Conteúdo

1. VISÃO GERAL E ENTENDENDO O DATASET

O objetivo deste trabalho é aplicar conhecimento de aprendizado indutivo de forma supervisionada, aplicando as tarefas de classificação e regressão sob um dataset controlado referente a câncer de mama diagnosticado no estado de Wisconsin nos Estados Unidos.

1.1 COMEÇANDO A EXPLORAÇÃO DOS DADOS

Importamos os arquivos .data do dataset, identificamos o arquivo wdbc.data, como sendo o que contia o dataset a ser trabalhado. Identificamos através do número de instâncias no mesmo e o especificado no link: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic) (Links para um site externo.).

Ao perceber que as colunas não possuíam um header, optamos por pesquisar sobre o dataset para dar significado a todos os atributos, para aplicarmos de forma correta o algoritmo KNN e árvore de decisão. Utilizamos outra especificação do mesmo dataset https://www.kaggle.com/uciml/breast-cancer-wisconsin-data, para assim renomear e dar sentido aos atributos que serão trabalhados.

1.2 ENTENDIMENTO DOS DADOS

Avaliando os atributos renomeados juntamente com os dados descritos no site do dataset entendemos que esse é o problema de pacientes que desenvolveram o câncer de mama, seja de forma maligna ou benigna como uma das features já classifica.

Os dados formam um dossiê do câncer observado no paciente, bem como espessura, tonalidade, média de diâmetro e etc.

1.3 ENTENDIMENTO DO PROBLEMA PROPOSTO

Ambas atividades visam prever através das features avaliadas no nosso estudo se o câncer desenvolvido pelo paciente é maligno ou benigno. Dessa forma a resultante seria essa distinção de benigno e maligno, além de poder gerar conhecimentos como quais são as principais características de tumores de mama que possibilitam diferenciar entre maligno ou benigno.

2. DESENVOLVIMENTO

Visamos mesclar o código e o visto em aula para preparar os dados para aplicação dos algoritmos de machine learning e realizar a diferenciação entre um cancer maligno e benigno de acordo com suas características obtendo assim uma classificação no caso com o algoritmo KNN que a partir de um conjunto de treino classifica os objetos de acordo com sua proximidade com objetos vizinhos considerando um parâmetro k que diz o quantidade de vizinhos que será considerado para a aplicação e obtenção dos resultados comparando com o conjunto de treino e teste. E também implementação do algoritmo de árvore de decisão que classifica instâncias baseado nos valores de seus atributos a fim de ter um nó raiz como melhor classificador a partir de candidatos que apresentam menor função de custo.

2.1 EXPLORAÇÃO ESTATÍSTICA DOS DADOS

Inicialmente tentamos verificar a quantidade de linhas e colunas e logo em seguida explorar a disposição dos dados, a dataset apresentou 569 linhas e 32 colunas. Observamos que o dataset inicialmente é importado com suas features do tipo caracter mesmo e que os dados eram quantitativo, desta forma foi feito um ajuste no importe no dataset para serem do tipo num e facilitar a manipulação da estrutura e geração de gráficos para exploração analitica.

Código

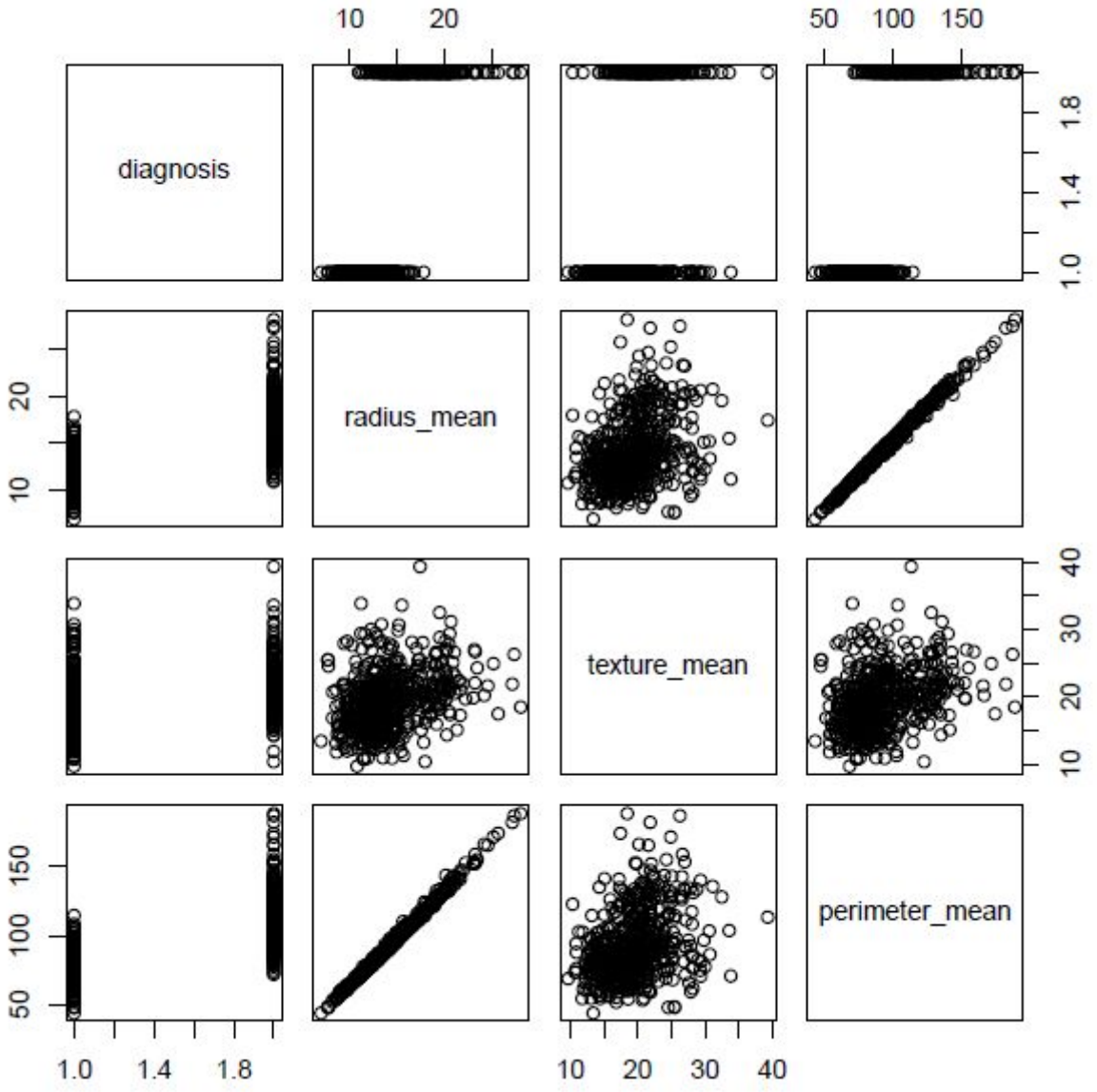
```
data <- read.csv2("wdbc.data", sep =",", na.strings =
c('','NA','na','N/A','n/a','NaN','nan'), header = FALSE,  dec=".",
stringsAsFactors=FALSE)

# Entendimento prévio do dataset
dim(cancers)
str(cancers)
```

Como mencionado as features inicialmente não tinham a especificação do que cada coluna correspondia no header, então após pesquisa foi feito a organização do cabeçalho para tratativa e possível trabalho com os dados.

