homework2_final.R

justinvarghese

2024-10-27

```
#Final model after multiple training and validation cycles
# Load libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(rpart)
library(class)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(readr)
library(rpart.plot)
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```r
library(e1071)
library(writexl)
set.seed(4)

# Import data
data <- read_csv("/Users/justinvarghese/Downloads/XYZData.csv", col_names = TRUE)
```

```
## Rows: 41540 Columns: 27
```

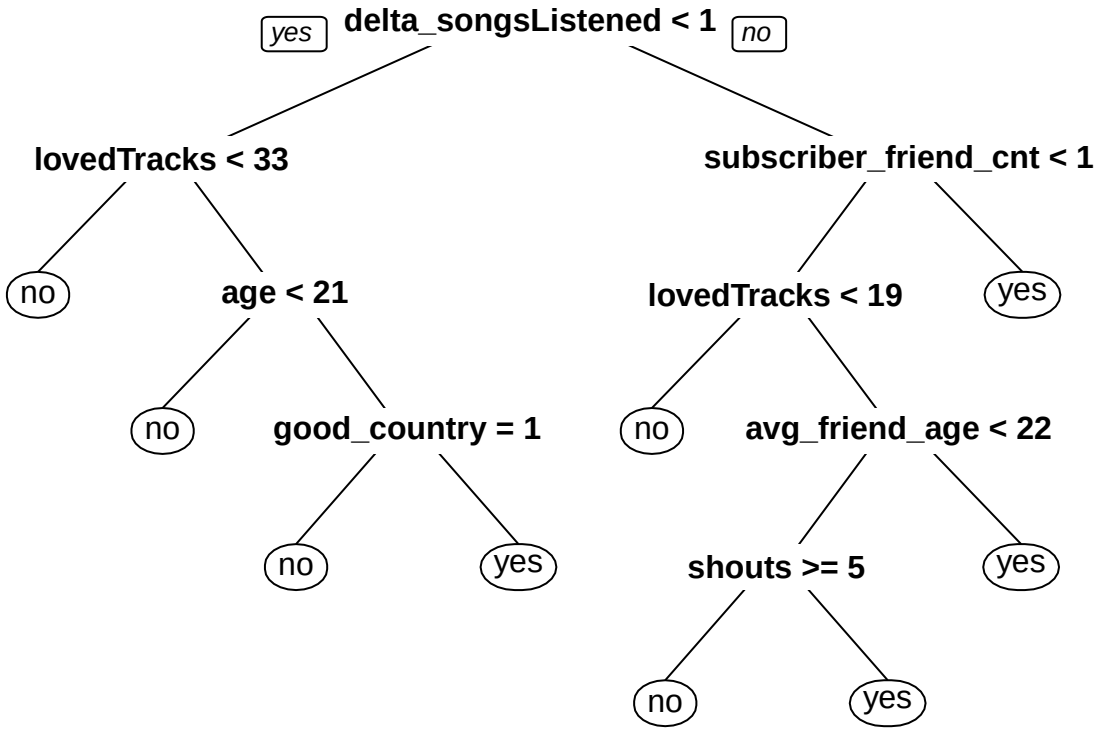
```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (27): user_id, age, male, friend_cnt, avg_friend_age, avg_friend_male, f...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Rename factor levels of adopter for compatibility
data$adopter <- factor(data$adopter, levels = c(0, 1), labels = c("no", "yes"))

# Balance the data using both under- and over-sampling with ROSE
data_both <- ovun.sample(adopter ~ ., data = data, method = "both", p = 0.5, nrow(data))$data

# Set up cross-validation with 10 folds
train_control <- trainControl(method = "cv", number = 10, classProbs = TRUE,
                              summaryFunction = twoClassSummary, savePredictions = "final")
tune_grid <- expand.grid(cp = seq(0.01, 0.1, by = 0.01))

# Model training with cross-validation
tree_model <- train(adopter ~ ., data = data_both[, 2:27],
                    method = "rpart",
                    trControl = train_control,
                    metric = "ROC",
                    tuneGrid = tune_grid)

# Display model details
print(tree_model)
```

```
## CART
##
## 41540 samples
##    25 predictor
##     2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 37386, 37387, 37386, 37386, 37386, 37386, ...
## Resampling results across tuning parameters:
##
##   cp    ROC        Sens       Spec
##   0.01  0.7681474  0.6639435  0.7917616
##   0.02  0.7147222  0.6268454  0.7619040
##   0.03  0.6790714  0.5428417  0.8153011
```

```
##   0.04  0.6790714  0.5428417  0.8153011
##   0.05  0.6790714  0.5428417  0.8153011
##   0.06  0.6790714  0.5428417  0.8153011
##   0.07  0.6790714  0.5428417  0.8153011
##   0.08  0.6790714  0.5428417  0.8153011
##   0.09  0.6790714  0.5428417  0.8153011
##   0.10  0.6790714  0.5428417  0.8153011
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.01.
```

```r
prp(tree_model$finalModel, varlen = 0)
```



