Telco customer churn

by Ketao Li, Kush Halani, Josue Romain, Juan Peña, Priyanka Patil

Abstract An abstract of less than 150 words.

Background

In an industry as competitive as Telecom, leading companies know that the key to success is not just about acquiring new customers, but rather, retaining existing ones. But how do you know which customers are at risk and why, and which negative experiences and interactions have the biggest impact on churn across touchpoints and channels over time

Objective

The objective of this research is to find a supervised, binary classification model that would provide accurate forecast of telco customer churn.

Data Analysis

The data set we are going to use for our research contains customer's attributes. There are over 7044 records. It has been sourced from Kaggle

Data Dictionary

Column Name	Column Description
customerID	Customer ID
gender	Whether the customer is a male or a female
SeniorCitizen	Whether the customer is a senior citizen or not (1, 0)
Partner	Whether the customer has a partner or not (Yes, No)
Dependents	Whether the customer has dependents or not (Yes, No)
tenure	Number of months the customer has stayed with the company
PhoneService	Whether the customer has a phone service or not (Yes, No)
MultipleLines	Whether the customer has multiple lines or not (Yes, No, No phone service)
InternetService	Customer's internet service provider (DSL, Fiber optic, No)
OnlineSecurity	Whether the customer has online security or not (Yes, No, No internet service)
OnlineBackup	Whether the customer has online backup or not (Yes, No, No internet service)
DeviceProtection	Whether the customer has device protection or not (Yes, No, No internet service)
TechSupport	Whether the customer has tech support or not (Yes, No, No internet service)
StreamingTV	Whether the customer has streaming TV or not (Yes, No, No internet service)
StreamingMovies	Whether the customer has streaming movies or not (Yes, No, No internet service)
Contract	The contract term of the customer (Month-to-month, One year, Two year)
PaperlessBilling	Whether the customer has paperless billing or not (Yes, No)

Column Name	Column Description
PaymentMethod	The customer's payment method (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic))
MonthlyCharges	The amount charged to the customer monthly
TotalCharges	The total amount charged to the customer
Churn	Whether the customer churned or not (Yes or No)

Data Exploration

Let's take a close look at the data set.

```
customerData = read.csv("../data/WA_Fn-UseC_-Telco-Customer-Churn.csv", header = TRUE, na.strings = c("NA","'
```

To have the full picture of the data let's print the data summary and sample.

customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0002-ORFBO: 1	Female:3488	Min. :0.0000	No :3641	No :4933	Min. : 0.00	No : 682	No :3390
0003-MKNFE: 1	Male :3555	1st Qu.:0.0000	Yes:3402	Yes:2110	1st Qu.: 9.00	Yes:6361	No phone service: 682
0004-TLHLJ: 1		Median :0.0000			Median :29.00		Yes :2971
0011-IGKFF: 1		Mean :0.1621			Mean :32.37		
0013-EXCHZ: 1		3rd Qu.:0.0000			3rd Qu.:55.00		
0013-MHZWF: 1		Max. :1.0000			Max. :72.00		
(Other) :7037							

InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	Stream
DSL :2421	No :3498	No :3088	No :3095	No :3473	No :2810	No :27
Fiber optic:3096	No internet service:1526	No internet service:1526	No internet service:1526	No internet service:1526	No internet service:1526	No int
No :1526	Yes :2019	Yes :2429	Yes :2422	Yes :2044	Yes :2707	Yes :27

PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
No :2872	Bank transfer (automatic):1544	Min. : 18.25	Min. : 18.8	No :5174
Yes:4171	Credit card (automatic) :1522	1st Qu.: 35.50	1st Qu.: 401.4	Yes:1869
	Electronic check :2365	Median : 70.35	Median :1397.5	
	Mailed check :1612	Mean : 64.76	Mean :2283.3	
		3rd Qu.: 89.85	3rd Qu.:3794.7	
		Max. :118.75	Max. :8684.8	
			NA's :11	

Table 2: `telco customer churn data Summary`

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtecti
1	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	No
2	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	No	Yes
3	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	No
4	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	No	Yes
5	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	No	No
6	9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic	No	No	Yes
7	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	Fiber optic	No	Yes	No
8	6713-OKOMC	Female	0	No	No	10	No	No phone service	DSL	Yes	No	No
9	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic	No	No	Yes
10	6388-TABGU	Male	0	No	Yes	62	Yes	No	DSL	Yes	Yes	No

TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
No	No	No	One year	No	Mailed check	56.95	1889.50	No
No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes
No	Yes	Yes	Month-to-month	Yes	Electronic check	99.65	820.50	Yes
No	Yes	No	Month-to-month	Yes	Credit card (automatic)	89.10	1949.40	No
No	No	No	Month-to-month	No	Mailed check	29.75	301.90	No
Yes	Yes	Yes	Month-to-month	Yes	Electronic check	104.80	3046.05	Yes
No	No	No	One year	No	Bank transfer (automatic)	56.15	3487.95	No

Table 3: `telco customer churn data`

```
#DATA EXPLORATION

#To see the names of the rows in the dataset
names(customerData)

#>  [1] "customerID"       "gender"           "SeniorCitizen"    "Partner"
#>  [5] "Dependents"       "tenure"           "PhoneService"     "MultipleLines"
#>  [9] "InternetService"  "OnlineSecurity"   "OnlineBackup"     "DeviceProtection"
#> [13] "TechSupport"      "StreamingTV"      "StreamingMovies"  "Contract"
#> [17] "PaperlessBilling" "PaymentMethod"    "MonthlyCharges"   "TotalCharges"
#> [21] "Churn"

#Display the dataset structure and sumary
str(customerData)

#> 'data.frame':    7043 obs. of  21 variables:
#>  $ customerID      : Factor w/ 7043 levels "0002-ORFBO","0003-MKNFE",..: 5376 3963 2565 5536 6512 6552 100
#>  $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 1 2 1 1 2 ...
#>  $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
#>  $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1 ...
#>  $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2 ...
#>  $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
#>  $ PhoneService    : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 2 1 2 2 ...
#>  $ MultipleLines   : Factor w/ 3 levels "No","No phone service",..: 2 1 1 2 1 3 3 2 3 1 ...
#>  $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2 2 2 1 2 1 ...
#>  $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3 3 3 1 1 1 3 1 3 ...
#>  $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1 3 1 1 1 3 1 1 3 ...
#>  $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3 1 3 1 3 1 1 3 1 ...
#>  $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 3 1 1 1 1 3 1 ...
#>  $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 3 1 3 1 ...
#>  $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 1 1 3 1 ...
#>  $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1 1 1 2 ...
#>  $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1 ...
#>  $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3 4 4 1 3 3 2 4 3 1 ...
#>  $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
#>  $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
#>  $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1 ...

#Display first rows of the dataset
head(customerData)

#>   customerID gender SeniorCitizen Partner Dependents tenure PhoneService
#> 1 7590-VHVEG Female             0     Yes         No      1           No
#> 2 5575-GNVDE   Male             0      No         No     34          Yes
#> 3 3668-QPYBK   Male             0      No         No      2          Yes
#> 4 7795-CFOCW   Male             0      No         No     45           No
```

```
#> 5 9237-HQITU Female             0      No         No      2          Yes
#> 6 9305-CDSKC Female             0      No         No      8          Yes
#>      MultipleLines InternetService OnlineSecurity OnlineBackup DeviceProtection
#> 1 No phone service             DSL             No          Yes               No
#> 2               No             DSL            Yes           No              Yes
#> 3               No             DSL            Yes          Yes               No
#> 4 No phone service             DSL            Yes           No              Yes
#> 5               No     Fiber optic             No           No               No
#> 6              Yes     Fiber optic             No           No              Yes
#>   TechSupport StreamingTV StreamingMovies       Contract PaperlessBilling
#> 1          No          No              No Month-to-month              Yes
#> 2          No          No              No       One year               No
#> 3          No          No              No Month-to-month              Yes
#> 4         Yes          No              No       One year               No
#> 5          No          No              No Month-to-month              Yes
#> 6          No         Yes             Yes Month-to-month              Yes
#>                PaymentMethod MonthlyCharges TotalCharges Churn
#> 1           Electronic check          29.85        29.85    No
#> 2               Mailed check          56.95      1889.50    No
#> 3               Mailed check          53.85       108.15   Yes
#> 4 Bank transfer (automatic)          42.30      1840.75    No
#> 5           Electronic check          70.70       151.65   Yes
#> 6           Electronic check          99.65       820.50   Yes
```

```
#To select just the continuous variables and summarise it
library(dplyr)
continues <- select_if(customerData, is.numeric)
#Sumarize the variables to find NA's and outliers
summary(continues)
```

```
#>  SeniorCitizen        tenure      MonthlyCharges    TotalCharges
#>  Min.   :0.0000   Min.   : 0.00   Min.   : 18.25   Min.   :  18.8
#>  1st Qu.:0.0000   1st Qu.: 9.00   1st Qu.: 35.50   1st Qu.: 401.4
#>  Median :0.0000   Median :29.00   Median : 70.35   Median :1397.5
#>  Mean   :0.1621   Mean   :32.37   Mean   : 64.76   Mean   :2283.3
#>  3rd Qu.:0.0000   3rd Qu.:55.00   3rd Qu.: 89.85   3rd Qu.:3794.7
#>  Max.   :1.0000   Max.   :72.00   Max.   :118.75   Max.   :8684.8
#>                                                    NA's   :11
```

```
#Display the factor columns and summarise it
factorColumns <- select_if(customerData, is.factor)
summary(factorColumns)
```

```
#>       customerID      gender     Partner    Dependents PhoneService
#>  0002-ORFBO:   1   Female:3488   No :3641   No :4933   No : 682
#>  0003-MKNFE:   1   Male  :3555   Yes:3402   Yes:2110   Yes:6361
#>  0004-TLHLJ:   1
#>  0011-IGKFF:   1
#>  0013-EXCHZ:   1
#>  0013-MHZWF:   1
#>  (Other)   :7037
#>           MultipleLines     InternetService             OnlineSecurity
#>  No              :3390   DSL        :2421   No                 :3498
#>  No phone service: 682   Fiber optic:3096   No internet service:1526
#>  Yes             :2971   No         :1526   Yes                :2019
#>
#>
#>
#>
#>               OnlineBackup             DeviceProtection
#>  No                 :3088   No                 :3095
#>  No internet service:1526   No internet service:1526
#>  Yes                :2429   Yes                :2422
#>
#>
```

```
#>
#>
#>               TechSupport                StreamingTV
#>  No                 :3473   No                 :2810
#>  No internet service:1526   No internet service:1526
#>  Yes                :2044   Yes                :2707
#>
#>
#>
#>
#>             StreamingMovies           Contract    PaperlessBilling
#>  No                 :2785   Month-to-month:3875   No :2872
#>  No internet service:1526   One year      :1473   Yes:4171
#>  Yes                :2732   Two year      :1695
#>
#>
#>
#>
#>                   PaymentMethod  Churn
#>  Bank transfer (automatic):1544   No :5174
#>  Credit card (automatic)  :1522   Yes:1869
#>  Electronic check         :2365
#>  Mailed check             :1612
#>
#>
#>
```

#Make a chart for each factor type column.

