



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS
ENGENHARIA DE COMPUTAÇÃO
Inteligência Artificial

Carlos Henrique Vieira Marques – 18720367
João Henrique Pereira –18712919

PROJETO 2
CLASSIFICAÇÃO POR ALGORITMO KNN E ÁRVORE DE DECISÃO

CAMPINAS
JUNHO 2021

SUMÁRIO

1. INTRODUÇÃO	3
2. DESENVOLVIMENTO	4
3. CONCLUSÃO	24
4. APÊNDICE	25

1. INTRODUÇÃO

O seguinte projeto visa apresentar as funcionalidades e resultados dos algoritmos de classificação KNN e Árvore de Decisão. Através de um Dataset fornecido por um acervo da UCI Machine Learning Repository, podemos realizar diversos testes e análises a respeito destes algoritmos. O dataset escolhido, denominado leaf, apresenta 40 espécies de plantas diferentes. A Tabela detalha os detalhes científicos de cada planta, nome e o número de espécimes de folhas disponíveis por espécies. Espécies numeradas de 1 a 15 e de 22 a 36 apresentam folhas simples e espécies numeradas de 16 a 21 e de 37 a 40 têm folhas complexas(não fornecidas).

O respectivo dataset, fornece 16 parâmetros para que os algoritmos realizem a classificação: Species, Specimen Number, Eccentricity, Aspect Ratio, Elongation, Solidity, Stochastic Convexity, Isoperimetric Factor, Maximal Indentation Depth, Lobedness, Average Intensity, Average Contrast, Smoothness, Third moment, Uniformity, Entrop.

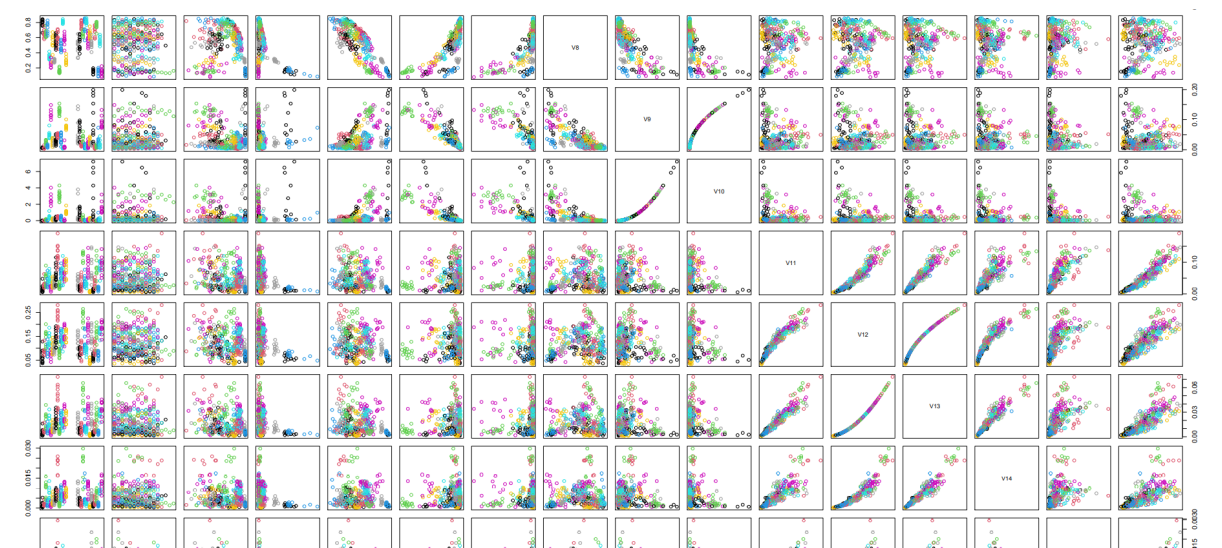
2. DESENVOLVIMENTO

Utilizando as ferramentas gráficas da linguagem R (boxplot e plot), foi possível realizar diversas pré-análises a respeito do Dataset, o que contribuiu para um melhor entendimento do Algoritmo. Ainda assim, teve-se algumas dificuldades em confirmar se os dados gerados estavam corretos, pois devido a grande quantidade de dados, a análise era complexa. Durante as fases de pesquisas para melhor entendimento do Algoritmos, descobriu-se uma ferramenta muito útil que analisa as features do dataset e retorna em forma de gráfico o quão semelhantes elas são, o que ajuda a interpretar quais features são mais predominantes para fazer a classificação dos dados.

