Gov 2018 Lab 1: k nearest neighbors

Adeline Lo

January 22, 2022

This exercise is based on:

Pager, Devah. (2003). "The Mark of a Criminal Record." *American Journal of Sociology* 108(5):937-975. You are also welcome to watch Professor Pager discuss the design and result here.

To isolate the causal effect of a criminal record for black and white applicants, Pager ran an audit experiment. In this type of experiment, researchers present two similar people that differ only according to one trait thought to be the source of discrimination. We will be using this dataset to see if certain applicant characteristics (including race) can help us predict out of sample callback rates for job applications.

In the data you will use `criminalrecord.csv` nearly all these cases are present, but 4 cases have been redacted. We've kept these unnecessary variables in the dataset because it is common to receive a dataset with much more information than you need.

Name	Description
`jobid`	Job ID number
`callback`	1 if tester received a callback, 0 if the tester did not receive a callback.
`black`	1 if the tester is black, 0 if the tester is white.
`crimrec`	1 if the tester has a criminal record, 0 if the tester does not.
`interact city`	1 if tester interacted with employer during the job application, 0 if tester does not interact with employer. 1 is job is located in the city center, 0 if job is located in the suburbs.
`distance`	Job's average distance to downtown.
`custserv`	1 if job is in the costumer service sector, 0 if it is not.
`manualskill`	1 if job requires manual skills, 0 if it does not.

For our classification exercise, we will consider `callback` the label of our dependent variable; all other variables are our explanatory variables.

KNN

Read in data and take a look at it:

Read in the data, and omit NAs. You should end up with 694 observations.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
data <- read_csv("criminalrecord.csv") %>%
  drop_na()
```

```
## Rows: 696 Columns: 9
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (9): jobid, callback, black, crimrec, interact, city, distance, custserv...
```

```
##
## 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.
```

Question 1: Create Training/Testing data split in R.

1. Set the seed to 2019.
2. Randomly sample 70% of the data for training and set aside the remaining 30% for testing.
3. Create a function that calculates the Euclidean distance between 2 points. As a reminder, the formula for Euclidean distance is:

$$d(x_1, x_2) = \sqrt{\sum_{p=0}^{n}(x_{1p} - x_{2p})^2} \tag{1}$$

```
set.seed(2019)

sample_num <- floor(0.70 * nrow(data))
index <- sample(seq_len(nrow(data)), size = sample_num)

train <- data[index, ]
test <- data[-index, ]

euclid_dist  <- function(x, y) {
  return(sqrt(sum((x-y)^2)))
}

a <- c(0,1,2,3,4,5)
b <- c(5,4,3,2,1,0)

print(euclid_dist(a,b))
```

```
## [1] 8.3666
```

Question 2: KNN prediction function

Write a function knn with the following characteristics:

- 3 arguments: test data, train data, and a value for k
- Loops over all records of test and train data
- Returns predicted class labels of test data

```
knn <- function(train, test, k) {
  classlabels <- c()
  for(i in 1:nrow(test)) {
    distances <- c()
    for(j in 1:nrow(train)) {
      distij <- euclid_dist(test[i,],train[j,])
      distances <- c(distances, distij)
    }
    #print(distances)
    combined <- train %>%
      add_column(distances = distances) %>%
      arrange(distances)


    combined$index <- 1:nrow(train)

    callback_sum <- combined %>%
      filter(index <= k) %>%
      summarise(sum = sum(callback))

    classlabels <- c(classlabels, round((1/k) * callback_sum))
  }
  return(classlabels)
}

#predictions <- knn(train, test, 2)
#predictions
```

Question 3 Accuracy calculation

Create a function called `accuracy` that calculates how accurate your predictions are for the test labels. The accuracy function should calculate the ratio of the number of correctly predicted labels to the total number of predicted labels.

```
accuracy <- function(test, predictions) {
  accuracy <- test %>%
    add_column(preds = predictions) %>%
    mutate(correct = ifelse(preds == callback, 1, 0)) %>%
    summarise(sum = sum(correct))

  return(accuracy/nrow(test))
}

#print(accuracy(test, predictions))
```

Question 4 Make your prediction, find your accuracy

Using K=5, predict your labels for the testing data using the `knn` function you wrote.

Append the prediction vector a column in your test dataframe and then using the `accuracy()` method you wrote up in the previous question print the accuracy of your KNN model. What is the accuracy rate of your classification?

```
preds <- knn(train,test,5)
print(accuracy(test,preds))
```

```
##         sum
## 1 0.8133971
```