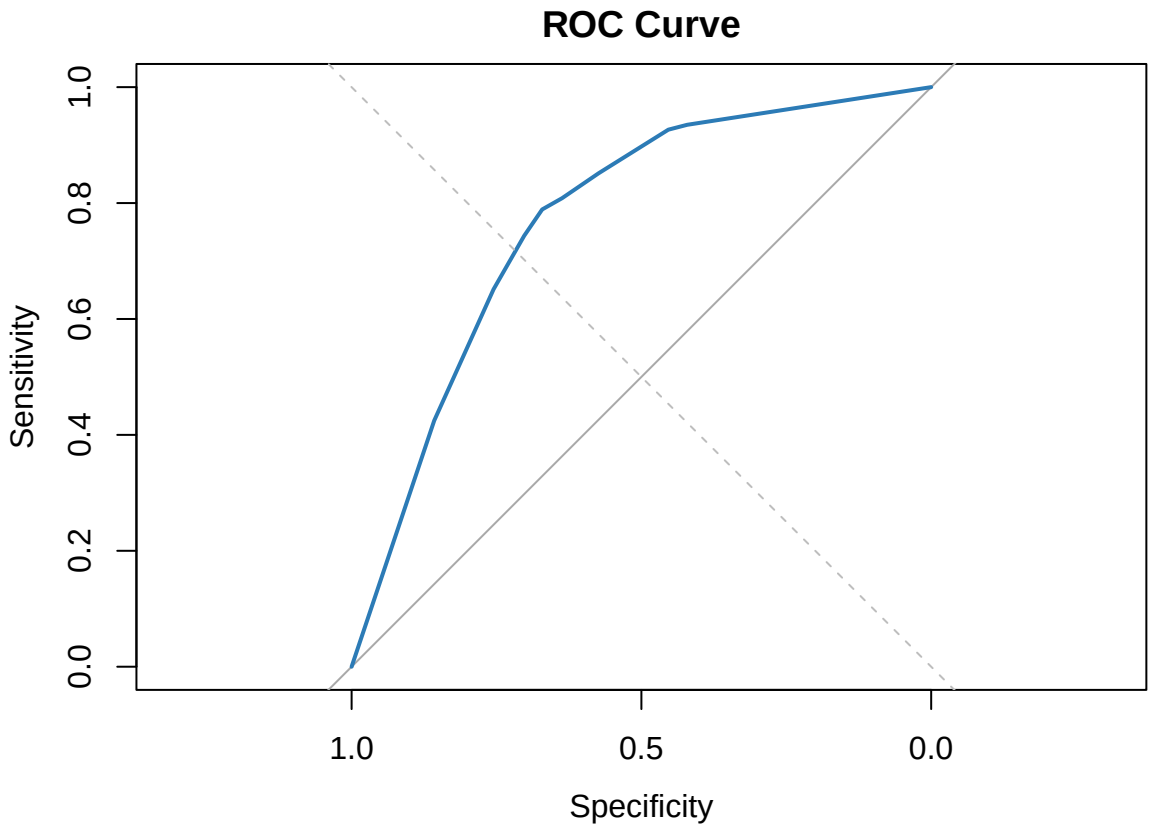

```r
# Predict on full dataset to get probabilities and predicted classes
data$predicted_prob_positive <- predict(tree_model, data[, 2:27], type = "prob")[, "yes"]
data$predicted_class <- predict(tree_model, data[, 2:27])

# Evaluate model performance on ROC and AUC
roc_obj <- roc(data$adopter, data$predicted_prob_positive)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```r
auc_value <- auc(roc_obj)
print(paste("AUC: ", auc_value))
```

