West Texas A&M University
Paul & Virginia Engler College of Business

Data Mining Final Project
Credit Card Churn Analysis

CIDM-6355-70: Data Mining Methods
Dr. Liang Chen

Gold Miners (Team A)
Don Warner
Dewayne Defoor
Ben Collier
Lauren Taylor
Bailee Gerlich
Jason Hardy

11/21/2023

Executive Summary

Customer retention is essential for credit card companies. Customer churn is the loss of a credit card customer through cancelation, non-renewal, or simply extended periods of non-use. Understanding the factors that contribute to customer churn can be critical in predicting and preventing churn. Steps such as customized incentives, personalized services, and improved communication, can be implemented to aid customer retention while simultaneously reducing the cost of bringing in new clients. By identifying the higher-risk customers, banks and credit card companies can proactively employ mitigation measures to improve customer retention and reduce customer churn.

Objective

The aim of this report is to examine past customer data, transactional patterns, and other attributes that could be indicative of churn and utilize data mining techniques, specifically classification methods, to predict credit card churn. Using decision tree modeling, logistic regression, Naïve Bayes, and neural network classification methods will allow us to find trends and relationships between attributes associated with existing customers and those who have previously churned, thus providing valuable insights into who and why credit cardholders would discontinue their services.

Dataset

The dataset used in this project created by (Chauhan, 2022) with 10,128 records and 18 attributes. The attributes provide a portfolio of the clientele with information such as age, gender, dependents, education level, marital status, income, and much more.

Data Mining Methodology

To address the business problem of predicting credit cardholder attrition and retention, we use four distinct data mining classification methods, namely Naive Bayes, Neural Networks, Logistic Regression, and Decision Tree Modeling. With both R and RapidMiner performing the analysis, the process begins with cleaning and sectioning our dataset into training and testing subsets. Next, each classification method is executed to generate multiple predictive models. These models are then evaluated using key performance measures, such as accuracy, precision, and recall. This evaluation allows us to assess the models' predictive capabilities and identify the most suitable method for this specific problem. By employing various methods and evaluating each, we aim to deploy a comprehensive and informed decision-making tool to proactively manage customer churn and retention strategies.

Key Findings

Recommendations

Conclusion

Introduction

In the dynamic landscape of the banking industry, the issue of customer attrition, particularly in credit card services, has become a growing concern for financial institutions. A manager at a reputable bank is disturbed by the increasing trend of customer churn within credit card services. For financial institutions, consistent loss of clientele poses a threat to its market share, and signals weakness in customer engagement strategies. This has instigated proactive measures to predict and address customer churn before it occurs.

The fundamental business problem revolves around predicting customer churn. The objective is to utilize predictive analytics to anticipate which customers will leave the bank's credit card services. By identifying the potential churners, the bank aims to intervene and create new strategies to prevent further customer churn. The stakeholders in this scenario include multiple sectors of the bank ranging from high-level management to marketing departments and operational teams. The other stakeholder is the customer itself. The bank aims to ensure customer satisfaction and loyalty by addressing their concerns before they contemplate leaving (Brownson, Pokora, & Frankel, 2023).

This initiative is motivated by two objectives. First, it focuses on maintaining the bank's competitive position and growth by retaining its customer base. Second, it wants to pursue better customer experience through services that are tailored to its customer's needs. This helps foster a more robust and loyal clientele.

The dataset was found on Kaggle.com, but it was originally obtained from Analyticca.com and worked by (Chauhan, 2022). This dataset contains information on 10,128 customers, which includes various attributes such as salary, marital status, age, and more crucial parameters. The key is to use data insights to deal with customers leaving. To achieve this, we need to use machine learning and data analysis to uncover patterns of customer behavior that indicate they might leave. We will use models such as decision trees, logistic regression, naïve bays, and neural networks to give us useful and accurate results to predict customer churn (Nie, Rowe, Zhang, Tian, & Shi, 2011).

Data Mining Techniques and Methodology

Data mining, the ongoing process of examining large amounts of data to discover meaningful patterns and rules, can be categorized by various tasks. In general, these common data mining tasks include profiling, co-occurrence grouping, clustering, data reduction, link prediction, similarity matching, casual modeling, regression, and classification. To present a comprehensive analysis of credit cardholder churn and retention prediction, classification is the main task that was used in this report, with the classification techniques being logistic regression, naïve Bayes, neural network, and decision tree modeling.

