Market Analysis - Predicting Customer Churn

Godfred Somua-Gyimah

January 20, 2017

Contents

Read in Dataset

```r
# Clean the environment
rm(list = ls())

# Read data file
df <- read.csv("F:/Data_Science_Files/Github/Market_Analysis-Predicting_Customer_Churn/Telco-Customer-Cl
```

Explore Data

```r
# Show the head of the dataset
head(df)
```

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

```r
# Show the structure of the dataset
str(df)
```

```
## 'data.frame':    7043 obs. of  21 variables:
##  $ CustomerID      : Factor w/ 7043 levels "0002-ORFBO","0003-MKNFE",..: 5376 3963 2565 5536 6512 65
##  $ 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 ...
```

```
# Summary statistics
summary(df)
```

```
##       CustomerID      Gender     SeniorCitizen    Partner    Dependents
##  0002-ORFBO:   1   Female:3488   Min.   :0.0000   No :3641   No :4933
##  0003-MKNFE:   1   Male  :3555   1st Qu.:0.0000   Yes:3402   Yes:2110
##  0004-TLHLJ:   1                 Median :0.0000
##  0011-IGKFF:   1                 Mean   :0.1621
##  0013-EXCHZ:   1                 3rd Qu.:0.0000
##  0013-MHZWF:   1                 Max.   :1.0000
##  (Other)   :7037
##      Tenure      PhoneService          MultipleLines     InternetService
##  Min.   : 0.00   No : 682     No              :3390   DSL        :2421
##  1st Qu.: 9.00   Yes:6361     No phone service: 682   Fiber optic:3096
##  Median :29.00                Yes             :2971   No         :1526
##  Mean   :32.37
##  3rd Qu.:55.00
##  Max.   :72.00
##
##              OnlineSecurity              OnlineBackup
##  No                 :3498   No                 :3088
##  No internet service:1526   No internet service:1526
##  Yes                :2019   Yes                :2429
##
##
##
##
##             DeviceProtection               TechSupport
##  No                 :3095    No                 :3473
##  No internet service:1526    No internet service:1526
##  Yes                :2422    Yes                :2044
##
```

```
##
##
##
##              StreamingTV              StreamingMovies
##  No                 :2810   No                 :2785
##  No internet service:1526   No internet service:1526
##  Yes                :2707   Yes                :2732
##
##
##
##
##            Contract    PaperlessBilling                   PaymentMethod
##  Month-to-month:3875   No :2872         Bank transfer (automatic):1544
##  One year      :1473   Yes:4171         Credit card (automatic)  :1522
##  Two year      :1695                    Electronic check         :2365
##                                         Mailed check             :1612
##
##
##
##  MonthlyCharges    TotalCharges    Churn
##  Min.   : 18.25   Min.   :  18.8   No :5174
##  1st Qu.: 35.50   1st Qu.: 401.4   Yes:1869
##  Median : 70.35   Median :1397.5
##  Mean   : 64.76   Mean   :2283.3
##  3rd Qu.: 89.85   3rd Qu.:3794.7
##  Max.   :118.75   Max.   :8684.8
##                   NA's   :11
```

From the summary statistics, there are missing values. Therefore, removing all missing values from the dataset;

```r
# Remove NAs
df <- na.omit(df)
```

Descriptive Modeling

We will explore the data to see if some patterns are obvious

```r
library("ggplot2")
```

Predictive Modeling

```r
# Load library
library(caret)
```

```
## Loading required package: lattice
```

In this section, we explore different methods to predict customer churn

- Logistic Regression
- Support Vector Machine (SVM)
- Gradient Boosted Machine (GBM)

Data Partitioning

Using a single 80/20% split for train/test sets;

```
set.seed(100)
trainIndex <- createDataPartition(df$Churn, p = .8, list = FALSE)
head(trainIndex)
```

```
##      Resample1
## [1,]         1
## [2,]         2
## [3,]         3
## [4,]         4
## [5,]         5
## [6,]         6
```

```
train_data <- df[ trainIndex,]
test_data  <- df[-trainIndex,]
```

3.1 Train Logistic Regression

We can use the train() method in caret package to easily train a regression (prediction) or classification model. Refer to the following link for all available models supported by the train() method.