Plot do Dataset:



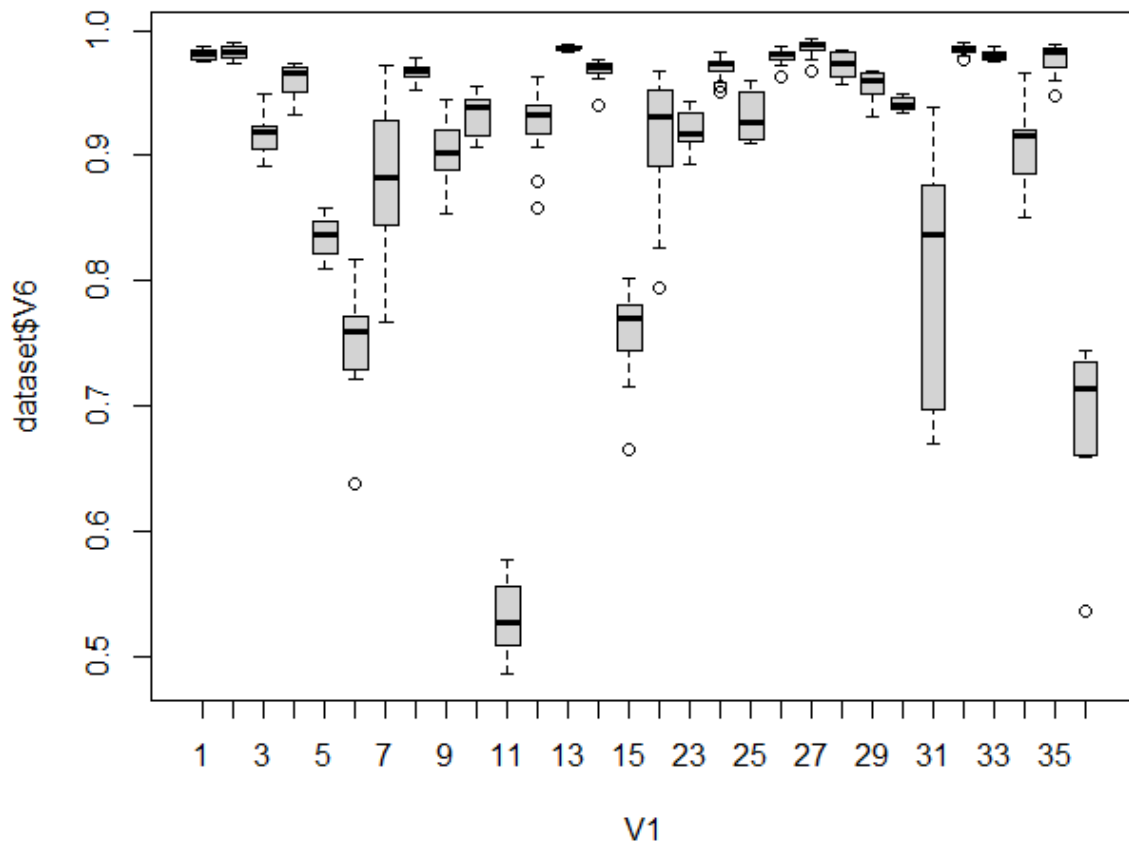
(Figura 1.1: Plot do Dataset)



(Figura 1.2: Plot do Dataset)

Devido a grande quantidade de amostra e de features, uma análise detalhada dos plot seria muito extensa, então faremos uma análise dos dados que são mais importantes para o grupo, junto disso utilizaremos o boxplot para auxiliar nas análises.