Classification Methods

Decision Tree Modeling. As an easily interpretable and fast method, decision tree modeling is an intuitive method that creates a hierarchical structure of decisions to discover nonlinear relationships.

Logistic Regression. With the use of binary dependent variables and probabilities, logistic regression offers a simplified and interpretable analysis for the classification of datasets.

Naïve Bayes. Similar to logistic regression, Naive Bayes is a statistical and probabilistic classification method. Rooted in the Bayesian Theorem, Naïve Bayes classification is fast and easy to implement because of its base assumption that features are conditionally independent.

Neural Networks. While capable of being used for both clustering and classification, neural networks when utilized for classification can handle large, unstructured data to make complex predictions.

Data Description

The dataset used for this problem was provided by (Chauhan, 2022) who created the data from an online database from Analtyica.com, in November of 2022. The data does not reveal any history such as how long they have been collecting the data.

The sample size consists of 10,128 customers. Each row represents a customer that is either existing or churned. The columns indicate the attributes in the dataset. There are over 18 attributes that describe the features of each customer, including age, gender, marital status income salary, baking activity, and more.

The following attributes are described as so:

- Clientnum: This is a unique identifier for each customer, holding an account. This is a polynomial datatype.
- Attrition_flag: An internal event that signifies if the account is closed or not. The datatype for this attribute is integer.
- Age: Demographic variable that resents customers age in years. The datatype is an integer.
- Dependent_count: Demographic variable signifying number of dependents. The datatype for this attribute is integer.
- Education_level: Demographic variable that describes the customer's education level such as high school, college, or college graduate. The datatype is polynomial.
- Marital_Status: Demographic variable that shows the customer's marital status. Values include married, single, divorced, and unknown. The datatype is polynomial.
- Income_Category: Demographic variable that puts each customer into an income category. Values include < \$40K, \$40K-\$60K, \$60K-\$80K, \$80K-\$120K, and \$120K+. The datatype is polynomial.
- Card_Category: Product variable that shows the card category of each customer. Card categories include Blue, Silver, Gold, and Platinum. The datatype is polynomial.
- Months_on_book: The number of months that each customer has been with the bank. The datatype is integer.
- Total_Relationship_count: The total number of products held by the customer. The datatype is integer.
- Months_Inactive_12_mon: Number of months inactive within the last 12 months. The datatype is integer.
- Contacts_Count_12_mon: Number of contacts within the last 12 months. The datatype is integer.
- Credit_Limit: Credit limit on each customer's card. Datatype is real.
- Total_Revolving_Bal: Total revolving balance on each customer's card. The datatype is integer.
- Avg_Open_To_Buy: The open to buy credit line average for each customer within the last 12 months. The datatype of this attribute is real.
- Total_Amt_Chng_Q4_Q1: The change in the transaction amount for each customer, quarter 4 over quarter 1. The datatype here is real.
- Total_Trans_Amt: Total transaction amount for each customer over the last 12 months. The datatype is integer.

- Total_Trans_Ct: Total transaction count for each customer over the last 12 months. The datatype is integer.
- Total_Ct_Chng_Q4_Q1: The change in transaction count for each customer, quarter 4 over quarter 1. The datatype of this attribute is real.
- Avg_Utilization_Ratio: Average card utilization ratio for each customer. The datatype for this attribute is real.
- Naïve_Bayes_Classifier_attribution: Naïve bayes value for each customer. Not to be used in our dataset at all.
- Naïve_Bayes_Classifier_attribution: Naïve bayes value for each customer. Not to be used in our dataset at all.

The dataset contains a variety of demographic, financial, and tractional attributes related to bank customers. These attributes are crucial for understanding customer behavior and predicting churn.

Data Preparation

When evaluating the data, some determinations were made about the particulars that should be examined and what would make them invalid. Specifically, we looked at the question of any unidentified data and whether we should include those records in our evaluation. We found "Unknown" in three attributes: Income_ Category, Marital_Status, and Education_Level.

After identifying these attributes, we evaluated the likely importance of each when considering churn. Income Category was essential that we have correct data, so it was one of the first that we looked at cleaning up.

Income Category: When evaluating the data, we found that there were 1,112 records with an unknown Income Category. Since income category is important to determine credit worthiness, we found it to be odd that there were so many records that were without an Income Category, and we decided that there may be additional quality issues with the records and that we should exclude them.

