Lab No. 10

Elizabeth Walter

```
knitr::opts_chunk$set(echo = TRUE,
                      fig.align = 'center')
```

```
library(tibble)
library(dplyr)
library(caret)

bank <- read.table("https://raw.githubusercontent.com/ajkirkpatrick/FS20/Spring2021/classdata/bank.csv"
                   header = TRUE,
                   sep = ",")
```

1. Split the data into an 80/20 train vs. test split. Make sure you explicitly set the seed for replicability, but do not share your seed with others in the class.

```
# change categorical variables to factor variables
bank$job <- as.factor(bank$job)
bank$marital <- as.factor(bank$marital)
bank$education <- as.factor(bank$education)
bank$default <- as.factor(bank$default)
bank$housing <- as.factor(bank$housing)
bank$loan <- as.factor(bank$loan)
bank$month <- as.factor(bank$month)
# change to dummy bc numeric not working with rmse/predict
bank$y <- ifelse(bank$y == "yes", 1, 0)


# Set seed
set.seed(7)
num_obs = nrow(bank)

# test/train split
train_index <- sample(num_obs, size = trunc(0.80 * num_obs))
train_data <- bank[train_index, ]
test_data <- bank[-train_index, ]

# estimation/validation split
bank_est_idx = sample(nrow(train_data), size = 0.8 * nrow(train_data))
bank_est = train_data[bank_est_idx, ]
bank_val = train_data[-bank_est_idx, ]
```

2. Run a series of KNN models with k ranging from 2 to 100. (You need not do every k between 2 and 100, but you can easily write a short function to do this; see the Content tab).

```
# function to do knn models for sequence of k values
fit_knn_mod <- function(neighbors) {
  knnreg(y ~ job + default + balance, data = bank_est, k = neighbors)
}

k_to_try <- seq(from = 2, to = 100, by = 2)

# list storing all 50 knn models
knn_mod_est_list <- lapply(k_to_try, fit_knn_mod)
```

3. Create a chart plotting the model complexity as the x-axis variable and RMSE as the y-axis variable for both the training and test data. What do you think is the optimal k?

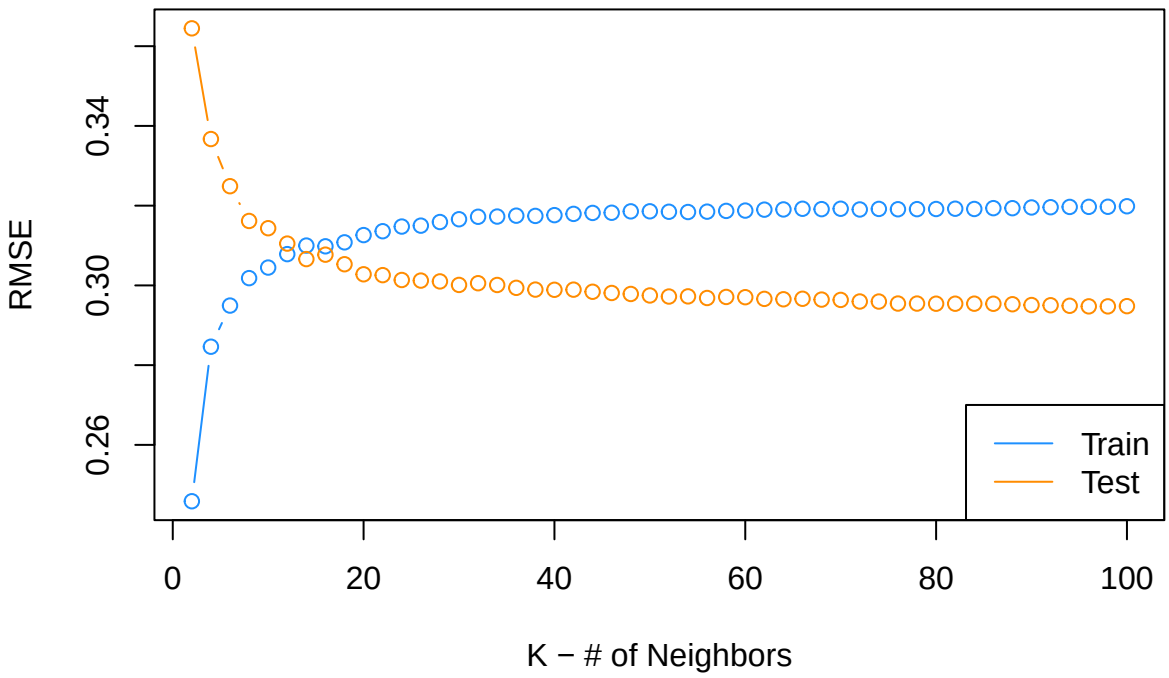
```
# Get prediction values for
knn_train_pred <- lapply(knn_mod_est_list, predict, bank_est)
knn_test_pred <- lapply(knn_mod_est_list, predict, bank_val)

calc_rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted) ^ 2))
}

# calc rmse of test and train data
trn_rmse <- sapply(knn_train_pred, calc_rmse, bank_est$y)
tst_rmse <- sapply(knn_test_pred, calc_rmse, bank_val$y)
tst_rmse
```

```
##  [1] 0.3644703 0.3367093 0.3248750 0.3161631 0.3143414 0.3104831 0.3066013
##  [8] 0.3077096 0.3053086 0.3027899 0.3025747 0.3013588 0.3012109 0.3009985
## [15] 0.3001153 0.3005469 0.3001080 0.2993905 0.2989457 0.2988994 0.2989345
## [22] 0.2984072 0.2980992 0.2978514 0.2974779 0.2972068 0.2972474 0.2968442
## [29] 0.2971017 0.2970413 0.2966213 0.2965223 0.2966446 0.2964417 0.2963564
## [36] 0.2959645 0.2959495 0.2954526 0.2954459 0.2954092 0.2954087 0.2954136
## [43] 0.2953935 0.2952737 0.2950734 0.2950192 0.2948878 0.2947431 0.2947396
## [50] 0.2947958
```

```
plot(k_to_try, trn_rmse, type = "b",
     ylim = c(min(c(trn_rmse, tst_rmse)),
              max(c(trn_rmse, tst_rmse))),
     col = "dodgerblue",
     xlab = "K - # of Neighbors",
     ylab = "RMSE",)
lines(k_to_try, tst_rmse, type = "b", col = "darkorange")
legend(x = 'bottomright', legend = c("Train", "Test"), col = c("dodgerblue", "darkorange"),
       lty = c(1,1))
```



k = 20. We chose a k value by evaluating the models on the validation data, so by looking at the Test RMSE, and want to minimize error without creating too inflexible of a model, as it is supposed to be used to predict on new data. As k is increasing from k = 2 to k = 20, the train RMSE is decreasing (apart from k = 16) by relatively large amounts before increasing and decreasing by much smaller amounts. This k minimizes test RMSE at 0.3027899 while limiting the chances of overfitting the data. This seems like possibly too large of a k value, though, when compared to the examples we've seen, and this may actually be too inflexible of a model to perform well on new data? If there was a point where the test RSME switched from generally decreasing and began generally increasing as k increased, it would be easier to determine the maximum appropriate k value. However here we see the changes in RMSE are quite small after k = 20, but RMSE is still overall decreasing, if only slightly, and I am not sure if further minimizing the test RMSE by at most 0.008 is worth a decrease in the flexibility of the model by another 60 neighbors.