V1 -Classe (Espécies) e V6 -Solidez:





(Figura 2: Boxplot V1 x V6)


Este gráfico pode nos auxiliar em determinar quais as cores se aplicam a cada classe. De início podemos ver que algumas classes são mais isoladas que outras, pois suas características são as que mais se diferem das demais. Infelizmente, devido à falta de demonstrativo de cores, o gráfico gerado pelo Plot utiliza 1 cor para referenciar 4 classes distintas, o que pode nos prejudicar em análises mais específicas quando formos usar dados mais detalhados. Nessas circunstâncias, decidimos juntar as 4 classes que são representadas por 1 cor e criar um grupo de classes com o intuito de sugerir que o dado apresentado nos gráficos pode ser referente entre as 4 classes do grupo.


Legenda:


 A cor verde pode representar as classes: 3 -> **Populus nigra**, 11 -> **Acer palmatum**, 25 -> **Arisarum vulgare** e 33 -> **Hydrangea sp.** Esse grupo possui a sigla de referência: **PAAH.**


 A cor roxa representa as classes: 6 -> **Crataegus monogyna** , 14 -> **Castanea sativa** , 28 -> **Magnolia soulangeana** e 36 -> **Geranium sp.** Esse grupo possui a sigla de referência: **CCMG.**

 A cor amarela representa as classes: 7 -> **Ilex aquifolium** , 15 -> **Populus alba** e 29 -> **Buxus sempervirens**. Esse grupo possui a sigla de referência: **IPB.**

 A cor preta representa as classes: 1 -> **Quercus suber** , 9 -> **Betula pubescens**, 23 -> **Erodium sp.** e 31 -> **Podocarpus sp.** Esse grupo possui a sigla de referência: **QBEP.**

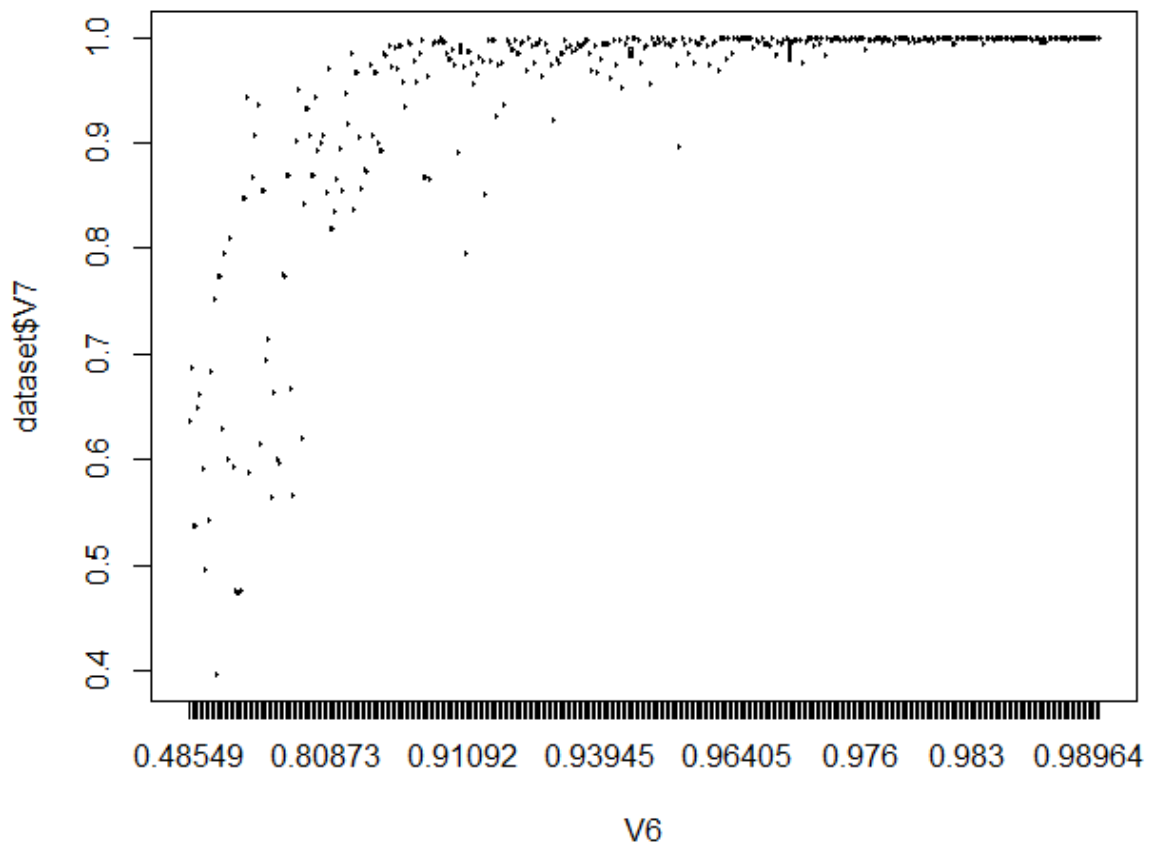
 A cor azul escuro representa as classes: 4 -> **Alnus sp** , 12 -> **Celtis sp** , 26 -> **Euonymus japonicus** e 34 -> **Euonymus japonicus**. Esse grupo possui a sigla de referência: **ACEE.**