http://topepo.github.io/caret/available-models.html

We can call getModelInfo() method to get model information.

```
# Get information of the "glm"" model
# getModelInfo("glm")
```

```
## Train a logistic regression model with 10-fold cross-validation
fitControl <- trainControl(method = "cv",number = 10)

set.seed(123)
logit_fit <- train(Churn ~ ., data = df[-1],
                   trControl = fitControl,
                   method="glm", family=binomial(link='logit'))
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
print(logit_fit)
```

```
## Generalized Linear Model
##
## 7032 samples
##   19 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6329, 6329, 6328, 6328, 6330, 6329, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.805319  0.4739409
##
##
```

```r
confusionMatrix(logit_fit)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction   No  Yes
##        No  65.8 11.9
##        Yes  7.6 14.7
##
##  Accuracy (average) : 0.8053
```

Please note that in the train() function call, we need to exclude customer ID as a predictor. Since customer ID is the first column in the dataset, we use "data = df[-1]" as a parameter of the train() function call to exclude customer ID. The same approach is applied to other models.

3.2 Train Support Vector Machine

```r
## Train Support Vector Machine (Radial Basis Function Kernel) with 10-fold Cross-Validation
## data=df[-1] implies that we are omitting the CustomerID category from the train data
set.seed(123)
svmRadial_fit <- train(Churn ~ ., data = df[-1],
```

```r
                        trControl = fitControl, method = "svmRadial",
                        verbose=FALSE)
```

```
## Loading required package: kernlab

##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

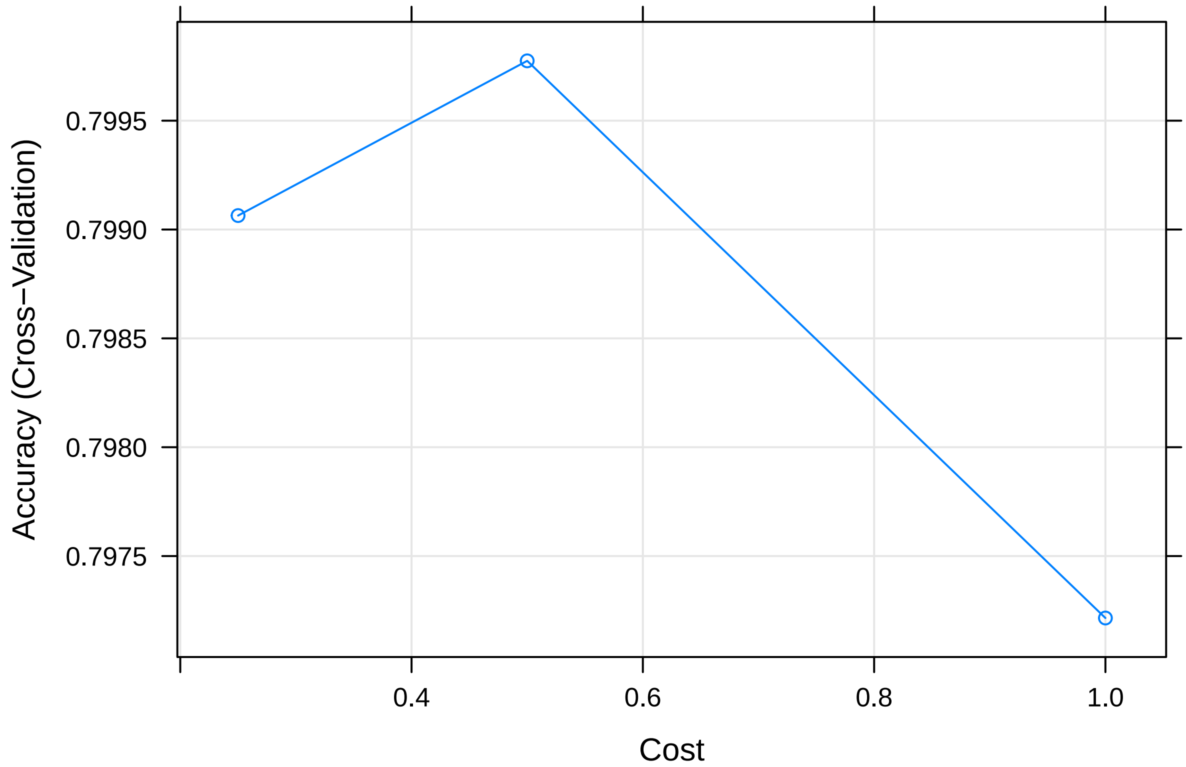
```r
print(svmRadial_fit)
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 7032 samples
##   19 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6329, 6329, 6328, 6328, 6330, 6329, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.7990642  0.4264567
##   0.50  0.7997748  0.4359961
##   1.00  0.7972154  0.4340415
##
## Tuning parameter 'sigma' was held constant at a value of 0.02379201
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.02379201 and C = 0.5.
```

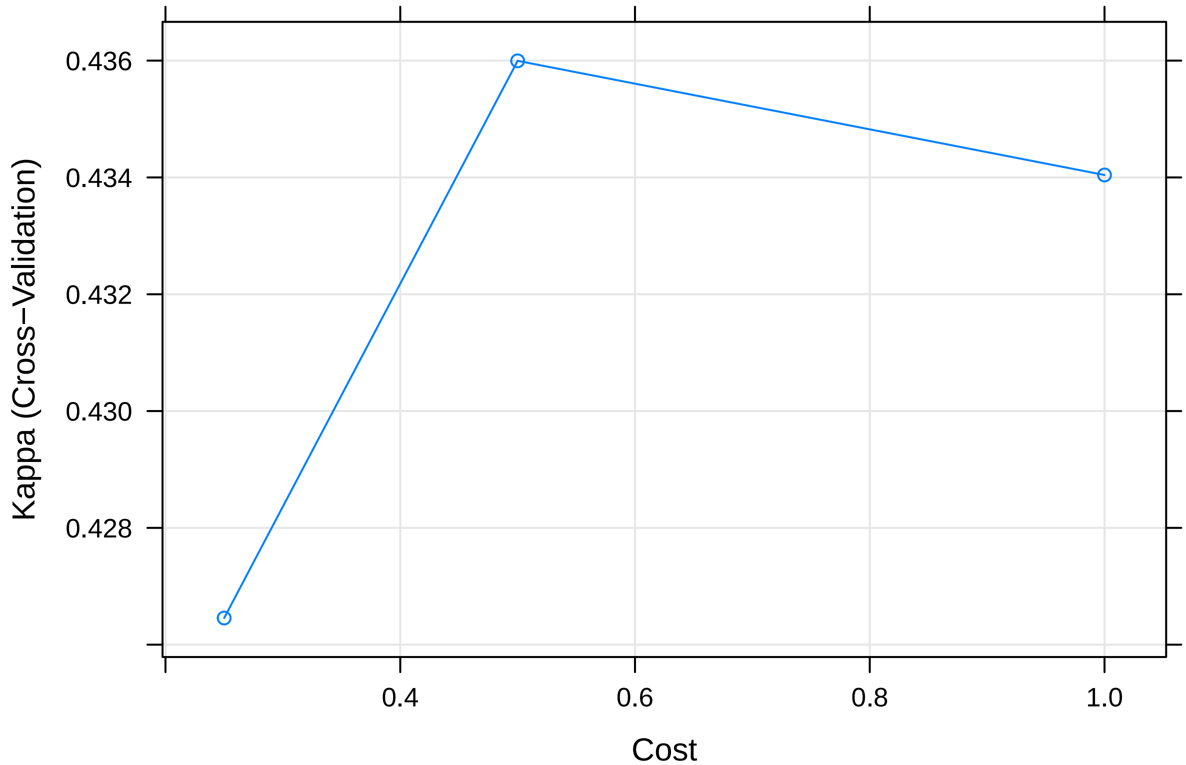
```r
confusionMatrix(svmRadial_fit)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction   No  Yes
##        No  67.2 13.8
##        Yes  6.3 12.8
##
##  Accuracy (average) : 0.7998
```

```r
# Plot resampling profile by accuracy
plot(svmRadial_fit)
```




```r
# Plot resampling profile by kappa statistic
plot(svmRadial_fit, metric = "Kappa")
```



3.3 Train Gradient Boosted Machine (GBM)

```
# Train GBM with 10-fold Cross-Validation
set.seed(123)
gbm_fit <- train(Churn ~ ., data = df[-1],
                 trControl = fitControl, method = "gbm",
                 verbose=FALSE)
```