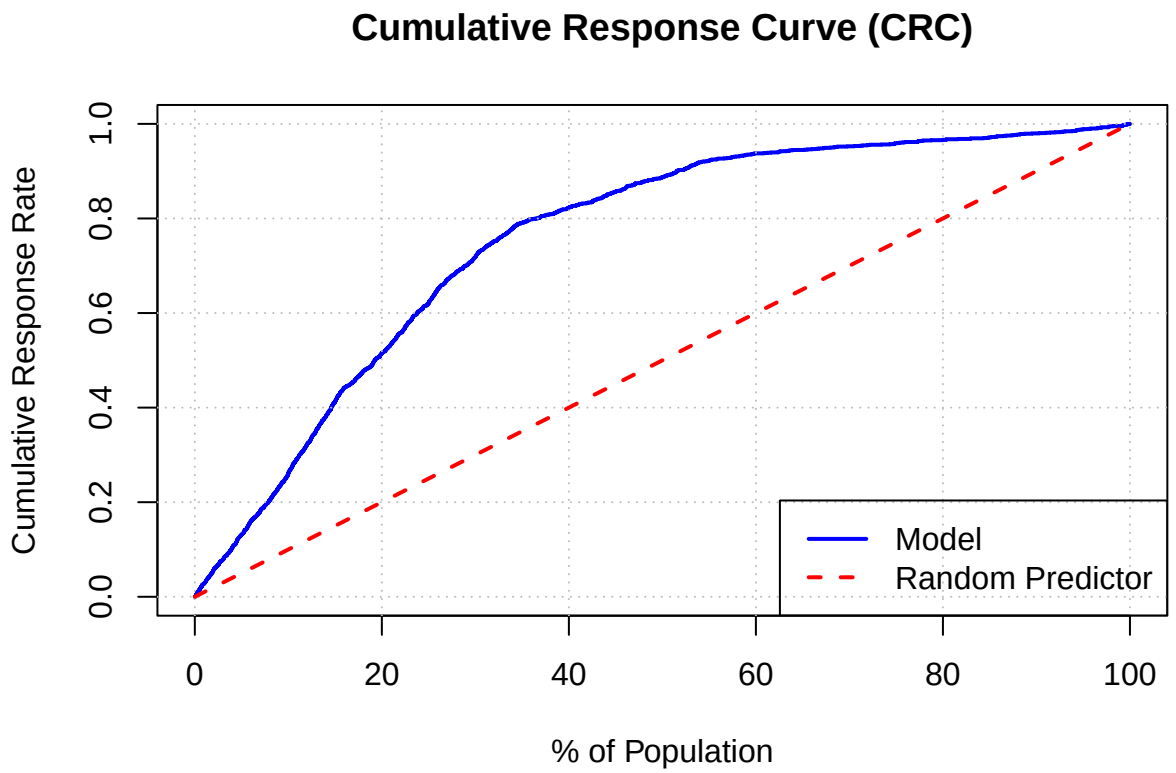
```
## [1] "AUC:  0.770272402597403"
```

```r
# Plot ROC curve
plot(roc_obj, col = "#2c7bb6", lwd = 2, main = "ROC Curve")
abline(a = 0, b = 1, col = "grey", lty = 2)
```




```r
# Cumulative Response Curve (CRC)
plot_df <- data %>% arrange(desc(predicted_prob_positive))

# Calculate cumulative response rate and % population
plot_df <- plot_df %>%
  mutate(cumulative_positive = cumsum(as.numeric(adopter) - 1),
         cumulative_rate = cumulative_positive / max(cumulative_positive),
         percent_population = (1:n()) / n() * 100)

# Plot CRC with % of Population
plot(plot_df$percent_population, plot_df$cumulative_rate, type = "l", col = "blue", lwd = 2,
     xlab = "% of Population", ylab = "Cumulative Response Rate",
     main = "Cumulative Response Curve (CRC)")
abline(h = seq(0, 1, by = 0.2), col = "gray", lty = 3)
abline(v = seq(0, 100, by = 20), col = "gray", lty = 3)
```

```r
lines(x = c(0, 100), y = c(0, 1), col = "red", lty = 2, lwd = 2)
legend("bottomright", legend = c("Model", "Random Predictor"),
       col = c("blue", "red"), lty = c(1, 2), lwd = 2)
```




```r
# Save results
write_xlsx(data, "/Users/justinvarghese/Downloads/data_output.xlsx")
#Saving the tree
save(tree_model, file = "/Users/justinvarghese/Downloads/tree_model.RData")
```