Marital Status: There were 749 records with an unknown marital status, 82 of which were also included in the income category. Since this data is supposed to represent actual customer information it seems highly suspect that the company would not know the marital status of the customers if they were to extend unsecured credit. Additionally, we thought that marital status would likely be an important factor when finding churn risk. With these factors in mind, we decided that eliminating the data would be necessary.

Education Status: There were 1,519 records with an unknown education status. This attribute was the most difficult to decide upon because we hypothesized that churn would be less likely to be based on education. In the end, we did eliminate the records because it seemed like an odd data point to be missing since education level is a common question when getting credit.

In total, we eliminated 3,046 records out of 10,127 for a total of 7,081 records evaluated. With that in mind, there were 334 records that had multiple unknown values. Ultimately, we decided that it is more important to have a very clean set of data that is the highest quality versus having some data that may cause variances or be of questionable trust levels.

We did some further analysis and ran simulations with the data filtered and unfiltered (see delta tables below) and in most cases we didn't see a large variance. Here is some of the analysis that we did to In the following cases, the negative value represents a higher value is filtered in RapidMiner and we didn't see a delta over 1%. With Logistic Regression, we saw a delta of -0.12% in precision, 0.87% in sensitivity, 0.18% in specificity, and 0.58% in accuracy. With Naïve Bayes, we saw a delta of -0.15% in precision, 0.38% in sensitivity, 0.12% in specificity, and 0.20% in accuracy. With Decision Tree, we saw a delta of -0.33% in precision, -0.08% in sensitivity, -0.88% in specificity, and -0.33% in accuracy.

RapidMiner LR: Delta	**Filtered**			**Unfiltered**		
Matrix		**P**	**N**		**P**	**N**
	P	1735	195	**P**	2494	284
	N	55	139	**N**	56	204
Summary		**Precision**	89.90%		**Precision**	89.78%
		Sensitivity	96.93%		**Sensitivity**	97.80%
		Specificity	41.62%		**Specificity**	41.80%
		Accuracy	88.23%		**Accuracy**	88.81%

RapidMiner NB: Delta	**Filtered**			**Unfiltered**		
Matrix		**P**	**N**		**P**	**N**
	P	1673	164	**P**	2393	239
	N	117	170	**N**	157	249
Summary		**Precision**	91.07%		**Precision**	90.92%
		Sensitivity	93.46%		**Sensitivity**	93.84%
		Specificity	50.90%		**Specificity**	51.02%
		Accuracy	86.77%		**Accuracy**	86.97%

RapidMiner DT: Delta	**Filtered**			**Unfiltered**		
Matrix		**P**	**N**		**P**	**N**
	P	1710	136	**P**	2434	203
	N	80	198	**N**	116	285
Summary		**Precision**	92.63%		**Precision**	92.30%
		Sensitivity	95.53%		**Sensitivity**	95.45%
		Specificity	59.28%		**Specificity**	58.40%
		Accuracy	89.83%		**Accuracy**	89.50%

The delta results were a bit more varied with R. With Logistic Regression, we saw a delta of 0.23% in precision, -0.21% in sensitivity, 1.26% in specificity, and 0.03% in accuracy. With Naïve Bayes, we saw a delta of 0.27% in precision, 0.42% in sensitivity, 1.48% in specificity, and 0.58% in accuracy. With Decision Tree, we saw a delta of 7.90% in precision, -0.90% in sensitivity, 1.87% in specificity, and 1.40% in accuracy. With Decision Tree we believe the larger delta might be due to the positive results being flipped because the negative precision for the unfiltered was 95.18% and 95.28%. The delta for this use case is much more in line with the results we have seen elsewhere at -0.10%.

R LR Delta	**Filtered**			**Unfiltered**		
Matrix		**P**	**N**		**P**	**N**
	P	1555	65	**P**	2217	87
	N	225	280	**N**	327	408
Summary		**Precision**	95.99%		**Precision**	96.22%
		Sensitivity	87.36%		**Sensitivity**	87.15%

	Specificity	81.16%		Specificity	82.42%
	Accuracy	86.35%		Accuracy	86.38%

R NB Delta	Filtered			Unfiltered		
Matrix		P	N		P	N
	P	1657	125	P	2379	172
	N	123	220	N	165	323
Summary		Precision	92.99%		Precision	93.26%
		Sensitivity	93.09%		Sensitivity	93.51%
		Specificity	63.77%		Specificity	65.25%
		Accuracy	88.33%		Accuracy	88.91%