 A cor azul claro representa as classes: 5 -> **Quercus robur** , 13 -> **Corylus avellana**, 27 -> **Ilex perado ssp. azorica** , 35 -> **Magnolia grandiflora**. Esse grupo possui a sigla de referência: **QCIM.**

 A cor vermelha representa as classes: 2 -> **Salix atrocinera** , 10 -> **Tilia tomentosa** , 24 -> **Bougainvillea sp.** , 32 -> **Acca sellowiana**. Esse grupo possui a sigla de referência: **STBA.**

Obs: De acordo com o Dataset **Lobedness** é o valor da curvatura da borda da folha.

V7 -Convexidade estocástica e V6 -Solidez



(Figura 3: Boxplot V6 x V7)

De acordo com o Plox e o Boxplot de V6 e V7, existem 2 grupos de classificação de folhas que são predominantes quando o valor de V7 é abaixo de 0.9. Podemos indicar que os grupos são CCMJ e PAAH. Tendo uma maior predominância do grupo PAAH quando V6 tem valores inferiores a 0.7 e o mesmo ocorre ao grupo CCMJ quando V6 é superior a 0.7. Além disso, é possível identificar mais 3 grupos que se sobressaem nesse gráfico, que são os QBEP, IPB e QCIM. Apesar da análise ser mais complexa podemos deduzir que entre 0.9 a 0.99 do valor de V6 temos destaque do grupo QBEP, entre 0.70 a 0.8 temos o grupo IPB, e por fim entre 0.8 a 0.91 temos o grupo QCIM.

