Riley Breske
CSC-161

Political Party Classification Using Random Forest

Project Overview:

In this project, I aimed to build a classification model to predict the political party affiliation of individuals based on a dataset consisting of 30 observations across 9 features. The goal was to create an accurate model by selecting the most relevant features and then applying a Random Forest classifier to achieve the best performance.

Feature Reduction:

Initially, I explored the relationships between the various features and reduced the number of features to enhance the model's accuracy. Through backward stepwise regression, four features—income, residence, job satisfaction (jobhappy), and gender—were removed due to their low predictive power. The final model was built using five features: ID, number of children, age, TV hours, and radio hours.

Random Forest Model:

Using the Random Forest algorithm, the model was able to achieve a very high level of accuracy with a minimal error rate of just 0.2%. The confusion matrix below illustrates the model's performance, showing nearly perfect classification with no misclassifications in most categories.

Confusion Matrix:

```
              Democrat Independent Other Republican class.error
Democrat        45         0          1      0          0.02173913
Independent     0          23         0      0          0.00000000
Other           0          0          36     0          0.00000000
Republican      0          0          0      45         0.00000000
```

When tested on the validation dataset, the model continued to perform perfectly, as demonstrated by the AUC values for each curve, all of which were 1. This indicates flawless performance of the model.

Conclusion:

The Random Forest model proved to be highly effective in predicting political party affiliation with exceptional accuracy. The feature selection process played a crucial role in this, ensuring that only the most relevant features were used, thereby preventing overfitting and improving the model's performance.

```r
### Library Installation ###
install.packages("boot", dep=TRUE)
library(boot)
install.packages('psych')
library(psych)
install.packages('ROCR')
library(ROCR)
install.packages('randomForest')
library(randomForest)

### EDA ###
# Read in the data
MyData = read.csv("smallsurveyQ1.csv")
summary(MyData)
describe(MyData)

### Feature Reduction with Backward Stepwise Regression ###
MyData$politicalparty = as.factor(MyData$politicalparty)
MyData$gender = as.factor(MyData$gender)
MyData$residence = as.factor(MyData$residence)

NumericMyData = MyData
NumericMyData$politicalparty = as.numeric(MyData$politicalparty)
NumericMyData$gender = as.numeric(MyData$gender)
NumericMyData$residence = as.numeric(MyData$residence)

# Initial Model
NumericMyData.lm = lm(politicalparty ~ id + gender + residence +
numbchildren + age + income + jobhappy + tvhours + radiohours, data =
NumericMyData)
summary(NumericMyData.lm)

# Stepwise Feature Removal
NumericMyData.lm = lm(politicalparty ~ id + numbchildren + age + tvhours +
radiohours, data = NumericMyData)
summary(NumericMyData.lm)
```

```r
####### Random Forest Model ###############
# Subset data for Random Forest
RandomForestData <- data.frame(politicalparty = MyData$politicalparty, id =
MyData$id, numbchildren = MyData$numbchildren, age = MyData$age, tvhours =
MyData$tvhours, radiohours = MyData$radiohours)

# Split data into build and validate samples
TotalSamples = sample(1:30, 300, replace = T)
ValidateSample = TotalSamples[1:150]
BuildSample = TotalSamples[151:300]

Build = RandomForestData[BuildSample,]
Validate = RandomForestData[ValidateSample,]

# Train Random Forest Model
MyModel = randomForest(politicalparty ~ ., data=Build, ntree=500, mtry=2,
importance=TRUE)
MyModel
varImpPlot(MyModel)

# Test with holdout data
MyPredictions = predict(MyModel, Validate[,-1])
table(observed=Validate[,1], predicted=MyPredictions)

# ROC Curve and AUC
install.packages("ROCR")
library(ROCR)

ROC_Predictions= predict(MyModel, Validate[,-1], type="prob")
Types = c("Green","Yellow","Red","Blue")
polparty = levels(Validate$politicalparty)

for (i in 1:4) {
    true_values = ifelse(Validate[,1] == polparty[i], 1, 0)
    pred = prediction(ROC_Predictions[,i], true_values)
    perf = performance(pred, "tpr", "fpr")
    if (i == 1) {
        plot(perf, col=Types[i], main="ROC for Political Party")
    } else {
        plot(perf, main="ROC", col=Types[i], add=TRUE)
    }
    AUC = performance(pred, measure = "auc")
    print(AUC@y.values)
}
```