Código

```
cancers <- data %>%
  rename(
    id = V1,
    diagnosis = V2, # Significado: Câncer de mama M -Maligno, B -
Beligno
    radius_mean = V3, # Significado: Raio medio do câncer
    texture_mean = V4, # Significado: Média de escala de cinza do
câncer
    perimeter_mean = V5, # Significado: Media de perimetro do câncer
    area_mean = V6, # Significado: Média de area do câncer
    smoothness_mean = V7,
    compactness_mean = V8,
    concavity_mean = V9,
    concave_points = V10,
    symmetry_mean = V11,
    fractal_dimension = V12,
    radius_se = V13,
    texture_se = V14,
    perimeter_se = V15,
    area_se = V16,
    smoothness_se = V17,
    compactness_se = V18,
    concavity_se = V19,
    concave_points_se = V20,
```

```
    symmetry_se = V21,
    fractal_dimension_se = V22,
    radius_worst = V23,
    texture_worst = V24,
    perimeter_worst = V25,
    area_worst = V26,
    smoothness_worst = V27,
    compactness_worst = V28,
    concavity_worst = V29,
    concave_points_worst = V30,
    symmetry_worst = V31,
    fractal_dimension_worst = V32
  )

# Plot geral das features
minCancers <- cancers[ , 2:4]
plot(minCancers)
```

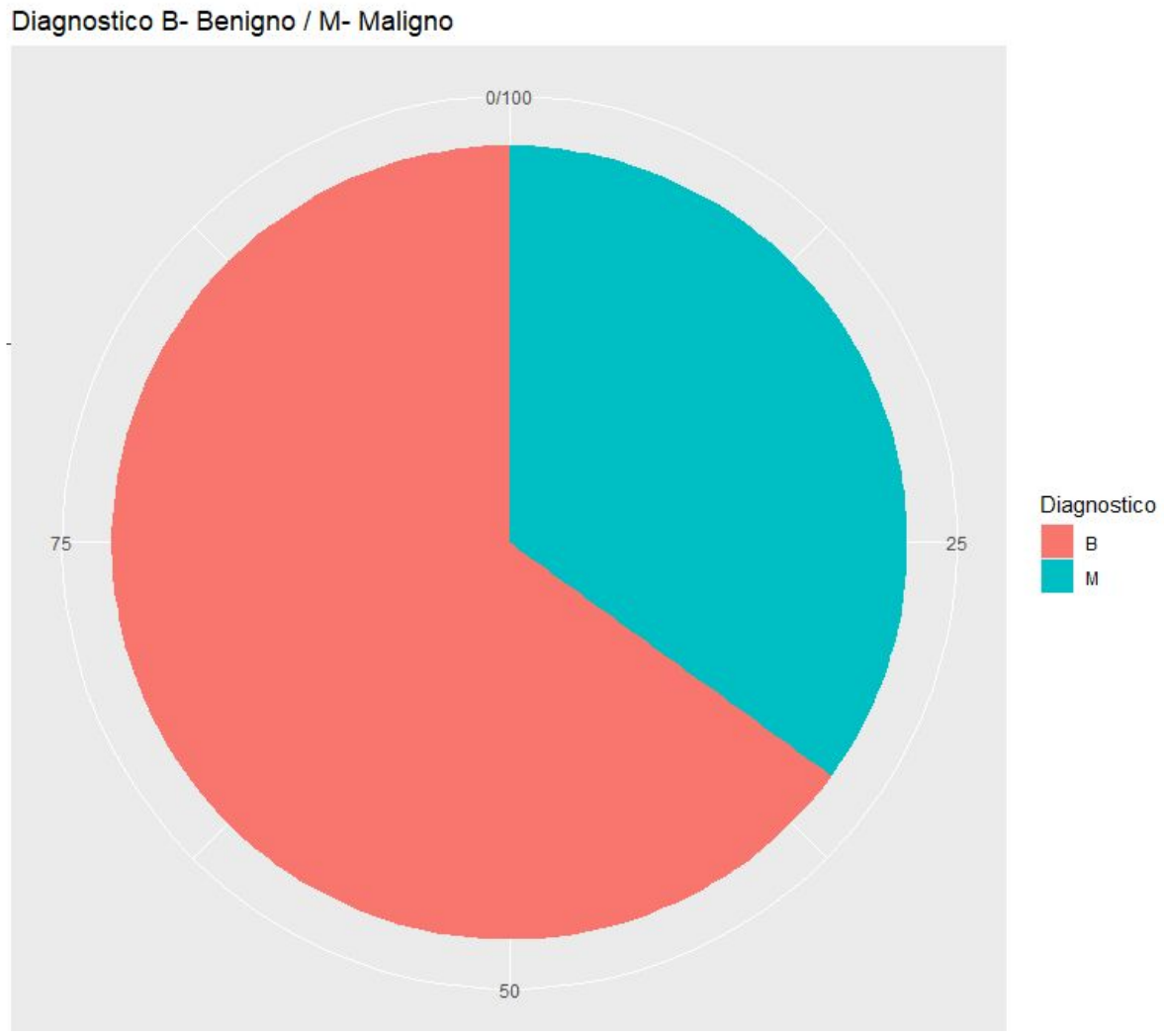


Foi realizada uma explorações centradas ao problema, observando quantos diagnósticos do tipo benigno e maligno havia no dataset. Da mesma forma uma análise para ver se o dataset é balanceado para o classificador, para isso primeiramente foi feito uma verificação se existiam dados vazios e perdidos para alguma feature e a contabilização de quantos dados são classificados em câncer maligno sendo de 212 e dados para câncer benigno sendo de 357, melhor observado no gráfico a seguir.

Código

```
# Verificar se existem dados vazios
sum(is.na(cancers)) # Nenhum valor de atributo ausente
```

```
sum(str_count(cancers$diagnosis, "M")) # 212 Maligno
sum(str_count(cancers$diagnosis, "B")) # 357 Benigno
```



Desta forma notamos que o dataset está desbalanceado com mais exemplares do tipo benigno podendo gerar uma estrutura de árvore prejudicada quando aplicar o algoritmo para a árvore de decisão.

