Classification Problem using KNN Algorithm

Ogundepo Ezekiel Adebayo

May 28, 2018

Overview of KNN Algorithm

K Nearest Neighbour also known as KNN is one of the machine learning algorithm for classification of a binary response variable given set of explanatory variables. It is a non parametric lazy learning algorithm and it doesn't make any assumptions on the underlying the distribution of the data. Since KNN is a lazy algorithm,it does not use the training data points to do any generalization, there is no explicit training phase. The training phase is faster compare to other learning methods.

There is a clause known as no free lunch, the disadvantage of KNN algorithm is the cost in terms of both time and memory. It needs more time to calculate the distance among the neighbours, all data points might take point in decision. More memory is needed as we need to store all training data.

Data Set Information:

We got the dataset from the study of the donor database of Blood Transfusion Service Center in Hsin-Chu City in Taiwan from the **link**. To build KNN model, we used the data stored in the database. The data have 748 donor records, the variables included Recency - months since last donation, Frequency - total number of donation, Monetary - total blood donated in c.c.), Time - months since first donation), and a binary variable representing whether he/she donated blood in March 2007 (1 stand for donating blood; 0 stands for not donating blood).

```
library(tidyverse)
```

```
## -- Attaching packages -------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
library(class)
```

Data loading

```
b_transfusion=read.csv('blood_transfusion.csv')
```

First of all, we want to have a dataset that is easy to read. the first data cleaning is about replacing the "0" value with 'not donating blood' and "1" value with 'donating blood' in donated. this replacement makes the data to be more informative. Hence we employ below code:

```
b_transfusion$donated=as.factor(b_transfusion$donated)
levels(b_transfusion$donated)=c('not donating blood','donating blood')
```

Overview of the dataset

```
head(b_transfusion)
```

```
##   recency frequency monetary time            donated
## 1       2        50    12500   98     donating blood
## 2       0        13     3250   28     donating blood
```

```
## 3       1        16     4000   35     donating blood
## 4       2        20     5000   45     donating blood
## 5       1        24     6000   77 not donating blood
## 6       4         4     1000    4 not donating blood
```

```
glimpse(b_transfusion)
```

```
## Observations: 748
## Variables: 5
## $ recency   <int> 2, 0, 1, 2, 1, 4, 2, 1, 2, 5, 4, 0, 2, 1, 2, 2, 2, 2...
## $ frequency <int> 50, 13, 16, 20, 24, 4, 7, 12, 9, 46, 23, 3, 10, 13, ...
## $ monetary  <int> 12500, 3250, 4000, 5000, 6000, 1000, 1750, 3000, 225...
## $ time      <int> 98, 28, 35, 45, 77, 4, 14, 35, 22, 98, 58, 4, 28, 47...
## $ donated   <fct> donating blood, donating blood, donating blood, dona...
```

This is a tall and skinny dataset. The number of observations is 748 while the number of variables in the dataset is 5.

```
summary(b_transfusion)
```

```
##     recency         frequency         monetary          time
##  Min.   : 0.000   Min.   : 1.000   Min.   :  250   Min.   : 2.00
##  1st Qu.: 2.750   1st Qu.: 2.000   1st Qu.:  500   1st Qu.:16.00
##  Median : 7.000   Median : 4.000   Median : 1000   Median :28.00
##  Mean   : 9.507   Mean   : 5.515   Mean   : 1379   Mean   :34.28
##  3rd Qu.:14.000   3rd Qu.: 7.000   3rd Qu.: 1750   3rd Qu.:50.00
##  Max.   :74.000   Max.   :50.000   Max.   :12500   Max.   :98.00
##                donated
##  not donating blood:570
##  donating blood    :178
##
##
##
##
```

