeared to be a weak measurement of the predictive power of the logistic regression. Where in model three, the hit rate remained equal to the other estimations, the Gini coefficient and top decile lift decreased.

7.2 Managerial implications

Out of all the methods evaluated in this paper in the context of estimating churn, bagging and boosting are most advisable for managers to estimate churn. Especially due to the high ability to discover the customers that are most likely to churn (TDL).

There are numerous methods available to estimate a binary decision (i.e., churn). Out of these methods, managers have to evaluate each method in terms of multiple performance measures. Just evaluating the estimations in terms of hit rate is not sufficient. A variety of performance measurements should also be included in the analysis of which estimation method has the highest predictive power.

Managers should profit from the understandability of decision trees. They are easy to interpret or evaluate why customers churn, due to its structure. Neural networks are harder to interpret as it is a collection of adjustments by nodes and biases that lead to a probability.

7.3 Limitations

First, this paper did not focus on the staying power of predictive models. This means that the model estimates are not evaluated over a certain time period. This implies that the predictive power of boosting and bagging may decline over time. Neslin, et al. (2006)

describes that model last at least three months. In that time practitioners do not need to develop a new model. However, future research should test for longer time periods.

Second, the results of the predictive power of the estimations may be dependent on the data provided. Trees tend to perform better on larger data sets (Perlich, Provost, and Simonoff 2004).

Third, the dataset on churn is limited, in this case, the data was from the telecom industry. However, future research should also investigate other variables, data and industries. Also, new estimation methods should be evaluated and be compared in terms of its predictive power.

Finally, there might be interaction effects among the variables used, which have not been discovered. For example, the account length might moderate the minutes called of customers. Future research could look for interaction effects, besides looking at the main effects.

References

Ahn, J.-H., Han, S.-P., & Lee, Y.-S. 2006. Customer churn analysis: Churn determinants and mediation effects of partian defection in the Korean mobile telecommunications service industry. ***Telecommunications Policy*** (30), 552-568.

Akaike, H. 1974. A new look at the statistical model identification. **IEEE Transactions on Automatic Control**, 19(6): 716–723.

Au, W. H., Chan, K., & Yao, X., 2003. A novel evolutionary data mining algorithm with applications to churn prediction. ***IEEE Transactions on Evolutionary Computation***, 7(6), 532–545.

Bijmolt, T. H. A., Leeflang, P. S. H., Block, F., Eisenbeiss, M., Hardie, B. G. S., et al. 2010. Analytics for Customer Engagement. ***Journal of Service Research***, 13(3): 341–356.

Bishop, C. M. 2006. ***Pattern recognition and machine learning***. New York.: Springer.

Breiman, L. 1996. Bagging predictors. ***Machine Learning***, 24(2): 123–140.

Günther, C.-C., Tvete, I. F., Aas, K., Sandnes, G. I., & Borgan, Ø. 2011. Modelling and predicting customer churn from an insurance company. ***Scandinavian Actuarial Journal***, 2014(1): 58–71.

Ha, K., Cho, S., & Maclachlan, D. 2005. Response models based on bagging neural networks. ***Journal of Interactive Marketing***, 19(1): 17–30.

Holtrop, N., Wieringa, J. E., Gijsenberg, M. J., & Verhoef, P. C. 2017. No future without the past? Predicting churn in the face of customer privacy. ***International Journal of Research in Marketing***, 34(1): 154–172.

Hssina, B., Merbouha, A., Ezzikouri, H., & Erritali, M. 2014. A comparative study of decision tree ID3 and C4.5. ***International Journal of Advanced Computer Science and Applications***, 4(2). http://doi.org/10.14569/specialissue.2014.040203.

Kim, H.-S., & Yoon, C.-H. 2004. Determinants of subscriber churn and customer loyalty in the Korean mobile telephony market. ***Telecommunications Policy***, 28(9-10): 751–765.

Lecun, Y., Bengio, Y., & Hinton, G. 2015. Deep learning. ***Nature***, 521(7553): 436–444.