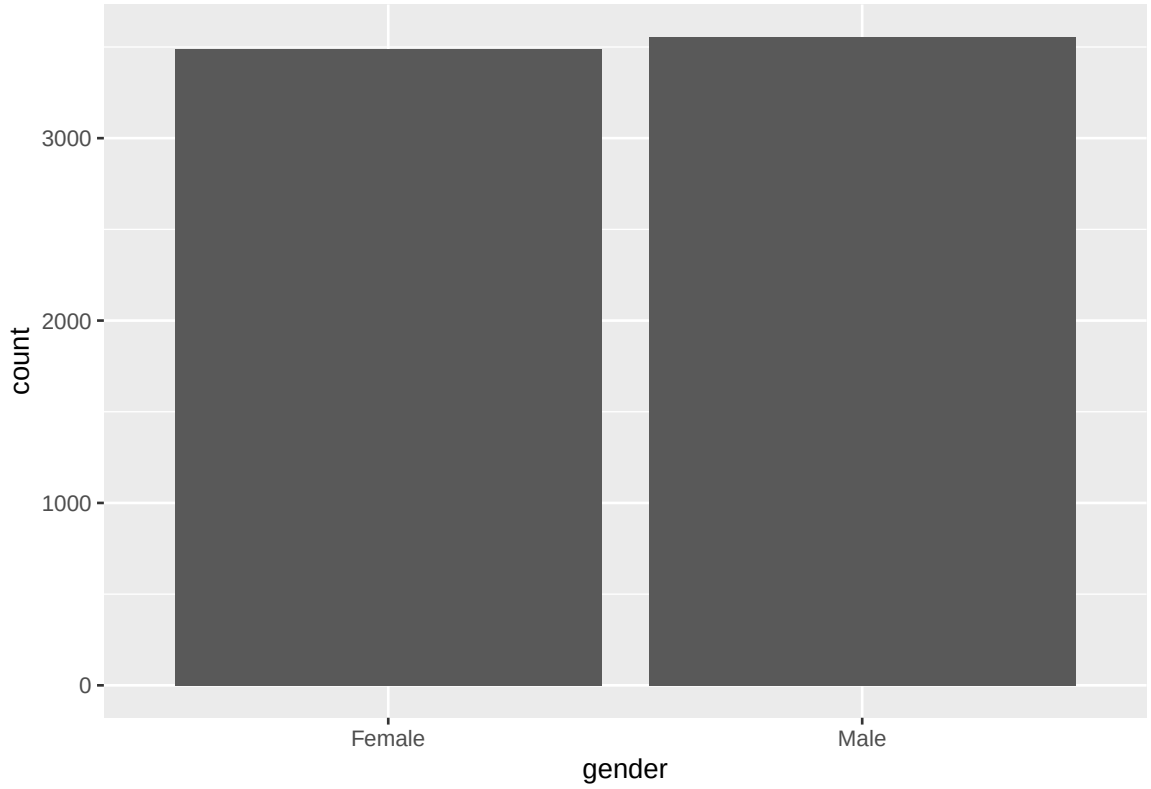

Figure 1: Test1

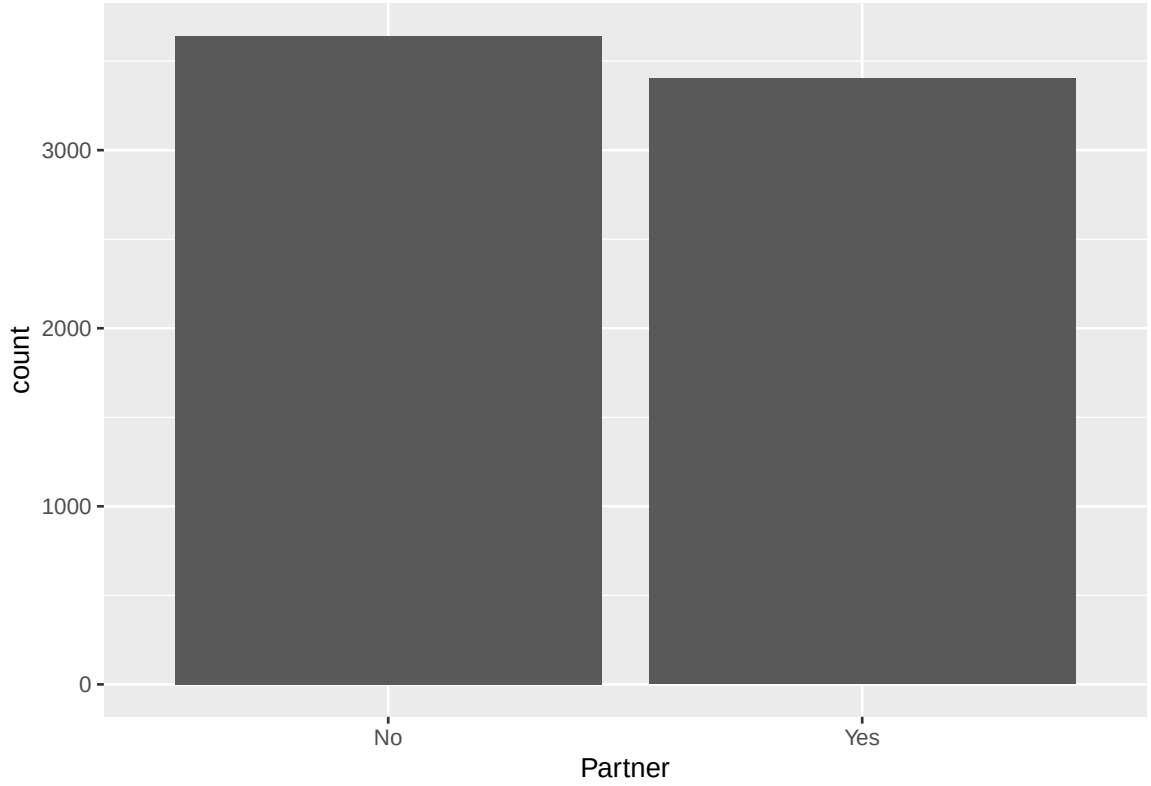

Figure 2: Test2

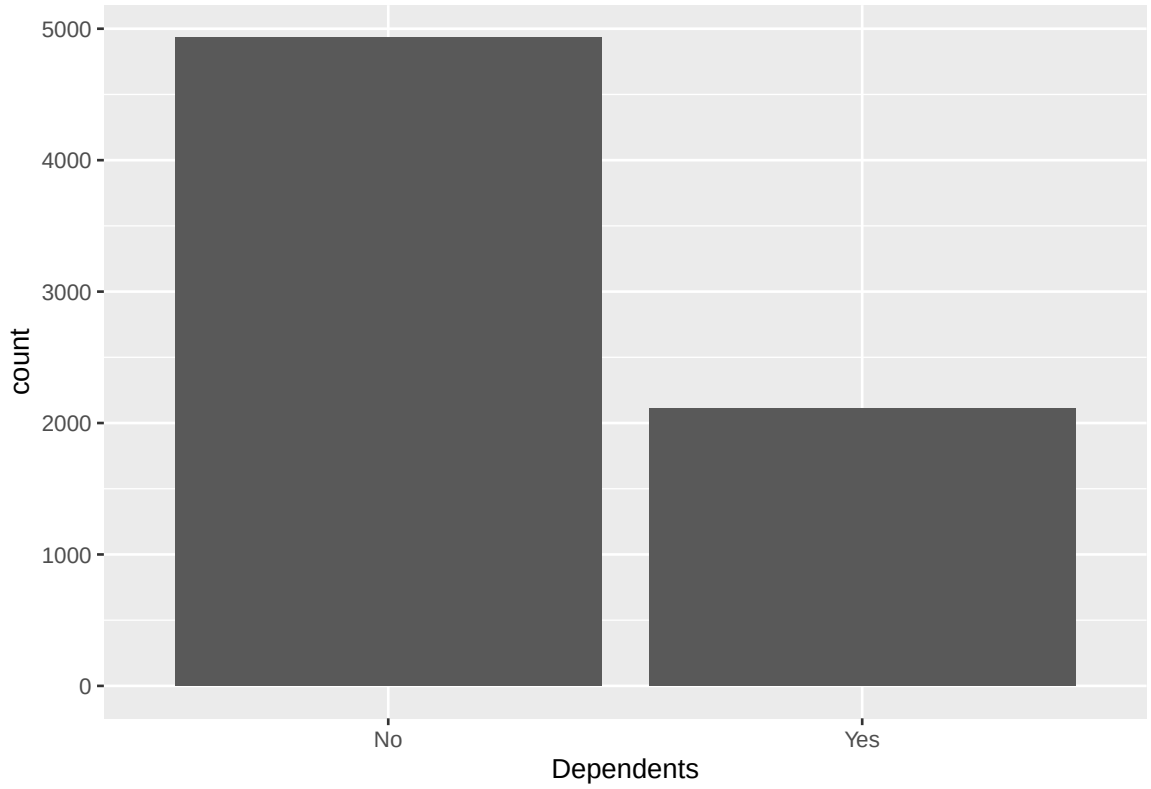

Figure 3: Test3

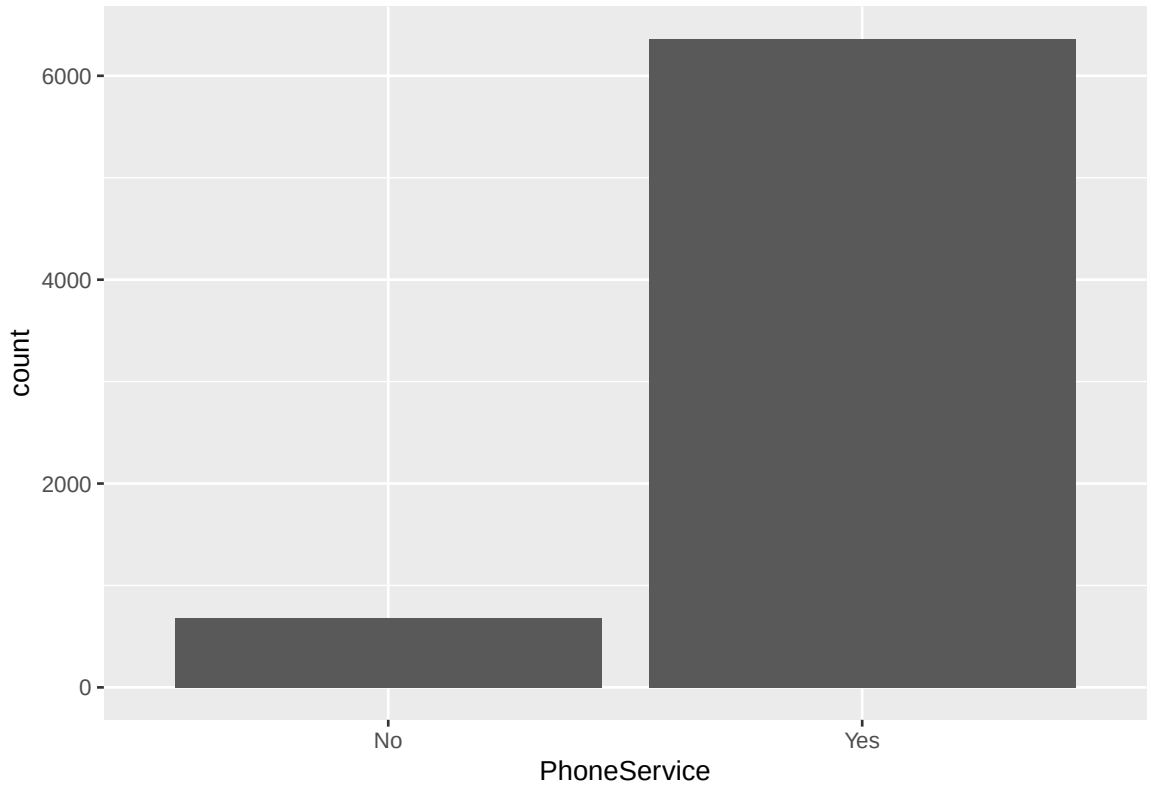

Figure 4: Test4

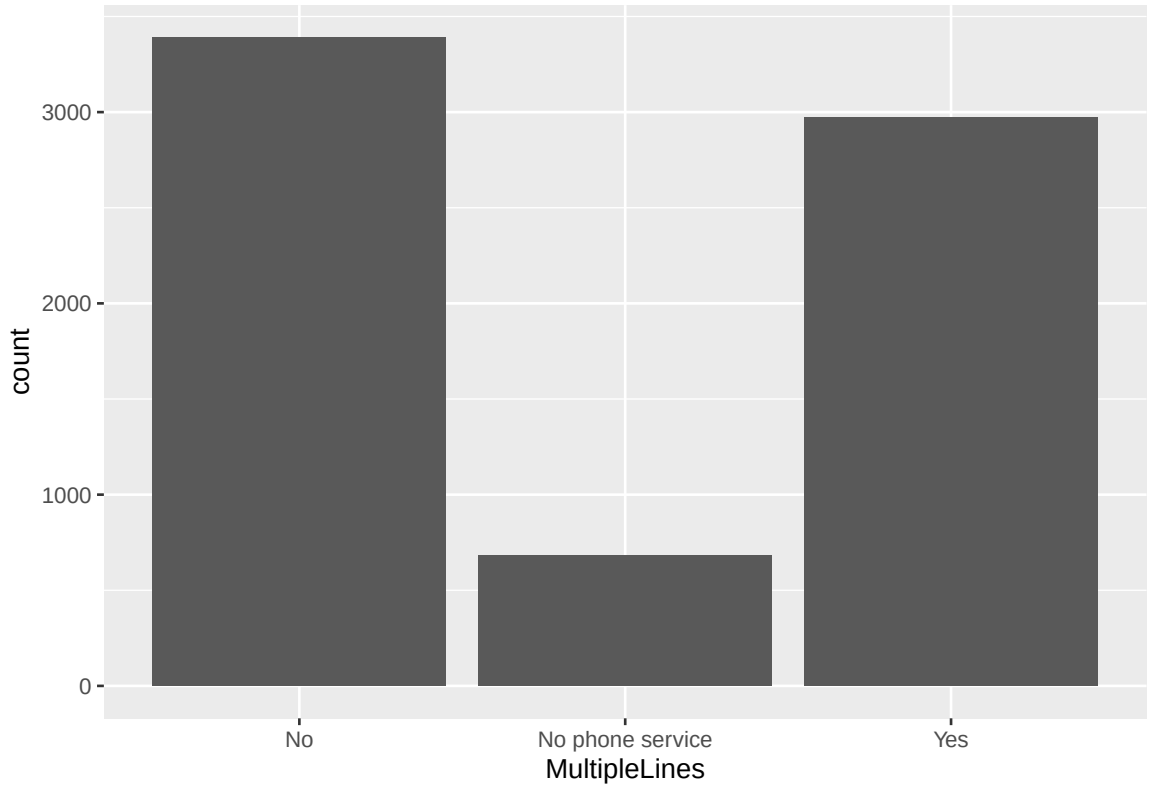

Figure 5: Test5