Código -> Gráfico de pizza exemplares para benigno e maligno

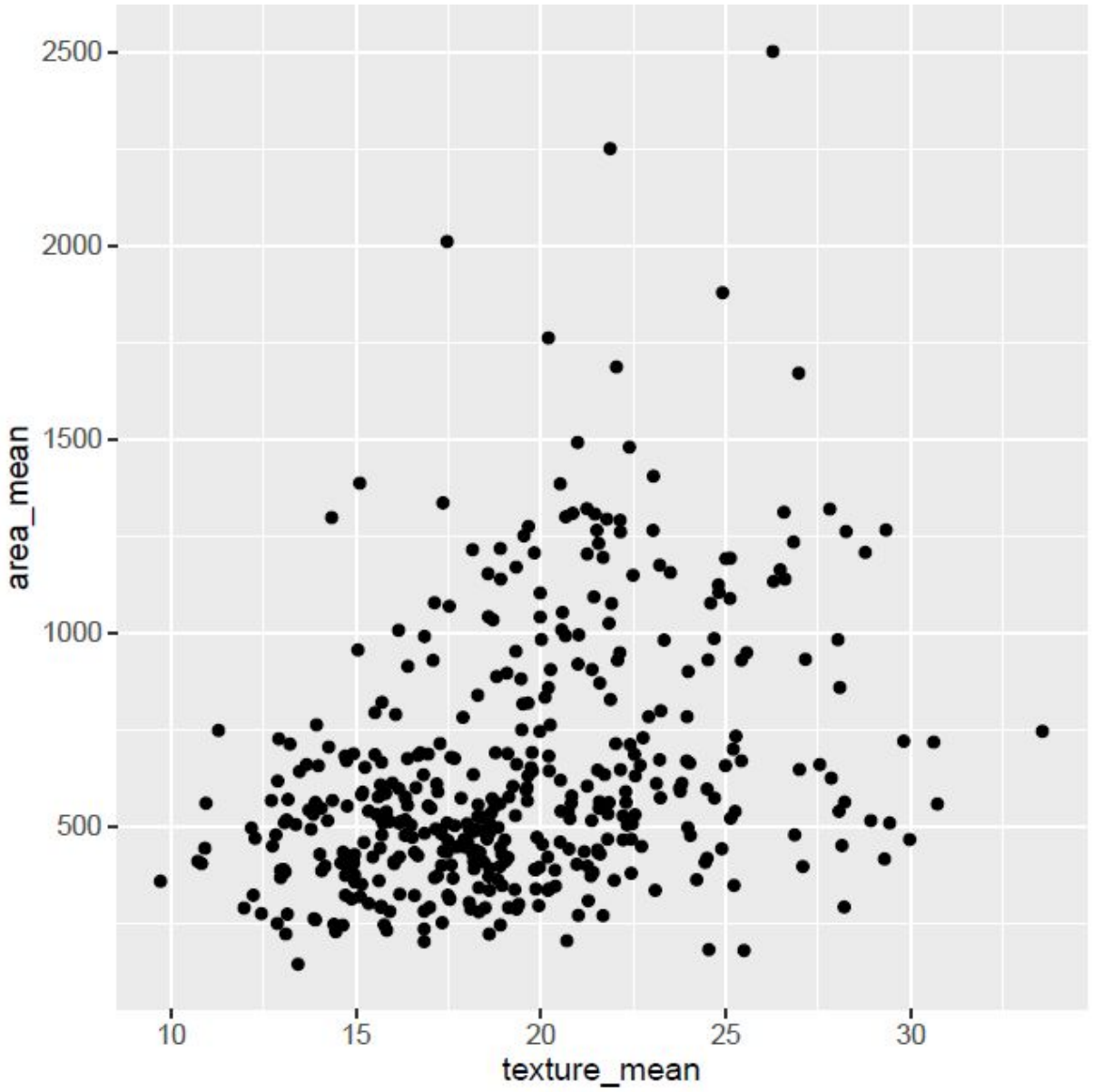
```
# Gráfico de pizza - quantidade de cancer diagnosticado como
maligno ou benigno
diagnostico_frequencia <-table(train$diagnosis)
diagnostico_porcentagem <-
```

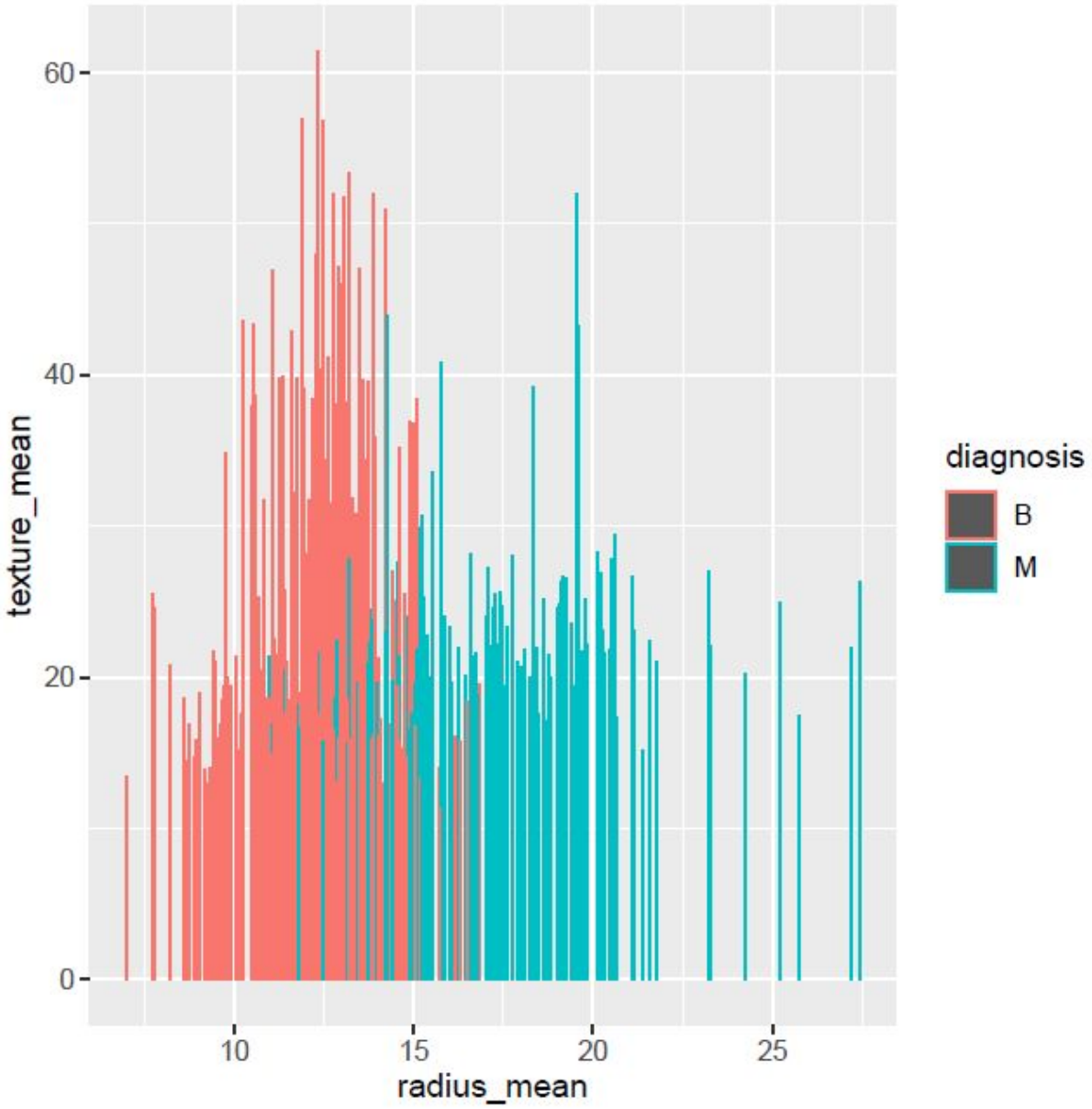
```
round(prop.table(diagnostico_frequencia)*100) # "% of total sum of
row of column"
diagnostico_porcentagem # B -> 65% e M -> 35%
diagnostico_tabela <-as.data.frame(diagnostico_porcentagem)
colnames(diagnostico_tabela)[1] <- "Diagnostico

pie <- bp + coord_polar("y", start=0) +
  labs(x=NULL,y=NULL,title="Diagnostico B- Benigno / M- Maligno")
pie
```