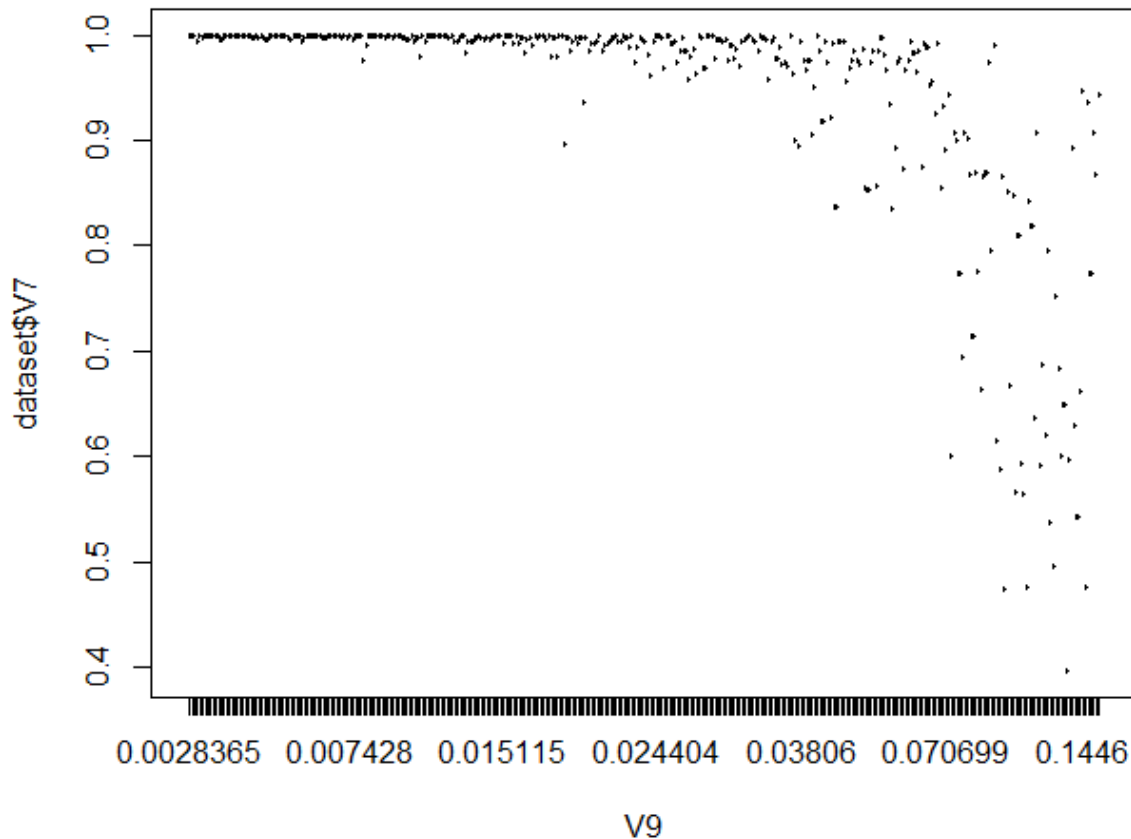
Análise :

O grupo PAAH pode possuir folhas com Solidez tanto baixa quanto alta, porém todas possuem uma convexidade estocástica baixa.

Já o grupo CCMJ possui folhas cuja Solidez e convexidade estocástica não são altas e nem baixas e sim medianas.

Os grupos QBEP, IPB e QCIM possuem folhas com solidez muito alta, sendo seus principais diferenciais nesse gráfico o nível de convexidade estocástica.

V7 -Convexidade estocástica e V9 -Profundidade máxima de recuo :



(Figura 4: Boxplot V9 x V7)

Nesta ocasião temos uma semelhante predominância de grupos, sendo eles o [CCMJ](#) e [PAAH](#) os mais destacados. Porém podemos ver outra semelhança entre as classes [QBEP](#) e [IPB](#) tendo poucas diferenças baseadas entre os baixos valores de V9 e os maiores valores de V7. Apesar do plot não apresentar valores numéricos entre V7 e V9 o Boxplot pode nos ajudar a interpretar qual o valor que cada coluna e linha está tendo para aquele ponto. Por exemplo, podemos identificar a presença do grupo [QCIM](#) nas escala mais baixas de V7, próximo a 0.6, e nas escalas altas de V9 com valores próximos a 0.8.

Análise:

O grupo **CCMJ** apresentou folhas com um nível médio de Convexidade estocástica, porém com grandes variações em seus dados de Profundidade máxima de recuo.