R DT Delta	Filtered			Unfiltered		
Matrix		P	N		P	N
	P	261	85	P	370	74
	N	84	1695	N	125	2470
Summary		Precision	75.43%		Precision	83.33%
		Sensitivity	75.65%		Sensitivity	74.75%
		Specificity	95.22%		Specificity	97.09%
		Accuracy	92.05%		Accuracy	93.45%

The results of the unfiltered data were slightly better in most cases for both RapidMiner and R. We believe that the small delta is a result of the larger sample size versus the unfiltered dataset versus a better quality model, including the unknown values. Because of this, we decided that it was appropriate to go with the filtered results, which we viewed as being more realistic to a real scenario for this scenario.

In terms of highly correlated attributes, RM picked up that the Avg_Open_To_Buy column was collinear and automatically excluded it. This is without using the correlation matrix operator and removing them based on that. Upon doing so with a confidence of 0.8, the Total_Trans_Ct column was also removed along with Avg_Open_To_Buy.

Some additional data preparation that we did include:

- Set the CLIENTNUM column as the ID in RM and the row name in R
- R
    - Used unfactored strings for Linear Regression and Naïve Bays analysis when importing. This was needed because we needed to transform the "Attrition Flag" into a noncategorical value. When we attempted to simply swap the values when they were not factored, we received errors, so we transformed the values from strings to either 0 or 1 (0 for Existing customers and 1 for Attired customers). This allowed us to process the data normally. However, we did run into a problem with the positive case being identified correctly. We believe that this may have been due to the factoring; however, we were unable to get R to swap the test case.
- RM
    - We set the CLIENTNUM column as the ID of the data frame.
    - We used nominal to binomial to convert the "Attrition Flag" column (applicable to all models).

- We used the Nominal to Numeric operator to convert the nominal values to a numerical counterpart for the Linear Regression analysis so it would have the numeric values it needs to make its model.

Modeling

Decision Tree: For RM, the decision tree model worked quite well. Changing the maximal depth to 10 was ideal in this case. Going any higher in depth made the decision tree difficult to interpret, which does make sense due to the number of records in our chosen data set.

(Possible) Exclusion of Neural Net: With not all our data being strictly numerical, neural net would not be applicable, with the data in its current state. Changes could be made to possibly make the data fit the neural net model which has not been completed yet.

Logistic Regression and Naïve Bayes are TBA.

Model Results

As previously mentioned the results here are based on the filtered record set which excluded 3,046 records because of unknown values. Since we are trying to identify instances where the credit card customers are likely to churn a higher level of sensitivity, which (Lantz, 2013) tells us the proportion of those that were classified correctly.

For RapidMiner we see that the sensitivity results are the best performing across all of the models with Logistic Regression performing the best at 96.93% and Naïve Bayes the worst at 93.46%. For R we saw much more varying results between the various models. Sensitivity ranged from 75.65% (Decision Tree) to 93.09% (Naïve Bayes). This means the best model in terms of sensitivity in R was lower than the lowest-performing model in RapidMiner. We did notice that the lowest in RapidMiner was also the highest in R which was interesting.

RapidMiner: Filtered	**Logistic Regression**			**Naïve Bayes**			**Decision Tree**		
Matrix		**P**	**N**		**P**	**N**		**P**	**N**
	P	1735	195	**P**	1673	164	**P**	1710	136
	N	55	139	**N**	117	170	**N**	80	198
Summary		**Precision**	89.90%		**Precision**	91.07%		**Precision**	92.63%
		Sensitivity	96.93%		**Sensitivity**	93.46%		**Sensitivity**	95.53%
		Specificity	41.62%		**Specificity**	50.90%		**Specificity**	59.28%
		Accuracy	88.23%		**Accuracy**	86.77%		**Accuracy**	89.83%

R - Filtered	**Logistic Regression**			**Naïve Bayes**			**Decision Tree**		
Matrix		**P**	**N**		**P**	**N**		**P**	**N**
	P	1555	65	**P**	1657	125	**P**	261	85
	N	225	280	**N**	123	220	**N**	84	1695
Summary		**Precision**	95.99%		**Precision**	92.99%		**Precision**	75.43%
		Sensitivity	87.36%		**Sensitivity**	93.09%		**Sensitivity**	75.65%
		Specificity	81.16%		**Specificity**	63.77%		**Specificity**	95.22%
		Accuracy	86.35%		**Accuracy**	88.33%		**Accuracy**	92.05%