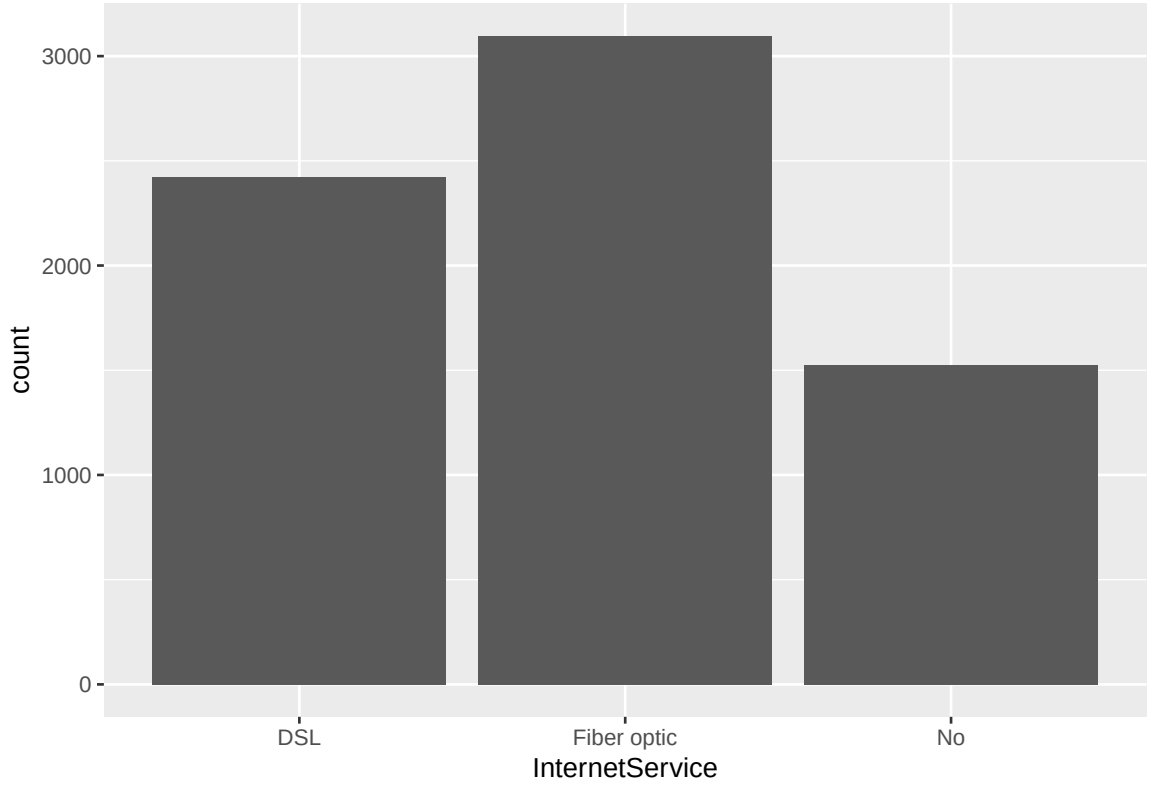

Figure 6: Test6

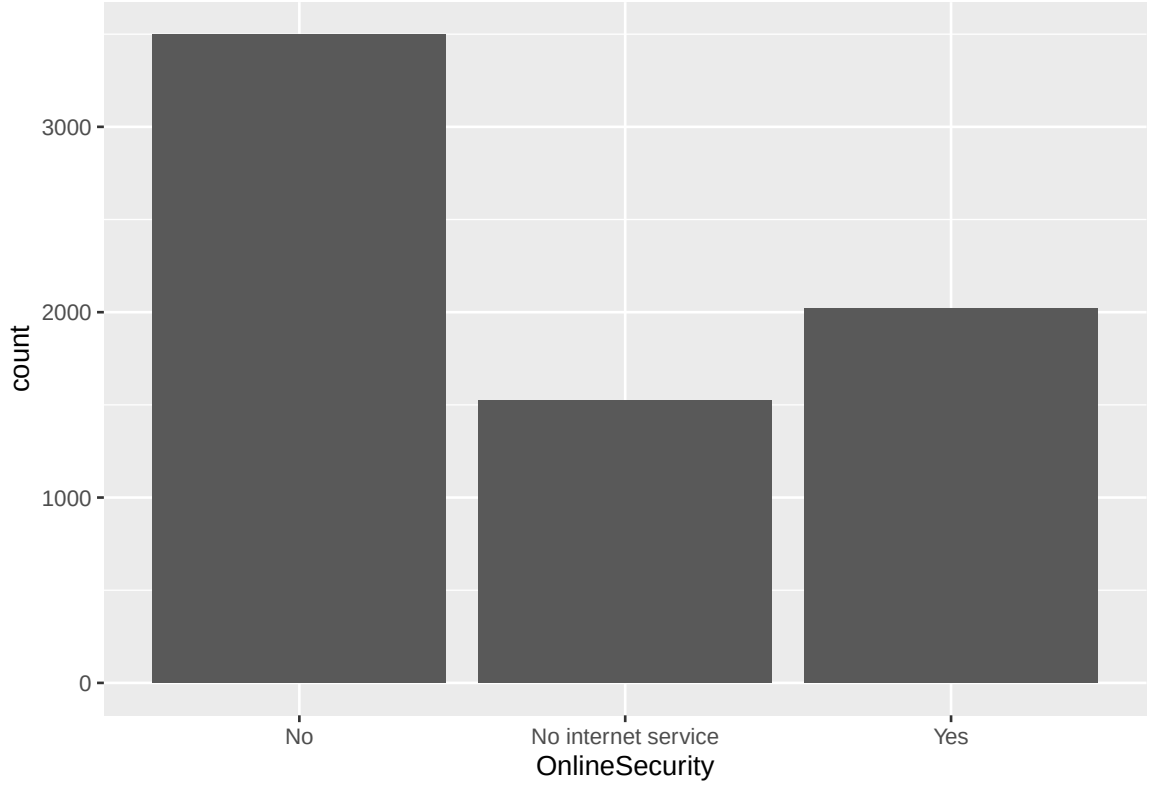

Figure 7: Test7

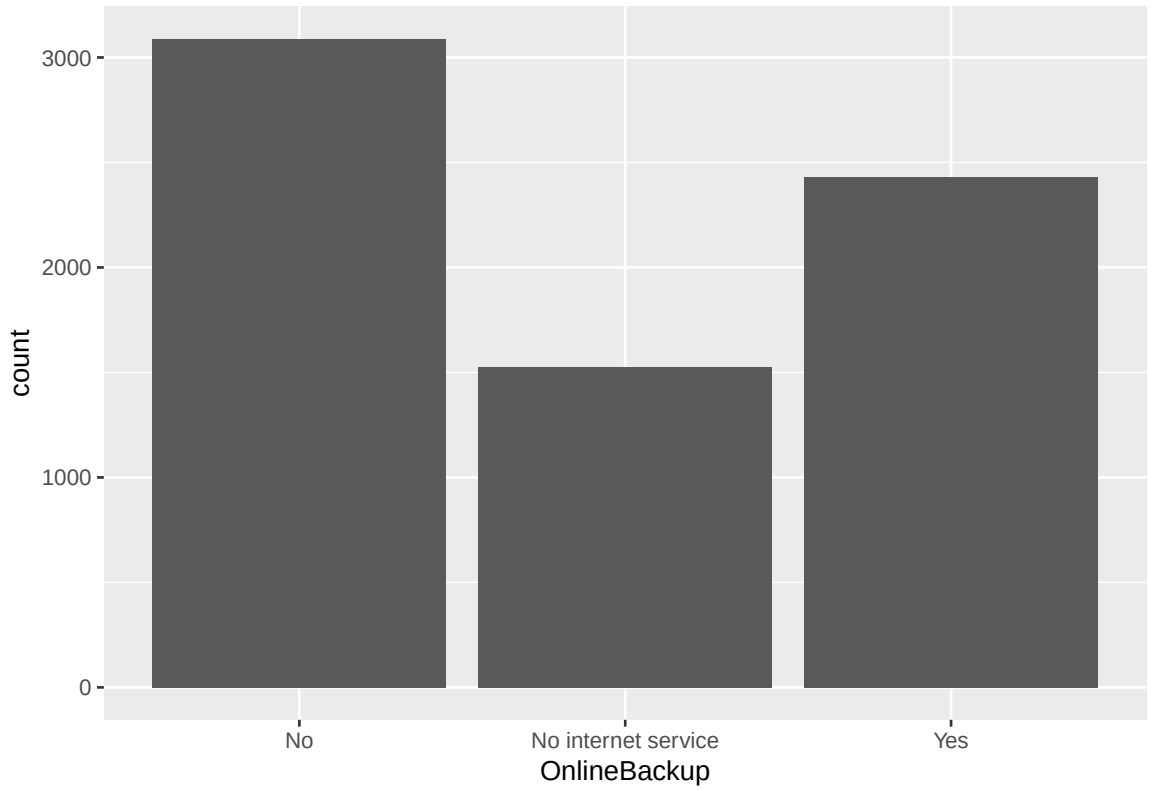

Figure 8: Test8

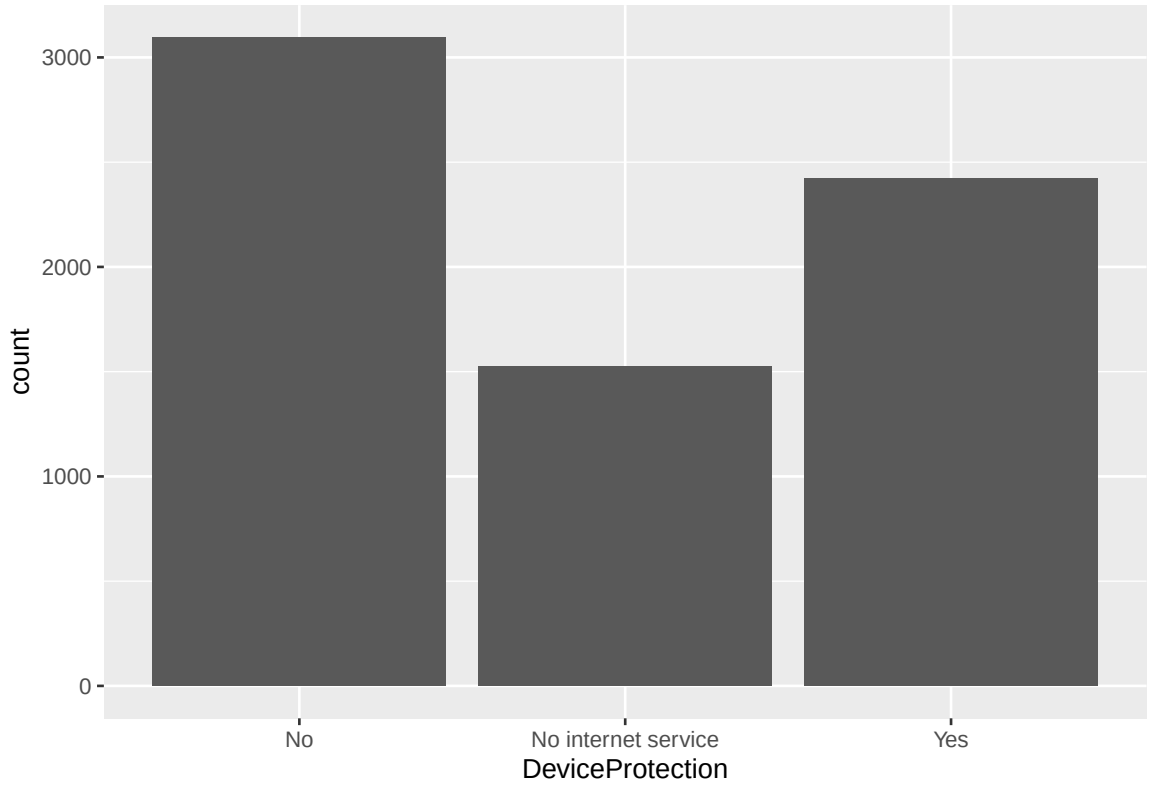

Figure 9: Test9

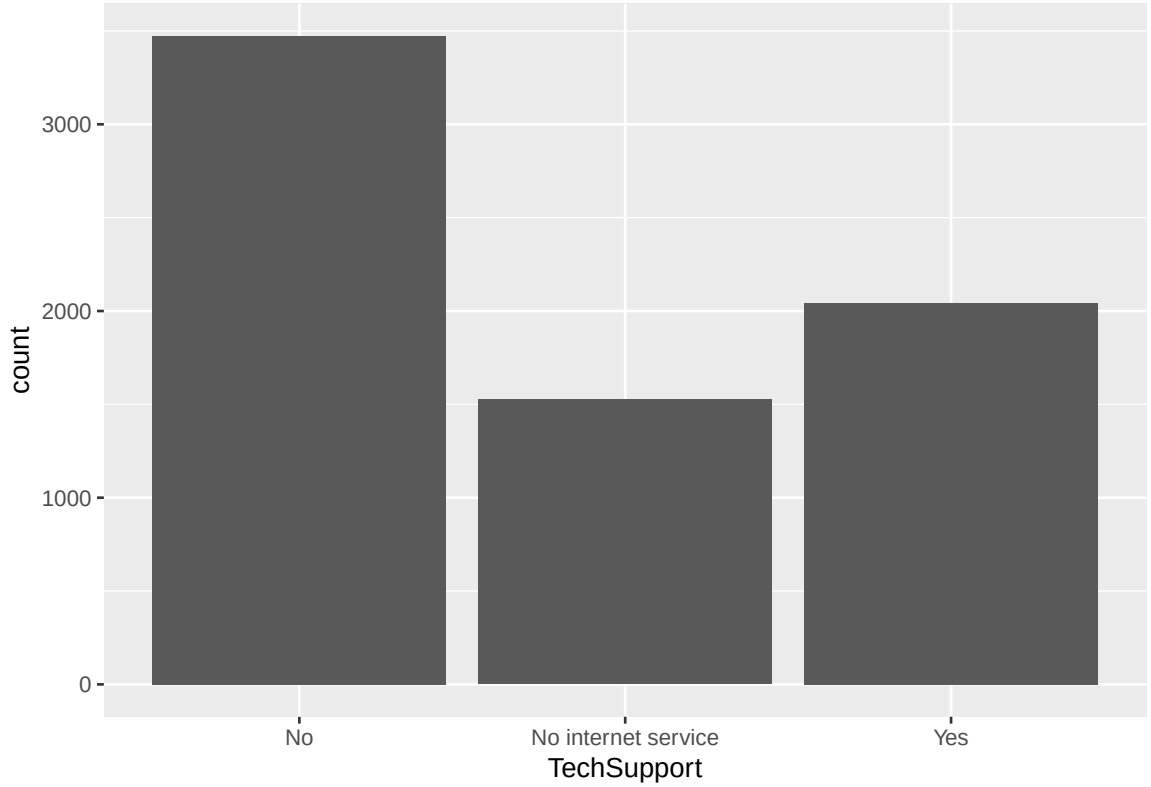

Figure 10: Test10

