Random Forests on Telecom Churn Data

Alexander Vaillant

9/7/2021

Environment Setup

Load Libraries

```
library(caret) # GridSearch?
```

```
## Warning: package 'caret' was built under R version 4.1.1
## Loading required package: lattice
## Loading required package: ggplot2
```

```
library(caTools) # AUC
```

```
## Warning: package 'caTools' was built under R version 4.1.1
```

```
library(ranger) #RandomForest
```

```
## Warning: package 'ranger' was built under R version 4.1.1
```

Data Gathering

```
# Import the raw dataset
url <- "C:/Users/tedda/Desktop/Data Science Portfolio/Machine Learning/Supervised Learning/Classificati
churn_data <- read.csv(url, header = TRUE)
```

Data Preparation

```
# Remove customer demographics by indexing
churn_dummies <- churn_data[20:50]

# Export entire prepped dataset
write.csv(churn_dummies, "C:/Users/tedda/Desktop/Data Science Portfolio/Machine Learning/Supervised Lea

# Set seed for random sampling of data
set.seed(111)

# Create the index for the training dataset
sample_size <- round(0.8*nrow(churn_dummies))
train_ind <- sample(1:nrow(churn_dummies), size = sample_size)

# Split the training and testing datasets from the prepped dataset
churn_train <- churn_dummies[train_ind,]
write.csv(churn_train, "C:/Users/tedda/Desktop/Data Science Portfolio/Machine Learning/Supervised Learn
```

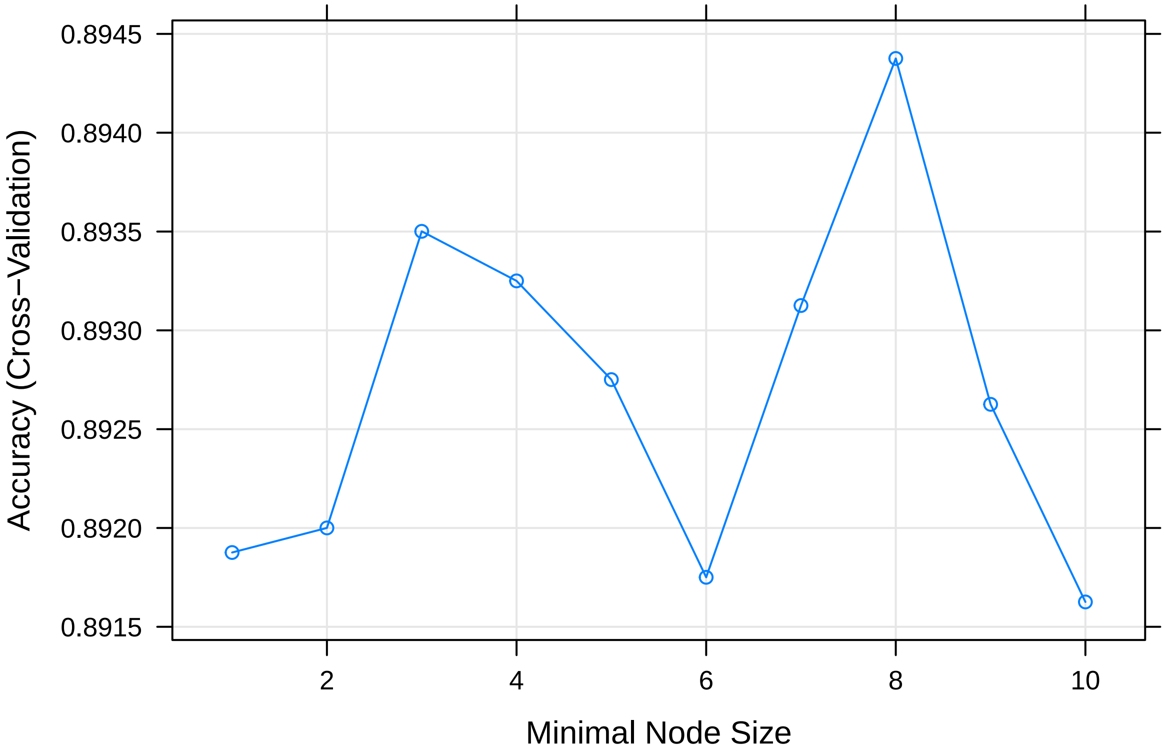
```r
churn_test <- churn_dummies[-train_ind,]
write.csv(churn_test, "C:/Users/tedda/Desktop/Data Science Portfolio/Machine Learning/Supervised Learni

# Create the "actual" results datasets for both the training and test datasets
churn_train_actual <- churn_train[,'Churn']
churn_test_actual <- churn_test[,'Churn']
```