The following tables demonstrate the deltas between the filtered models and the tool used (RapdiMiner or R). We found it surprising how different some of the results were. With Logistic Regression, an enormous change in specificity, which (Lantz, 2013) describes as the proportion of the results that were correctly identified as false. R performed 39.54% better than RapidMiner for Logistic Regression. Decision Tree also had a large change in specificity with 35.94% in favor of R. With Naïve Bayes we still saw a large difference but it was 12.87% better in R. In general, Naïve Bayes seemed to be the closest in terms of results across all of the results delivered. The sensitivity of the two were very close at 0.37%.

Logistic Regression	**Precision**	**Sensitivity**	**Specificity**	**Accuracy**
RM	89.90%	96.93%	41.62%	88.23%
R	95.99%	87.36%	81.16%	86.35%
Delta	-6.09%	9.57%	-39.54%	1.88%

Naïve Bayes	**Precision**	**Sensitivity**	**Specificity**	**Accuracy**

RM	91.07%	93.46%	50.90%	86.77%
R	92.99%	93.09%	63.77%	88.33%
Delta	-1.92%	0.37%	-12.87%	-1.56%

Decision Tree	**Precision**	**Sensitivity**	**Specificity**	**Accuracy**
RM	92.63%	95.53%	59.28%	89.83%
R	75.43%	75.65%	95.22%	92.05%
Delta	17.20%	19.88%	-35.94%	-2.22%

Evaluation

Discussion

- Lauren, Dewayne

References

AL-Najjar, D., Al-Rousan, N., & AL-Najjar, H. (2022). Machine Learning to Develop Credit Card Customer Churn Prediction. *Journal of Theoretical and Applied Electronic Commerce Research, 17*(4), 1529. doi:10.3390/jtaer17040077

Brownson, J., Pokora, B., & Frankel, R. S. (2023, January 24). *What Is Credit Card Churning?* Retrieved from Forbes: https://www.forbes.com/advisor/credit-cards/what-is-credit-card-churning/

Chauhan, A. (2022, November). *Credit Card Customers Prediction*. Retrieved from Kaggle: https://www.kaggle.com/datasets/whenamancodes/credit-card-customers-prediction

Dickinson, A. (2021, October 21). Ask Amy: Credit card accounts churn up concern. Chicago, IL, US. Retrieved from https://shorturl.at/gxFKW

Lantz, B. (2013). *Machine Learning with R.* Packt Publishing, Limited.

Nie, G., Chen, Y., Zhang, L., & Guo, Y. (2010). Credit card customer analysis based on panel data clustering. *Procedia Computer Science, 1*(1), 2489-2497. doi:https://doi.org/10.1016/j.procs.2010.04.281

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. doi:https://doi.org/10.1016/j.eswa.2011.06.028

Rajamohamed, R., & Manokaran, J. (2018, March). Improved credit card churn prediction based on rough clustering and supervised learning techniques. *Cluster Computing, 21*, 65-77. doi:https://doi.org/10.1007/s10586-017-0933-1

Stango, V., & Zinman, J. (2016, April). Borrowing High versus Borrowing Higher: Price Dispersion and Shopping Behavior in the U.S. Credit Card Market. *The Review of Financial Studies, 29*(4), 979-1006. Retrieved from http://www.jstor.org/stable/43866041

Wang, G., Liu, L., Peng, Y., Nie, G., Kou, G., & Shi, Y. (2010). Predicting Credit Card Holder Churn in Banks of China Using Data Mining and MCDM. *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. doi:10.1109/WI-IAT.2010.237

