Cancer de Mama: Exemplo da Variacao de Performance Atraves da Mudanca de Parametros

Rafael Bicudo Rosa

May 31, 8

Prevendo a Ocorrencia de Cancer

Este trabalho e uma releitura de um projeto integrante do curso Big Data Analytics com R e Microsoft Azure da Formacao Cientista de Dados. O objetivo e analisar dados reais sobre exames de cancer de mama realizados com mulheres nos EUA, usar um modelo 'knn' para prever a ocorrencia de novos casos, e ver a variacao de performance com o ajustamento do valor de um dos parametros.

Os dados de cancer de mama incluem 569 observacoes de biopsias, cada uma com 32 caracteristicas (variaveis), sendo a 1a um numero de identificacao (ID), a 2a o diagnostico do tumor ('B' indicando benigno e 'M' maligno), e o restante 30 medidas laboratoriais numericas. Todas as informacoes foram retiradas do repositorio online da Universidade de Irvine, California (http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29).

Todo o projeto sera descrito de acordo com suas etapas.

Etapa 1 - Coletando os Dados

Assim como descrito acima, os dados serão retirados de um repositorio online contendo a base em si no formato csv, e a informacao de cada uma das caracteristicas.

```
# Coletando dados

# link para os dados
link_dados <- 'http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.da

# definicao dos nomes das features
names_bc = c("id", "diagnosis", "radius_mean", "texture_mean", "perimeter_mean", "area_mean", "smoothnes
          "compactness_mean", "concavity_mean", "points_mean", "symmetry_mean", "dimension_mean", "radiu
          "texture_se", "perimeter_se", "area_se", "smoothness_se", "compactness_se", "concavity_se", "p
          "symmetry_se", "dimension_se", "radius_worst", "texture_worst", "perimeter_worst", "area_worst
          "smoothness_worst", "compactness_worst", "concavity_worst", "points_worst", "symmetry_worst",

dados <- read.csv(link_dados, stringsAsFactors = F, col.names = names_bc)
str(dados)
```

```
## 'data.frame':    568 obs. of  32 variables:
##  $ id               : int  842517 84300903 84348301 84358402 843786 844359 84458202 844981 84501001 8
##  $ diagnosis        : chr  "M" "M" "M" "M" ...
##  $ radius_mean      : num  20.6 19.7 11.4 20.3 12.4 ...
##  $ texture_mean     : num  17.8 21.2 20.4 14.3 15.7 ...
##  $ perimeter_mean   : num  132.9 130 77.6 135.1 82.6 ...
##  $ area_mean        : num  1326 1203 386 1297 477 ...
##  $ smoothness_mean  : num  0.0847 0.1096 0.1425 0.1003 0.1278 ...
##  $ compactness_mean : num  0.0786 0.1599 0.2839 0.1328 0.17 ...
##  $ concavity_mean   : num  0.0869 0.1974 0.2414 0.198 0.1578 ...
##  $ points_mean      : num  0.0702 0.1279 0.1052 0.1043 0.0809 ...
```

```
##  $ symmetry_mean    : num  0.181 0.207 0.26 0.181 0.209 ...
##  $ dimension_mean   : num  0.0567 0.06 0.0974 0.0588 0.0761 ...
##  $ radius_se        : num  0.543 0.746 0.496 0.757 0.335 ...
##  $ texture_se       : num  0.734 0.787 1.156 0.781 0.89 ...
##  $ perimeter_se     : num  3.4 4.58 3.44 5.44 2.22 ...
##  $ area_se          : num  74.1 94 27.2 94.4 27.2 ...
##  $ smoothness_se    : num  0.00522 0.00615 0.00911 0.01149 0.00751 ...
##  $ compactness_se   : num  0.0131 0.0401 0.0746 0.0246 0.0335 ...
##  $ concavity_se     : num  0.0186 0.0383 0.0566 0.0569 0.0367 ...
##  $ points_se        : num  0.0134 0.0206 0.0187 0.0188 0.0114 ...
##  $ symmetry_se      : num  0.0139 0.0225 0.0596 0.0176 0.0216 ...
##  $ dimension_se     : num  0.00353 0.00457 0.00921 0.00511 0.00508 ...
##  $ radius_worst     : num  25 23.6 14.9 22.5 15.5 ...
##  $ texture_worst    : num  23.4 25.5 26.5 16.7 23.8 ...
##  $ perimeter_worst  : num  158.8 152.5 98.9 152.2 103.4 ...
##  $ area_worst       : num  1956 1709 568 1575 742 ...
##  $ smoothness_worst : num  0.124 0.144 0.21 0.137 0.179 ...
##  $ compactness_worst: num  0.187 0.424 0.866 0.205 0.525 ...
##  $ concavity_worst  : num  0.242 0.45 0.687 0.4 0.535 ...
##  $ points_worst     : num  0.186 0.243 0.258 0.163 0.174 ...
##  $ symmetry_worst   : num  0.275 0.361 0.664 0.236 0.399 ...
##  $ dimension_worst  : num  0.089 0.0876 0.173 0.0768 0.1244 ...
```

