Project #3 Report

St. Clair, Ryan Gorey, Junxiong Liu, Il Shan Ng

Due: 8am Wed., Nov. 16

1. Data Cleaning

First, we confirm that the dataset provided does not contain any missing values. We also replace the levels `Dem` and `Rep` of the response with `1` and `0` respectively.

```
> train <- read_csv("https://people.carleton.edu/~kstclair/data/train.csv")
> which(is.na(train))     # no NA values
integer(0)
> train2 <- train %>%
+   mutate(winner = as.factor(ifelse(winner == "Dem", 1, 0)))     # 1 if elected Democrat
```

We then create our own test dataset by randomly selecting 20 percent of the cases present in the `train2` data frame, making the remaining cases the training dataset. This test set will be used to compute accuracy rates for the four classifiers that we create.

```
> set.seed(1)
> N <- nrow(train2)
> train_index <- sample(1:N, size = round(0.8*N))
> train_df <- train2[train_index, ]
> test_df <- train2[-train_index, ]
```

2. Random Forest Model

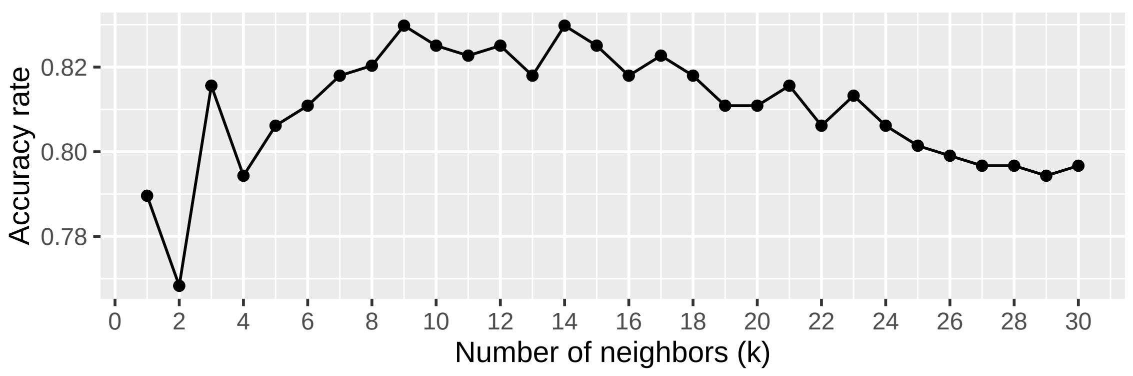
Here, we fit a random forest model using all 51 variables in the `train_df` data frame. We bootstrapped 10,000 trees to improve the model's accuracy subject to limitations in computing power. We follow the empirical literature and choose $\sqrt{51} \approx 7$ for the number of predictors to be considered at each split. We then store the predictions for the test dataset in a new data frame called `test_pred`.

```
> rf1 <- randomForest(winner ~ ., data = train_df, ntree = 10000, mtry = 7)
> test_pred <- test_df %>%
+   transmute(winner = winner,
+             prob_rf = predict(rf1, newdata = test_df, type = "prob")[,2],
+             pred_rf = predict(rf1, newdata = test_df, type = "class"))
```

We attempted some variable selection by refitting the model with only variables whose inclusion led to the greatest mean decreases in Gini coefficient. However, we found that the full model with 51 variables still gave the best accuracy.

3. K-nearest Neighbors Model

In this section, we fit a k-nearest neighbors (k-nn) model. Before doing so, we find the optimal number of neighbors by creating a plot of accuracy rate against k. We find that $k = 9$ and $k = 14$ give the highest accuracy rates, and choose the former to reduce computing times.



Now, we fit the knn model with $k = 9$. Bearing in mind the need to standardize the scales of all predictors, we center and scale all predictor variables before fitting the model. We then store the predictions for the test dataset in `test_pred`.

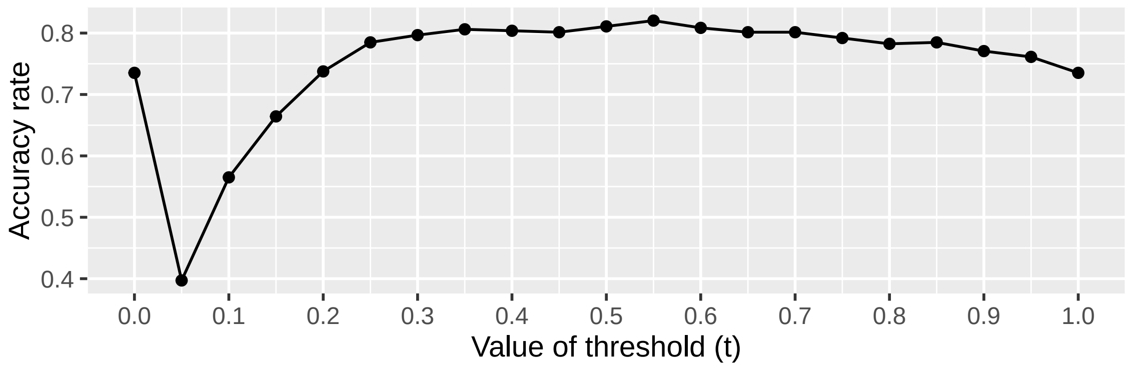
```
> train_df2 <- data.frame(scale(train_df[ ,-52]))
> test_df2 <- data.frame(scale(test_df[ ,-52]))
> knn1 <- knn(train = train_df2, test = test_df2, cl = train_df$winner, k = 9)
>
> test_pred <- test_pred %>%
+   mutate(pred_knn = knn1)
```