Appendices

Figure 1 - RapidMiner Setup

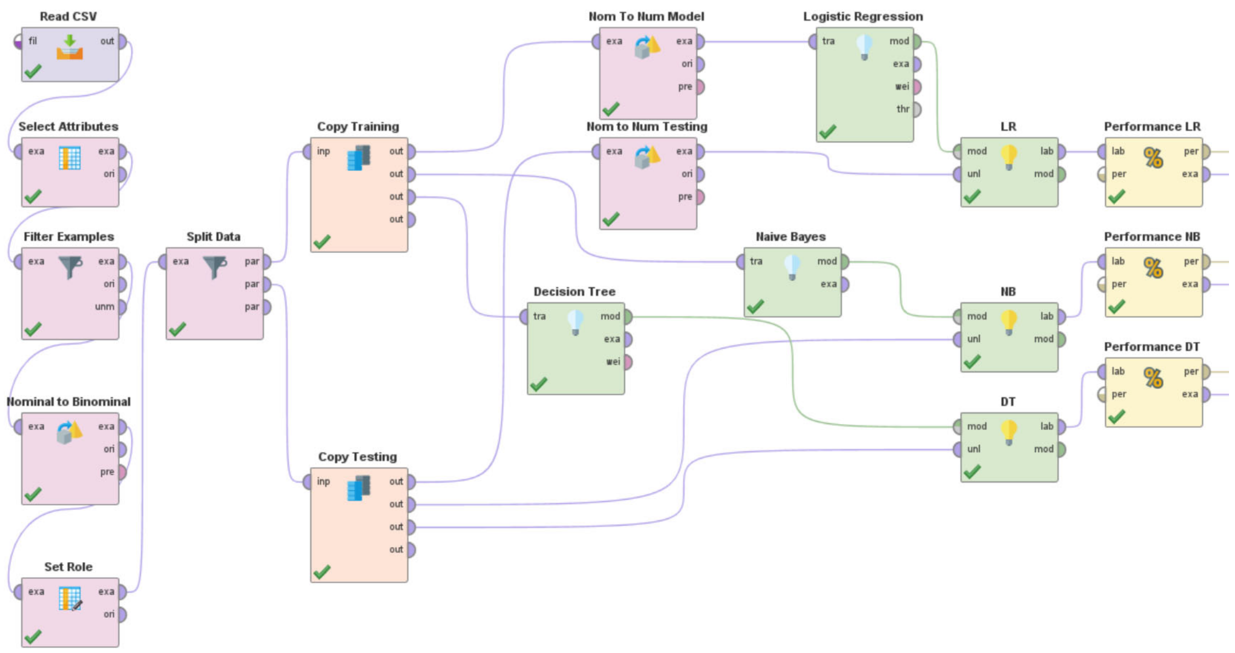
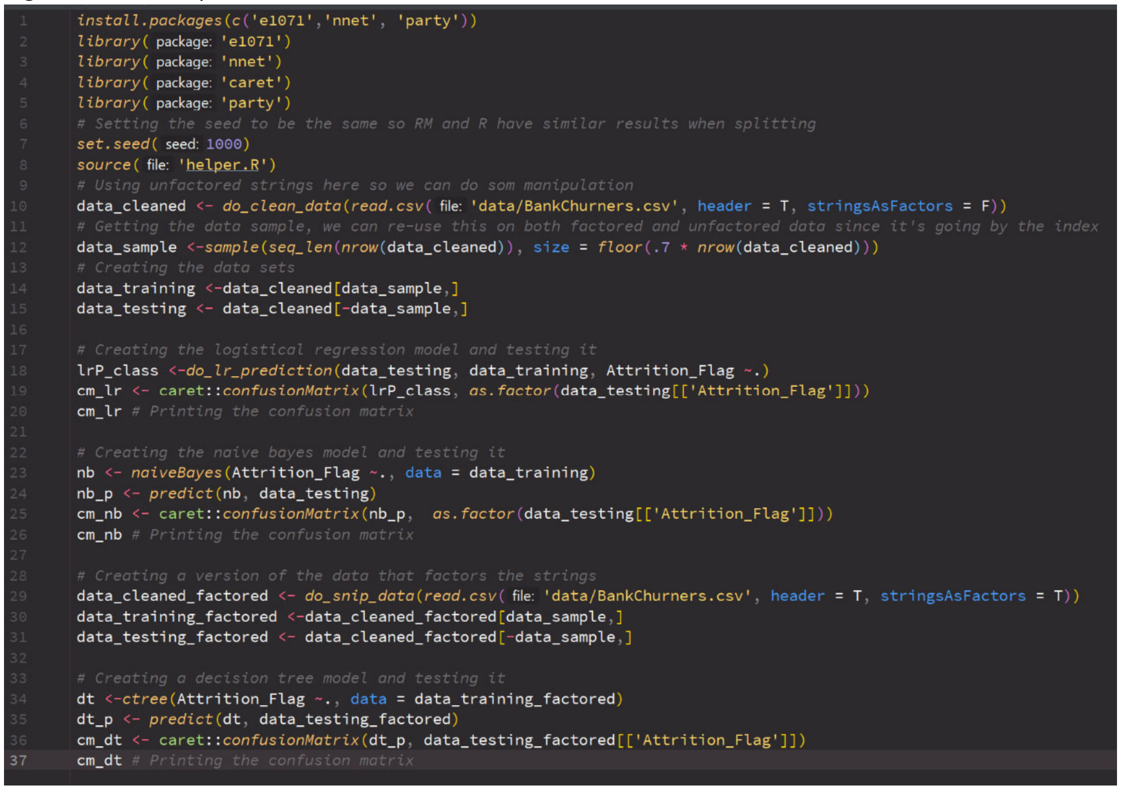
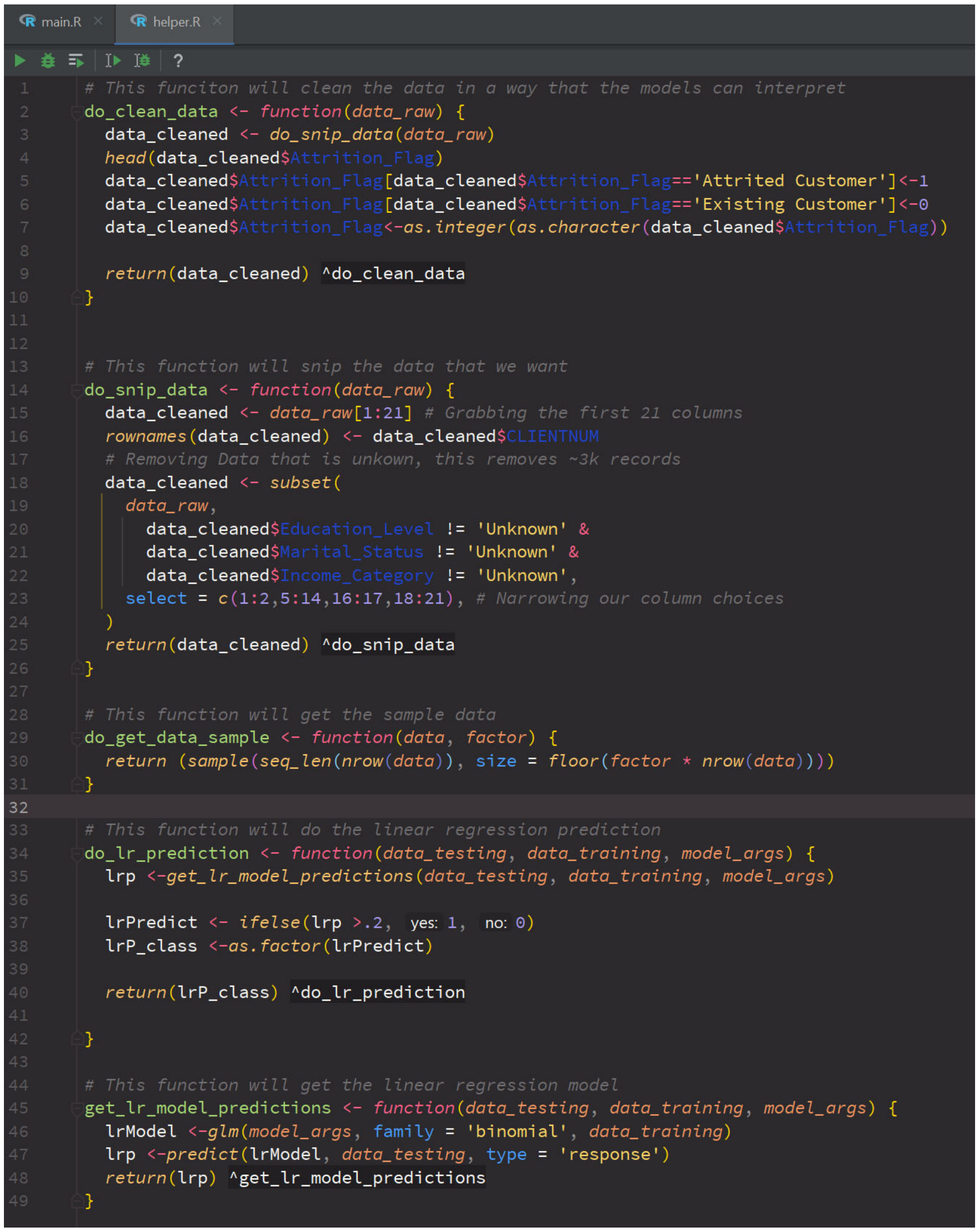