Model Building

```r
# Build the model and hyperparameter tuning grid
train_ctrl <- trainControl(method = "cv", number = 5, classProbs = TRUE, verboseIter = TRUE)
tuneGrid <- data.frame(.mtry = sqrt(30),.splitrule = c("gini"), .min.node.size = c(1:10))
rfc_fit <- train(x = churn_train[-1], y = churn_train_actual, method = "ranger", metric = c("Accuracy")
                 tuneGrid = tuneGrid, trControl = train_ctrl)
```

```
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 1
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 1
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 2
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 2
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 3
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 3
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 4
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 4
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 5
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 5
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 6
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 6
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 7
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 7
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 8
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 8
## + Fold1: mtry=5.477, splitrule=gini, min.node.size= 9
## - Fold1: mtry=5.477, splitrule=gini, min.node.size= 9
## + Fold1: mtry=5.477, splitrule=gini, min.node.size=10
## - Fold1: mtry=5.477, splitrule=gini, min.node.size=10
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 1
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 1
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 2
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 2
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 3
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 3
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 4
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 4
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 5
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 5
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 6
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 6
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 7
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 7
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 8
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 8
## + Fold2: mtry=5.477, splitrule=gini, min.node.size= 9
## - Fold2: mtry=5.477, splitrule=gini, min.node.size= 9
## + Fold2: mtry=5.477, splitrule=gini, min.node.size=10
## - Fold2: mtry=5.477, splitrule=gini, min.node.size=10
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 1
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 1
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 2
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 2
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 3
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 3
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 4
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 4
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 5
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 5
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 6
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 6
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 7
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 7
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 8
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 8
## + Fold3: mtry=5.477, splitrule=gini, min.node.size= 9
## - Fold3: mtry=5.477, splitrule=gini, min.node.size= 9
## + Fold3: mtry=5.477, splitrule=gini, min.node.size=10
## - Fold3: mtry=5.477, splitrule=gini, min.node.size=10
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 1
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 1
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 2
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 2
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 3
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 3
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 4
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 4
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 5
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 5
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 6
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 6
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 7
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 7
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 8
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 8
## + Fold4: mtry=5.477, splitrule=gini, min.node.size= 9
## - Fold4: mtry=5.477, splitrule=gini, min.node.size= 9
## + Fold4: mtry=5.477, splitrule=gini, min.node.size=10
## - Fold4: mtry=5.477, splitrule=gini, min.node.size=10
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 1
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 1
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 2
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 2
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 3
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 3
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 4
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 4
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 5
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 5
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 6
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 6
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 7
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 7
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 8
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 8
## + Fold5: mtry=5.477, splitrule=gini, min.node.size= 9
## - Fold5: mtry=5.477, splitrule=gini, min.node.size= 9
## + Fold5: mtry=5.477, splitrule=gini, min.node.size=10
## - Fold5: mtry=5.477, splitrule=gini, min.node.size=10
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 5.48, splitrule = gini, min.node.size = 8 on full training set
```

```r
# Plot the finished model to show Accuracy of each min.node.size
plot(rfc_fit)
```



Save and Load Model

```r
# Save and Load the Model
model_url <- "C:/Users/tedda/Desktop/Data Science Portfolio/Machine Learning/Supervised Learning/Classi
saveRDS(rfc_fit, model_url)
rfc_model <- readRDS(model_url)
```

