Predict Member Churn Report

Group Ben Lorica
Jiamei Xu
Pranav Nair
Xinye Li
Zhicheng Zhu

Percentage of Effort Contributed by Jiamei: _____25%_____
Percentage of Effort Contributed by Pranav: _____25%_____
Percentage of Effort Contributed by Xinye: _____25%_____
Percentage of Effort Contributed by Zhicheng: _____25%_____

Signature of Jiamei: _____ *Jiamei Xu* **_____**
Signature of Pranav: _____ Pranav Nair **_____**
Signature of Xinye: _____ Xinye Li**_____**
Signature of Zhicheng: __ Zhicheng Zhu**_____**

Submission Date: _____4/25/2020_____

Contents

Introduction

Imagining you are a CEO of your company, what kind of information do you want based on the existing data? How to improve the company's business performance? In this project, we will use several predictive models to see how to predict the customer member churn.

At the very beginning, we want to illustrate our project by asking several questions:

- What's the problem we're trying to solve?
- What's our solution to the problem?
- How can we evaluate our model?

Let's answer these questions in detail:

What's the problem we're trying to solve?

In order to define our problem, we need to know what the churn rate is. The churn rate is the annual percentage at which customers stop subscribing to a service. The data we used comes from a large retail company in America, so customer member churn occurs here when customers stop subscribing their membership cards with this company. We want to know what factors are really essential to predict the customer member churn. If we have already known the actual factors, how can we suggest the company to take some measures to decrease the member churn rate?

What's our solution to the problem?

We can try to use our existing data to build a machine learning model to predict the member churn. In this project, we will use logistic regression, decision tree, random forest and neural network to do the prediction. We use R studio to do the data processing and run the model.

How can we evaluate our model?

Having the ability to accurately predict future churn rate is necessary because it helps your business gain a better understanding of future expected revenue. So, we'll try to use a

series of machine learning evaluation metrics, such as confusionMatrix, F1-score, AUROC to evaluate and improve our models.

Data Sources And Data Description

First, let's look at the dimension of the dataset.

```
## [1] "[120,450 x 22]"
##  [1] "Renew"      "MembNo"     "AccType"    "ExecMemb"   "BsnType"
##  [6] "Region"     "WhNo"       "Age"        "Tenure"     "Zip"
## [11] "MemCount""Dist"      "LastShop"           "HomeWh"           "RecentMove"
## [16] "SHOP1YR"            "SHOP6M"             "SHOP3M"           "ECOMSHOP"       "GASSHOP"
## [21] "MEDICALSHOP" "GROCERYSHOP"
```

Taking a look we see that there are 22 features and 120,450 rows of observances. The column names are confusing, so we rename it to make it more understandable. The target feature we'll be attempting to predict is "Renew". We can dig a little deeper and take a look at the data types of the features.

```
 $ Renew      : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 1 1 ...
 $ MembNo     : int  280928 280100 279886 279912 279896 279912 279904 279904 279922 280282 ...
 $ AccType    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ ExecMemb   : Factor w/ 2 levels "E","N": 2 2 2 2 1 1 2 2 2 1 ...
 $ BsnType    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Region     : Factor w/ 10 levels "BA","BD","BO",..: 6 3 3 9 1 2 8 4 8 5 ...
 $ WhNo       : int  1078 847 847 185 472 823 673 769 691 376 ...
 $ Age        : int  42 61 52 32 46 36 34 45 52 32 ...
 $ Tenure     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Zip        : int  20715 77346 91024 32789 93960 94544 89822 90047 85340 48316 ...
 $ MemCount   : int  2 2 2 2 2 1 2 2 2 2 ...
 $ Dist       : num  7.53 6.05 7.89 3.43 26.29 ...
 $ LastShop   : int  75 320 350 137 41 53 38 363 53 64 ...
 $ HomeWh     : Factor w/ 2 levels "N","Y": 2 1 1 1 1 1 1 1 1 2 ...
 $ RecentMove : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 1 1 ...
 $ SHOP1YR    : num  1385 3500 114 997 12579 ...
 $ SHOP6M     : num  827.7 0 0 23.2 73 ...
 $ SHOP3M     : num  253 0 0 0 73 ...
 $ ECOMSHOP   : num  0 1 0 0 0 0 0 0 0 0 ...
 $ GASSHOP    : num  0.0293 0 0 0.0251 0 ...
 $ MEDICALSHOP: num  0.0173 0 0.30377 0 0.00818 ...
 $ GROCERYSHOP: num  0.523 0 0.234 0.405 0.936 ...
```

figure 1.1 data types of features

The data contains various numerical features that are double types, such as the "AccType" feature, that has values of either "2" or "1". We can create dummy variables for this feature if necessary for any specific model. There are also character types, which include

“RECENTMOVING” and “HOMEFCTYCHANGE”. These values just represent “Yes” and “No” so we’ll convert these to a factor. And we also have “ECOMSHOP”,“GASSHOP” and it has a value between 0 and 1, which just represents the percentage of this specific shopping in total shopping. We will do discretization for these variables which make it more compatible with the random forest model. And we will remove A2ACCIPK because it’s just an account ID.

Next, we will check if there are any missing values in the dataset and will remove them.

```
    Renew      MembNo     AccType    ExecMemb     BsnType      Region        WhNo         Age      Tenure
        0           0           0           0           0           0           0           0           0
      Zip    MemCount        Dist    LastShop     HomeWh  RecentMove     SHOP1YR      SHOP6M      SHOP3M
        0           0           0           0           0           0           0           0           0
ECOMSHOP     GASSHOP MEDICALSHOP GROCERYSHOP
        0           0           0           0
```

figure 1.2 check missing values

Data Exploration

It looks like there is no missing value for any columns. And we have imported the data and done some cleaning, let’s start to explore the data.
To make this analysis more understandable, We are going to set up some questions and answer them by the following visualization.