Figure 2 – R Setup



```r
install.packages(c('e1071','nnet', 'party'))
library( package: 'e1071')
library( package: 'nnet')
library( package: 'caret')
library( package: 'party')
# Setting the seed to be the same so RM and R have similar results when splitting
set.seed( seed: 1000)
source( file: 'helper.R')
# Using unfactored strings here so we can do som manipulation
data_cleaned <- do_clean_data(read.csv( file: 'data/BankChurners.csv', header = T, stringsAsFactors = F))
# Getting the data sample, we can re-use this on both factored and unfactored data since it's going by the index
data_sample <-sample(seq_len(nrow(data_cleaned)), size = floor(.7 * nrow(data_cleaned)))
# Creating the data sets
data_training <-data_cleaned[data_sample,]
data_testing <- data_cleaned[-data_sample,]

# Creating the logistical regression model and testing it
lrP_class <-do_lr_prediction(data_testing, data_training, Attrition_Flag ~.)
cm_lr <- caret::confusionMatrix(lrP_class, as.factor(data_testing[['Attrition_Flag']]))
cm_lr # Printing the confusion matrix

# Creating the naive bayes model and testing it
nb <- naiveBayes(Attrition_Flag ~., data = data_training)
nb_p <- predict(nb, data_testing)
cm_nb <- caret::confusionMatrix(nb_p,  as.factor(data_testing[['Attrition_Flag']]))
cm_nb # Printing the confusion matrix

# Creating a version of the data that factors the strings
data_cleaned_factored <- do_snip_data(read.csv( file: 'data/BankChurners.csv', header = T, stringsAsFactors = T))
data_training_factored <-data_cleaned_factored[data_sample,]
data_testing_factored <- data_cleaned_factored[-data_sample,]

# Creating a decision tree model and testing it
dt <-ctree(Attrition_Flag ~., data = data_training_factored)
dt_p <- predict(dt, data_testing_factored)
cm_dt <- caret::confusionMatrix(dt_p, data_testing_factored[['Attrition_Flag']])
cm_dt # Printing the confusion matrix
```

