Praticando K-NN

Contents

Objetivo e Prelúdio

Nesse laboratório, nós vamos investigar a utilidade de algoritmos de ML para detecção de câncer ao aplicar o algoritmo k-NN, comparando entre diferentes K_i, para fazer a melhor predição de diagnóstico, se é `Maligno` ou `Benigno`.

```
library(tidyverse)
library(class)
library(gt)
library(knitr)
library(kableExtra)
```

Os Dados

Importei os dados, chamando-o de `dt`. Retirei a variável `id`. Transformei nossa variável alvo em fator. Por fim, listei e sumarizei as variáveis.

```
dt <- read_csv("wisc_bc_data.csv") %>% as_tibble()
```

```
dt$id <- NULL
dt$diagnosis <- factor(dt$diagnosis,
                       levels = c('B', 'M'),
                       labels = c('Benigno', 'Maligno'))
glimpse(dt)
```

```
## Rows: 569
## Columns: 31
## $ diagnosis         <fct> Benigno, Benigno, Benigno, Benigno, Benigno, Benigno~
## $ radius_mean       <dbl> 12.32, 10.60, 11.04, 11.28, 15.19, 11.57, 11.51, 13.~
## $ texture_mean      <dbl> 12.39, 18.95, 16.83, 13.39, 13.21, 19.04, 23.93, 23.~
## $ perimeter_mean    <dbl> 78.85, 69.28, 70.92, 73.00, 97.65, 74.20, 74.52, 91.~
## $ area_mean         <dbl> 464.1, 346.4, 373.2, 384.8, 711.8, 409.7, 403.5, 597~
## $ smoothness_mean   <dbl> 0.10280, 0.09688, 0.10770, 0.11640, 0.07963, 0.08546~
## $ compactness_mean  <dbl> 0.06981, 0.11470, 0.07804, 0.11360, 0.06934, 0.07722~
## $ concavity_mean    <dbl> 0.039870, 0.063870, 0.030460, 0.046350, 0.033930, 0.~
## $ points_mean       <dbl> 0.037000, 0.026420, 0.024800, 0.047960, 0.026570, 0.~
```

```
## $ symmetry_mean     <dbl> 0.1959, 0.1922, 0.1714, 0.1771, 0.1721, 0.2031, 0.13~
## $ dimension_mean    <dbl> 0.05955, 0.06491, 0.06340, 0.06072, 0.05544, 0.06267~
## $ radius_se         <dbl> 0.2360, 0.4505, 0.1967, 0.3384, 0.1783, 0.2864, 0.23~
## $ texture_se        <dbl> 0.6656, 1.1970, 1.3870, 1.3430, 0.4125, 1.4400, 2.90~
## $ perimeter_se      <dbl> 1.670, 3.430, 1.342, 1.851, 1.338, 2.206, 1.936, 3.9~
## $ area_se           <dbl> 17.43, 27.10, 13.54, 26.33, 17.72, 20.30, 16.97, 52.~
## $ smoothness_se     <dbl> 0.008045, 0.007470, 0.005158, 0.011270, 0.005012, 0.~
## $ compactness_se    <dbl> 0.011800, 0.035810, 0.009355, 0.034980, 0.014850, 0.~
## $ concavity_se      <dbl> 0.016830, 0.033540, 0.010560, 0.021870, 0.015510, 0.~
## $ points_se         <dbl> 0.012410, 0.013650, 0.007483, 0.019650, 0.009155, 0.~
## $ symmetry_se       <dbl> 0.01924, 0.03504, 0.01718, 0.01580, 0.01647, 0.01868~
## $ dimension_se      <dbl> 0.002248, 0.003318, 0.002198, 0.003442, 0.001767, 0.~
## $ radius_worst      <dbl> 13.50, 11.88, 12.41, 11.92, 16.20, 13.07, 12.48, 19.~
## $ texture_worst     <dbl> 15.64, 22.94, 26.44, 15.77, 15.73, 26.98, 37.16, 41.~
## $ perimeter_worst   <dbl> 86.97, 78.28, 79.93, 76.53, 104.50, 86.43, 82.28, 12~
## $ area_worst        <dbl> 549.1, 424.8, 471.4, 434.0, 819.1, 520.5, 474.2, 115~
## $ smoothness_worst  <dbl> 0.1385, 0.1213, 0.1369, 0.1367, 0.1126, 0.1249, 0.12~
## $ compactness_worst <dbl> 0.12660, 0.25150, 0.14820, 0.18220, 0.17370, 0.19370~
## $ concavity_worst   <dbl> 0.124200, 0.191600, 0.106700, 0.086690, 0.136200, 0.~
## $ points_worst      <dbl> 0.09391, 0.07926, 0.07431, 0.08611, 0.08178, 0.06664~
## $ symmetry_worst    <dbl> 0.2827, 0.2940, 0.2998, 0.2102, 0.2487, 0.3035, 0.21~
## $ dimension_worst   <dbl> 0.06771, 0.07587, 0.07881, 0.06784, 0.06766, 0.08284~
```

```
summary(dt)
```

```