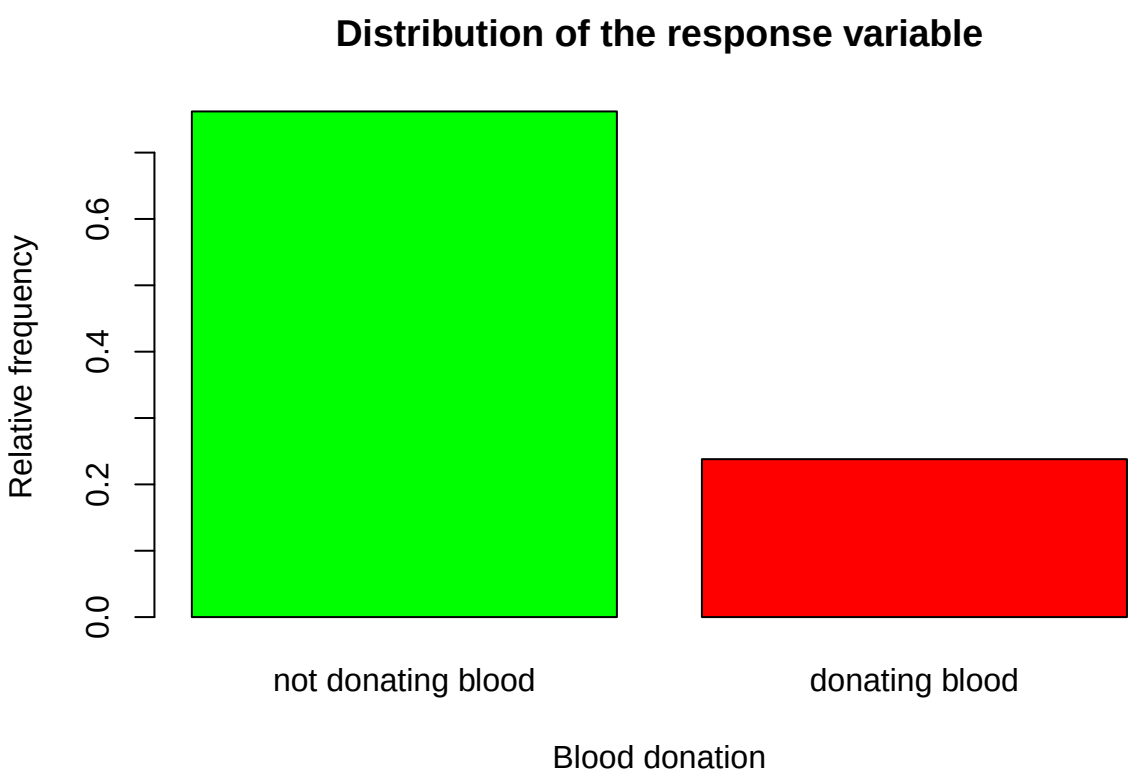
The five number summary which include max, median, mean. 1st Qu, and 3rd Qu were also displayed for the explanatory variables.

The distribution of the response variable :

```
table(b_transfusion$donated)
```

```
##
## not donating blood     donating blood
##                570                178
```

A binary variable blood donation representing whether he/she donated blood in March 2007. It can be seen from the table that 570 (76.2%) did not donate blood while only 178 (23.8%) donated blood. The probability that an individual will donate the blood is approximately 23.8%.



It is important to normalize the dataset for the data to be in the same scale. for instance for recency all numbers are between 0 to 74 while for column time is between 2 to 98. To perform prediction analysis using KNN, it is important all numbers should be in same scale.

Normalization can be done by the function below:

```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }
```

Now we are going to apply this function in all numeric columns in blood transfusion dataset. There is a function in R that apply a function over a dataset:

```
b_transfusion=b_transfusion %>% mutate_at(vars(1:4),~normalize(.))
```

"mutate_at" gets the dataset and function name, then apply the function on all dataset. in this case because the fifith column is text (donating), we apply "normalize" function on columns 1 to 4.Now our data is ready for creating a KNN model.

From machine learning process we need a dataset for training model and another for testing model.