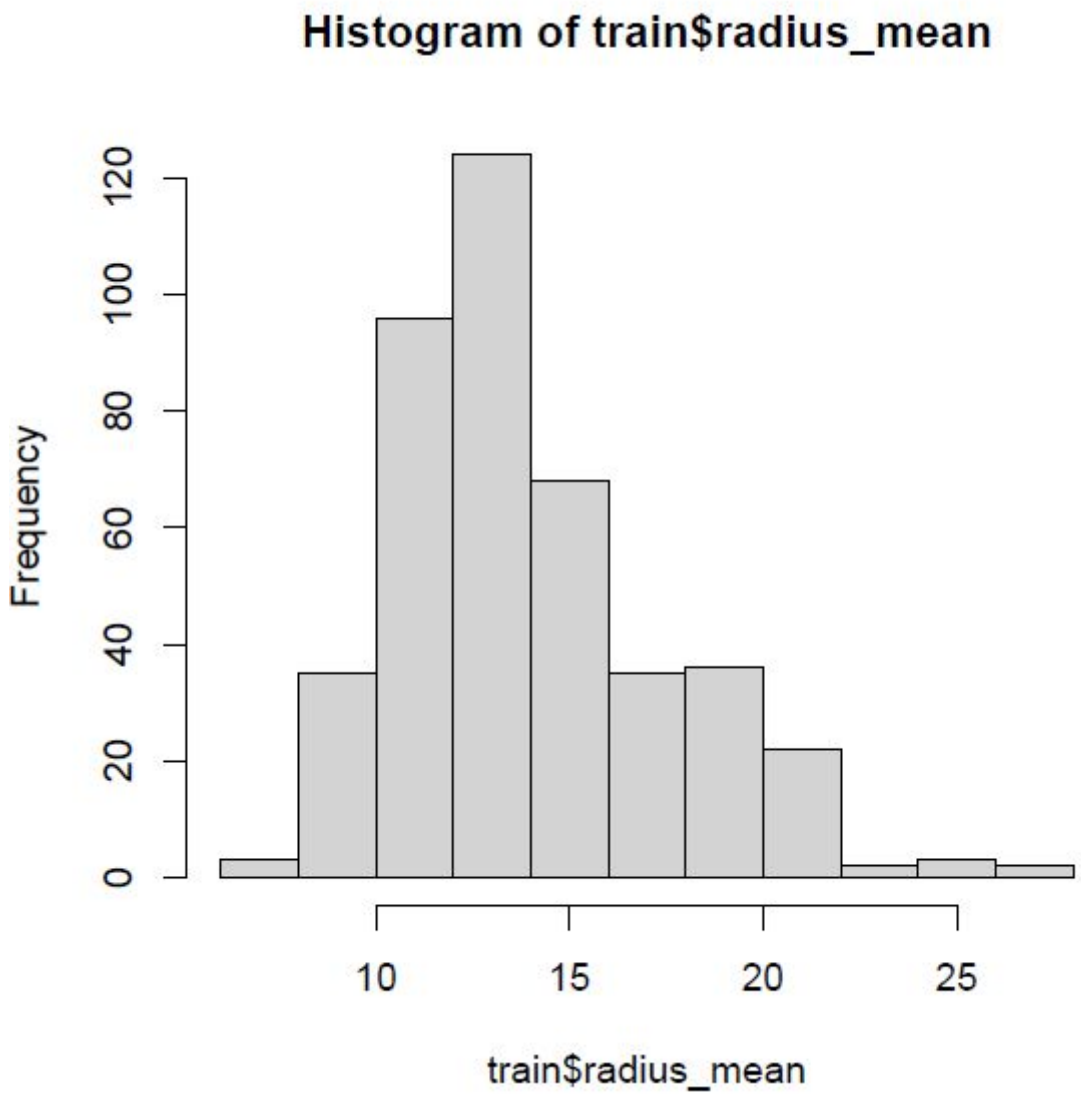
Para uma melhor análise notamos que a feature ID continha duplicatas e seus valores não eram interessante para as gerações dos gráficos e análises nos algoritmos de IA, portanto foi decidido removê la do dataset.

```
# Remoção da feature id para não ser considerado em plots e em
algoritmos de IA
cancers=subset(cancers,select = -id)
```

O restante das análises centramos nas features em si, observando desde espalhamento a comportamento de valores individuais. Tendo observar relações em features como área média de um câncer com sua textura, podendo considerar que existe relação de acordo com a concentração de nuvem de pontos gerada, ou seja, a indícios que os cancer tem apresentam medidas semelhantes assim como área nos exemplares do dataset em análise.