##    diagnosis    radius_mean      texture_mean   perimeter_mean
##  Benigno:357   Min.   : 6.981   Min.   : 9.71   Min.   : 43.79
##  Maligno:212   1st Qu.:11.700   1st Qu.:16.17   1st Qu.: 75.17
##                Median :13.370   Median :18.84   Median : 86.24
##                Mean   :14.127   Mean   :19.29   Mean   : 91.97
##                3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:104.10
##                Max.   :28.110   Max.   :39.28   Max.   :188.50
##    area_mean      smoothness_mean   compactness_mean  concavity_mean
##  Min.   : 143.5   Min.   :0.05263   Min.   :0.01938   Min.   :0.00000
##  1st Qu.: 420.3   1st Qu.:0.08637   1st Qu.:0.06492   1st Qu.:0.02956
##  Median : 551.1   Median :0.09587   Median :0.09263   Median :0.06154
##  Mean   : 654.9   Mean   :0.09636   Mean   :0.10434   Mean   :0.08880
##  3rd Qu.: 782.7   3rd Qu.:0.10530   3rd Qu.:0.13040   3rd Qu.:0.13070
##  Max.   :2501.0   Max.   :0.16340   Max.   :0.34540   Max.   :0.42680
##   points_mean      symmetry_mean    dimension_mean      radius_se
##  Min.   :0.00000   Min.   :0.1060   Min.   :0.04996   Min.   :0.1115
##  1st Qu.:0.02031   1st Qu.:0.1619   1st Qu.:0.05770   1st Qu.:0.2324
##  Median :0.03350   Median :0.1792   Median :0.06154   Median :0.3242
##  Mean   :0.04892   Mean   :0.1812   Mean   :0.06280   Mean   :0.4052
##  3rd Qu.:0.07400   3rd Qu.:0.1957   3rd Qu.:0.06612   3rd Qu.:0.4789
##  Max.   :0.20120   Max.   :0.3040   Max.   :0.09744   Max.   :2.8730
##    texture_se      perimeter_se       area_se        smoothness_se
##  Min.   :0.3602   Min.   : 0.757   Min.   :  6.802   Min.   :0.001713
##  1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850   1st Qu.:0.005169
##  Median :1.1080   Median : 2.287   Median : 24.530   Median :0.006380
##  Mean   :1.2169   Mean   : 2.866   Mean   : 40.337   Mean   :0.007041
##  3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190   3rd Qu.:0.008146
##  Max.   :4.8850   Max.   :21.980   Max.   :542.200   Max.   :0.031130
##  compactness_se      concavity_se       points_se         symmetry_se
```

```
##  Min.   :0.002252   Min.   :0.00000   Min.   :0.000000   Min.   :0.007882
##  1st Qu.:0.013080   1st Qu.:0.01509   1st Qu.:0.007638   1st Qu.:0.015160
##  Median :0.020450   Median :0.02589   Median :0.010930   Median :0.018730
##  Mean   :0.025478   Mean   :0.03189   Mean   :0.011796   Mean   :0.020542
##  3rd Qu.:0.032450   3rd Qu.:0.04205   3rd Qu.:0.014710   3rd Qu.:0.023480
##  Max.   :0.135400   Max.   :0.39600   Max.   :0.052790   Max.   :0.078950
##   dimension_se        radius_worst   texture_worst   perimeter_worst
##  Min.   :0.0008948   Min.   : 7.93   Min.   :12.02   Min.   : 50.41
##  1st Qu.:0.0022480   1st Qu.:13.01   1st Qu.:21.08   1st Qu.: 84.11
##  Median :0.0031870   Median :14.97   Median :25.41   Median : 97.66
##  Mean   :0.0037949   Mean   :16.27   Mean   :25.68   Mean   :107.26
##  3rd Qu.:0.0045580   3rd Qu.:18.79   3rd Qu.:29.72   3rd Qu.:125.40
##  Max.   :0.0298400   Max.   :36.04   Max.   :49.54   Max.   :251.20
##    area_worst     smoothness_worst  compactness_worst concavity_worst
##  Min.   : 185.2   Min.   :0.07117   Min.   :0.02729   Min.   :0.0000
##  1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720   1st Qu.:0.1145
##  Median : 686.5   Median :0.13130   Median :0.21190   Median :0.2267
##  Mean   : 880.6   Mean   :0.13237   Mean   :0.25427   Mean   :0.2722
##  3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910   3rd Qu.:0.3829
##  Max.   :4254.0   Max.   :0.22260   Max.   :1.05800   Max.   :1.2520
##   points_worst     symmetry_worst   dimension_worst
##  Min.   :0.00000   Min.   :0.1565   Min.   :0.05504
##  1st Qu.:0.06493   1st Qu.:0.2504   1st Qu.:0.07146
##  Median :0.09993   Median :0.2822   Median :0.08004
##  Mean   :0.11461   Mean   :0.2901   Mean   :0.08395
##  3rd Qu.:0.16140   3rd Qu.:0.3179   3rd Qu.:0.09208
##  Max.   :0.29100   Max.   :0.6638   Max.   :0.20750
```