Hence, we should have two different dataset for train and test. in this example, we going to have 499 for training and creating model and 249 for testing the model.

```
n <- nrow(b_transfusion);n       # Sample size
```

```
## [1] 748
```

```
  p <- ncol(b_transfusion) - 1;p   # Dimensionality of the input space
```

```
## [1] 4
```

```
  pos <- p+1;pos          # Position of the response
```

```
## [1] 5
```

```
  x   <- b_transfusion[,-pos]    # Data matrix: n x p matrix
  y   <- b_transfusion[, pos]    # Response vector

set.seed(200) #Set seed for random number generation to be reproducible
n=nrow(b_transfusion)
  epsilon <- 1/3               # Proportion of observations in the test set
  nte     <- round(n*epsilon)  # Number of observations in the test set
  ntr     <- n - nte

  id.tr   <- sample(sample(sample(n)))[1:ntr]   # For a sample of ntr indices from {1,2,..,n}
  id.te   <- setdiff(1:n, id.tr)

  k        <- 23 #Number of nearest neighbour
  y.te     <- y[id.te]                                  # True responses in test set
```

The data is ready, now we are going to train model and create KNN algorithm. We want to create a prediction models for a new donor with specific blood transfusion data, we want to predict whether this donor will donate blood or not.

For using KNN there is a need to install package "Class". Now we able to call function KNN to predict the blood donation. KNN function accept the training dataset and test dataset as second arguments. moreover the prediction label also need for result. To identify the number K (K nearest Neighbour), we calculate the square root of observation. here for 469 observation the K is 23.

```
b_transfusion_test_pred <- knn(x[id.tr,], x[id.te,], y[id.tr], k=k) # Predicted responses in test set
```

The result is "b_transfusion_test_pred" holds the result of the KNN prediction.

```
table(b_transfusion_test_pred)
```

```
## b_transfusion_test_pred
## not donating blood     donating blood
##                223                 26
```

We want to evaluate the result of the model by "gmodels" a packages that shows the evaluation performance. We employ a function name "CrossTable". it gets label as first input, the prediction result as second argument.

```
library(gmodels)
CrossTable(x = y.te, y = b_transfusion_test_pred ,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  249
##
```

```
##
##                    | b_transfusion_test_pred
##                y.te | not donating blood |     donating blood |          Row Total |
## --------------------|--------------------|--------------------|--------------------|
## not donating blood |                187 |                 11 |                198 |
##                    |              0.944 |              0.056 |              0.795 |
##                    |              0.839 |              0.423 |                    |
##                    |              0.751 |              0.044 |                    |
## --------------------|--------------------|--------------------|--------------------|
##     donating blood |                 36 |                 15 |                 51 |
##                    |              0.706 |              0.294 |              0.205 |
##                    |              0.161 |              0.577 |                    |
##                    |              0.145 |              0.060 |                    |
## --------------------|--------------------|--------------------|--------------------|
##       Column Total |                223 |                 26 |                249 |
##                    |              0.896 |              0.104 |                    |
## --------------------|--------------------|--------------------|--------------------|
##
##
```

We have 249 observation. the tables show the result of evaluation and see how much the KNN prediction is accurate. the first row and first column shows the true negative (TN) cases, means the cases that already do not donate blood and KNN correctly classify them as do not donate blood. The first row and second column shows number of cases that already do not donate blood and KNN misclassify them as donating blood (FP). The second row and first column are those that donate blood in real world but KNN predict they are not donating blood (FN). finally the last column and last row is True Postive (TP) that means cases that they are donating blood and KNN predict as donating blood.

So as much as TP and FN is higher the prediction is better. In our study TP is 15 and TN is 187, moreover the FN and FP are just 36 and 11 which is good.

To calculate the accuracy we should follow the below formula:

$accuracy = \frac{TP+TN}{TP+FN+FP+TN}$

Accuracy will be $\frac{15+187}{15+36+11+187} = 79.5\%$

References

James, G, Witten, D, Hastie, T and Tibshirani, R (2013). An Introduction to Statistical Learning with Applications in R. Springer, New York, (e-ISBN: 978-1-4614-7138-7),(2013)

Clarke, B, Fokou?e, E and Zhang, H (2009). Principles and Theory for Data Mining and Machine Learning. Springer Verlag, New York, (ISBN: 978-0-387-98134-5), (2009)