Etapa 2 - Preparacao dos Dados

Durante esta etapa, far-se-ao todas as trasformacoes necessarias a aplicacao do modelo, bem como observacoes interessantes acerca da amostra.

Independentemente do metodo de aprendizagem de maquina, deve-se sempre excluir variaveis de indentificacao (ID). Embora possuam funcao importante durante etapas de limpeza e organizacao dos dados, sua utilizacao durante a aprendizagem pode levar a resultados equivocados, pois as ID atuariam como preditoras das observacoes existentes embora nao possuam nenhuma informacao relevante além da própria identificacao em si, levando a um problema de sobreidentificacao (overfitting).

Em seguida, o proximo passo e a fatorizacao da caracteristica alvo: se o tumor e benigno ou maligno. Sua transformacao em variavel qualitativa e necessaria ao funcionamento do algoritimo, bem como permite a visualizacao das proporcoes originais atraves de uma tabela.

Por fim, realiza-se a sumarizacao dos atributos com o intuito de identificar a existencia de anomalias, como outliers ou valores missing. Com a percepccao da inexistencia de anomalias, procedeu-se a normalizacao das variaveis numericas, pois, ao se analizar as estatisticas descritivas, percebeu-se como suas grandezas numericas variam, o que poderia causar distorcoes nas relacoes entre as variaveis.

```
## Etapa 2 - Explorando os Dados

# Excluindo a coluna ID
dados <- subset(dados, select = - id)

# Realizado o processo de Factoring em nossa variável resposta (por boa parte dos algorítimos exigir)
dados$diagnosis <- factor(dados$diagnosis, levels = c('B', 'M'), labels = c('Benigno', 'Maligno'))

# Verificado a proporção dos meus dados alvo
round(prop.table(table(dados$diagnosis))*100, digits = 1)
```

```
##
```

```
## Benigno Maligno
##    62.9    37.1
```

```
# Normalização dos dados
summary(dados)
```

```
##    diagnosis    radius_mean      texture_mean   perimeter_mean
##  Benigno:357   Min.   : 6.981   Min.   : 9.71   Min.   : 43.79
##  Maligno:211   1st Qu.:11.697   1st Qu.:16.18   1st Qu.: 75.14
##                Median :13.355   Median :18.86   Median : 86.21
##                Mean   :14.120   Mean   :19.31   Mean   : 91.91
##                3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:103.88
##                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.: 420.2   1st Qu.:0.08629   1st Qu.:0.06481   1st Qu.:0.02954
##  Median : 548.8   Median :0.09587   Median :0.09252   Median :0.06140
##  Mean   : 654.3   Mean   :0.09632   Mean   :0.10404   Mean   :0.08843
##  3rd Qu.: 782.6   3rd Qu.:0.10530   3rd Qu.:0.13040   3rd Qu.:0.12965
##  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 :0.03345   Median :0.1792   Median :0.06152   Median :0.3240
##  Mean   :0.04875   Mean   :0.1811   Mean   :0.06277   Mean   :0.4040
##  3rd Qu.:0.07373   3rd Qu.:0.1956   3rd Qu.:0.06612   3rd Qu.:0.4773
##  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.:0.8331   1st Qu.: 1.605   1st Qu.: 17.850   1st Qu.:0.005166
##  Median :1.1095   Median : 2.285   Median : 24.485   Median :0.006374
##  Mean   :1.2174   Mean   : 2.856   Mean   : 40.138   Mean   :0.007042
##  3rd Qu.:1.4743   3rd Qu.: 3.337   3rd Qu.: 45.017   3rd Qu.:0.008151
##  Max.   :4.8850   Max.   :21.980   Max.   :542.200   Max.   :0.031130
##  compactness_se      concavity_se       points_se
##  Min.   :0.002252   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:0.013048   1st Qu.:0.01506   1st Qu.:0.007634
##  Median :0.020435   Median :0.02587   Median :0.010920
##  Mean   :0.025437   Mean   :0.03186   Mean   :0.011789
##  3rd Qu.:0.032218   3rd Qu.:0.04176   3rd Qu.:0.014710
##  Max.   :0.135400   Max.   :0.39600   Max.   :0.052790
##   symmetry_se        dimension_se        radius_worst   texture_worst
##  Min.   :0.007882   Min.   :0.0008948   Min.   : 7.93   Min.   :12.02
##  1st Qu.:0.015128   1st Qu.:0.0022445   1st Qu.:13.01   1st Qu.:21.09
##  Median :0.018725   Median :0.0031615   Median :14.96   Median :25.43
##  Mean   :0.020526   Mean   :0.0037907   Mean   :16.25   Mean   :25.69
##  3rd Qu.:0.023398   3rd Qu.:0.0045258   3rd Qu.:18.77   3rd Qu.:29.76
##  Max.   :0.078950   Max.   :0.0298400   Max.   :36.04   Max.   :49.54
##  perimeter_worst    area_worst      smoothness_worst  compactness_worst
##  Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729
##  1st Qu.: 84.10   1st Qu.: 515.0   1st Qu.:0.11660   1st Qu.:0.14690
##  Median : 97.66   Median : 685.5   Median :0.13130   Median :0.21185
##  Mean   :107.13   Mean   : 878.6   Mean   :0.13232   Mean   :0.25354
##  3rd Qu.:125.17   3rd Qu.:1073.5   3rd Qu.:0.14600   3rd Qu.:0.33760
##  Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800
```

```
##  concavity_worst   points_worst     symmetry_worst   dimension_worst
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.1565   Min.   :0.05504
##  1st Qu.:0.1145   1st Qu.:0.06473   1st Qu.:0.2504   1st Qu.:0.07141
##  Median :0.2266   Median :0.09984   Median :0.2821   Median :0.08002
##  Mean   :0.2714   Mean   :0.11434   Mean   :0.2898   Mean   :0.08388
##  3rd Qu.:0.3814   3rd Qu.:0.16132   3rd Qu.:0.3177   3rd Qu.:0.09206
##  Max.   :1.2520   Max.   :0.29100   Max.   :0.6638   Max.   :0.20750
```

```
# função base R para normalização -> scale
dados_normalizados <- as.data.frame(scale(dados[2:31]))

# Fazendo uma comparação entre algumas features antes e após
summary(dados[c("radius_mean", "area_mean", "smoothness_mean")])
```

```
##   radius_mean       area_mean      smoothness_mean
##  Min.   : 6.981   Min.   : 143.5   Min.   :0.05263
##  1st Qu.:11.697   1st Qu.: 420.2   1st Qu.:0.08629
##  Median :13.355   Median : 548.8   Median :0.09587
##  Mean   :14.120   Mean   : 654.3   Mean   :0.09632
##  3rd Qu.:15.780   3rd Qu.: 782.6   3rd Qu.:0.10530
##  Max.   :28.110   Max.   :2501.0   Max.   :0.16340
```

```
summary(dados_normalizados[c("radius_mean", "area_mean", "smoothness_mean")])
```

```
##   radius_mean        area_mean       smoothness_mean
##  Min.   :-2.0263   Min.   :-1.4514   Min.   :-3.1106
##  1st Qu.:-0.6877   1st Qu.:-0.6652   1st Qu.:-0.7142
##  Median :-0.2173   Median :-0.2999   Median :-0.0325
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.4710   3rd Qu.: 0.3647   3rd Qu.: 0.6392
##  Max.   : 3.9704   Max.   : 5.2475   Max.   : 4.7756
```

Etapa 3 - Treinando o modelo

Com os dados devidamente preparados, pode-se, agora, comecar o processo de treinamento do modelo. Para isso, carregam-se os pacotes necessarios a execucao, dividi-se nosso conjunto em dados de treino e de teste, e se inicia a criacao do 1o modelo com os parametros padroes.

```
## Etapa 3: Treinando o modelo

# Carregando os pacotes necessários
# install.packages("class")
# install.packages("caTools")
library(caTools)
library(class)

# Criando os dados de treino e os de teste (obs.: neste dataset em especial não seria
# necessário por ser randomizado originalmente)
set.seed(69)
amostra <- sample.split(dados$diagnosis, SplitRatio = 0.70)
dados_treino <- as.data.frame(subset(dados_normalizados, amostra == T))
dados_teste <- as.data.frame(subset(dados_normalizados, amostra == F))

# Criando os labels para identificação no modelo
dados_treino_labels <- subset(dados[1], amostra == T)[,1]
```

```r
dados_teste_labels <- subset(dados[1], amostra == F)[,1]

# Criando o Modelo
modelo <- knn(train = dados_treino,
              test = dados_teste,
              cl = dados_treino_labels)
```