Fiz uma nova base de dados escalonando positivamente as variáveis numéricas em relação à maior diferença entre elas.

```
scale1 <- function(x){ return( (x-min(x)) / (max(x)-min(x)) ) }
dt1 <- as.data.frame(lapply(dt[2:31],scale1)) %>% as_tibble()
dt1$diagn <- dt$diagnosis
```

Divisão de Amostras

Escolhi, com aleatoriedade fixa em 777 (para fins de replicação), uma amostra de treino, sobre o qual o modelo será feito, de 80%. Portanto, o teste será feito na amostra restante, 20%.

```
set.seed(777)
train <- dt1 %>% sample_frac(.,0.8)
sid   <- as.numeric(rownames(train))
test  <- dt1[-sid,]
remove(sid)
```

Modelo e Comparações entre K's

Rodarei um primeiro modelo seguindo a regra de bolso $k = \sqrt{(numero\ \ de\ \ observacoes\ \ do\ \ treino)} = \sqrt{(455)} \approx 21$.
As amostras de Treino e Teste devem conter apenas variáveis numéricas, por isso, especifiquei que treino e teste devem ser realizados da coluna 1 até a 30, e.g., `tain[1:30]`. Em `cl =` é que colocarmos a variável-fator, que queremos prever.

Primeiro Modelo

```
knn0 <- knn(train = train[1:30],
            test = test[1:30],
            cl = train$diagn,
            k = 21)
```

Para fins de comparação, criei um novo banco de dados chamado `rslt0`; comparei em `t0` a acurácia do $k(21) - nn$ e salvei três resultados interessantes: a acurácia global (`Acc0`), o falso benigno (`F_Ben0`) e o falso maligno (`F_Mal0`):

```
rslt0 <- data.frame(Original = c(test$diagn),
                    Predito = c(knn0))
t0 <- table(rslt0$Original, rslt0$Predito)
Acc0 <- sum(diag(t0))/sum(t0)
F_Ben0 <- sum(t0[2,1])/sum(t0[,1])
F_Mal0 <- sum(t0[1,2])/sum(t0[,2])
t0
```

```
##
##      1  2
##   1 69  1
##   2  3 41
```