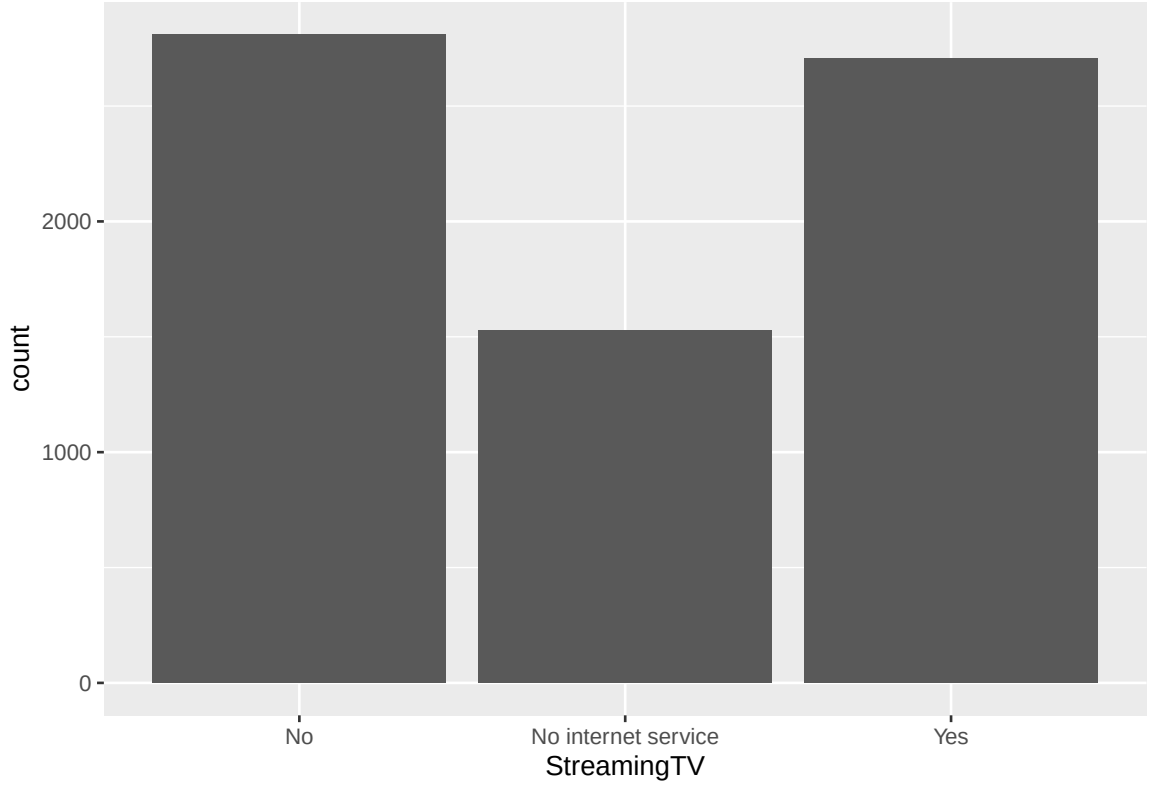

Figure 11: Test11

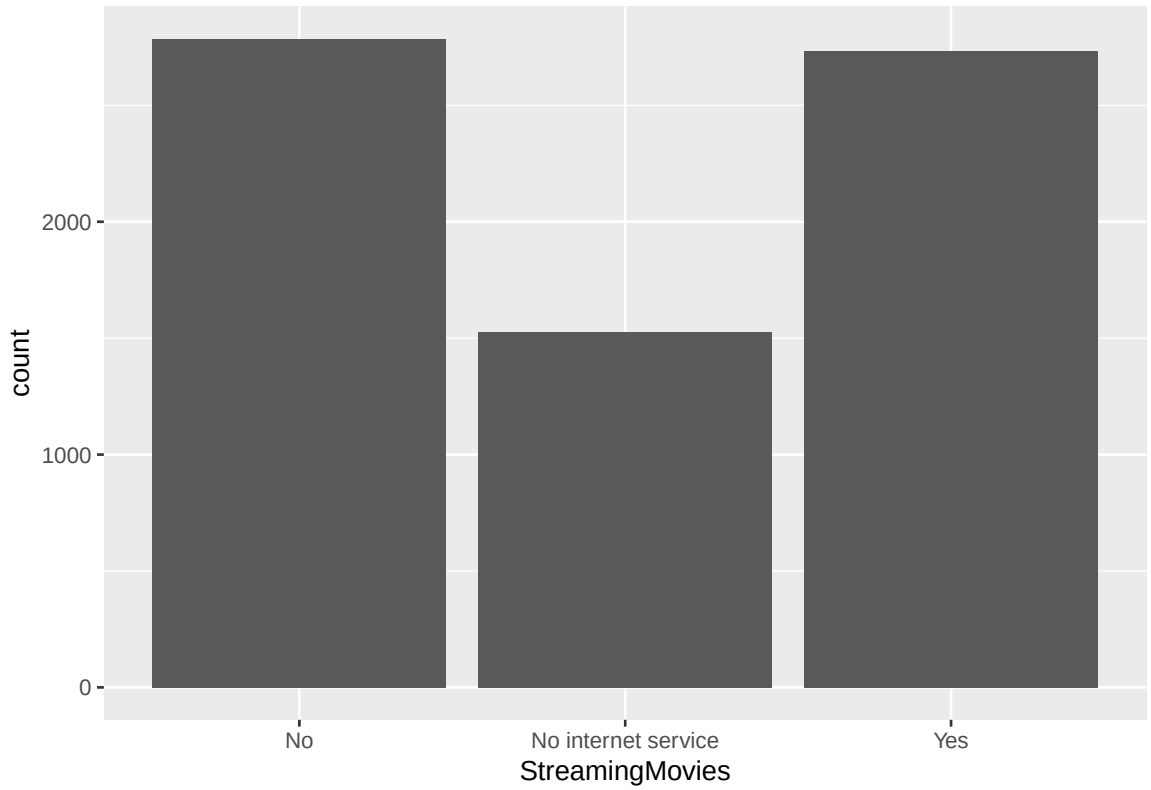

Figure 12: Test12

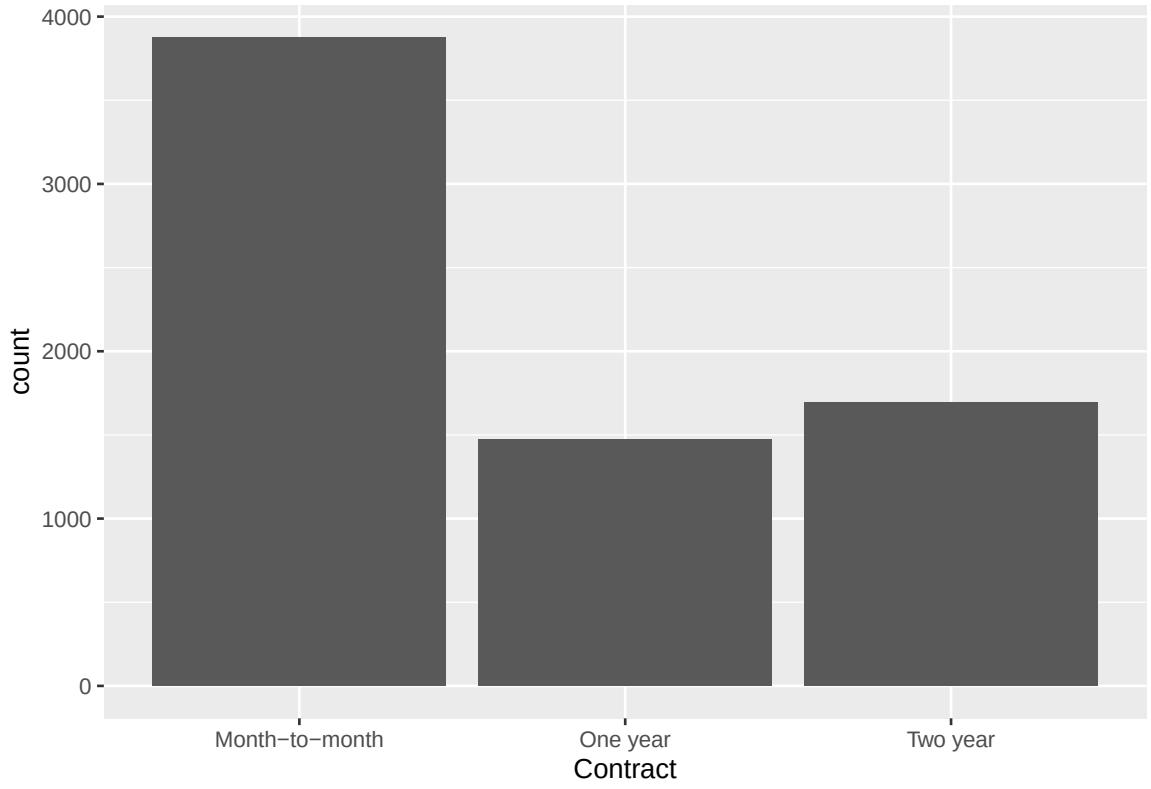

Figure 13: Test13

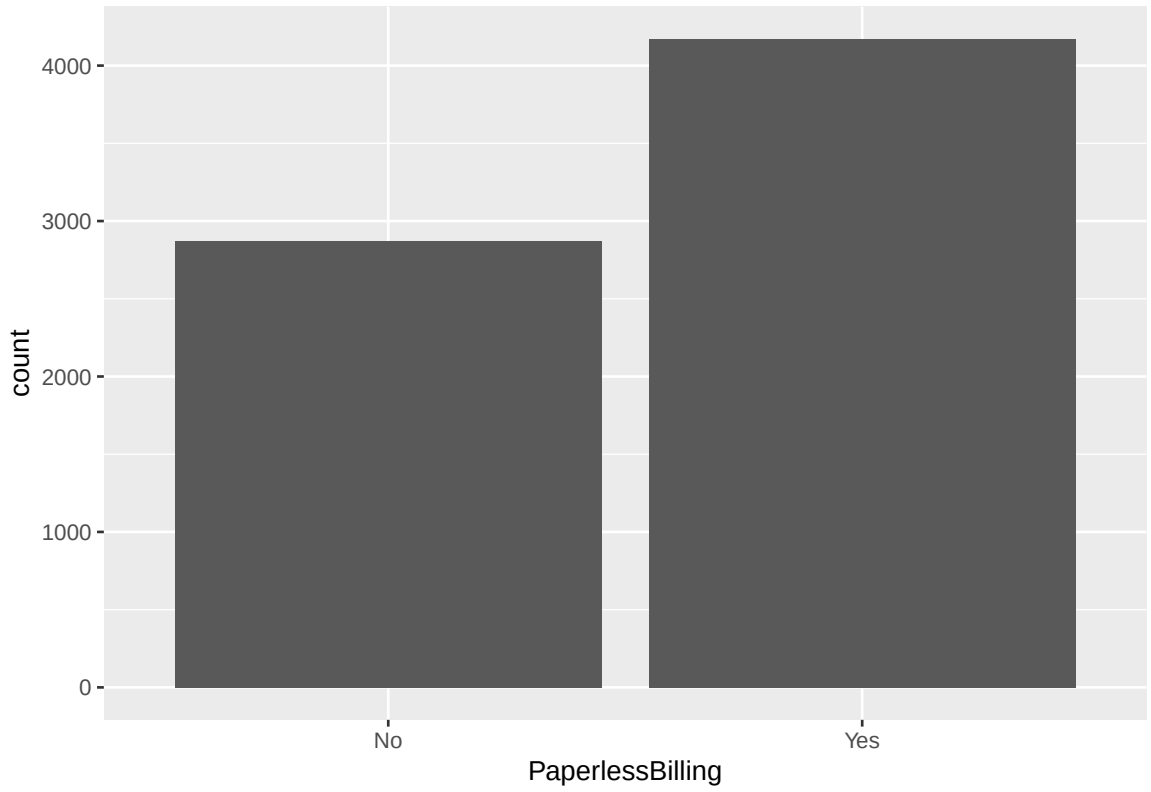

Figure 14: Test14

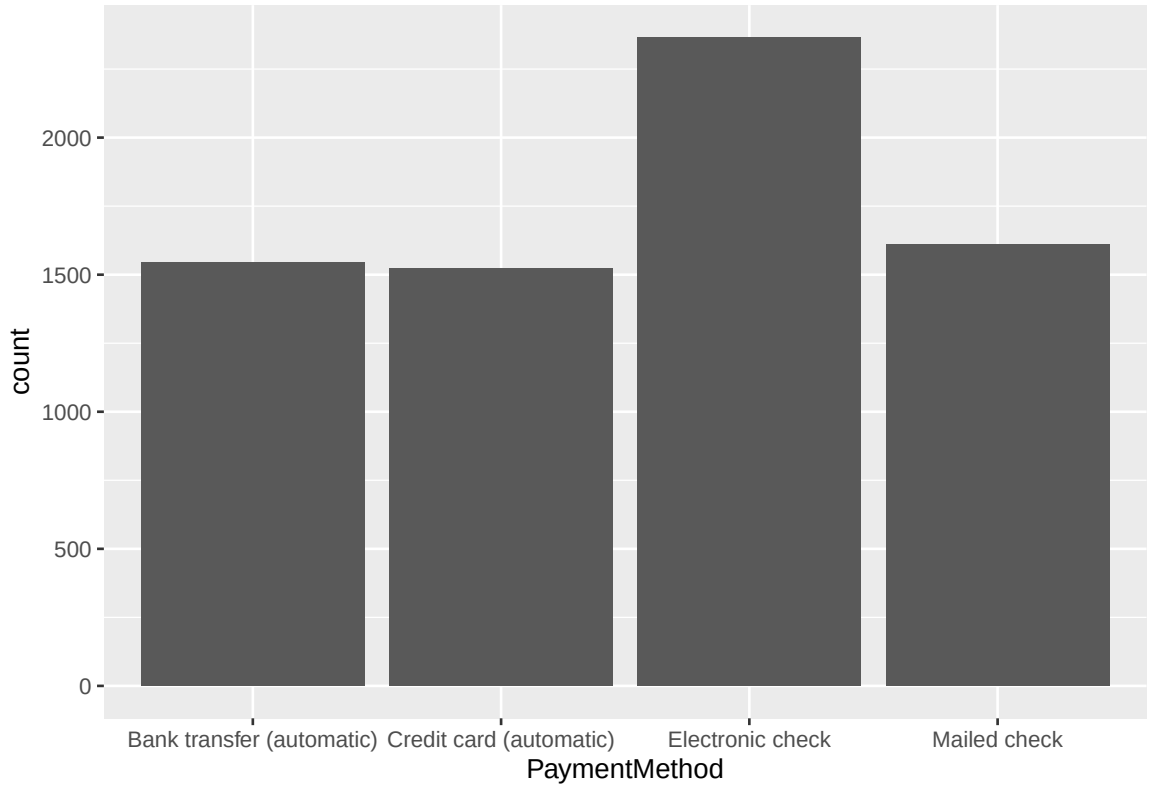