```
## Loading required package: gbm

## Loading required package: survival

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

## Loading required package: splines

## Loading required package: parallel

## Loaded gbm 2.1.1

## Loading required package: plyr
```

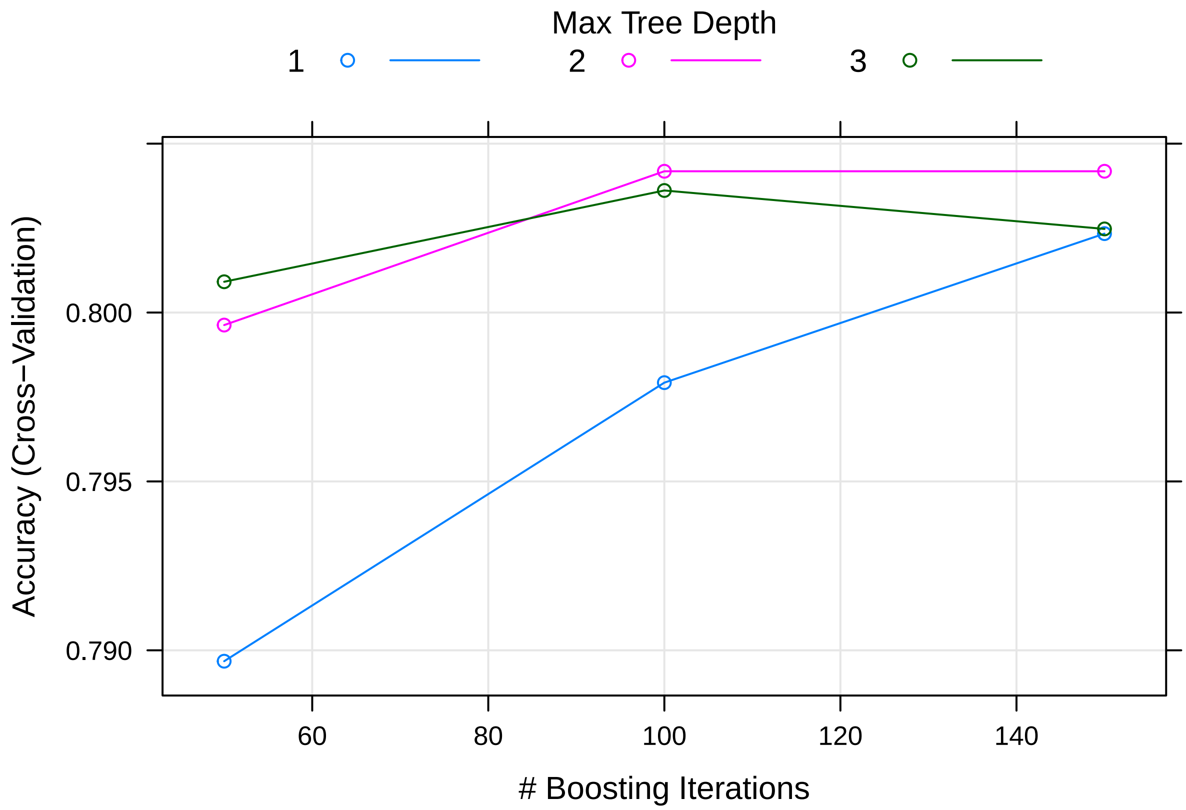
```
print(gbm_fit)
```

```
## Stochastic Gradient Boosting
##
## 7032 samples
##   19 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6329, 6329, 6328, 6328, 6330, 6329, ...
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa
##   1                   50      0.7896767  0.3641618
##   1                  100      0.7979242  0.4193485
##   1                  150      0.8023359  