Salvarei esse resultado e os demais que se seguirão em `Resultados` como se segue:

```
Resultados <- data.frame( K = 21,
                          Acurácia = Acc0,
                          Falso_Benigno = F_Ben0,
                          Falso_Maligno = F_Mal0)
```

Comparando os Demais K

Farei um *loop* de 1 a 30 onde cada índice será o número de vizinhos, $k = (1 : 30)$. Em cada i, o *loop* fará um modelo k-nn; salvará temporariamente os diagnósticos preditos vis-a-vis os diagnósticos originais em `rslt`; calcularei e salvarei temporariamente os três resultados que quero; farei uma nova linha com os três resultados e a adicionarei em `Resultados`:

```
Ki <- 1:30

for (i in Ki) {
  modelo_KNN <- knn(train = train[1:30],
                    test = test[1:30],
                    cl = train$diagn,
                    k = i)
  rslt <- data.frame(Original = c(test$diagn),
                     Predito = c(modelo_KNN))
  tab <- table(rslt$Original, rslt$Predito)
  Acc   <- sum(diag(tab))/sum(tab)
  F_Ben <- sum(tab[2,1])/sum(tab[,1])
  F_Mal <- sum(tab[1,2])/sum(tab[,2])
  nova_linha <- data.frame(i, Acc, F_Ben, F_Mal)
  names(nova_linha) <- c('K', 'Acurácia', 'Falso_Benigno', 'Falso_Maligno')
  Resultados <- rbind(Resultados, nova_linha)
}
```

Table 1: Taxa de Acurácia e Falso Resultado Por K Vizinhos Próximos

K	Acurácia	Falso_Benigno	Falso_Maligno
21	0.9649	0.0417	0.0238
1	0.9912	0.0000	0.0222
2	0.9737	0.0145	0.0444
3	0.9912	0.0141	0.0000
4	0.9825	0.0278	0.0000
5	0.9912	0.0141	0.0000
6	0.9912	0.0141	0.0000
7	0.9825	0.0143	0.0227
8	0.9912	0.0141	0.0000
9	0.9825	0.0143	0.0227
10	0.9912	0.0141	0.0000
11	0.9912	0.0141	0.0000
12	0.9737	0.0411	0.0000
13	0.9825	0.0278	0.0000
14	0.9825	0.0278	0.0000
15	0.9825	0.0278	0.0000
16	0.9737	0.0282	0.0233
17	0.9649	0.0417	0.0238
18	0.9737	0.0411	0.0000
19	0.9649	0.0417	0.0238
20	0.9649	0.0417	0.0238
21	0.9649	0.0417	0.0238
22	0.9737	0.0411	0.0000
23	0.9649	0.0417	0.0238
24	0.9649	0.0417	0.0238
25	0.9737	0.0411	0.0000
26	0.9649	0.0417	0.0238
27	0.9737	0.0411	0.0000
28	0.9649	0.0541	0.0000
29	0.9737	0.0411	0.0000
30	0.9649	0.0541	0.0000

Visualizando os Resultados.

```
Resultados %>% round(4) %>% kbl(booktabs = T, caption= 'Taxa de Acurácia e Falso Resultado Por K Vizinho
```

Table 2: Taxa(s) Máxima(s) de Acurácia Por K Vizinhos Próximos

K	Acurácia	Falso_Benigno	Falso_Maligno
1	0.9912	0.0000	0.0222
3	0.9912	0.0141	0.0000
5	0.9912	0.0141	0.0000
6	0.9912	0.0141	0.0000
8	0.9912	0.0141	0.0000
10	0.9912	0.0141	0.0000
11	0.9912	0.0141	0.0000

Por fim, escolheremos os modelos com melhor acurácia.

```
Resultados %>% slice_max(Resultados$Acurácia) %>% round(4) %>% kbl(booktabs = T, caption='Taxa(s) Máxima
```