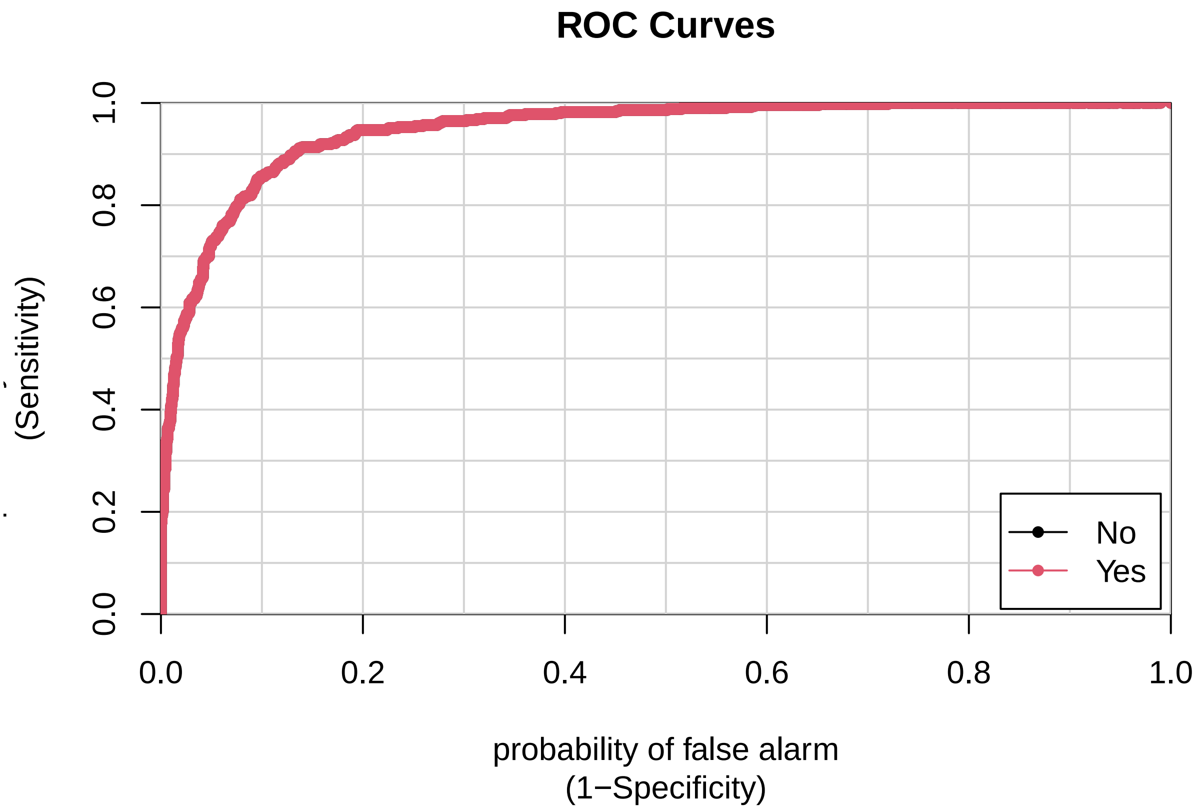
Model Evaluation

```r
# Create a confusion matrix to show the Accuracy and other metrics of our final model
pred <- predict(rfc_model, newdata = churn_test[-1])
```

```r
confusionMatrix(pred,as.factor(churn_test$Churn))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1412  137
##        Yes   79  372
##
##                Accuracy : 0.892
##                  95% CI : (0.8776, 0.9053)
##     No Information Rate : 0.7455
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7043
##
##  Mcnemar's Test P-Value : 0.0001052
##
##             Sensitivity : 0.9470
##             Specificity : 0.7308
##          Pos Pred Value : 0.9116
##          Neg Pred Value : 0.8248
##              Prevalence : 0.7455
##          Detection Rate : 0.7060
##    Detection Prevalence : 0.7745
##       Balanced Accuracy : 0.8389
##
##        'Positive' Class : No
##
```

```r
# Plot the AUC of our final model
pred_ROC <- predict(rfc_model, newdata = churn_test[-1], type = "prob")
colAUC(X = pred_ROC, y = churn_test_actual, plotROC = TRUE)
```



```
##                      No      Yes
## No vs. Yes 0.9474443 0.947445
```

```
# Print the importance of the top 20 variables in our model
varImp(rfc_model)
```

```
## ranger variable importance
##
##   only 20 most important variables shown (out of 30)
##
##                      Overall
## Tenure               100.000
## Bandwidth_GB_Year     83.487
## MonthlyCharge         67.955
## Contract              44.135
## StreamingMovies       29.005
## StreamingTV           18.192
## Outage_sec_perweek    15.368
## Email                 10.359
## InternetService        8.076
## Multiple               5.719
## Item5                  5.116
## Item4                  5.032
## Item7                  4.933
## Item8                  4.836
## Item3                  4.579
## Item2                  4.503
## Item1                4.404
## Item6                4.244
## PaymentMethod        4.099
## Contacts             3.854
```