0.4470039
##   2                   50      0.7996291  0.4275072
##   2                  100      0.8041829  0.4584840
##   2                  150      0.8041831  0.4607798
##   3                   50      0.8009114  0.4413300
##   3                  100      0.8036133  0.4564048
##   3                  150      0.8024759  0.4563331
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 150,
##  interaction.depth = 2, shrinkage = 0.1 and n.minobsinnode = 10.
```

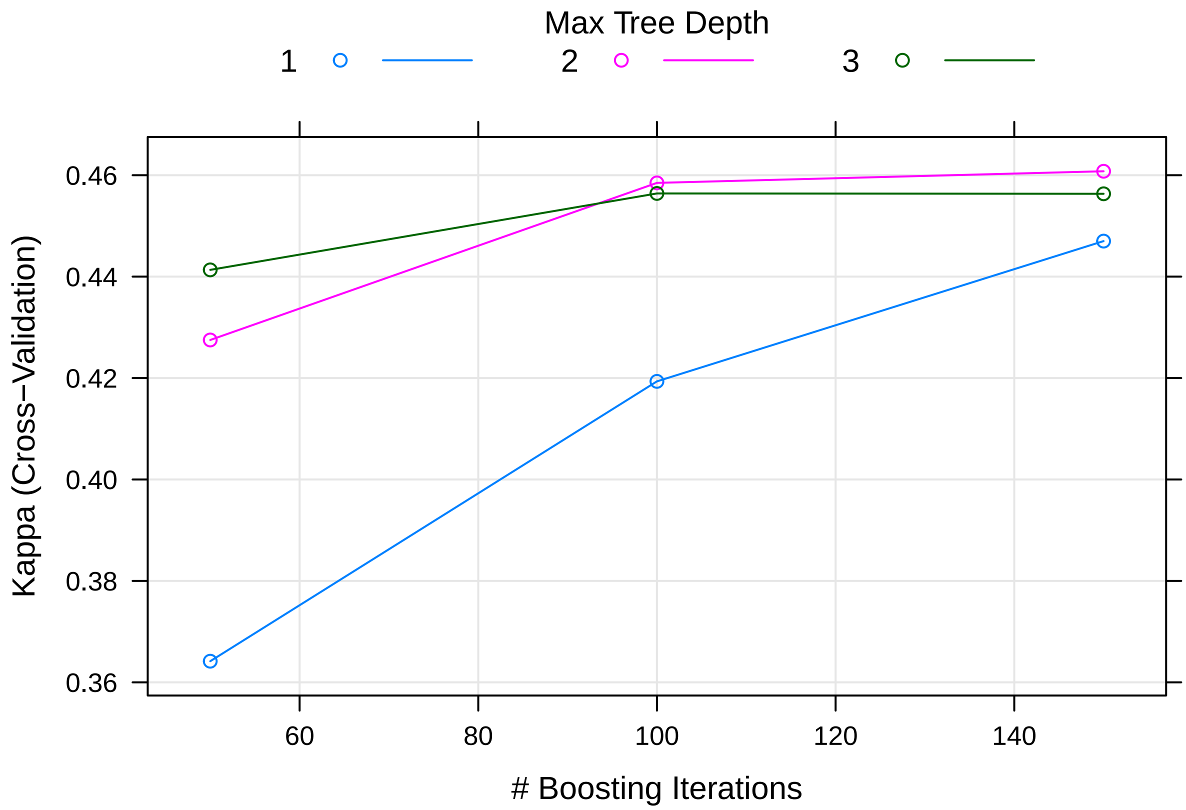
```r
confusionMatrix(gbm_fit)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction   No  Yes
##        No  66.5 12.7
##        Yes  6.9 13.9
##
##  Accuracy (average) : 0.8042
```