2500
2000
1500
1000
500
area_mean
10
15
20
25
30
texture_mean

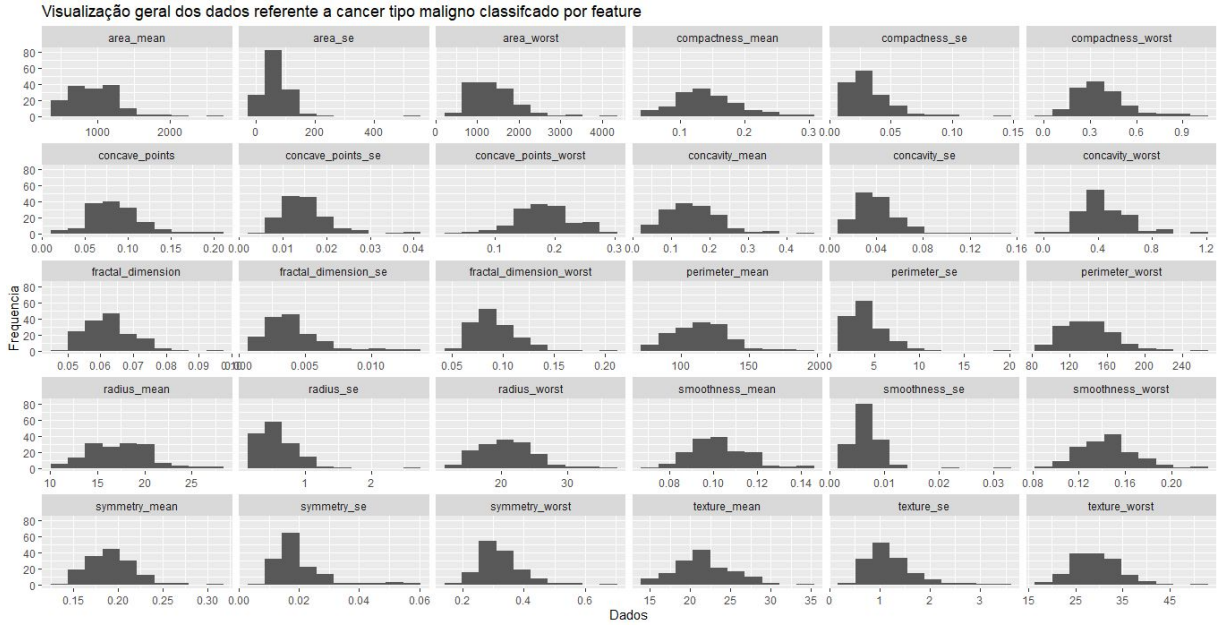

60
40
20
0
texture_mean
10
15
20
25
radius_mean
diagnosis
B
M



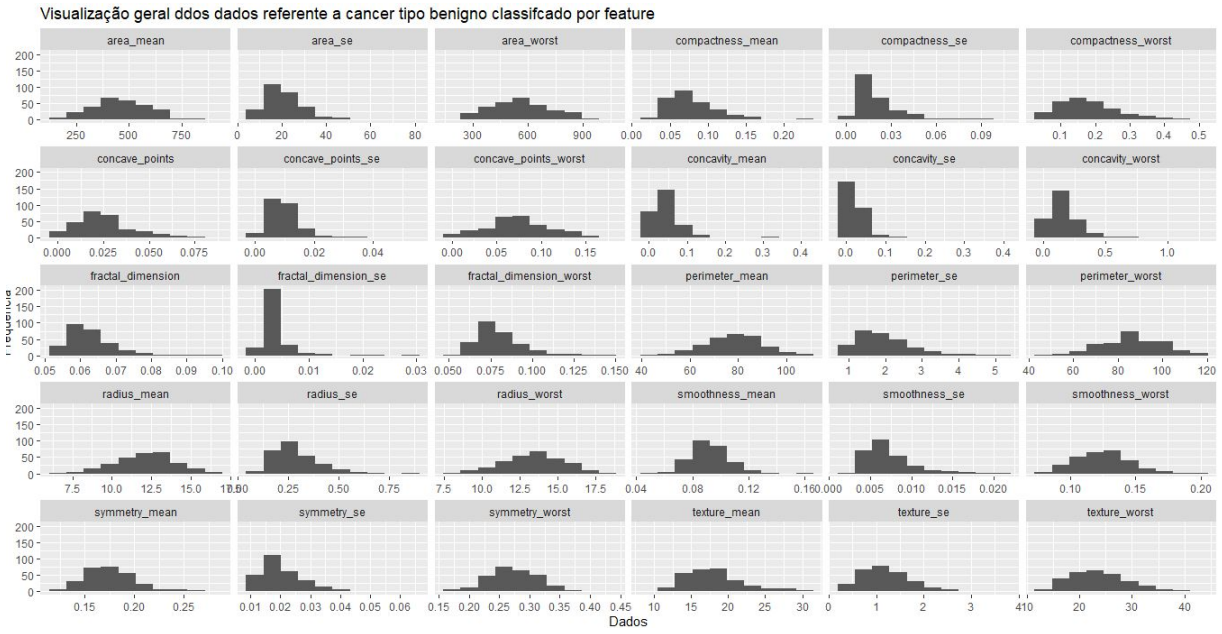
Observações	Código
A tendência que observamos é de grande concentração de ambas as classes para algumas features, isso de início, para aplicação do algoritmo KNN pode ter um impacto linear nos resultados, pois haveria uma proporção de cálculo de distância Euclidiana.	`hist(train$radius_mean)` `ggplot(data=train, aes(x=radius_mean, y=texture_mean, group=diagnosis, color=diagnosis)) +` `  geom_bar(stat='identity')` `ggplot(train, aes(x=texture_mean, y=area_mean)) + geom_point()`

As últimas observações estatísticas foram histogramas de todas as features separadas por classificação, buscando visão macro do comportamento do dataset. Assim foi possível observar que as distribuições de frequência para os exemplares são diferentes pois tem menos exemplares para câncer tipo maligno, no entanto ao analisar ambas distribuições de frequência em geral tem o mesmo comportamento e semelhança na curva nas features.

Maligno	```train_M <- filter(train, diagnosis == "M") ggplot(gather(train_M), aes(x = value)) + geom_histogram(bins = 10) + facet_wrap(~key, scales = 'free_x') + labs(x = "Dados", y = "Frequencia") + ggtitle("Visualização geral dos dados referente a cancer tipo maligno classificado por feature")```



Benigno	```train_B <- filter(train, diagnosis == "B") ggplot(gather(train_B), aes(x = value)) + geom_histogram(bins = 10) + facet_wrap(~key, scales = 'free_x') + labs(x = "Dados", y = "Frequencia") + ggtitle("Visualização geral dos dados referente a cancer tipo benigno classificado por feature")```



3. Aplicação dos algoritmos

Foi separado o dataset em conjunto de treino e teste, utilizando 75% dos exemplares para treino e 25% para teste.

```
# Dataset de treino 75%  da base original e teste 25% base original.

set.seed(123)
smp_size <- floor(0.75 * nrow(cancers))
train_ind <- sample(seq_len(nrow(cancers)), size = smp_size)

train <- cancers[train_ind, ]
test <- cancers[-train_ind, ]
```

3.1 KNN

Enunciado proposto: Executar o algoritmo KNN com k = 1,3,5 e 11 vizinhos.

Solução: Implementamos uma função geradora do KNN, similar ao proposto em forma de exercício em aula, dessa forma, executamos os testes com K vizinhos coletando os resultados. Com a remoção da coluna ID que não beneficiava a análise em questão, a coluna diagnosis passou a ser a primeira coluna do conjunto, desta forma para otimizar o processamento realizamos uma função genérica de verificação knn e que poderia ser implementada a outros conjuntos ou classificadores pois é passado como parâmetro a coluna na qual quer ser trabalhada para a classificação.