Figure 15: Test15

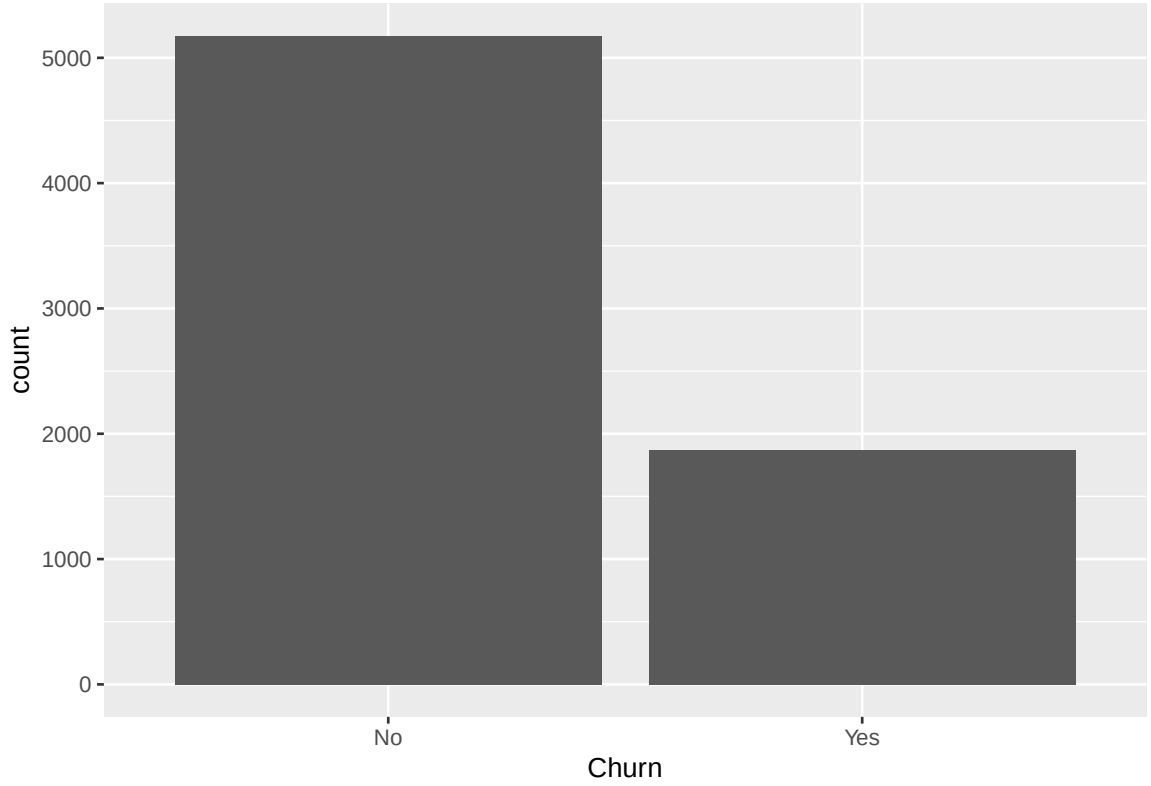

Figure 16: Test16

#As we can see in the continues' summary, there are 11 NA's in the Totalcharges column and some #outliers in Monthlycharges and TotalCharges. # Let's take a look of those columns in a graph to better understanding