```r
# Plot resampling profile by accuracry
plot(gbm_fit)
```




```r
# Plot resampling profile by kappa statistic
plot(gbm_fit, metric = "Kappa")
```



3.4 Compare Different Predictive Models

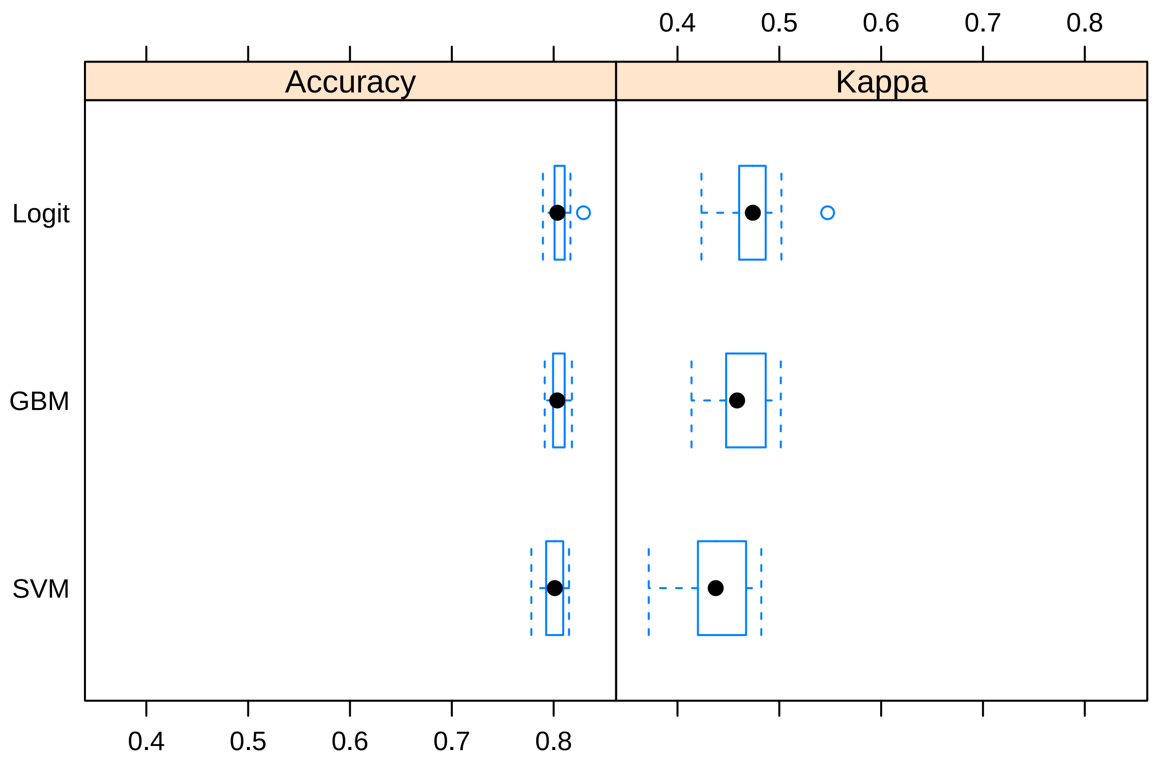
```
# Collect resamples
resamps <- resamples(list(Logit=logit_fit, SVM=svmRadial_fit, GBM = gbm_fit))
```

```
# Summarize the resamples
summary(resamps)
```

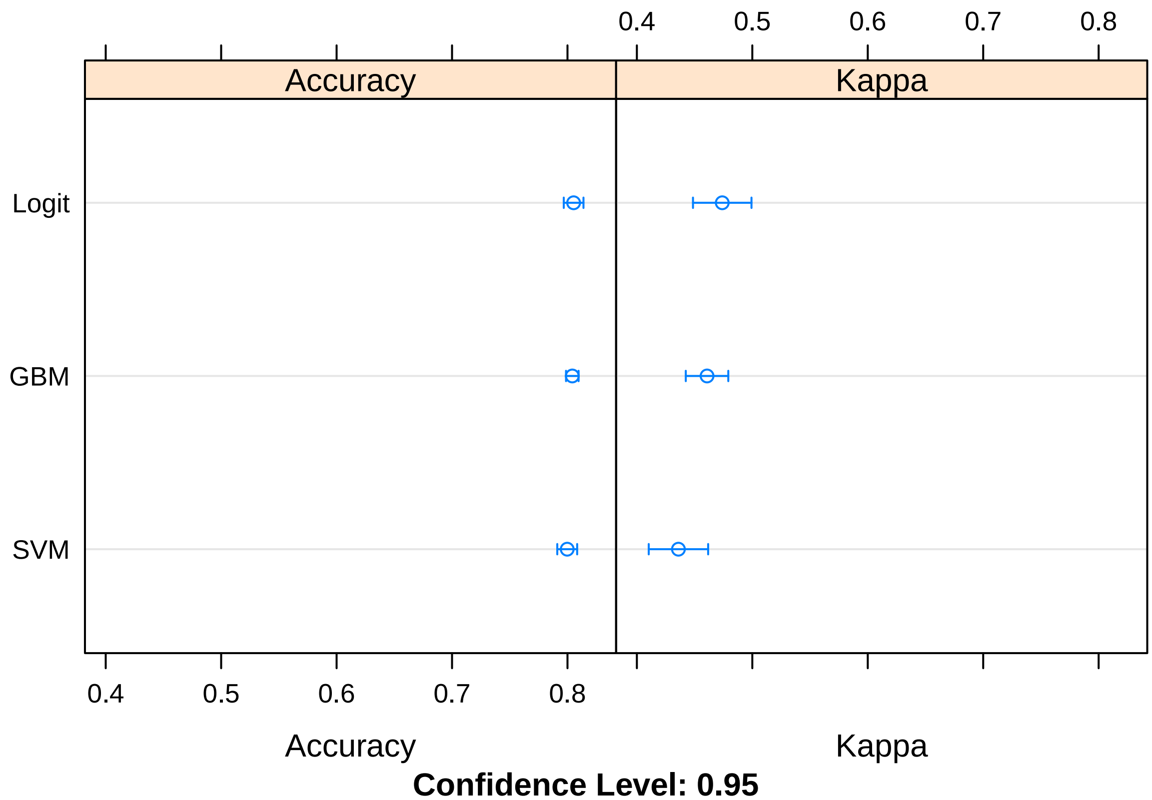
```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: Logit, SVM, GBM
## Number of resamples: 10
##
## Accuracy
##          Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## Logit 0.7895  0.8009 0.8037 0.8053  0.8102 0.8293    0
## SVM   0.7781  0.7933 0.8011 0.7998  0.8090 0.8151    0
## GBM   0.7912  0.7998 0.8036 0.8042  0.8095 0.8179    0
##
## Kappa
##          Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## Logit 0.4235  0.4606 0.4740 0.4739  0.4851 0.5474    0
## SVM   0.3717  0.4213 0.4375 0.4360  0.4656 0.4823    0
```