Além deste parâmetro a função foi composta pelo, dataset de treino, dataset de teste e o vetor K requerido, obtendo como saída a matriz confusão e o índice de acerto dado o conjunto de teste em comparação ao conjunto de treino.

```
# train[,1] representa coluna diagnosis

verificaknn <- function(datasetTrain, datasetTest, vetorK,
posicaoClassificador){
  classesTrain <- datasetTrain[ ,posicaoClassificador]
  datasetTrain <- datasetTrain[ , -posicaoClassificador]

  classesTest <- datasetTest[ , posicaoClassificador]
  datasetTest <- datasetTest[ , -posicaoClassificador]

  result <- knn(datasetTrain, datasetTest, classesTrain, vetorK)

  # Matriz de confusão
  print("Matriz confusão:")
  print(as.matrix(table(classesTest, result)))
  matriz <- as.matrix(table(classesTest, result))

  # Indíce de acerto
  acc <- sum(diag(matriz))/nrow(datasetTest)
  print("Indice de acerto:")
  print(acc)
```

```
}
```

3.1.1 RESULTADOS KNN

Saída para k = 1

```
> # k = 1
> verificaknn(train, test, 1, 1)

"Matriz confusão:"
           result
classesTest  B  M
          B 77  3
          M  8 55

"Indice de acerto:"
0.9230769
```

Saída para k = 3

```
> # k = 3
> verificaknn(train, test, 3, 1)

"Matriz confusão:"
           result
classesTest  B  M
          B 78  2
          M  5 58

"Indice de acerto:"
0.951049
```

Saída para k = 5

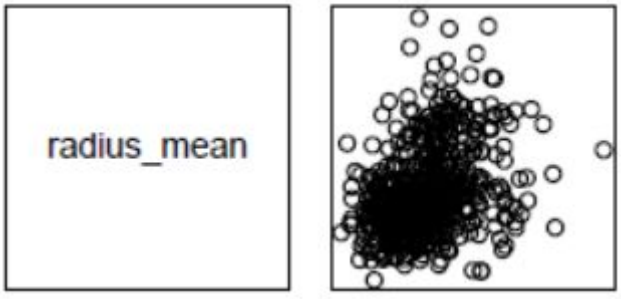
```
> # k = 5
> verificaknn(train, test, 5, 1)

"Matriz confusão:"
           result
classesTest  B  M
          B 77  3
          M  4 59
```

	"Indice de acerto:" 0.951049

Saída para k = 11	> # k = 11 > verificaknn(train, test, 11, 1) "Matriz confusão:" result classesTest B M B 78 2 M 5 58 "Indice de acerto:" 0.951049

Os resultados apresentaram um alto índice de acerto na predição, mostra que a classificação de benigno e maligno foi aplicada com sucesso pelo algoritmo de classificação KNN.

A partir de um valor de K >=3, o índice de acerto se manteve o mesmo. Nossa hipótese, seria de que na exploração estatística, verificamos a densidade dos dados e proximidade de valores no plano, como segue:



Portanto ao aumentar a quantidade de K's avaliados pelo algoritmo, supomos que a diferencia de distância desses elementos para cálculo de distância Euclidiana acontece de forma linear a partir desse valor.

3.1 ÁRVORE DE DECISÃO

Enunciado proposto:

- Executar o algoritmo de árvore de decisão.
- Verificar a taxa de acertos da árvore.
- Verificar a ordem de importância das características do problema no modelo da árvore. Alguma característica não foi utilizada no modelo? Comente.

Solução:

Para a utilização do algoritmo da árvore de decisão, montamos uma função que gerava de forma automática a matriz de confusão e o nosso índice de acerto do modelo, usando o dataset de teste.

Código	

```
modelo <- rpart(diagnosis~., train, method = "class", control =
rpart.control(minisplit = 1))
plot <- rpart.plot(modelo, type = 3)

verificaDesicionTree <- function(modelo, datasetTest,
posicaoClassificador){
  classesTest <- datasetTest[ ,posicaoClassificador]
  datasetTest <- datasetTest[ , -posicaoClassificador]

  pred <- predict(modelo, datasetTest, type = "class")

  # Matriz de confusão
  print("Matriz confusão:")
  matriz <- as.matrix(table(classesTest, pred))
  print(matriz)

  # Indíce de acerto
```

```
  acc <- sum(diag(matriz))/nrow(datasetTest)
  print("Indice de acerto:")
  print(acc)
}
```

Para a verificação da taxa de acertos fizemos o experimento de executar a função da árvore de decisão implementada acima para diferentes conjuntos do dataset, trabalhando o conceito de, uma vez que eu tenha o modelo, sua predição possa ser aplicada a qualquer subconjunto do dataset e não havendo uma dependência apenas do treino.

Resultados:

- **Dataset teste**

Terminal

```
> verificaDesicionTree(modelo, test, 1)

"Matriz confusão:"
           pred
classesTest  B  M
          B 74  6
          M  6 57

"Indice de acerto:"
0.9160839
```

O conjunto de teste contém 80 exemplares benignos acertando 74 e 6 destes benigno classificou como maligno. Da mesma forma o conjunto contém 63 exemplares malignos e a árvore de decisão classificou 57 como malignos e 6 destes como benigno.

- **Dataset completo**

Terminal

```
> verificaDesicionTree(modelo, cancers, 1)

"Matriz confusão:"
           pred
classesTest   B   M
```

	`` B 346 11 M 20 192 "Indice de acerto:" 0.9455185 ```

O dataset completo contém 357 exemplares benignos acertando 346 e 11 destes benigno classificou como maligno. Da mesma forma o conjunto completo contém 212 exemplares malignos e a árvore de decisão classificou 192 como malignos e 20 destes como benigno.