1. Are business type customers more likely to renew than normal type?
2. Are elder customers more likely to renew?
3. Does a customer with guests renew more than individuals?

We will start with business type,

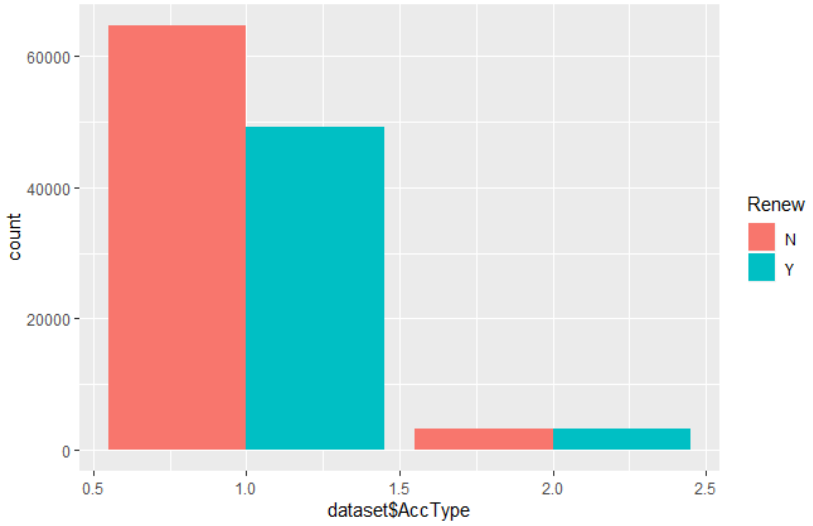

figure 2.1 Account type and Renew status

According to the graph, customers in individual types roughly have 10% fewer than the non-renew group and roughly the same for business types in both groups. And we can find the majority of customers are group 1 which means individual type. And we can also take a look at the average consumption of each group. We find we have 85562 individual customers and 4777 business customers and 2253.271 dollar and 40358.87 dollar average consumption for each group. The result shows us both groups are important for our business. Next we can have a look at the distribution of people from different ages and their membership status.

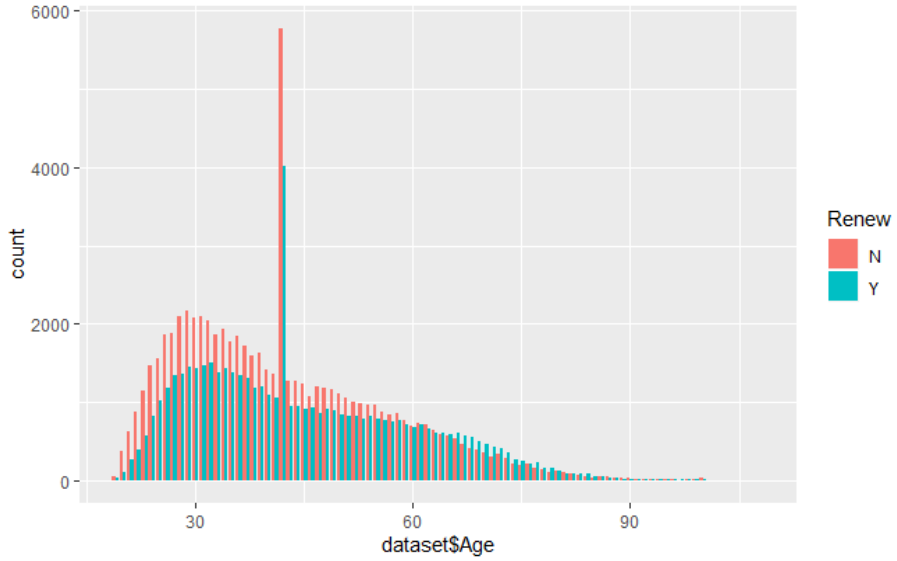

figure 2.2 Age and Renew status

This variable shows a much more meaningful relationship. The group after 60 years old tend to have more people renew their membership and for the group, before 60 more people tend to not renew, and the majority of our customer age belongs to the 40s group and for 20s and 30s group have the highest non-renew rate.

Next, we will take a look at the customer with guests.

For customers with more guests, they tend to have more people renew their membership and for these customers with more than 3 guests have roughly the same number in both renew and non-renew groups.

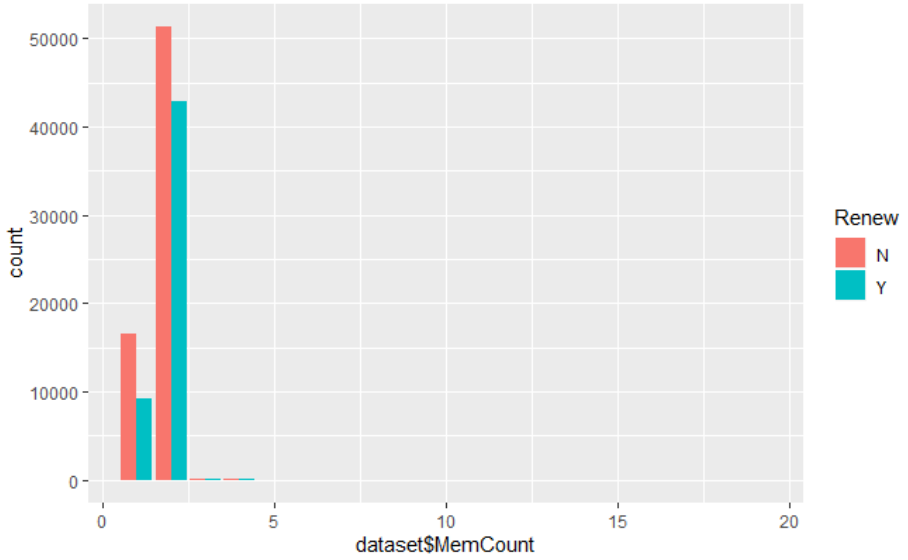

figure 2.3 Member count and Renew

From the next graph, we see that 43% of people renew their membership. Imbalanced classification problems are known to be more skewed with a binary class distribution of 90% to 10%. so our dataset will be fine. And the graph shows the regional distribution of our customer, we can find the top three regions are Midwest, Northeast and Southeast. And above half percent of our customers are executive members and roughly 25% of our customers go to the home city warehouse.