Lu, N., Lin, H., Lu, J., & Zhang, G. 2014. A Customer Churn Prediction Model in Telecom Industry Using Boosting. ***IEEE Transactions on Industrial Informatics***, 10(2): 1659–1665.

Malhotra, N. K. 2010. ***Marketing research: an applied orientation***. New York: Pearson.

Mitchell, T. 1997. ***Machine learning.*** New York: McGraw-Hill.

Mozer, M.C., Wolniewicz, R, Grimes, D.B., Johnson E., Kaushansky H. 2000. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry, ***IEEE Trans. Neural Netw***. 11(3) 690–696.

Neslin, S. A., Gupta, S., Kamakura, W., Lu, J., & Mason, C. H. 2006. Defection Detection: Measuring and Understanding the Predictive Accuracy of Customer Churn Models. ***Journal of Marketing Research***, 43(2): 204–211.

Nie, G., Rowe, W., Zhang, L., Tian, Y., & Shi, Y. 2011. Credit card churn forecasting by logistic regression and decision tree. ***Expert Systems with Applications***, 38(12): 15273–15285.

Perlich, C., Provost, F., & Simonoff, J. S. 2003. Tree Induction vs. Logistic Regression: A Learning-Curve Analysis. **Journal of Machine Learning Research**, 4: 211–255.

Ricci, F., Rokach, L., Shapria, B., & Kantor, P. B. 2010. ***Recommender systems handbook.*** Springer: New York Inc.

Risselada, H., Verhoef, P. C., & Bijmolt, T. H. 2010. Staying Power of Churn Prediction Models. ***Journal of Interactive Marketing***, 24(3): 198–208.

Sola, J., & Sevilla, J. 1997. Importance of input data normalization for the application of neural networks to complex industrial problems. ***IEEE Transactions on Nuclear Science***, 44(3): 1464–1468.

Torres-Reyna, O. 2014, December. Logit, Probit and Multinomial Logit models in R. ***Data & statistical services***. Lecture presented at the Logit, Probit and Multinomial Logit models in R.

Tsai, C.-F., & Lu, Y.-H. 2009. Customer churn prediction by hybrid neural networks. **Expert Systems with Applications**, 36(10): 12547–12553.

Vafeiadis, T., Diamantaras, K. I., Sarigiannidis, G., & Chatzisavvas, K. C. 2015. A comparison of machine learning techniques for customer churn prediction. ***Simulation Modelling Practice and Theory***, 55: 1–9.

Verhoef, P. P. C., & Wieringa, D. J. E. 2011. ***Churn: Welke klanten dreigen weg te lopen?*** https://www.rug.nl/cic/downloads/rugcic_rapport_201101_churn.pdf.