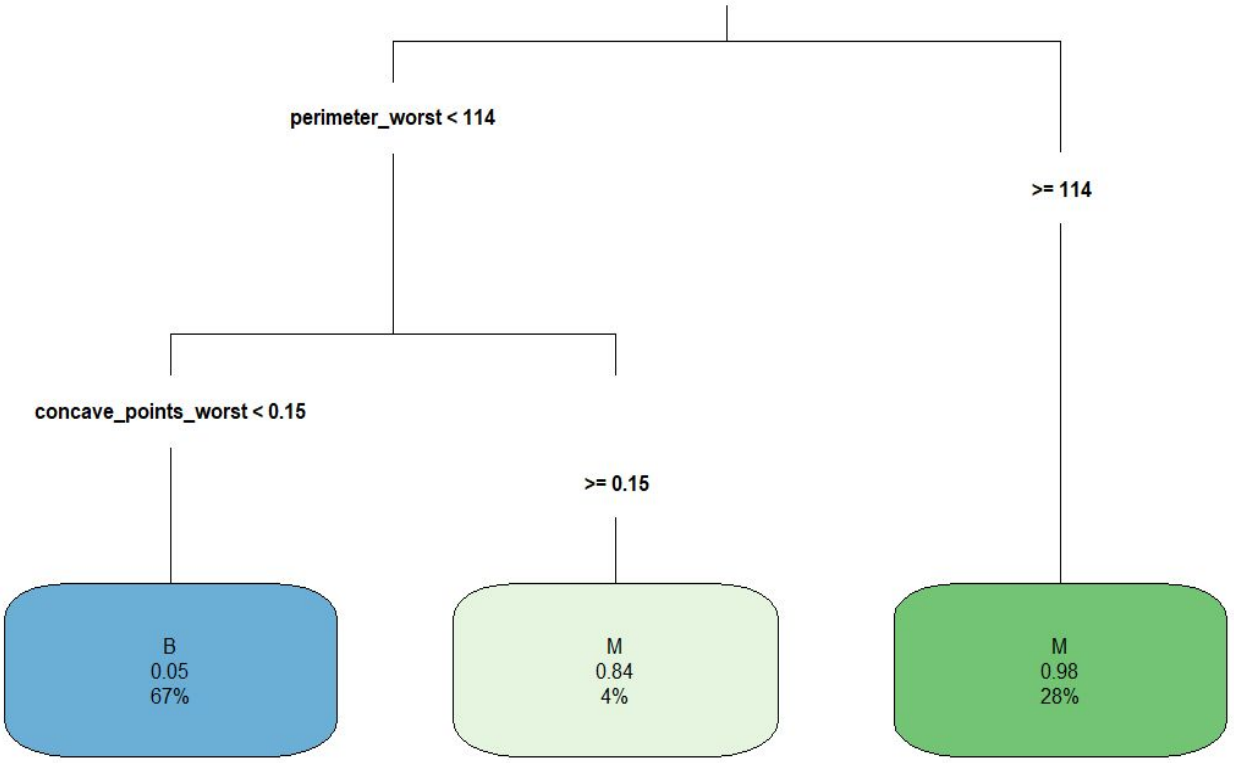
- **Dataset de treino**

Terminal	``` > verificaDesicionTree(modelo, train, 1) "Matriz confusão:" pred classesTest B M B 272 5 M 14 135 "Indice de acerto:" 0.9553991 ```

O dataset de treino contém 277 exemplares benignos acertando 272 e 5 destes benigno classificou como maligno. Da mesma forma o conjunto de treino contém 149 exemplares malignos e a árvore de decisão classificou 135 como malignos e 14 destes como benigno.

Nosso modelo fora gerado da seguinte forma, gerando a seguinte árvore de decisão.

Código	``` modelo <- rpart(diagnosis~., train, method = "class", control = rpart.control(minisplit = 1)) plot <- rpart.plot(modelo, type = 3) ```



Assim obtivemos os seguintes resultados para os conjuntos aplicados na árvore de decisão.

- Dataset de teste: 0.9160839 ou 91% índice de acerto
- Dataset de completo: 0.9455185 ou 94% índice de acerto
- Dataset de treino: 0.9553991 ou 95% índice de acerto

Portanto olhando os resultados para diferentes conjuntos consideremos uma acurácia relativamente boa, em geral acima de 90% assim, o modelo gerado a partir de uma árvore de decisão teria grande chances de classificar corretamente um exemplar como benigno ou maligno. Por ser um conjunto de dataset originalmente desbalanceado com mais exemplares benignos do que malignos pode ocorrer algum tipo de influência na estrutura de árvore, indicando a causa da pequena variação de erro no índice de acerto, não atingindo acurácias próximas a 100% de acerto, assim como medida corretiva para uma melhor análise uma estratégia pensada seria aplicar um balanceamento do dataset inicial antes de separar em conjunto de treino e teste.

Analisando a estrutura da árvore de decisão gerada é possível notar que as melhores features que contribui para classificação entre um diagnóstico de câncer

maligno ou benigno é o perímetro e concavidade da média tipo worst. Essa features em conjunto é possível realizar uma classificação para o diagnóstico segundo a árvore, para isso plotamos um gráfico para comprovar e analisar se a estrutura é confiavél.

Desta forma é possível notar que existe uma separação bem estabelecida e definida com uma nuvem de pontos para benigno e outra nuvem de pontos para maligno. Porém dá para notar uma sobreposição onde os pontos de ambos tipos se misturam indicando uma das causas do resultado gerado pelo modelo conter erros no índice de acerto. Além de notar um ponto outlier da classe maligno entre a nuvem de pontos benigno.

Analisando a ordem de importância gerada pela estrutura da árvore de decisão não foi aproveitado todas as características presentes no dataset, uma hipótese é que a função de custos dessas features teve um valor baixo, sendo considerada não relevante para diferenciar um diagnóstico tipo maligno e benigno, visto que o dataset é complexo considerando o número de informações presentes. Desta forma a função de custo para a característica de medidas de perímetro e concavidade teve um valor alto, sendo comprovado pelo plot gráfico, nesse sentido

a análise dessas duas característica por um médico poderia auxiliá-lo no diagnóstico de tipo de câncer de mama.

4. CONCLUSÃO E ANÁLISE DE RESULTADOS

Em classificação por KNN o índice de acerto foi relativamente baixo com 63% utilizando o melhor K, isso devido a muitos dados de diferentes features com distâncias muito próximas, nos revelando uma possível tendência a ter diversas features que contribuem para o surgimento de resposta à doença cardíaca. Pensando assim, o aprendizado de máquina por KNN, teve uma acurácia em mais da metade dos dados comparados com o dataset de teste, ou seja, esse processo de machine learning poderia contribuir com mais da metade de chance de acerto na predição de um diagnóstico de pacientes ter ou não a doença, e se alinhar as outras porcentagem conhecimento próprio do médico, poderia ter bons resultado em predição desse tipo de diagnóstico.

Na classificação utilizando a árvore de decisão aplicando o conjunto de teste para ser classificado, o resultado obtido foi um pouco menor que ao KNN com 59% de taxa de acerto, e como nesse algoritmo não há uma memorização para ocorrer a comparação, podemos aplicar outros conjuntos de dados, assim utilizamos o conjunto de treino e obtivemos uma acurácia de 83%, nos mostrando que com um conjunto maiores de dados sobre esse modelo de árvore gerado, a taxa de acerto obtém os melhores resultado para classificação.

Com a árvore de decisão foi possível entender que as features que mais contribuem para diferenciar se há resposta para doença cardíaca são respectivamente são consumo de tabaco, colesterol de lipoproteína de baixa densidade, pressão arterial sistólica, histórico familiar, comportamento do tipo A e idade.

O principal objetivo do trabalho era analisar os dados de um paciente e predizer se tem ou não doença cardíaca, assim como o grau de tendência a ser desenvolvido. Por isso a ideia da aplicação de métodos de classificação de machine learning para predizer esse tipo de diagnóstico. A expectativa inicial era atingir alto índice de acurácia acima de 90%, por entender que o dataset apresenta um bom

classificador para a questão que estávamos procurando responder, o classificador chd, no entanto, o nível de acurácia foi mais baixo. Procuramos entender o motivo para o baixo índice de acerto, adotamos primeiramente o balanceamento do dataset em relação a chd, onde foi mantido o nível de acerto. Ao analisar a árvore gerada entendemos que contém muitos ramos, ou seja, muitas features com alta função de custo. Desta forma, seria errado dizer que se um paciente tem alto consumo de tabaco e está em idade avançada que ele vai ter a doença cardíaca coronariana, pois isso depende de mais fatores como qual o seu nível de gordura localizada entre outros.

Portanto entendemos que a classificação oferecida pelo machine learning tem grande relevância para suporte na área da saúde uma vez que são mais de um tipo de característica que pode levar um pessoa ter uma doença cardíaca coronariana, assim juntamente com o conhecimento de um médico esse trabalho de classificação sobre tais dados podem contribuir em situações reais ligadas a area da saude.