Figure 3 – R Data Manipulation



```r
# This funciton will clean the data in a way that the models can interpret
do_clean_data <- function(data_raw) {
  data_cleaned <- do_snip_data(data_raw)
  head(data_cleaned$Attrition_Flag)
  data_cleaned$Attrition_Flag[data_cleaned$Attrition_Flag=='Attrited Customer']<-1
  data_cleaned$Attrition_Flag[data_cleaned$Attrition_Flag=='Existing Customer']<-0
  data_cleaned$Attrition_Flag<-as.integer(as.character(data_cleaned$Attrition_Flag))

  return(data_cleaned) ^do_clean_data
}


# This function will snip the data that we want
do_snip_data <- function(data_raw) {
  data_cleaned <- data_raw[1:21] # Grabbing the first 21 columns
  rownames(data_cleaned) <- data_cleaned$CLIENTNUM
  # Removing Data that is unkown, this removes ~3k records
  data_cleaned <- subset(
    data_raw,
      data_cleaned$Education_Level != 'Unknown' &
      data_cleaned$Marital_Status != 'Unknown' &
      data_cleaned$Income_Category != 'Unknown',
    select = c(1:2,5:14,16:17,18:21), # Narrowing our column choices
  )
  return(data_cleaned) ^do_snip_data
}

# This function will get the sample data
do_get_data_sample <- function(data, factor) {
  return (sample(seq_len(nrow(data)), size = floor(factor * nrow(data))))
}

# This function will do the linear regression prediction
do_lr_prediction <- function(data_testing, data_training, model_args) {
  lrp <-get_lr_model_predictions(data_testing, data_training, model_args)

  lrPredict <- ifelse(lrp >.2,  yes: 1,  no: 0)
  lrP_class <-as.factor(lrPredict)

  return(lrP_class) ^do_lr_prediction

}

# This function will get the linear regression model
get_lr_model_predictions <- function(data_testing, data_training, model_args) {
  lrModel <-glm(model_args, family = 'binomial', data_training)
  lrp <-predict(lrModel, data_testing, type = 'response')
  return(lrp) ^get_lr_model_predictions
}
```