4. Logistic Regression Model

Here, we fit our third classifier, a binary logistic regression model with all 51 predictor variables. As with our random forest model, we attempted some variable selection by refitting the model with only the significant variables. However, we found that doing so increased the 5-fold cross validation error rate, and so went with our original full model.

```
> # fit logistic model with all variables
> elections.glm <- glm(winner ~ ., data = train2, family = "binomial")
>
> # fit logistic model with only significant variables
> sig_vars <- str_c(c("PST045214", "POP715213", "EDU635213", "HSG096213", "HSG495213", "INC910213",
+                     "BZA010213", "SBO315207", "SBO415207"), collapse = "+")
> myForm <- as.formula(str_c("winner ~ ", sig_vars))
> elections.glm2 <- glm(myForm, data = train2, family = "binomial")
>
> # cross validation on logistic model
> cost <- function(y, pi) mean(abs(y - pi) > 0.55)
> cv.glm1 <- cv.glm(train2, elections.glm, cost, K=5)
> cv.glm2 <- cv.glm(train2, elections.glm2, cost, K=5)
> cv.glm1$delta[1]     # error rate
[1] 0.1798391
> cv.glm2$delta[1]
[1] 0.1893043
```

Next, we found the optimal threshold t by creating a plot of accuracy rate against t, finding that a threshold of 0.55 gave the highest accuracy. Using this threshold, we then add the binary predictions of the logistic model to the `test_pred` data frame.



```
> test_pred <- test_pred %>%
+   mutate(pred_glm = ifelse(prob_glm > 0.55, 1, 0))
```

5. Ensemble Model

Rather than compare the above three models to choose the one that gives the highest accuracy, we use an ensemble model, as outlined in Section 8.3 of our textbook. To determine the prediction for each case in the test dataset, the ensemble method takes the majority vote of the other three classifiers' predictions, as shown in the code below. Ensemble classifiers can outperform other classifiers as long as their components are independent. This is because the chances of the majority of those independent classifiers being wrong may be lower than the chance of any particular classifier making a wrong prediction.

```
> test_pred <- test_pred %>%
+   mutate(pred_ens = ifelse((pred_rf == 1) +
+                            (pred_knn == 1) +
+                            (pred_glm == 1) >= 2, 1, 0))
```

6. Accuracy Rates

Here, we compute the accuracy rates for all four models. The output shows that the ensemble model produces the highest accuracy rate, and so we use the ensemble model as our final model.

```
> preds <- c("pred_rf", "pred_knn", "pred_glm", "pred_ens")
> sapply(preds, function(i, df) {
+   conf_mat <- with(df, table(winner, get(i)))
+   return(sum(diag(prop.table(conf_mat))))
+ }, df = test_pred)
  pred_rf  pred_knn  pred_glm  pred_ens
0.8179669 0.8297872 0.8203310 0.8345154
```

7. Submission of Predictions

In this final section, we add predictions for the test dataset provided by the Professor, and write the resulting dataset into a .csv file for submission.

```
> testNoY <- read_csv("https://people.carleton.edu/~kstclair/data/testNoY.csv")
> test_pred2 <- testNoY %>%
+   transmute(
+     pred_rf = predict(rf1, newdata = testNoY, type = "class"),
+     pred_knn = knn(train = train_df2, test = data.frame(scale(testNoY)),
+                    cl = train_df$winner, k = 9),
+     prob_glm = predict(elections.glm, type = "response", newdata = testNoY),
+     pred_glm = ifelse(prob_glm > 0.55, 1, 0),
+     pred_winner = ifelse((pred_rf == 1) +
+                          (pred_knn == 1) +
+                          (pred_glm == 1) >= 2, "Dem", "Rep")
+     )
> testNoY <- bind_cols(testNoY, test_pred2["pred_winner"])
> write_csv(testNoY, "testNoY_gorey_liu_ng.csv")
```