Appendices

Appendix A – R-code

```
#clear workspace

rm(list = ls())

set.seed(100)


## set wd and load data

setwd("C:/Users/Gilia/Documents/")

Churn_data <- read.csv2("Churn.csv",header=TRUE, stringsAsFactors = F)

Churn_data$DayCalls <- as.numeric(Churn_data$DayCalls)

Churn_data$DayMins <- as.numeric(Churn_data$DayMins)

Churn_data$DayCharge <- as.numeric(Churn_data$DayCharge)

Churn_data$EveMins <- as.numeric(Churn_data$EveMins)

Churn_data$EveCalls <- as.numeric(Churn_data$EveCalls)

Churn_data$EveCharge <- as.numeric(Churn_data$EveCharge)

Churn_data$NightMins <- as.numeric(Churn_data$NightMins)

Churn_data$NightCalls <- as.numeric(Churn_data$NightCalls)

Churn_data$NightCharge <- as.numeric(Churn_data$NightCharge)

Churn_data$IntlMins <- as.numeric(Churn_data$IntlMins)

Churn_data$IntlCalls <- as.numeric(Churn_data$IntlCalls)

Churn_data$IntlCharge <- as.numeric(Churn_data$IntlCharge)


## selection of variables

Churn_data$TotalCharge = as.numeric(Churn_data$DayCharge) + as.numeric(Churn_data$EveCharge) +
as.numeric(Churn_data$NightCharge) + as.numeric(Churn_data$IntlCharge) #variable total charge

Churn_data$TotalCalls = as.numeric(Churn_data$DayCalls) + as.numeric(Churn_data$EveCalls) +
as.numeric(Churn_data$NightCalls) + as.numeric(Churn_data$IntlCalls) #variable total calls

Churn_data$TotalMinutes = as.numeric(Churn_data$DayMins) + as.numeric(Churn_data$EveMins) +
as.numeric(Churn_data$NightMins) + as.numeric(Churn_data$IntlMins)

Churn_data$ChurnString[Churn_data$Churn==1]="Churn"

Churn_data$ChurnString[Churn_data$Churn==0]="Stay"

Churn_data$ChurnString <- as.factor(Churn_data$ChurnString)


# Check for multicollinearity

library(corrplot)

plot(Churn_data$TotalCharge, Churn_data$TotalMinutes)

m <- Churn_data
```

```
m$State <- NULL
m$AreaCode <- NULL
m$Phone <- NULL
m$ChurnString <- NULL
m <- cor(m)
corrplot(m, method = "circle")

# Deleting some unrelevant variables
Churn_data$State <- NULL
Churn_data$AreaCode <- NULL
Churn_data$Phone <- NULL
ChurnString <- Churn_data$ChurnString
Churn_data$ChurnString <- NULL
Churn <- Churn_data$Churn
Churn_data$Churn <- NULL

## Standardizing data for the NN
maxs <- apply(Churn_data, 2, max)
mins <- apply(Churn_data, 2, min)
scaled <- as.data.frame(scale(Churn_data, center = mins, scale = maxs - mins))
Churn_data <- cbind(scaled, Churn)
Churn_data <- cbind(Churn_data, ChurnString)

# Split randomly
x <- Churn_data[sample(1:nrow(Churn_data), nrow(Churn_data), replace = F),]
x.train <- x[1:floor(nrow(x)*.75), ]
x.evaluate <- x[(floor(nrow(x)*.75)+1):nrow(x), ]

# Function of a liftplot
makeLiftPlot <- function(Prediction, Evaluate, ModelName){
  iPredictionsSorted <- sort(Prediction,index.return=T,decreasing=T)[2]$ix #extract the index order according to predicted retention
  CustomersSorted <- Evaluate$ChurnString[iPredictionsSorted] #sort the true behavior of customers according to predictions
  SumChurnReal<- sum(Evaluate$ChurnString == "Churn") #total number of real churners in the evaluation set
  CustomerCumulative=seq(nrow(Evaluate))/nrow(Evaluate) #cumulative fraction of customers
  ChurnCumulative=apply(matrix(CustomersSorted=="Churn"),2,cumsum)/SumChurnReal #cumulative fraction of churners

  ProbTD = sum(CustomersSorted[1:floor(nrow(Evaluate)*.1)]=="Churn")/floor(nrow(Evaluate)*.1) #probability of churn in 1st decile

  ProbOverall = SumChurnReal / nrow(Evaluate) #overall churn probability

  TDL = ProbTD / ProbOverall

  GINI = sum((ChurnCumulative-CustomerCumulative)/(t(matrix(1,1,nrow(Evaluate))-CustomerCumulative)),na.rm=T)/nrow(Evaluate)

  plot(CustomerCumulative,ChurnCumulative,type="l",main=paste("Lift curve of", ModelName),xlab="Cumulative fraction of customers (sorted by predicted churn probability)",ylab="Cumulative fraction of churners")