Etapa 4 - Avaliando a Performance do Modelo

Nesta etapa, acontecerá a analise da eficacia do modelo. Para se chegar a esse resultado, o pacote 'gmodels' será carregado e utilizado para construir uma matriz de confusao, ou tabela cruzada, com o objetivo de se identifcar os casos corretamente previsto, no caso, com 4 falso negativos ou 97,6 % de acuracia.

```r
# Carregando pacote necessario
# install.packages("gmodels")
library(gmodels)

# Criando uma tabela cruzada dos dados previstos x dados atuais, ou seja, uma ConfusionMatrix e analisa
CrossTable(x = dados_teste_labels, y = modelo, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  170
##
##
##                     | modelo
## dados_teste_labels |   Benigno |   Maligno | Row Total |
## -------------------|-----------|-----------|-----------|
##            Benigno |       104 |         3 |       107 |
##                    |     0.972 |     0.028 |     0.629 |
##                    |     0.972 |     0.048 |           |
##                    |     0.612 |     0.018 |           |
## -------------------|-----------|-----------|-----------|
##            Maligno |         3 |        60 |        63 |
##                    |     0.048 |     0.952 |     0.371 |
##                    |     0.028 |     0.952 |           |
##                    |     0.018 |     0.353 |           |
## -------------------|-----------|-----------|-----------|
##       Column Total |       107 |        63 |       170 |
##                    |     0.629 |     0.371 |           |
## -------------------|-----------|-----------|-----------|
##
##
```

```r
taxa_erro_inicial = mean(dados_teste_labels != modelo)
```

Etapa 5 - Otimizacao do Modelo

Por último, como objetivo do trabalho, analizar-se-a a mudanca na performance do modelo atraves da variacao do parametro k, ou seja, o numero de vizinhos mais proximos (em distancia euclidiana) utilizados para definir a classificacao. Assim será feito um plot, com o uso do pacote 'ggplot2', demonstrando como a performance, de fato, altera-se consideravelmente com uma adocao de 'k' variando de 1 ate 25.

```r
## Otimizacao do Modelo

# Carregando pacote necessario
# install.package('ggplot2')
library(ggplot2)

# Calculando função taxa de erro em relação ao tamanho do k
prev = NULL
taxa_erro = NULL
k_values = 1:25
#obs.: sempre que for realizar um loop, é bom costume começá-los vazios para garantir isso
suppressWarnings(
  for(i in k_values){
    set.seed(101)
    prev = knn(train = dados_treino,
               test = dados_teste,
               cl = dados_treino_labels,
               k = i)
    taxa_erro[i] = mean(dados_teste_labels != prev)
  })

df_erro <- data.frame(taxa_erro, k_values)
df_erro
```

```
##     taxa_erro k_values
## 1  0.03529412        1
## 2  0.02941176        2
## 3  0.01764706        3
## 4  0.03529412        4
## 5  0.01764706        5
## 6  0.02352941        6
## 7  0.01764706        7
## 8  0.02352941        8
## 9  0.02352941        9
## 10 0.01764706       10
## 11 0.02352941       11
## 12 0.02941176       12
## 13 0.02941176       13
## 14 0.02941176       14
## 15 0.03529412       15
## 16 0.02941176       16
## 17 0.02941176       17
## 18 0.02941176       18
## 19 0.02352941       19
```

```
## 20 0.02352941       20
## 21 0.02352941       21
## 22 0.02352941       22
## 23 0.02352941       23
## 24 0.02352941       24
## 25 0.02352941       25
```

```r
# Plotando a relação entre as duas variáveis
ggplot(df_erro, aes(x = k_values, y = taxa_erro)) +
  geom_point()+
  geom_line(lty = "dotted", color = 'red') +
  labs(title = 'Taxa de Erro em Função dos Valores de K',
       y = 'Taxa de Erro', x = 'Valores de K') +
  theme_classic()
```