```
## GBM   0.4138  0.4491 0.4586 0.4608  0.4807 0.5014    0
```

```r
# Boxplots of resamples
bwplot(resamps)
```




```r
# Dot plots of resamples
dotplot(resamps)
```



Comparing the three models, we found that logistic regression model is the best since it has the highest levels of both accuracy and Kappa coefficient.

We can compute the differences between models, then use a simple t-test to evaluate the null hypothesis that there is no difference between models.

```
difValues <- diff(resamps)
difValues
```

```
##
## Call:
## diff.resamples(x = resamps)
##
## Models: Logit, SVM, GBM
## Metrics: Accuracy, Kappa
## Number of differences: 3
## p-value adjustment: bonferroni
```

```
summary(difValues)
```

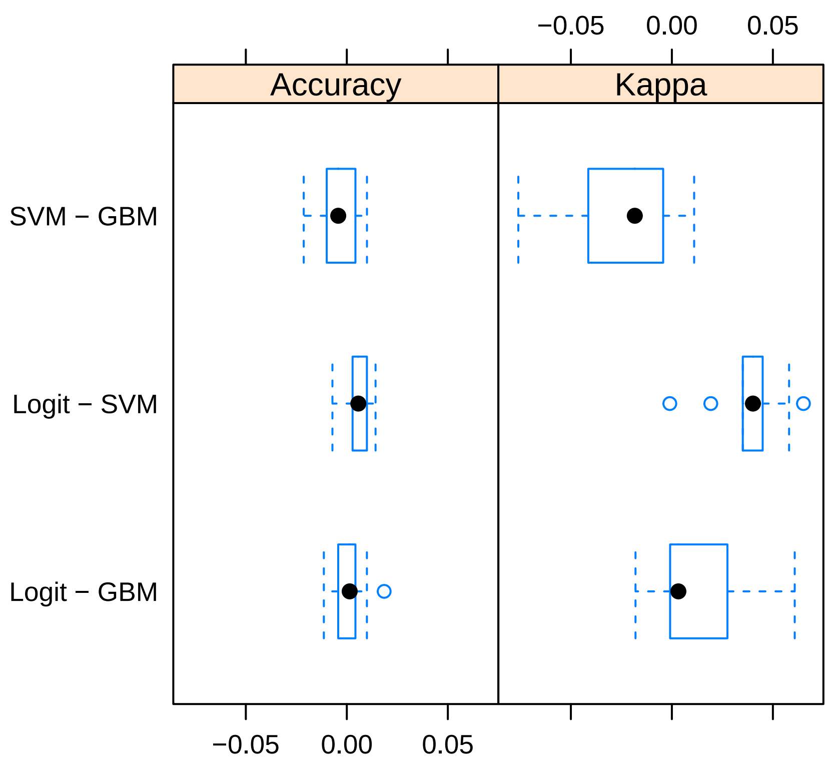
```
##
## Call:
## summary.diff.resamples(object = difValues)
##
## p-value adjustment: bonferroni
## Upper diagonal: estimates of the difference
## Lower diagonal: p-value for H0: difference = 0
##
```