Figure 17: Test17

#There is no normal distribution in Monthlycharges

#Let's see now Totalchrges

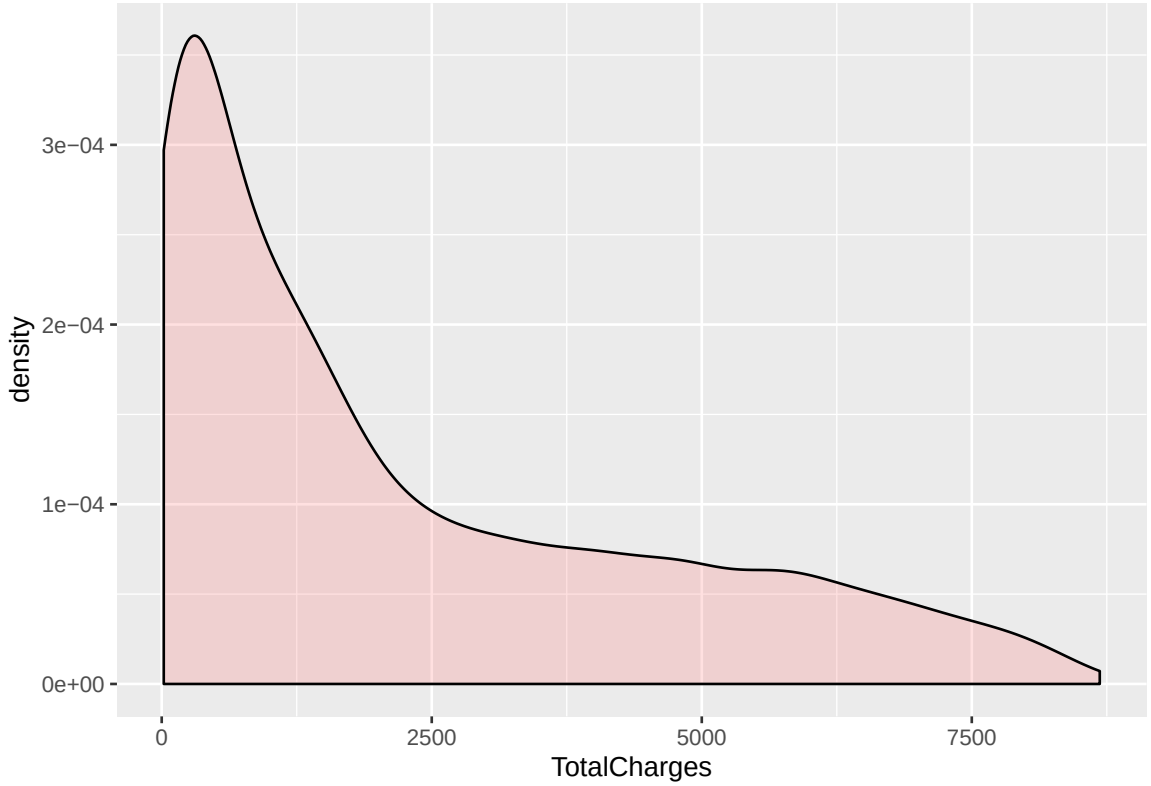

Figure 18: Test18

#Can we make a hypothesis about the relation of charges total and monthly with the client churn?

```
customerData = customerData %>%filter(complete.cases(.))
customerData = subset(customerData,select = -customerID)
```

Modeling and Evalutation

Finally we have reached the stage where we can start training and evaluating classification models. At this point we have clear understanding of our data. We have gotten rid of the features that did not present much value. We have filled the gaps in our data set employing sophisticated imputation technique.

Feature Selection

Generally speaking feature evaluation methods can be separated into two groups: those that use the model information and those that do not. Clearly at this stage of our research the models are not ready. Thus we will be exploring the methods that do not require model.

This group of the method could be spit further as follows:

- wrapper methods that evaluate multiple models adding and/or removing predictors. These are some examples:
  - recursive feature elimination
  - genetic algorithms
  - simulated annealing
- filter methods which evaluate the relevance of the predictors outside of the predictive models.

Before we proceed any further let's ensure that all categorical values get converted back to the factors. This is useful for dimentiality reduction algorithms and model training.

```
customerData = mutate(customerData,
          gender = as.factor(unclass(gender)),
          Partner = as.factor(unclass(Partner)),
          Dependents = as.factor(unclass(Dependents)),
          PhoneService = as.factor(unclass(PhoneService)),
          MultipleLines = as.factor(unclass(MultipleLines)),
          InternetService = as.factor(unclass(InternetService)),
          OnlineSecurity = as.factor(unclass(OnlineSecurity)),
          OnlineBackup = as.factor(unclass(OnlineBackup)),
          DeviceProtection = as.factor(unclass(DeviceProtection)),
          TechSupport = as.factor(unclass(TechSupport)),
          StreamingTV = as.factor(unclass(StreamingTV)),
          StreamingMovies = as.factor(unclass(StreamingMovies)),
          Contract = as.factor(unclass(Contract)),
          PaperlessBilling = as.factor(unclass(PaperlessBilling)),
          PaymentMethod = as.factor(unclass(PaymentMethod)),
          Churn = as.factor(unclass(Churn)))
```

It is time to run feature selection algorithm.

```
predictors = subset(customerData,select = -Churn)
label = customerData[,20]

# run the RFE algorithm
rfePrediction = rfe(predictors, label, sizes=c(1:19),
                    rfeControl = rfeControl(functions=rfFuncs, method="cv", number=3))
print(rfePrediction)

#>
#> Recursive feature selection
#>
#> Outer resampling method: Cross-Validated (3 fold)
#>
#> Resampling performance over subset size:
#>
#>  Variables Accuracy  Kappa AccuracySD KappaSD Selected
#>          1   0.7537 0.2324  0.0002463 0.03448
#>          2   0.7671 0.3427  0.0125907 0.04812
#>          3   0.7766 0.3476  0.0104877 0.05413
#>          4   0.7662 0.3692  0.0094340 0.01931
#>          5   0.7868 0.4065  0.0084718 0.01321
#>          6   0.7929 0.4257  0.0055515 0.01507
#>          7   0.7954 0.4352  0.0091234 0.01266
#>          8   0.7961 0.4415  0.0055461 0.01515
#>          9   0.7881 0.4289  0.0066275 0.02784
#>         10   0.7900 0.4342  0.0085466 0.02744
#>         11   0.7898 0.4279  0.0060983 0.02090
#>         12   0.7944 0.4420  0.0079699 0.02233
#>         13   0.7929 0.4360  0.0097689 0.03018
#>         14   0.7975 0.4445  0.0060684 0.01973
#>         15   0.7965 0.4422  0.0021331 0.01174
#>         16   0.7965 0.4399  0.0051725 0.01372
#>         17   0.7964 0.4399  0.0023496 0.01032
#>         18   0.7961 0.4369  0.0041142 0.01590
#>         19   0.7981 0.4422  0.0034483 0.01359        *
#>
#> The top 5 variables (out of 19):
#>    tenure, TotalCharges, MonthlyCharges, Contract, TechSupport
```

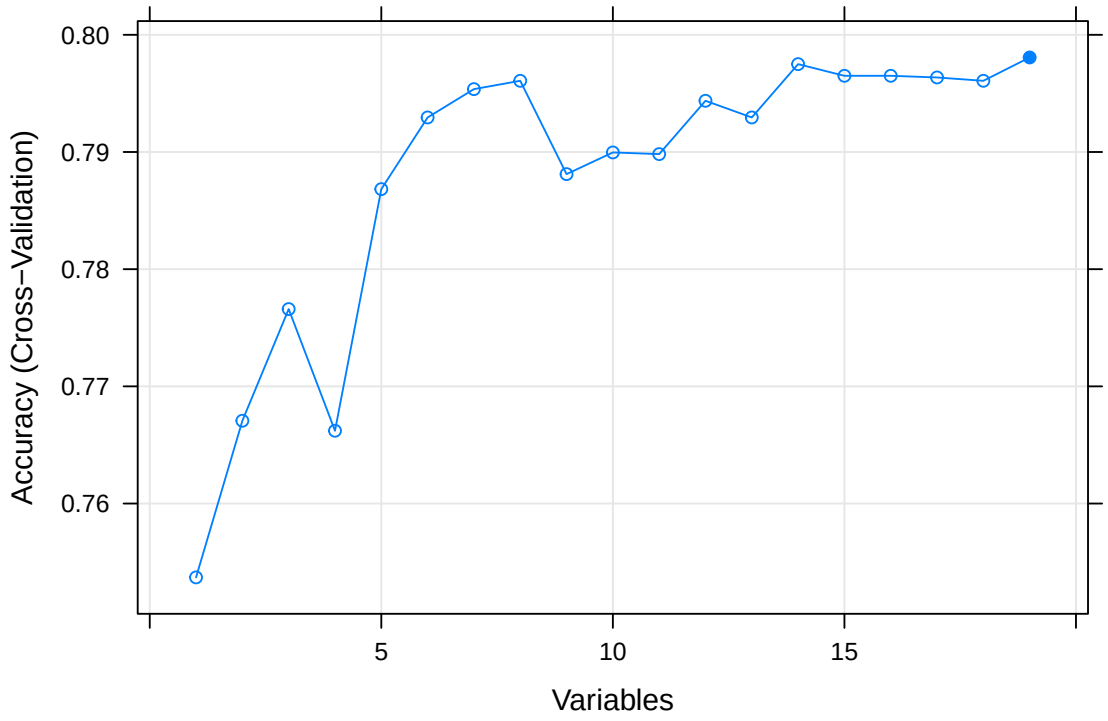

Figure 19: Number of Predictors vs Accuracy

Figure 19 illustrates that the accuracy practically flattens out when a number of predictors reaches 8. The accuracy improves a bit more when a number of features reaches 15 but the gain is negligible. Here is the list of features ordered by importance. We take first nine for model training.

```
#>  [1] "tenure"           "TotalCharges"     "MonthlyCharges"   "Contract"
#>  [5] "TechSupport"      "InternetService"  "OnlineSecurity"   "OnlineBackup"
#>  [9] "PaymentMethod"    "MultipleLines"    "PaperlessBilling" "StreamingTV"
#> [13] "StreamingMovies"  "DeviceProtection" "SeniorCitizen"    "PhoneService"
#> [17] "Partner"          "Dependents"       "gender"
```

Data Upsampling

There is one more step to make before we get to the model training. As shown in Figure **??** our data set is unbalanced. This could cause model over-fitting. So let's split the data into the training and testing sets and up-sample the training set.

```
set.seed(1608)

# keep only the selected features
finalSample = customerData %>% dplyr::select(c(selectedPredictors,"Churn"));

splitIdx = createDataPartition(finalSample$Churn, p=0.7, list = F)  # 70% training data
trainData = finalSample[splitIdx, ]
testData = finalSample[-splitIdx, ]

set.seed(590045)
columns = colnames(trainData)
trainData = upSample(x = trainData[, columns[columns != "Churn"] ],
      y = trainData$Churn, list = F, yname = "Churn")

rm(splitIdx, columns, finalSample)
print(table(trainData$Churn))
```

```
#>
#>    1    2
#> 3615 3615
```

As we can see now the training set is balanced.

Thus we have prepared our training and test data sets. We have identified the most important features. We are ready to work on the prediction models.