  lines(c(0,1),c(0,1),col="blue",type="l",pch=22, lty=2)

  legend(.66,.2,c("According to model","Random selection"),cex=0.8,  col=c("black","blue"), lty=1:2)

  text(0.15,1,paste("TDL = ",round(TDL,2), "; GINI = ", round(GINI,2) ))

  return(data.frame(TDL,GINI))

}



# Formulas used for estimation

FormulaAll <- as.formula(Churn ~ AccountLength + IntlPlan + VMailPlan + VMailMessage + DayMins + DayCalls + DayCharge + EveMins + EveCalls + EveCharge + NightMins + NightCalls + NightCharge + IntlMins + IntlCalls + IntlCharge + CustServCalls + TotalCharge + TotalCalls + TotalMinutes)

FormulaLogit <- as.formula(Churn ~ TotalCharge + TotalCalls + TotalMinutes + IntlPlan + VMailPlan + DayMins + DayCalls + EveMins + EveCalls + NightCalls + CustServCalls)

FormulaTree <- as.formula(ChurnString ~ AccountLength + TotalCharge + TotalCalls + TotalMinutes + IntlPlan + VMailPlan + VMailMessage + DayMins + DayCalls + DayCharge + EveMins + EveCalls + EveCharge + NightMins + NightCalls + NightCharge + CustServCalls)

FormulaTreetest <- as.formula(Churn ~ TotalCharge + DayMins + TotalMinutes + VMailMessage + VMailPlan + CustServCalls + EveMins)


######### LOGIT

ptm <- proc.time()

Churn_logit <- glm(FormulaLogit , data = x.train, family = "binomial")

summary(Churn_logit)

x.evaluate$predictionlogit <- predict(Churn_logit, newdata=x.evaluate, type = "response")

x.evaluate$predictionlogitclass[x.evaluate$predictionlogit>.5] <- "Churn"

x.evaluate$predictionlogitclass[x.evaluate$predictionlogit<=.5] <- "Stay"

x.evaluate$correctlogit <- x.evaluate$predictionlogitclass == x.evaluate$ChurnString

print(paste0(mean(x.evaluate$correctlogit),"% of predicted classifications correct"))

LogitOutput <- makeLiftPlot(x.evaluate$predictionlogit,x.evaluate,"Logit")
```

```
TimeAux <- proc.time() - ptm
LogitOutput$TimeElapsed <- TimeAux[3]
LogitOutput$PercCorrect <- mean(x.evaluate$correctlogit)*100
rm(TimeAux)
###

## marginal effects
library(mfx)
marginaleffects <- logitmfx(FormulaLogit , data = x.train)

########## TREE
ptm <- proc.time()
library(rpart)                          # Popular decision tree algorithm
library(rattle)                         # Fancy tree plot
library(rpart.plot)                     # Enhanced tree plots
library(RColorBrewer)                   # Color selection for fancy tree plot
#library(party)                         # Alternative decision tree algorithm
#library(partykit)                      # Convert rpart object to BinaryTree
#library(caret)                         # Just a data source for t

tree.2 <- rpart(FormulaTree,x.train)                          # A more reasonable tree
prp(tree.2, type = 4, extra = 8, xflip = T)                   # A fast plot
fancyRpartPlot(tree.2, palettes=c("Reds", "Greys"))           # A fancy plot from rat
#plot(tree.2)
x.evaluate$predictionTree <- predict(tree.2, newdata = x.evaluate, type = "vector")
x.evaluate$predictionTreeString[x.evaluate$predictionTree==1]="Churn"
x.evaluate$predictionTreeString[x.evaluate$predictionTree==2]="Stay"
x.evaluate$predictionTreeString <- as.factor(x.evaluate$predictionTreeString)
x.evaluate$correctTree <- x.evaluate$predictionTreeString == x.evaluate$ChurnString
print(paste(round(mean(x.evaluate$correctTree)*100,2), "% of predicted classifications correct"))
x.evaluate$probabilitiesTree <- predict(tree.2, newdata = x.evaluate, type = "prob")[,1]
TreeOutput <- makeLiftPlot(x.evaluate$probabilitiesTree,x.evaluate,"Tree")
TimeAux <- proc.time() - ptm
TreeOutput$TimeElapsed <- TimeAux[3]
TreeOutput$PercCorrect <- mean(x.evaluate$correctTree)*100
rm(TimeAux)
```

```
#plot the variable importance

variable_importance <- head(tree.2$variable.importance,-2)

barplot(variable_importance, ylim = c(0,250), las = 2, cex.names=0.8, names.arg=c("TotalCharge",