```
## Accuracy
##       Logit   SVM       GBM
## Logit         0.005544  0.001136
## SVM   0.04856           -0.004408
## GBM   1.00000 0.60144
##
## Kappa
##       Logit     SVM       GBM
## Logit           0.03794   0.01316
## SVM   0.0003456           -0.02478
## GBM   0.4300187 0.0542964
```

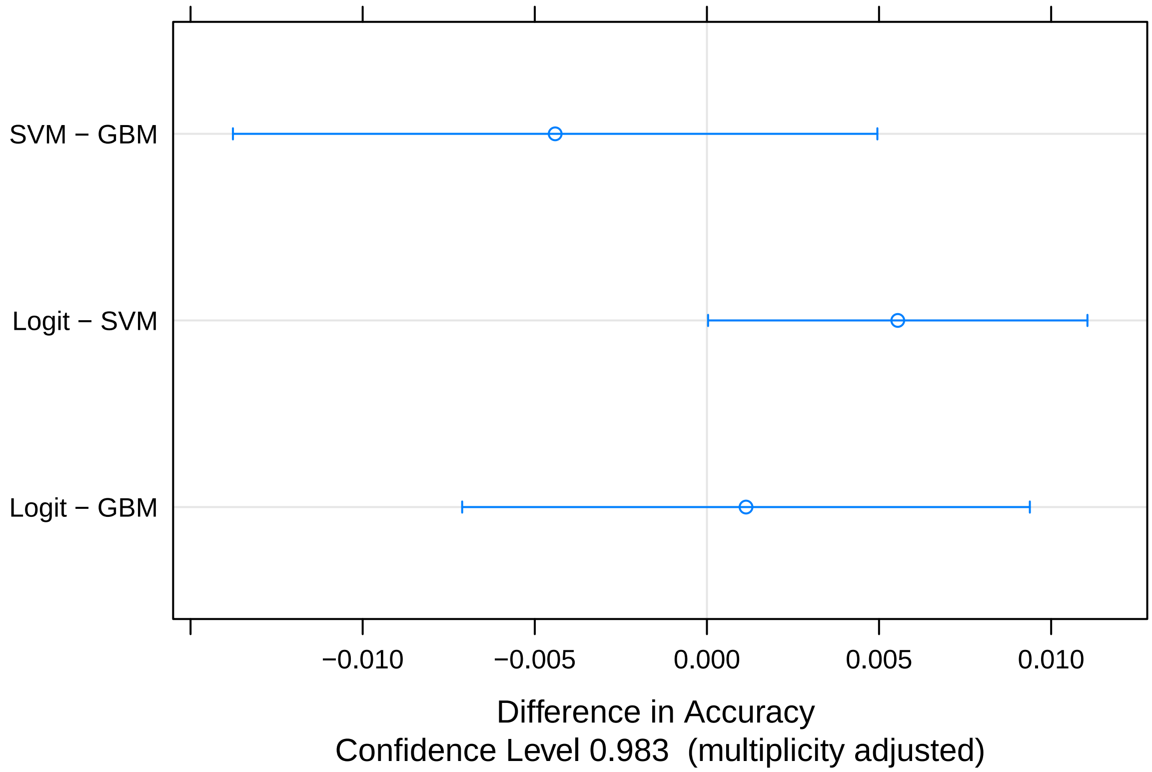
From the above hypothesis test, we can conclude that logit model has better performance than SVM in terms of prediction accuracy and inter-rater agreement (p-value < 0.05). The difference between logit and GBM is not statistically significant.

We can also plot the difference between models.

```
bwplot(difValues, layout = c(3, 1))
```




```
dotplot(difValues)
```


SVM − GBM
Logit − SVM
Logit − GBM
−0.010
−0.005
0.000
0.005
0.010
Difference in Accuracy
Confidence Level 0.983 (multiplicity adjusted)