Decision Tree Model

Decision Tree algorithm is simple to understand, interpret and visualize. Effort required for data preparation is minimal. This is probably why the Decision Tree model tends to be the method of choice for predictive modeling of many.

```
#> Setting levels: control = no, case = yes
```

```
#> Setting direction: controls < cases
```

```
#> Conditional Inference Tree
#>
#> 7230 samples
#>    8 predictor
#>    2 classes: 'no', 'yes'
#>
#> No pre-processing
#> Resampling: Cross-Validated (5 fold)
#> Summary of sample sizes: 5784, 5784, 5784, 5784, 5784
#> Resampling results across tuning parameters:
#>
#>   mincriterion  ROC        Sens       Spec
#>   0.01          0.8552514  0.7125864  0.8420470
#>   0.50          0.8506961  0.7195021  0.8320885
#>   0.99          0.8392769  0.6829876  0.8406639
#>
#> ROC was used to select the optimal model using the largest value.
#> The final value used for the model was mincriterion = 0.01.
```

```
confusionMatrix(data = pred.decisionTreeModel.raw, testDataCopy$Churn)
```

```
#> Confusion Matrix and Statistics
#>
#>           Reference
#> Prediction   no  yes
#>        no  1050  107
#>        yes  498  453
#>
#>                Accuracy : 0.713
#>                  95% CI : (0.6932, 0.7322)
#>     No Information Rate : 0.7343
#>     P-Value [Acc > NIR] : 0.9871
#>
#>                   Kappa : 0.3984
#>
#>  Mcnemar's Test P-Value : <2e-16
#>
#>             Sensitivity : 0.6783
#>             Specificity : 0.8089
#>          Pos Pred Value : 0.9075
#>          Neg Pred Value : 0.4763
#>              Prevalence : 0.7343
#>          Detection Rate : 0.4981
#>    Detection Prevalence : 0.5489
#>       Balanced Accuracy : 0.7436
#>
#>        'Positive' Class : no
#>
```

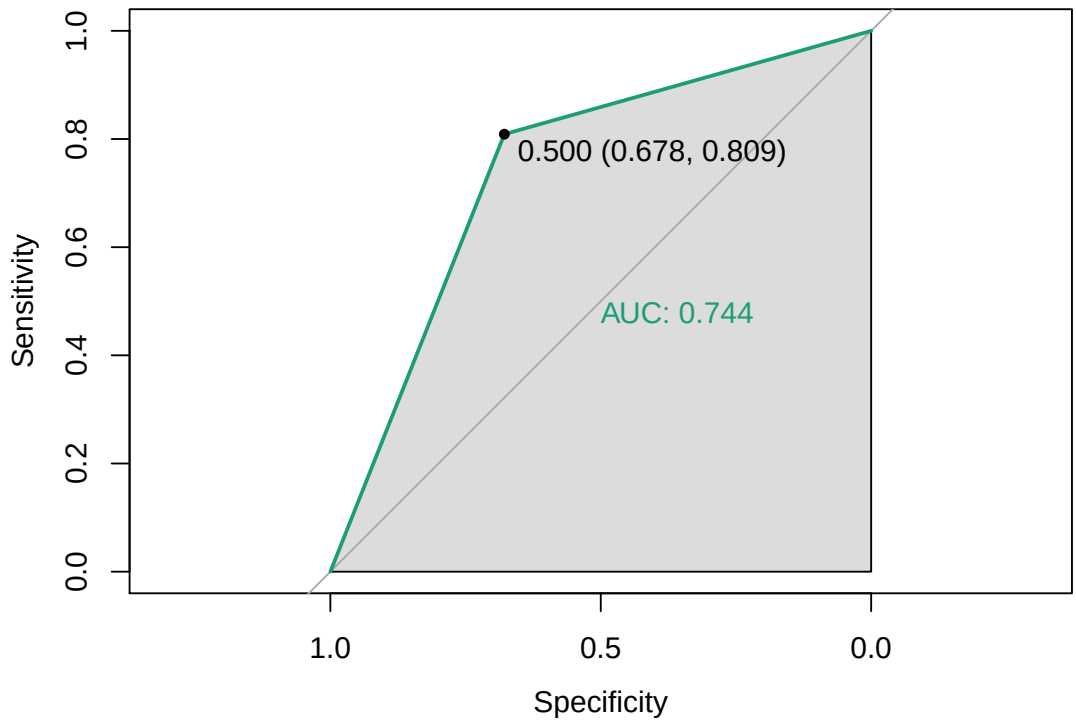

Figure 20: Classification Tree Model AUC and ROC Curve

Naive Bayes Model

Naïve Bayes classification is a kind of simple probabilistic classification methods based on Bayes' theorem with the assumption of independence between features.

It is simple (both intuitively and computationally), fast, performs well with small amounts of training data, and scales well to large data sets. The greatest weakness of the naïve Bayes classifier is that it relies on an often-faulty assumption of equally important and independent features which results in biased posterior probabilities. Although this assumption is rarely met, in practice, this algorithm works surprisingly well and accurate; however, on average it rarely can compete with the accuracy of advanced tree-based methods (random forests & gradient boosting machines) but is definitely worth having in our toolkit.

```
#> Naive Bayes
#>
#> 7230 samples
#>    8 predictor
#>    2 classes: 'no', 'yes'
#>
#> No pre-processing
#> Resampling: Cross-Validated (5 fold)
#> Summary of sample sizes: 5784, 5784, 5784, 5784, 5784
#> Resampling results across tuning parameters:
#>
#>   usekernel  ROC        Sens       Spec
#>   FALSE      0.8262922  0.5836791  0.8824343
#>    TRUE      0.8297156  0.3695712  0.9380360
#>
#> Tuning parameter 'fL' was held constant at a value of 0
#> Tuning
#>  parameter 'adjust' was held constant at a value of 1
#> ROC was used to select the optimal model using the largest value.
#> The final values used for the model were fL = 0, usekernel = TRUE and adjust
#>  = 1.
```

```
confusionMatrix(data = pred.naiveBayesModel.raw, testDataCopy$Churn)

#> Confusion Matrix and Statistics
#>
#>           Reference
#> Prediction  no yes
#>        no  550  47
#>        yes 998 513
#>
#>                Accuracy : 0.5043
#>                  95% CI : (0.4827, 0.5258)
#>     No Information Rate : 0.7343
#>     P-Value [Acc > NIR] : 1
#>
#>                   Kappa : 0.176
#>
#>  Mcnemar's Test P-Value : <2e-16
#>
#>             Sensitivity : 0.3553
#>             Specificity : 0.9161
#>          Pos Pred Value : 0.9213
#>          Neg Pred Value : 0.3395
#>              Prevalence : 0.7343
#>          Detection Rate : 0.2609
#>    Detection Prevalence : 0.2832
#>       Balanced Accuracy : 0.6357
#>
#>        'Positive' Class : no
#>
```

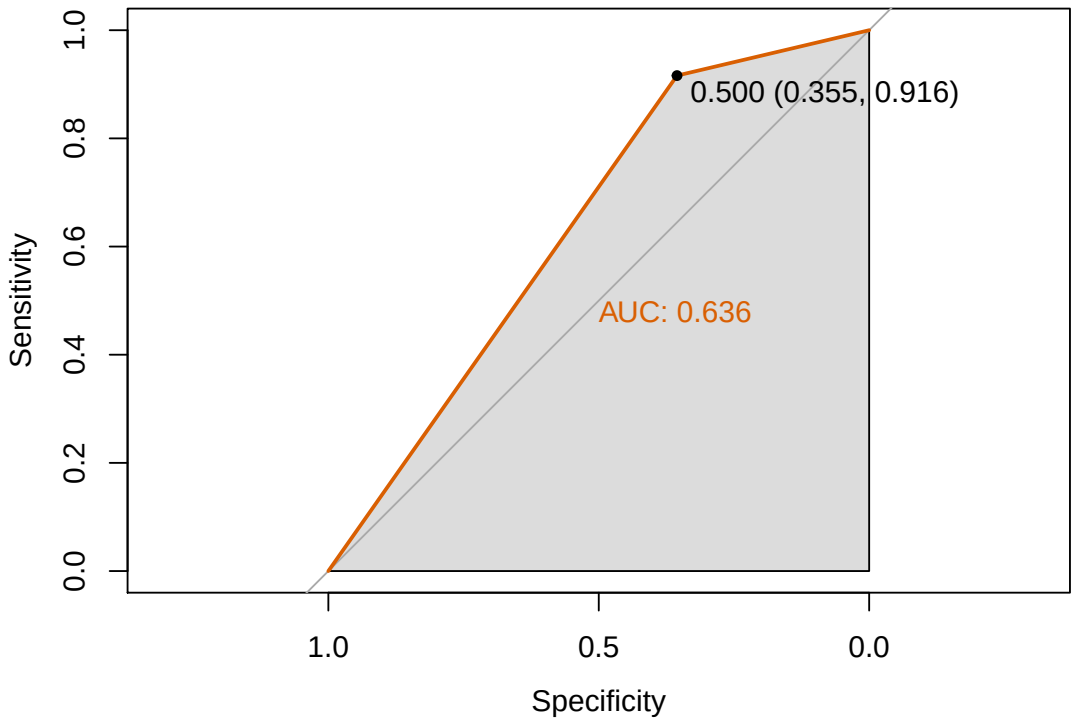

Figure 21: Naive Bayes Model AUC and ROC Curve

Random Forest Model

Random Forest is also considered as a very handy and easy to use algorithm, because it's default hyperparameters often produce a good prediction result. Random Forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. The main limitation of Random Forest is that a large number of trees can make the algorithm to slow and ineffective for real-time predictions. In general, these algorithms are fast to train, but quite slow to create predictions once they are trained.

```
#>    user  system elapsed
#>   28.53    0.35   28.89
```

```
#> Random Forest
#>
#> 7230 samples
#>    8 predictor
#>    2 classes: 'no', 'yes'
#>
#> No pre-processing
#> Resampling: Cross-Validated (3 fold)
#> Summary of sample sizes: 4820, 4820, 4820
#> Resampling results across tuning parameters:
#>
#>   mtry  ROC        Sens       Spec
#>    2    0.8548556  0.6874136  0.8517289
#>    7    0.9332004  0.7858921  0.9311203
#>   13    0.9308769  0.7872752  0.9244813
#>
#> ROC was used to select the optimal model using the largest value.
#> The final value used for the model was mtry = 7.
```

```
confusionMatrix(data = pred.randomForestModel.raw, testDataCopy$Churn)
```

```
#> Confusion Matrix and Statistics
#>
#>           Reference
#> Prediction   no  yes
#>        no  1263  211
#>        yes  285  349
#>
#>                Accuracy : 0.7647
#>                  95% CI : (0.746, 0.7827)
#>     No Information Rate : 0.7343
#>     P-Value [Acc > NIR] : 0.0007686
#>
#>                   Kappa : 0.4213
#>
#>  Mcnemar's Test P-Value : 0.0010462
#>
#>             Sensitivity : 0.8159
#>             Specificity : 0.6232
#>          Pos Pred Value : 0.8569
#>          Neg Pred Value : 0.5505
#>              Prevalence : 0.7343
#>          Detection Rate : 0.5991
#>    Detection Prevalence : 0.6992
#>       Balanced Accuracy : 0.7196
#>
#>        'Positive' Class : no
#>
```