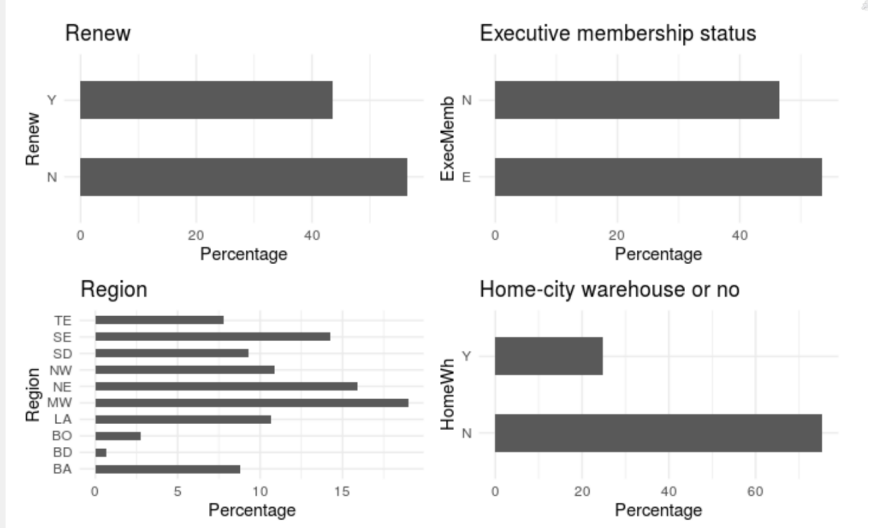

figure 2.4 Categorical variables visualization

Next, we will develop some predictive models. and find the best-fitted model for our prediction

Data Mining Tasks And Data Mining Models

1 Logistic Regression

1.1 Data Processing

To begin with, we drop the primary key MembNo since it will not be helpful for modeling. After using summary( ) and sum(is.na( )) to check the data types and missing values, we find that our dependent variable Renew, and our independent variables ExecMemb, Region, HomeWh, and RecentMove are categorical variables with factors, the remaining ones are all numerical variables. Meanwhile, there are no missing values.

Hence, the first thing we do is to change those categorical variables with two factors to 0 or 1. Since the variable Region has ten factors, we create dummy variables for it, split it into

ten different columns with two factors 0 or 1, aka ten different regions. The purpose of this step is to ensure the accuracy of logistic regression modeling.

1.2 Logistic Regression Modeling

After we set our seed to 2 and split our dataset into 60% for training and 40% for testing, we run our logistic regression model for three times. We remove non-significant variables AccType, BsnType, Region.BD, Region.BO, Region.TE, Tenure, MemCount, and Dist before running the second time. We give our second model ANOVA Check and remove non-significant variable RecentMove before running the final version. The summary of our final logistic regression model is shown below.

```
Call:
glm(formula = Renew ~ ., family = "binomial", data = train.df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-8.4904  -0.9495  -0.3877   1.0022   2.8905

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.278e+00  7.445e-02 -17.162  < 2e-16 ***
ExecMemb1   -4.843e-01  1.760e-02 -27.525  < 2e-16 ***
Region.BA1   2.379e-01  4.376e-02   5.436 5.44e-08 ***
Region.LA1   1.446e-01  4.080e-02   3.545 0.000393 ***
Region.MW1   4.418e-01  3.614e-02  12.226  < 2e-16 ***
Region.NE1   3.268e-01  5.571e-02   5.867 4.45e-09 ***
Region.NW1   1.656e-01  4.179e-02   3.962 7.44e-05 ***
Region.SD1   1.525e-01  4.043e-02   3.772 0.000162 ***
Region.SE1   1.308e-01  4.816e-02   2.715 0.006627 **
WhNo         6.502e-05  2.318e-05   2.805 0.005039 **
Age          2.127e-02  6.022e-04  35.324  < 2e-16 ***
Zip          2.147e-06  7.617e-07   2.818 0.004828 **
LastShop    -7.796e-03  1.536e-04 -50.766  < 2e-16 ***
HomeWh1      1.258e-01  1.971e-02   6.381 1.76e-10 ***
SHOP1YR      4.996e-05  7.360e-06   6.789 1.13e-11 ***
SHOP6M       1.654e-04  1.977e-05   8.365  < 2e-16 ***
SHOP3M       3.791e-04  3.075e-05  12.329  < 2e-16 ***
ECOMSHOP     3.386e-02  5.351e-03   6.328 2.49e-10 ***
GASSHOP      1.341e-02  3.639e-03   3.687 0.000227 ***
MEDICALSHOP  1.838e-02  3.354e-03   5.481 4.23e-08 ***
GROCERYSHOP -5.688e-02  3.952e-03 -14.391  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 98924  on 72269  degrees of freedom
Residual deviance: 81327  on 72249  degrees of freedom
AIC: 81369

Number of Fisher Scoring iterations: 7
```

figure 3.1.1 logistic regression summary

We can find all the significant independent variables from the summary. They are all marked with asterisks by RStudio. The significance of ExecMemb, Region.MW, Age,

Lastshop, SHOP6M, SHOP3M, and GROCERYSHOP are worth mentioning since their P values are extremely small, which could be used to compare with the rest of our models.