ANEXO: 4. CÓDIGO

```
library("tidyr")
library(dplyr)
library(ggplot2)
library(dslabs)
library(reshape2)
library(stringr)
library(class)
library(rpart)
library(rpart.plot)
```

```
data <- read.csv2("wdbc.data", sep =",", na.strings =
c('','NA','na','N/A','n/a','NaN','nan'), header = FALSE,  dec=".",
stringsAsFactors=FALSE)

# Renomeando as colunas com base no indicado no link do dataset
https://www.kaggle.com/uciml/breast-cancer-wisconsin-data
cancers <- data %>%
 rename(
   id = V1,
   diagnosis = V2, # Significado: Câncer de mama M -Maligno, B -
Beligno
   radius_mean = V3, # Significado: Raio medio do câncer
   texture_mean = V4, # Significado: Média de escala de cinza do câncer
   perimeter_mean = V5, # Significado: Media de perimetro do câncer
   area_mean = V6, # Significado: Média de area do câncer
   smoothness_mean = V7,
   compactness_mean = V8,
   concavity_mean = V9,
   concave_points = V10,
   symmetry_mean = V11,
   fractal_dimension = V12,
   radius_se = V13,
   texture_se = V14,
   perimeter_se = V15,
   area_se = V16,
   smoothness_se = V17,
   compactness_se = V18,
   concavity_se = V19,
   concave_points_se = V20,
   symmetry_se = V21,
   fractal_dimension_se = V22,
   radius_worst = V23,
   texture_worst = V24,
   perimeter_worst = V25,
   area_worst = V26,
   smoothness_worst = V27,
   compactness_worst = V28,
   concavity_worst = V29,
   concave_points_worst = V30,
   symmetry_worst = V31,
   fractal_dimension_worst = V32
 )
```

```r
# Verificar se existem dados vazios
sum(is.na(cancers)) # Nenhum valor de atributo ausente

# Entendimento previo do dataset
dim(cancers)
str(cancers)
sum(str_count(cancers$diagnosis, "M")) # 212 Maligno
sum(str_count(cancers$diagnosis, "B")) # 357 Benigno


# Remoção da feature id para não ser considerado em plots e em
algoritmos de IA
cancers=subset(cancers,select = -id)
#Plot geral das features
minCancers <- cancers[ , 1:4]
plot(minCancers )


# Dataset de treino 75%  da base original e teste 25% base original.
set.seed(123)
smp_size <- floor(0.75 * nrow(cancers))
train_ind <- sample(seq_len(nrow(cancers)), size = smp_size)

train <- cancers[train_ind, ]
test <- cancers[-train_ind, ]


# Gráficos para verificar se visualmente os dados são
classificaveis----------------------------------------------

# Gráfico de pizza - quantidade de cancer diagnosticado como maligno ou
benigno
diagnostico_frequencia <-table(train$diagnosis)
diagnostico_porcentagem <-
round(prop.table(diagnostico_frequencia)*100) # "% of total sum of row
of column"
diagnostico_porcentagem # B -> 65% e M -> 35%
diagnostico_tabela <-as.data.frame(diagnostico_porcentagem)
colnames(diagnostico_tabela)[1] <- "Diagnostico"

bp<- ggplot(diagnostico_tabela, aes(x="", y= Freq, fill=Diagnostico))+
```

```
 geom_bar(width = 1, stat = "identity")

pie <- bp + coord_polar("y", start=0) +
 labs(x=NULL,y=NULL,title="Diagnostico B- Benigno / M- Maligno")
pie

# plots de teste:
hist(train$radius_mean)

ggplot(data=train, aes(x=radius_mean, y=texture_mean, group=diagnosis,
color=diagnosis)) +
 geom_bar(stat='identity')

ggplot(train, aes(x=texture_mean, y=area_mean)) + geom_point()

# Gráfico histograma - frequência dos dados de cada feature filtrado
para maligno e outro benigno
train_M <- filter(train, diagnosis == "M")
train_B <- filter(train, diagnosis == "B")

train_M = subset(train_M,select = -diagnosis)
train_B = subset(train_B,select = -diagnosis)

ggplot(gather(train_M), aes(x = value)) +
 geom_histogram(bins = 10) +
 facet_wrap(~key, scales = 'free_x') +
 labs( x = "Dados", y = "Frequencia") +
 ggtitle("Visualização geral dos dados referente a cancer tipo maligno
classifcado por feature")

ggplot(gather(train_B), aes(x = value)) +
 geom_histogram(bins = 10) +
 facet_wrap(~key, scales = 'free_x') +
 labs( x = "Dados", y = "Frequencia") +
 ggtitle("Visualização geral ddos dados referente a cancer tipo benigno
classifcado por feature")


# ---------- Algoritmo KNN com k = 1,3,5 e 11 vizinhos ---------

# train[,1] representa coluna diagnosis
verificaknn <- function(datasetTrain, datasetTest, vetorK,
posicaoClassificador){
```

```
 classesTrain <- datasetTrain[ ,posicaoClassificador]
 datasetTrain <- datasetTrain[ , -posicaoClassificador]

 classesTest <- datasetTest[ , posicaoClassificador]
 datasetTest <- datasetTest[ , -posicaoClassificador]

 result <- knn(datasetTrain, datasetTest, classesTrain, vetorK)
 # Matriz de confusão
 print("Matriz confusão:")
 print(as.matrix(table(classesTest, result)))
 matriz <- as.matrix(table(classesTest, result))
  # Indíce de acerto
 acc <- sum(diag(matriz))/nrow(datasetTest)
 print("Indice de acerto:")
 print(acc)
}

# k = 1
verificaknn(train, test, 1, 1)
# k = 3
verificaknn(train, test, 3, 1)
# k = 5
verificaknn(train, test, 5, 1)
# k = 11
verificaknn(train, test, 11, 1)



# -------------- Algoritmo de árvore de decisão -------------------
modelo <- rpart(diagnosis~., train, method = "class", control =
rpart.control(minisplit = 1))

plot <- rpart.plot(modelo, type = 3)

verificaDesicionTree <- function(modelo, datasetTest,
posicaoClassificador){
 classesTest <- datasetTest[ ,posicaoClassificador]
 datasetTest <- datasetTest[ , -posicaoClassificador]
  pred <- predict(modelo, datasetTest, type = "class")
  # Matriz de confusão
 print("Matriz confusão:")
 matriz <- as.matrix(table(classesTest, pred))
 print(matriz)
  # Indíce de acerto
```

```
 acc <- sum(diag(matriz))/nrow(datasetTest)
 print("Indice de acerto:")
 print(acc)
}

verificaDesicionTree(modelo, test, 1)
verificaDesicionTree(modelo, cancers, 1) #Utilizando o modelo para
predizer todo o data set
verificaDesicionTree(modelo, train, 1) #Utilizando modelo para predizer
o data set de treino

p <- ggplot(cancers, aes(perimeter_worst, concave_points_worst,
group=diagnosis, colour = diagnosis))
p + geom_point()
```

5. REFERÊNCIAS

https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)