"DayCharge", "DayMins", "TotalMins", "VoiceMess", "VoicePlan", "CustServCalls", "EveCharge",
"EveMins"))


############ Bagging

ptm <- proc.time()

# Create a model using bagging ensemble algorithms

library(adabag)

library(party)

#Model1

x.modelBagging  <- bagging(FormulaTree, data=x.train, control = cforest_unbiased(mtry = 3))

plot(x.modelBagging)

barplot(x.modelBoosting$importance, ylim = c(0,20), las = 2, cex.names=0.7)


# Use the model to predict the evaluation.

x.evaluate$predictionBagging <- predict(x.modelBagging, newdata=x.evaluate)$class

# Calculate the overall accuracy.

x.evaluate$correctBagging <- x.evaluate$predictionBagging == x.evaluate$ChurnString

print(paste(round(mean(x.evaluate$correctBagging)*100,2), "% of predicted classifications correct"))

# Extract the class probabilities.

x.evaluate$probabilitiesBagging <- predict(x.modelBagging,newdata=x.evaluate)$prob[,1]

BaggingOutput <- makeLiftPlot(x.evaluate$probabilitiesBagging,x.evaluate,"Bagging")

TimeAux <- proc.time() - ptm

BaggingOutput$TimeElapsed <- TimeAux[3]

BaggingOutput$PercCorrect <- mean(x.evaluate$correctBagging)*100

rm(TimeAux)


############ Boosting

ptm <- proc.time()

# Create a model using boosting ensemble algorithms

#Model1

x.modelBoosting  <- boosting(FormulaTree, data=x.train, control = cforest_unbiased(mtry = 3))

x.modelBoosting$importance

# Use the model to predict the evaluation.
```

```
x.evaluate$predictionBoosting <- predict(x.modelBoosting, newdata=x.evaluate)$class

# Calculate the overall accuracy.

x.evaluate$correctBoosting <- x.evaluate$predictionBoosting == x.evaluate$ChurnString

print(paste(round(mean(x.evaluate$correctBoosting)*100,2), "% of predicted classifications correct"))

# Extract the class probabilities.

x.evaluate$probabilitiesBoosting <- predict(x.modelBoosting,newdata=x.evaluate)$prob[,1]

# Make a lift curve

BoostingOutput <- makeLiftPlot(x.evaluate$probabilitiesBoosting,x.evaluate,"Boosting")

TimeAux <- proc.time() - ptm

BoostingOutput$TimeElapsed <- TimeAux[3]

BoostingOutput$PercCorrect <- mean(x.evaluate$correctBoosting)*100

rm(TimeAux)


########## Neural network

ptm <- proc.time()

library(nnet)

library(caret)

library(e1071)

library(reshape)

x.train$Churn <- as.factor(x.train$Churn)

x.modelNNet <- train(FormulaTree, data=x.train, method='nnet', trControl=trainControl(method='cv'))

x.evaluate$predictionNNet <- predict(x.modelNNet, newdata = x.evaluate, type="raw")

x.evaluate$correctNNet <- x.evaluate$predictionNNet == x.evaluate$ChurnString

print(paste("% of predicted classifications correct", mean(x.evaluate$correctNNet)))

library(devtools)

source_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4
684a5/nnet_plot_update.r')

plot.nnet(x.modelNNet, cex.val = 0.5 ,y.lab = "Churn",  x.lab = c("Account Length", "Total Charge", "Total
Calls", "Total Minutes", "International Plan", "Voicemail Plan", "Voicemail Message", "Day Minutes", "Day
Calls", "Day Charge", "Evening Minutes", "Evening Calls", "Evening Charge", "Night Minutes", "Night Calls",
"Night Charge", "Customer Service Calls"))

x.evaluate$probabilitiesNNet <- predict(x.modelNNet, newdata = x.evaluate, type='prob')[,1]

NNetOutput <- makeLiftPlot(x.evaluate$probabilitiesNNet,x.evaluate,"Neural Network")

TimeAux <- proc.time() - ptm

NNetOutput$TimeElapsed <- TimeAux[3]

NNetOutput$PercCorrect <- mean(x.evaluate$correctNNet)*100

rm(TimeAux)
```

```
#import 'gar.fun' from Github

source_url('https://gist.githubusercontent.com/fawda123/6206737/raw/d6f365c283a8cae23fb20892dc223bc5764
d50c7/gar_fun.r')


# Attemps to improve the neural network

library(neuralnet)

Churn_data$State <- NULL

Churn_data$AreaCode <- NULL

Churn_data$Phone <- NULL

Churn_data$ChurnString <- NULL