1.3 Logistic Regression Performance Evaluation

First of all, we use the confusion matrix to make predictions and assess our logistic regression model. The table is shown below.

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 24594 12770
         1  2504  8312

               Accuracy : 0.683
                 95% CI : (0.6788, 0.6871)
    No Information Rate : 0.5624
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3191

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9076
            Specificity : 0.3943
         Pos Pred Value : 0.6582
         Neg Pred Value : 0.7685
             Prevalence : 0.5624
         Detection Rate : 0.5105
   Detection Prevalence : 0.7755
      Balanced Accuracy : 0.6509

       'Positive' Class : 0
```

figure 3.1.2 confusion matrix of logistic regression model

We can see the classification prediction accuracy is about 68%, which is not so good. The misclassification error rate is 32%.

2 Decision Tree

2.1 Decision Tree Modeling

In this part, we are going to use the “rpart” package in R to generate our tree.

First, since Member ID is unique to each customer, we remove the Member ID feature. Next, we split the data into training and testing sets, here we choose 70% for training and 30%

for testing. To avoid overfitting, we set the minimum number of records in a terminal node to 50(in R : minbucket=50). Finally, we plot the decision tree:

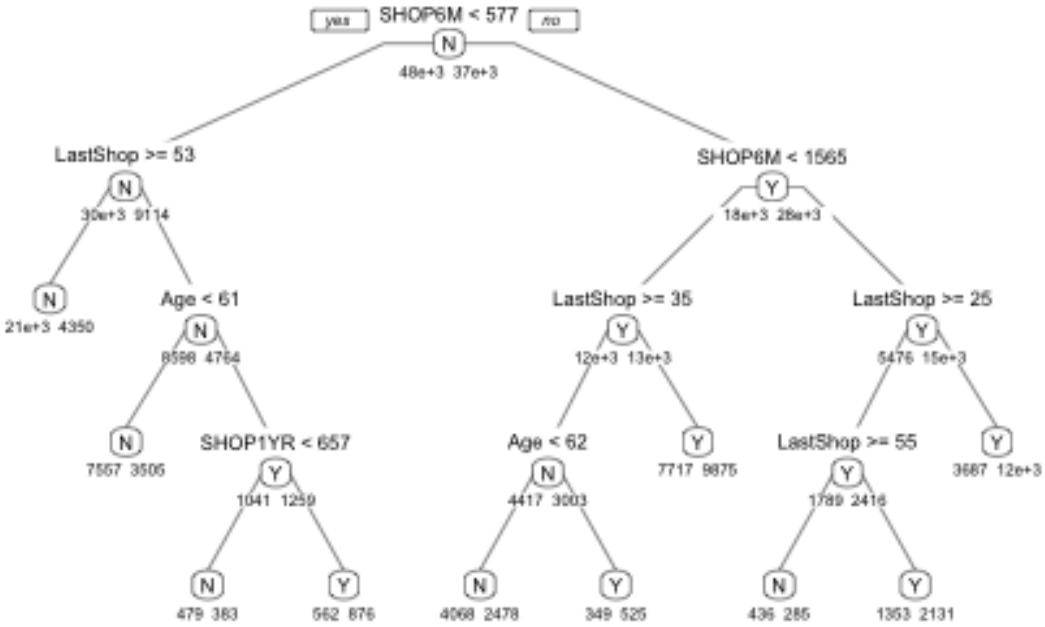

figure 3.2.1 Decision Tree

From the graph, we can see:

1. Among all the predictors, SHOP6M is the most important variable to predict customer churn or not churn.
2. For other variables, LastShop(the time from last shop), Age and SHOP1YR are important factors that influence customer churn or not churn. When we do predict, we could pay more attention to these factors.

3. Also as the graph shows, if a person has not shopped for 53 days and his total consumption for 6 months is less than 577, he would not renew.

2.2 Decision Tree Performance Evaluation

We are using all the variables to produce confusion matrix tables and make predictions. By using the testing data, we got:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     N     Y
##          N 14617  4889
##          Y  5885 10744
##
##                Accuracy : 0.7018
##                  95% CI : (0.6971, 0.7066)
##     No Information Rate : 0.5674
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3972
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7130
##             Specificity : 0.6873
##          Pos Pred Value : 0.7494
##          Neg Pred Value : 0.6461
##              Prevalence : 0.5674
##          Detection Rate : 0.4045
##    Detection Prevalence : 0.5398
##       Balanced Accuracy : 0.7001
##
##        'Positive' Class : N
```

The accuracy of this simple tree is not high, it is just 0.7018. Actually, even though we change some rpart control for the tree, the accuracy is still not high and is around 0.71. So, what we want to see next is whether a deeper tree could improve the accuracy.

2.3 Deeper Tree and Deeper Tree Evaluation

Since the deeper tree is too complex to plot , we are not going to do the visualization. We will show you the confusion matrix for the deeper tree:

```
## [1] 14178
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction     N     Y
##          N 14735  5668
##          Y  5767  9965
##
##                Accuracy : 0.6835
##                  95% CI : (0.6787, 0.6883)
##     No Information Rate : 0.5674
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.3559
##
##  Mcnemar's Test P-Value : 0.3594
##
##             Sensitivity : 0.7187
##             Specificity : 0.6374
##          Pos Pred Value : 0.7222
##          Neg Pred Value : 0.6334
##              Prevalence : 0.5674
##          Detection Rate : 0.4078
##    Detection Prevalence : 0.5646
##       Balanced Accuracy : 0.6781
##
##        'Positive' Class : N
##
```

As the results show, there are 14178 leaves in this tree. The accuracy of this deeper tree is 0.6835 which is less than the simple tree. The accuracy has hardly improved, let's see if we can do better using Random Forest.