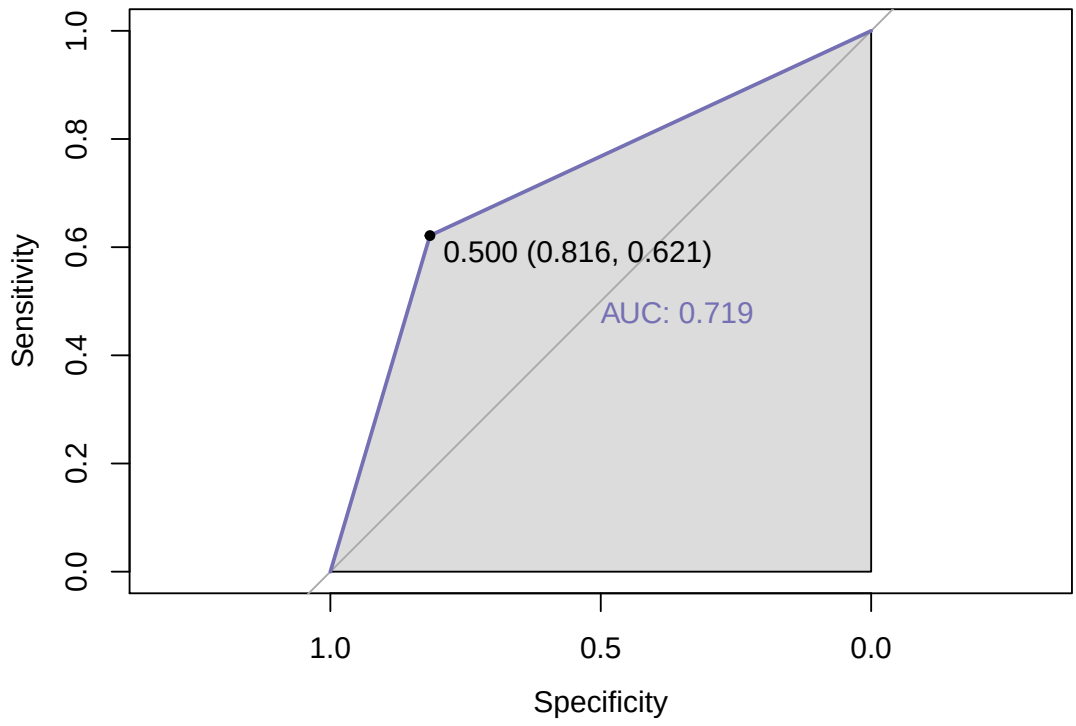

Figure 22: Random Forest Model AUC and ROC Curve

Logistic Regression Model

Logistic regression is an efficient, interpretable and accurate method, which fits quickly with minimal tuning. Logistic regression prediction accuracy will benefit if the data is close to Gaussian distribution. Thus we apply addition transformation to the training data set. We will also be employing 5-fold cross-validation resampling procedure to improve the model. In addition to the above we are going to convert *Location* categorical value to numeric data type. We could have used dummy encoding but having 49 locations such approach does not seem beneficial.

```
confusionMatrix(data = pred.logRegModel.raw, testDataCopy$Churn)
```

```
#> Confusion Matrix and Statistics
#>
#>           Reference
#> Prediction    1    2
#>          1 1129  125
#>          2  419  435
#>
#>                Accuracy : 0.7419
#>                  95% CI : (0.7227, 0.7605)
#>     No Information Rate : 0.7343
#>     P-Value [Acc > NIR] : 0.2228
#>
#>                   Kappa : 0.4335
#>
#>  Mcnemar's Test P-Value : <2e-16
#>
#>             Sensitivity : 0.7293
#>             Specificity : 0.7768
#>          Pos Pred Value : 0.9003
#>          Neg Pred Value : 0.5094
#>              Prevalence : 0.7343
#>          Detection Rate : 0.5356
```

```
#>    Detection Prevalence : 0.5949
#>       Balanced Accuracy : 0.7531
#>
#>        'Positive' Class : 1
#>
```

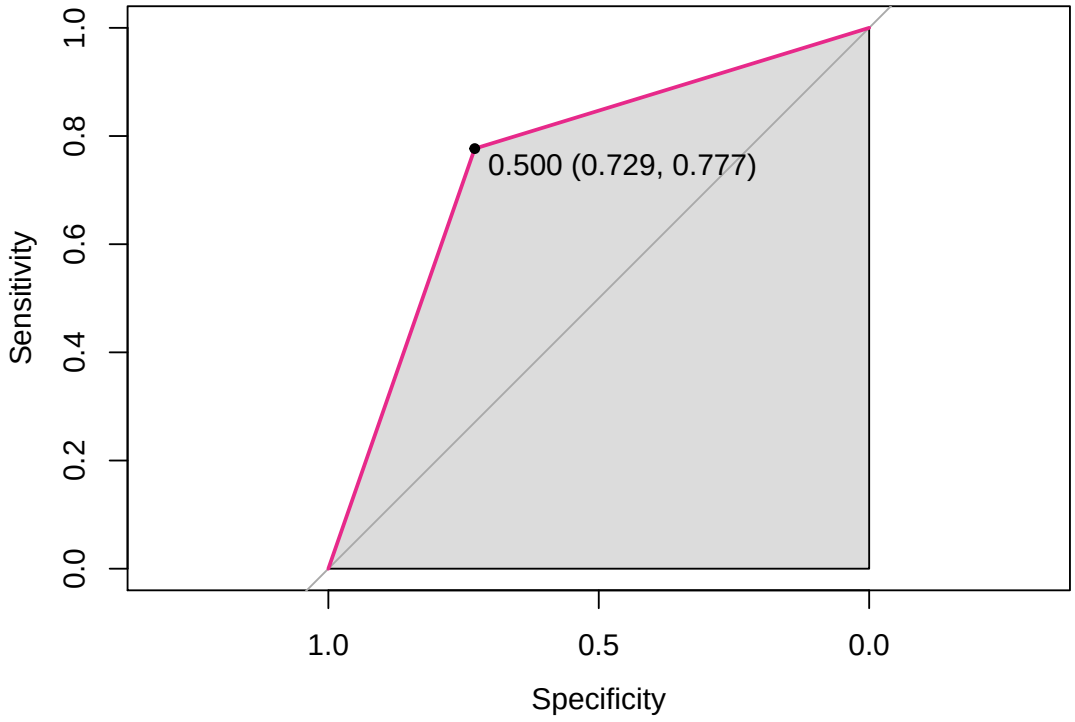

Figure 23: Logistic Regression Model AUC and ROC Curve

Model Comparison

Now it is time to compare the models side by side and pick a winner.

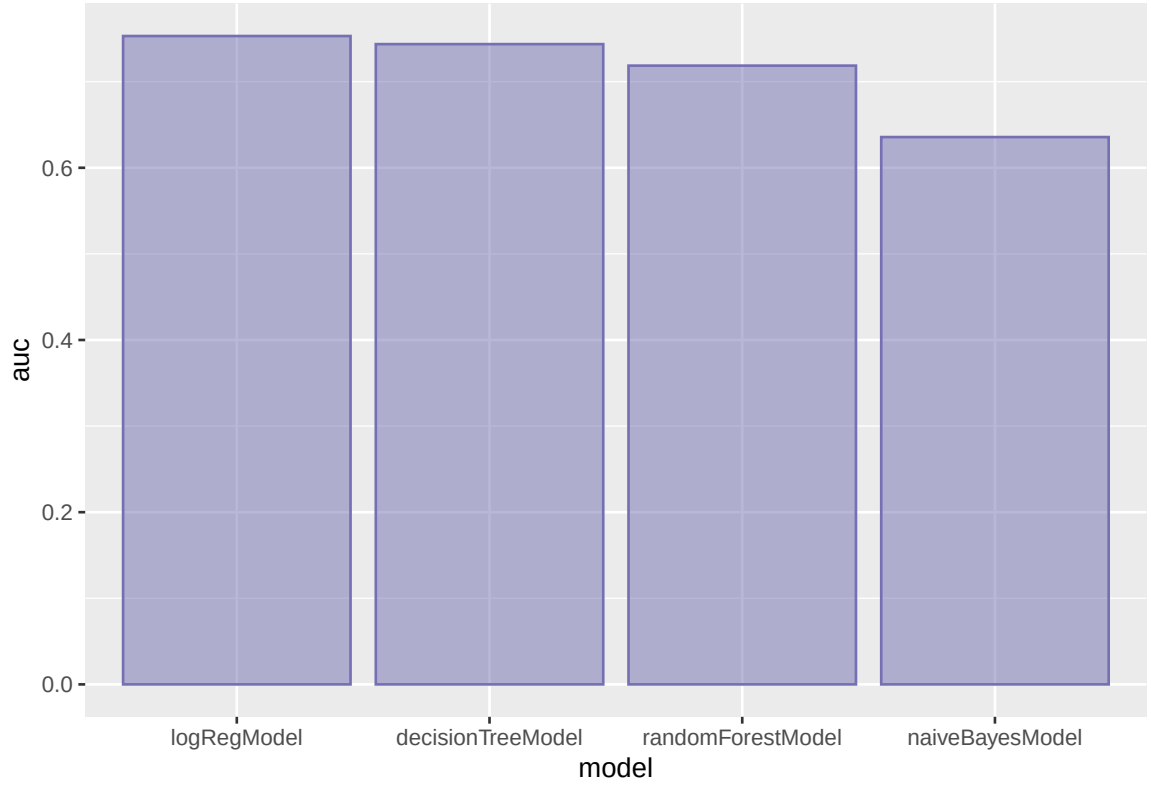

Figure 24: Model AUC Comparison

```
#>                model       auc
#> 1       logRegModel 0.7530569
#> 2 decisionTreeModel 0.7436116
#> 4 randomForestModel 0.7186600
#> 3   naiveBayesModel 0.6356843
```

AUC - ROC perfomance AUC stands for Area under the ROC Curve and ROC for Receiver operating characteristic curve. This is one of the most important KPIs of the classification algorithms. These two metrics measure how well the models distinguishing between the classes. The higher AUC the better model predicts positive and negative outcome.

Figures 20, 21, 22, 23 and accompanying data show that on the test data set all the models demonstrated very close resuts. Random Forest has the highest overall accuracy (85%) but performs poorly predicting rainy days (65%), thus the balanced accuracy is lower (about 78%).

Naive Bayes has lesser overall accuracy in comparison with the Random Forest but is more balanced, demonstrating consistent power to predict rainy and sunny days with almost equal accuracy. It scored over 78% on the balanced accuracy.

Logistic regression model scores the best having the highest AUC and all other metrics. It's balanced accuracy is 80%.

The Decision Tree performance is close to the other models with the balanced accuracy of 76%.

Model interpretibility Logistic Regression, Decision Tree and Naive Bayes are all highly interpreatable models. It is easy to explain to the business what impact each input parameter has. The decision tree could be visualized (provided if it is not too large).

Random Forest on the other hand is a black-box model, complex algorithm which is difficult to explain in simple terms.

Data Preparation Decision Tree, Random Forest and Naive Bayes can deal with missing data, outliers, numeric and alphanumeric values. Simply speaking they are not very demanding for data quality. It would be interesting to see how they perform on the original data set without data cleaning. But this is subject of another research...

Logistic regression does require conversion of alphanumeric values to numeric, struggles dealing with the outliers and performs best when fitted with the data that have normal distribution.

Verdict Despite sensitivity to data quality Logistic Regression outperforms other models in all other major categories. This is our choice!

Model Deployment

Without a doubt it would be a stretch to compare our model to the production numerical weather prediction models. But we do believe it might have a real live application as an educational tool. The model can demonstrate how various weather elements affect the probability of the rain.

It is simple to understand and deploy. The model does not require frequent updates because the weather patterns tend to be stable for a given geographical area (though this statement might be compromised in the context of the global warming). The model would benefit greatly if more complete data was available. Recall that we had to impute a lot of missing values.

Conclusion

Through exploring weather observations collected by 49 stations in Australia from 2007 to 2017 we selected and tuned a model to predict a rainy day tomorrow employing current day observations and historical data.

We commenced our research analyzing and understanding available data and geography of the weather stations. Then we identified the missing data, its distribution and feasibility of imputing it. We applied sophisticated data imputation algorithm to attack the problem. We continued our research selecting the most impactful data attributes to use as an input for our future model. Again we apply the feature identification algorithm to do the job.

When the data preparation phase was finished we picked and analysed four different classification models: Decision Tree, Naive Bayes, Random Forest and Logistic Regression. We conducted comparative analysis of the models, reviewed their strength and weaknesses. We fitted each model using K-fold cross-validation technique. Subsequently we evaluated performance of each model applying them to the test data set and comparing AUC - ROC and balanced accuracy metrics.

Finally we moved to identifying a winning model. In order to so we reviewed each model from different angles namely:

- performance
- interpretability
- data quality sensitivity and data preparation effort

The winning model scored the highest in the majority of the categories. It was Logistic Regression, which we employed to build a Shiny App Web application.

We consider the project to be a success.

Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

Ketao Li
York University

liketao@yahoo.com

Kush Halani
York University

kush.halani@ontariotechu.net

Josue Romain
York University

josue.rolland.romain@gmail.com

Juan Peña
York University

jppena62@my.yorku.ca

Priyanka Patil
York University

priyanka181994@gmail.com