Churn <- Churn_data$Churn

Churn_data$Churn <- NULL

## Standardizing data for the NN

maxs <- apply(Churn_data, 2, max)

mins <- apply(Churn_data, 2, min)

scaled <- as.data.frame(scale(Churn_data, center = mins, scale = maxs - mins))

Churn_data <- cbind(scaled, Churn)


x <- scaled[sample(1:nrow(scaled), nrow(scaled), replace = F),]

x.trainscaled <- x[1:floor(nrow(x)*.75), ]

x.evaluatescaled <- x[(floor(nrow(x)*.75)+1):nrow(x), ]

neuralnetwork <- neuralnet(FormulaAll, data = x.trainscaled, hidden = 8, linear.output = F, threshold = .1, err.fct
= 'ce', likelihood = T, stepmax = 1e6)

plot.nnet(neuralnetwork)

Churn <- x.evaluatescaled$Churn

x.evaluatescaled$Churn <- NULL

neuralnetwork$model.list

names(x.evaluatescaled)

probabilitiesNN2 <- neuralnet::compute(neuralnetwork, x.evaluatescaled)

results <- as.data.frame(probabilitiesNN2$net.result)

# Examine accuracy of test set

resultsrounded <- round(results)

results$correctNNet <- Churn == resultsrounded

mean(results$correctNNet)

x.evaluatescaled <- cbind(x.evaluatescaled, Churn)

probabilitiesNN2 <- as.data.frame(probabilitiesNN2)

NNOutput <- makeLiftPlot(probabilitiesNN2, Churn,"Neural network with 8 layers")
```

```
# SOME Summarizing plots:

barplot(c(LogitOutput$TDL,TreeOutput$TDL,BaggingOutput$TDL,BoostingOutput$TDL,NNetOutput$TDL),
names.arg = c("Logit","Tree","Bagging","Boosting","Neural Network"), main="Top Decile Lifts of the models")

barplot(c(LogitOutput$GINI,
TreeOutput$GINI,BaggingOutput$GINI,BoostingOutput$GINI,NNetOutput$GINI), names.arg =
c("Logit","Tree","Bagging","Boosting","Neural Network"), main="GINI coefficients of the models")

OverallTDL <-
c(LogitOutput$TDL,TreeOutput$TDL,BaggingOutput$TDL,BoostingOutput$TDL,NNetOutput$TDL)

OverallGINI <-
c(LogitOutput$GINI,TreeOutput$GINI,BaggingOutput$GINI,BoostingOutput$GINI,NNetOutput$GINI)

ForGraph <- data.frame(OverallTDL,OverallGINI)

myLeftAxisLabs <- pretty(seq(0, max(ForGraph$OverallTDL), length.out = 10))

myRightAxisLabs <- pretty(seq(0, max(ForGraph$OverallGINI), length.out = 10))

myLeftAxisAt <- myLeftAxisLabs/max(ForGraph$OverallTDL)

myRightAxisAt <- myRightAxisLabs/max(ForGraph$OverallGINI)

ForGraph$OverallTDL1 <- ForGraph$OverallTDL/max(ForGraph$OverallTDL)

ForGraph$OverallGINI1 <- ForGraph$OverallGINI/max(ForGraph$OverallGINI)

op <- par(mar = c(5,4,4,4) + 0.1)

barplot(t(as.matrix(ForGraph[, c("OverallTDL1", "OverallGINI1")])), beside = TRUE, yaxt = "n", names.arg =
c("Logit","Tree","Bagging","Boosting","Neural Network"), ylim=c(0, max(c(myLeftAxisAt, myRightAxisAt))),
ylab =    "Top Decile Lift", legend = c("TDL","GINI"), main="Performance of the Machine Learning
Algorithms")

axis(2, at = myLeftAxisAt, labels = myLeftAxisLabs)

axis(4, at = myRightAxisAt, labels = myRightAxisLabs)

mtext("GINI Coefficient", side = 4, line = 3, cex = par("cex.lab"))

mtext(c(paste(round(LogitOutput$TimeElapsed,digits=2),"sec"),

        paste(round(TreeOutput$TimeElapsed,digits=2),"sec"),

        paste(round(BaggingOutput$TimeElapsed,digits=2),"sec"),

        paste(round(BoostingOutput$TimeElapsed,digits=2),"sec"),

        paste(round(NNetOutput$TimeElapsed,digits=2),"sec")), side = 1, line = 3, cex = par("cex.lab"), at =
c(2,5,8,11,14,17,20,23,26))

mtext(c(paste(round(LogitOutput$PercCorrect,digits=0),"%"),

        paste(round(TreeOutput$PercCorrect,digits=0),"%"),

        paste(round(BaggingOutput$PercCorrect,digits=0),"%"),

        paste(round(BoostingOutput$PercCorrect,digits=0),"%"),

        paste(round(NNetOutput$PercCorrect,digits=0),"%")), side = 1, line = 4, cex = par("cex.lab"), at =
c(2,5,8,11,14,17,20,23,26))

mtext("Calc. time", side = 1, line = 3, cex = par("cex.lab"), at = -.8)

mtext("% correct", side = 1, line = 4, cex = par("cex.lab"), at = -.8)
```