3 Neural Network

3.1 Data Processing

To begin with, we find that after processing our association rules (which is not covered in this project), we now have the variables Renew, ExecMemb, Region, HomeWh, and RecentMove as characters, and the other variables as doubles. The "recipes" package will be used to fix the data type issues and create design matrices based on our current dataset in order to prepare for neural network modeling.

3.2 Neural Network Modeling

After we set our seed to 100 and split our dataset into 80% for training and 20% for testing, we begin to do the preprocessing and run our neural network model. The data recipe we created is shown below.

```
## Data Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         20
##
## Training data contained 96361 data points and no missing data.
##
## Operations:
##
## Log transformation on Dist [trained]
## Dummy variables from ExecMemb, Region, HomeWh, RecentMove [trained]
## Centering for AccType, BsnType, WhNo, Age, Tenure, ... [trained]
## Scaling for AccType, BsnType, WhNo, Age, Tenure, ... [trained]
```

figure 3.3.1 Data Recipe

We do a log transformation on the Dist variable, and create dummy variables for the categorical variables. There is no missing value. We use the bake( ) function to take our trained data recipe and apply the operations to our dataset to create our design matrix. In this project, we use the "keras" package to make our neural network model. The training/validation history of our keras model is shown below.

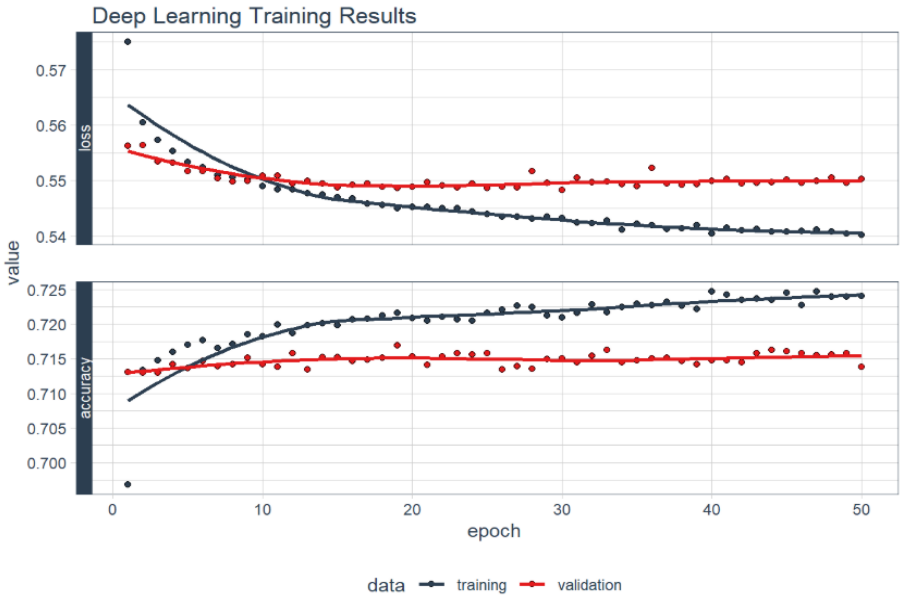

figure 3.3.2 NN Model

We can find the accuracy after the final training is not exceeding 72.4% , which is not too bad.

3.3 Neural Network Performance Evaluation

The AUC value is also used to evaluate the performance of our neural network model. The result is shown below.

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.791
```

figure 3.3.3 NN Model AUC Value

We can see the AUC score is 0.791, which is greater than 0.5. This means the test quality is good.

4 Random Forest

4.1 Random Forest Modeling

```
Call:
 randomForest(formula = Renew ~ ., data = train, importance = T)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 23.59%
Confusion matrix:
      N     Y class.error
N 41493  9510   0.1864596
Y 11805 27531   0.3001068
```

figure 3.4.1 Random Forest Model

The error rate is relatively low when predicting “No”, and the error rate is much higher when predicting “Yes”.

4.2 Random Forest Performance Evaluation

```
Confusion Matrix and Statistics

          Reference
Prediction     N     Y
         N 13760  3939
         Y  3240  9172

               Accuracy : 0.7616
                 95% CI : (0.7567, 0.7664)
    No Information Rate : 0.5646
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5121

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6996
            Specificity : 0.8094
         Pos Pred Value : 0.7390
         Neg Pred Value : 0.7774
             Prevalence : 0.4354
         Detection Rate : 0.3046
   Detection Prevalence : 0.4122
      Balanced Accuracy : 0.7545

       'Positive' Class : Y

       F1
0.7931064
```

figure 3.4.2 Random Forest Confusion Matrix

The accuracy of the model is 76.16%, the true positive rate is 69.96%, and the true negative rate is 80.94%. It looks like it is higher than logistic regression and decision tree. The ROC curve and AUC values are shown below.

ROC curve:

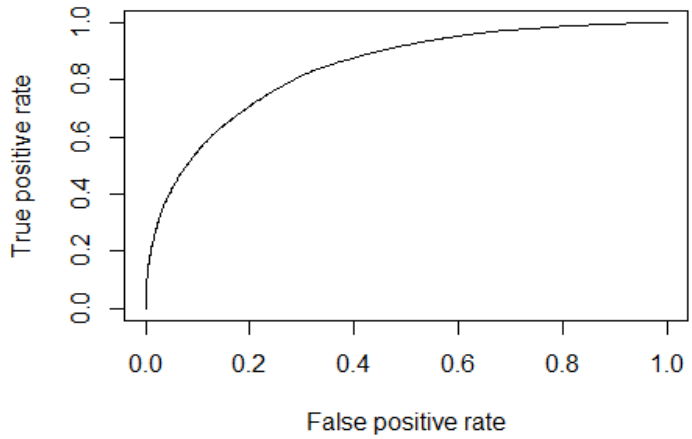

figure 3.4.3 Random Forest ROC curve

AUC value:

```
## [1] 0.8417671
```

The AUC value is 0.8414, which is higher than the logistic regression model.

Now, let's look at the feature importance of variables in random forest:

	N	Y	MeanDecreaseAccuracy	MeanDecreaseGini
AccType	25.859906	20.847562	30.810238	160.937051
ExecMemb	97.755844	79.753818	97.194125	722.221301
BsnType	24.348230	18.411033	29.402077	131.228233
Region	96.550290	95.465949	141.152437	2089.915782
WhNo	98.240298	82.621792	122.739919	3028.484087
Age	126.214873	151.791842	171.805469	3484.993642
Tenure	4.128989	1.564425	4.193639	3.805641
Zip	87.775949	78.435193	113.579291	3154.738308
MemCount	32.996196	37.199474	40.187926	457.019819
Dist	87.170048	99.007857	122.347592	3339.054341
LastShop	70.416244	124.869569	132.191852	4696.723704
HomeWh	44.386630	33.643705	44.942215	402.812328
RecentMove	-24.341300	-20.315381	-29.415356	198.306401
SHOP1YR	95.760386	74.546134	126.501327	4975.852988
SHOP6M	59.556901	84.022900	115.265582	5765.478105
SHOP3M	42.739812	93.570484	89.824004	5208.059517
ECOMSHOP	49.255332	46.886630	52.614756	736.466590
GASSHOP	49.085138	57.892365	58.348220	1539.269605
MEDICALSHOP	59.370121	57.810298	63.845080	1853.415467
GROCERYSHOP	76.423269	75.389085	103.304649	2090.050170

figure 3.4.4 RF Feature Importance

This plot will help us learn better:

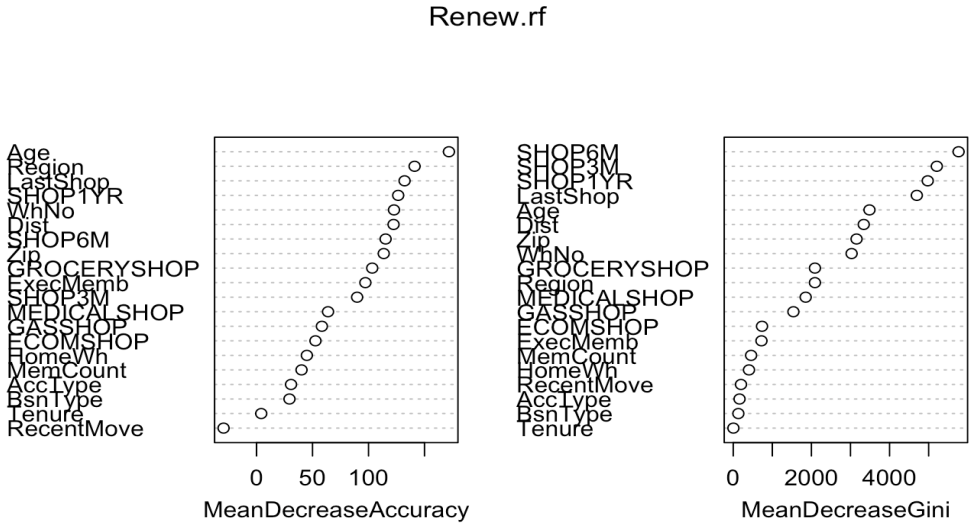

figure 3.4.5 RF Feature Importance

The two measures of feature importance are Mean Decrease in Accuracy and Mean Decrease in Gini. From the graph we can see that SHOP6M, LastShop and Age are important features that influence our prediction. This result is similar with decision trees.

4.3 Random Forest Parameter Tuning

Next, we will try to change some of the parameters of the random forest model to see if it improves the accuracy.

First, change the number of variables:

```
Confusion Matrix and Statistics

          Reference
Prediction     N     Y
         N 13702  3959
         Y  3298  9152

               Accuracy : 0.759
                 95% CI : (0.7541, 0.7638)
    No Information Rate : 0.5646
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.507

 Mcnemar's Test P-Value : 9.367e-15

            Sensitivity : 0.6980
            Specificity : 0.8060
         Pos Pred Value : 0.7351
         Neg Pred Value : 0.7758
             Prevalence : 0.4354
         Detection Rate : 0.3039
   Detection Prevalence : 0.4135
      Balanced Accuracy : 0.7520

       'Positive' Class : Y

       F1
0.7906292
```

figure 3.4.6 RF(mtry=20) confusion matrix

We have set the mtry value equals to 8, 12, 16 and 20, and the accuracy of each model is 0.7573, 0.757, 0.756 and 0.759. Here we just show the result of 20. The accuracy of the model(mtry=20) is 0.759, which is not improved. And the AUC value has also decreased slightly. We can see clearly that changing the number of variables could not improve our model.

Next, we'll try changing the number of trees:

```
Confusion matrix:
      N     Y class.error
N 41493  9510   0.1864596
Y 11925 27411   0.3031574
Confusion Matrix and Statistics

          Reference
Prediction     N     Y
         N 13783  3953
         Y  3217  9158

               Accuracy : 0.7619
                 95% CI : (0.757, 0.7667)
    No Information Rate : 0.5646
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5126

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6985
            Specificity : 0.8108
         Pos Pred Value : 0.7400
         Neg Pred Value : 0.7771
             Prevalence : 0.4354
         Detection Rate : 0.3041
   Detection Prevalence : 0.4110
      Balanced Accuracy : 0.7546

       'Positive' Class : Y

       F1
0.7935859
```

figure 3.4.7 RF(ntree = 250) confusion matrix

We have set the number of trees equal to 25, 250,500 and 750. The highest accuracy is 0.7619 with the number of trees equals to 250, which is slightly increased compared to our original model. So actually, from all the models we have created, the accuracy is around 0.76 no matter how we change the parameters and could not be improved significantly. Let's look at the ROC curve and the feature importance of variables when ntree=250:

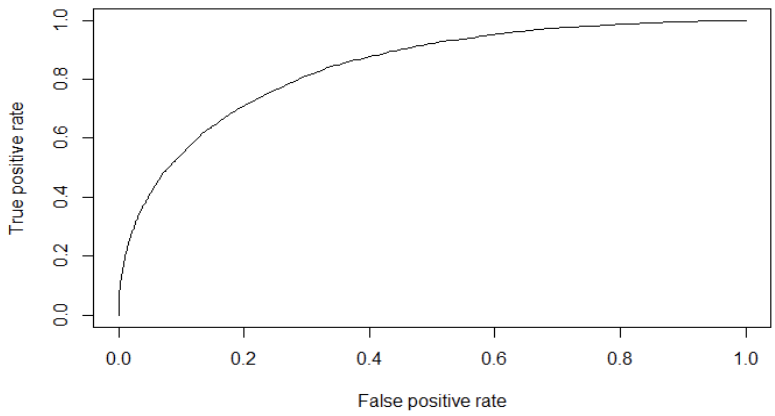

figure 3.4.8 RF(ntree=250) ROC curve

The feature importance of different variables:

	N	Y	MeanDecreaseAccuracy	MeanDecreaseGini
BsnType	27.028868	20.509941	35.672691	92.47685
Region	112.075396	112.477437	160.250762	2070.92285
WhNo	115.913778	99.070813	143.584896	2957.32145
Age	149.998314	176.071805	200.838675	3416.80834
Tenure	5.373861	3.331207	6.009646	3.68634
Zip	109.201679	92.568226	146.451485	3079.76778
MemCount	42.542907	43.292607	48.808254	447.99754
Dist	108.713617	119.380179	149.887542	3244.37303
LastShop	89.572384	149.283444	172.182707	4638.73962
HomeWh	53.983101	44.073407	55.281622	404.77894
RecentMove	-31.816678	-25.771427	-38.670406	206.35753
SHOP1YR	114.945377	91.385736	153.583055	4873.56290
SHOP6M	76.299630	100.105022	137.306588	5720.70351
SHOP3M	50.285390	113.971202	109.361656	4884.63585
ECOMSHOP	65.342439	57.149446	69.574572	736.07024
GASSHOP	58.147321	67.795084	69.468560	1532.90765
MEDICALSHOP	70.922881	69.458179	76.630395	1893.35664
GROCERYSHOP	92.799597	95.261316	126.613964	2068.81480
MemberDataAccType1	29.949027	23.259851	34.795418	127.45308
MemberDataAccType2	32.104207	20.607750	34.073229	126.20997
MemberDataExecMembE	42.766897	39.945507	43.342391	433.82133
MemberDataExecMembN	42.245534	40.067590	43.058740	443.06829

figure 3.4.9 RF(ntree = 250) Feature importance

Project Results

1. Logistic Regression

From our logistic regression model, we achieved an accuracy of 68%, and a misclassification error rate of 32 percent.

The ROC curve and AUC value is also used to evaluate the performance of our logistic regression model. The plot and result are shown below.

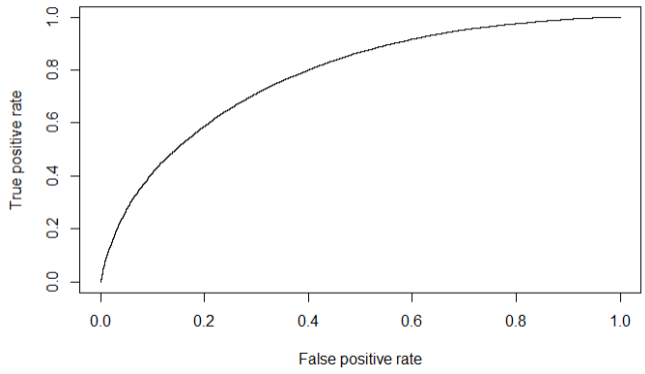

figure 4.1.1 Logistic regression ROC curve

AUC value: [1] 0.7782006

The AUC value is .78, which means while the model accuracy is not ideal, the system is doing more than just random guessing, in which case the AUC value would have been 0.5.

F1 score: [1] 0.7630378

The F1 score is .76, which also verifies our conclusion.

2. Decision Tree

While fitting the data into our Decision Tree Model, we noticed that the simpler tree with lesser leaves had greater accuracy and F1 score than the deeper tree, while the accuracies remained more or less the same at 68 percent. Below is the F1-score comparison between the simpler tree and the deeper tree.

The F1 score for deeper tree:

```
       F1
0.7204498
```

figure 4.2.1 DT F1 Score

The F1 score for simple tree :