Appendix B – Check for highly correlated variables

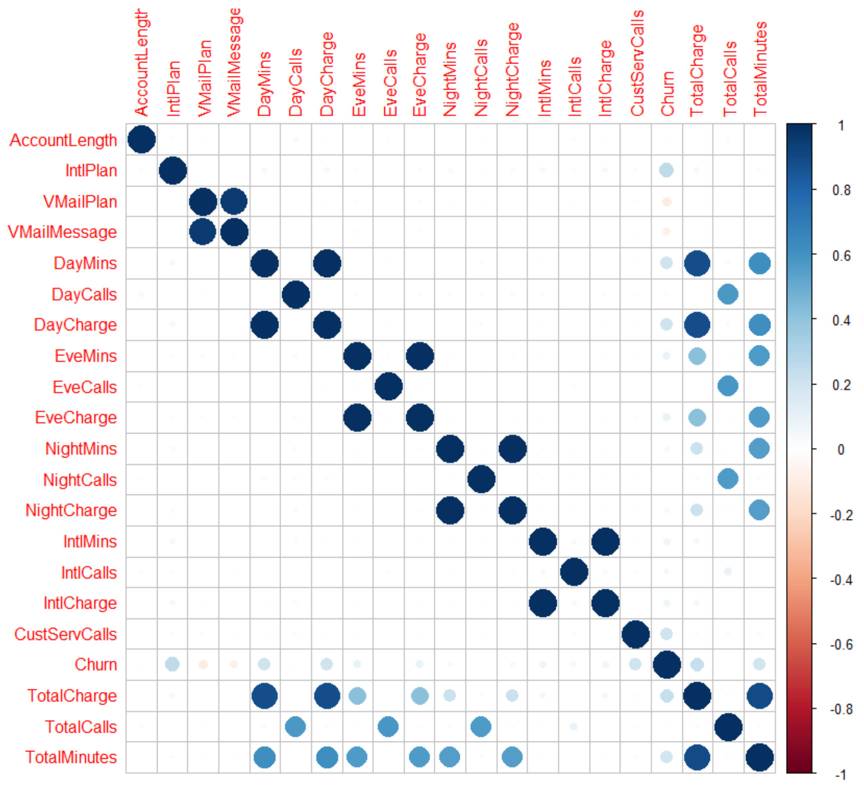

Figure 8, correlation plot.