```
       F1
0.7307039
```

figure 4.2.2 DT F1 Score

The F1 score has slightly decreased for the deeper tree, which means that the precision and recall are balanced better for the simpler tree than the deeper tree.

3. Neural Network

Our Neural Network model had a best accuracy of 73%, with 2 hidden layers of 32 Neurons each and 50 epochs. We tried increasing the number of hidden layers to 3 and the number of neurons to 64 per hidden layer, but the accuracy remained at around 71%. The number of epochs was raised to 100 but this too did not have much of an effect on the accuracy. For further scope, we could try k-fold cross-validation to increase the accuracy of the model. The correlation scores that the model calculated are shown below:

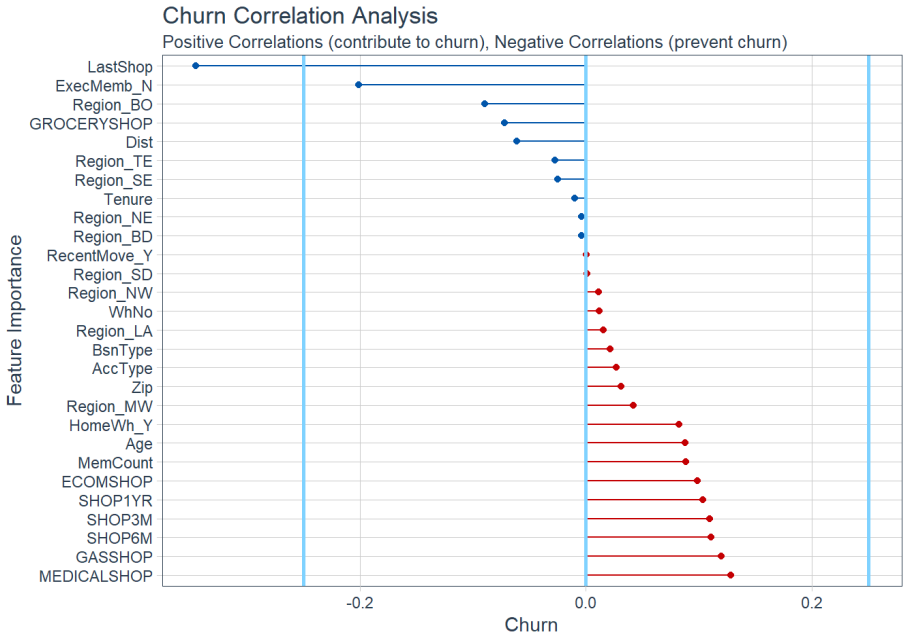

figure 4.3.1 NN Correlation Analysis

Curiously, we see that SHOP1YR, SHOP3M etc have a negative effect on the churn.

The precision and recall are shown below:

.metric <chr>	.estimator <chr>	.estimate <dbl>
precision	binary	0.7420465

figure 4.3.2 NN precision

.metric <chr>	.estimator <chr>	.estimate <dbl>
recall	binary	0.775088

figure 4.3.3 NN recall

The precision value is 74.2%, which means that 74.2% of our results are relevant. We can see that the recall value is 77.5% which means it is accurately classifying 77.5 of the total relevant results.

4. Random Forest

We achieved the highest accuracy of all our models with random forest. We achieved an accuracy of 76%. Here's a look at the feature importances that the model computed:

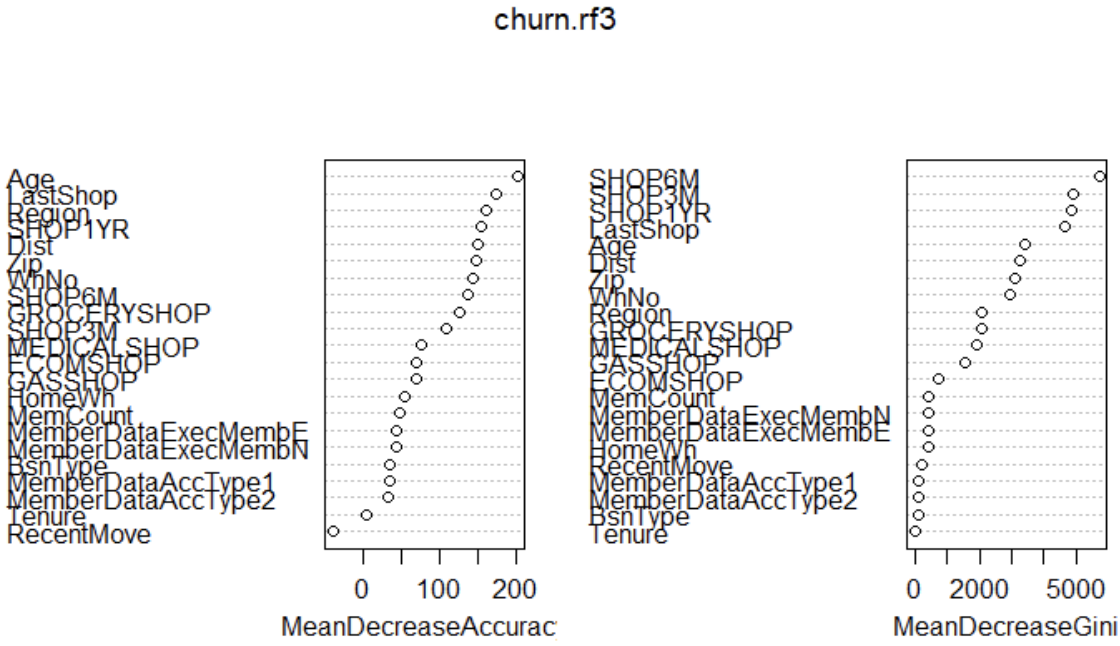

figure 4.4.1 RF Feature importance

The most important variables are SHOP6M, SHOP1YR, Age, and LastShop. The F1-score is shown below.

```
##   F1 0.79356
```

This shows that the Random Forest Model has high Precision and Recall, making it a highly viable model for predicting churn.

Impact of the Project Outcomes

Customer churn is a costly problem. The good news is that machine learning can solve churn problems, making the organization more profitable in the process**.** We have implemented 4 basic classification models without too much cost and got results that are viable, and with much scope for further improvement. From across all the models, we can safely conclude that “LastShop”, “SHOP6M”, “SHOP1YR” and “Dist” are the features that were the most useful in diagnosing member churn.

If customers have not shopped for a while, they seem to be more reluctant to renew their membership. Costco can come up with incentives or additional discounted prices exclusively for people who have not shopped for a while to induce them into shopping more. The lesser the shopping and the more days the customer goes without shopping, there is more possibility of not renewing. Frequently asking for feedback from non-frequenting customers would also help prevent the churn.