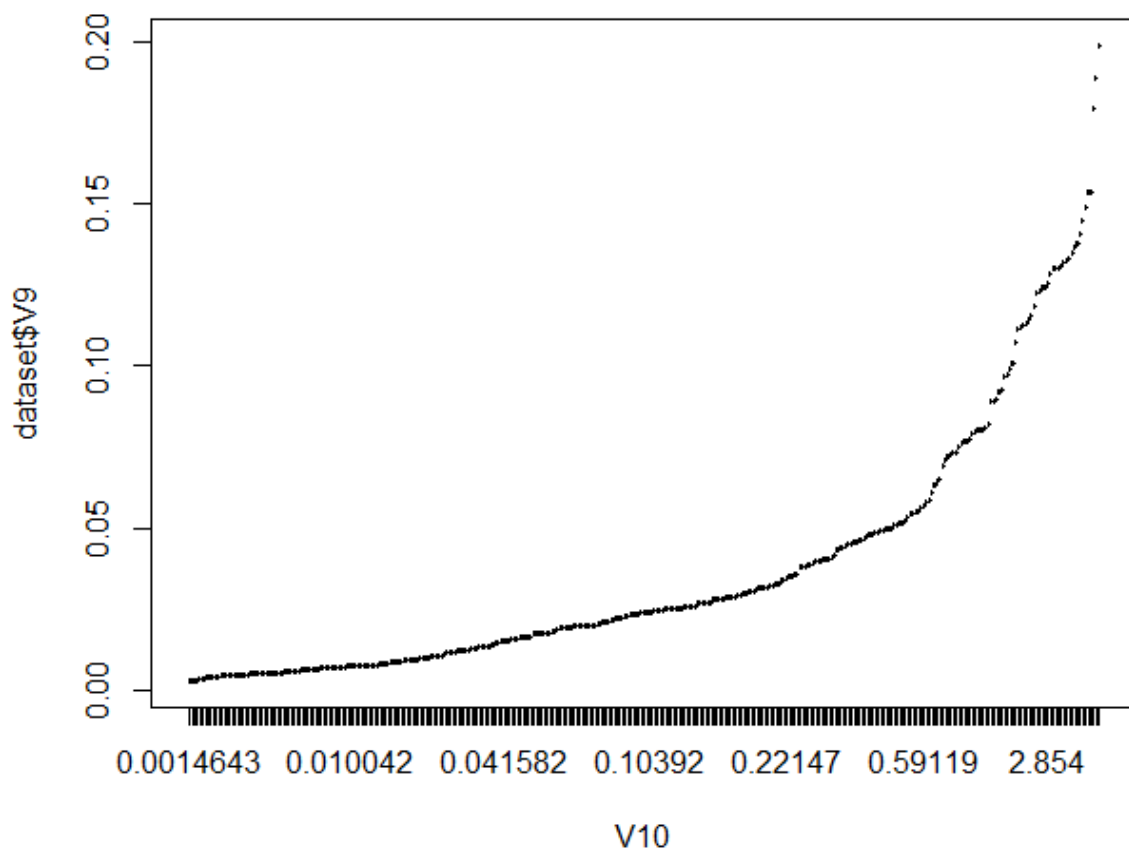
O grupo **PAAH** apresentou folhas com um nível alto de Convexidade estocástica, e também uma grande variação em seus dados de profundidade máxima de recuo.

O grupo **QBEP** apresentou folhas com nível médio e baixo de Convexidade estocástica, mas por outro lado eles possuem altos níveis de profundidade máxima de recuo.

O grupo **IPB** apresentou folhas com nível de convexidade estocástica baixa, mas também revelou que possui grandes valores de profundidade máxima de recuo em suas folhas.

O grupo **QCIM** revelou possuir os níveis mais baixos de convexidade estocástica, mas possui um dos níveis mais altos de Profundidade máxima de recuo.

V9- Profundidade máxima de recuo e V10- Lobedness :



(Figura 5: Boxplot V10 e V9)

Na análise para V9 e V10 podemos ver um modelo de gráfico com crescimento gradativo tendo no marco 0 a classificação do grupo **STBA** que logo em seguida é substituída pelo grupo **QCIM** que mantém um domínio até o valor de 0.41 de V10. Em seguida após V10 passar de 0.041 ocorre classificações do grupo **QBEP**, que logo depois é ofuscado pelo grupo **CCMJ** entre 0.22 a 0.50 de V10. Após isso, temos uma mesclagem entre os grupos **CCMJ** e **PAAH** até o final do gráfico com o fim tendo amostras de **QBEP**.

Análise:

*obs: Lembrando que **Lobedness** se trata da curvatura da borda da folha.*

O grupo **STBA** revelou possuir folhas com os níveis mais baixos de Profundidade máxima de recuo e de Lobedness.

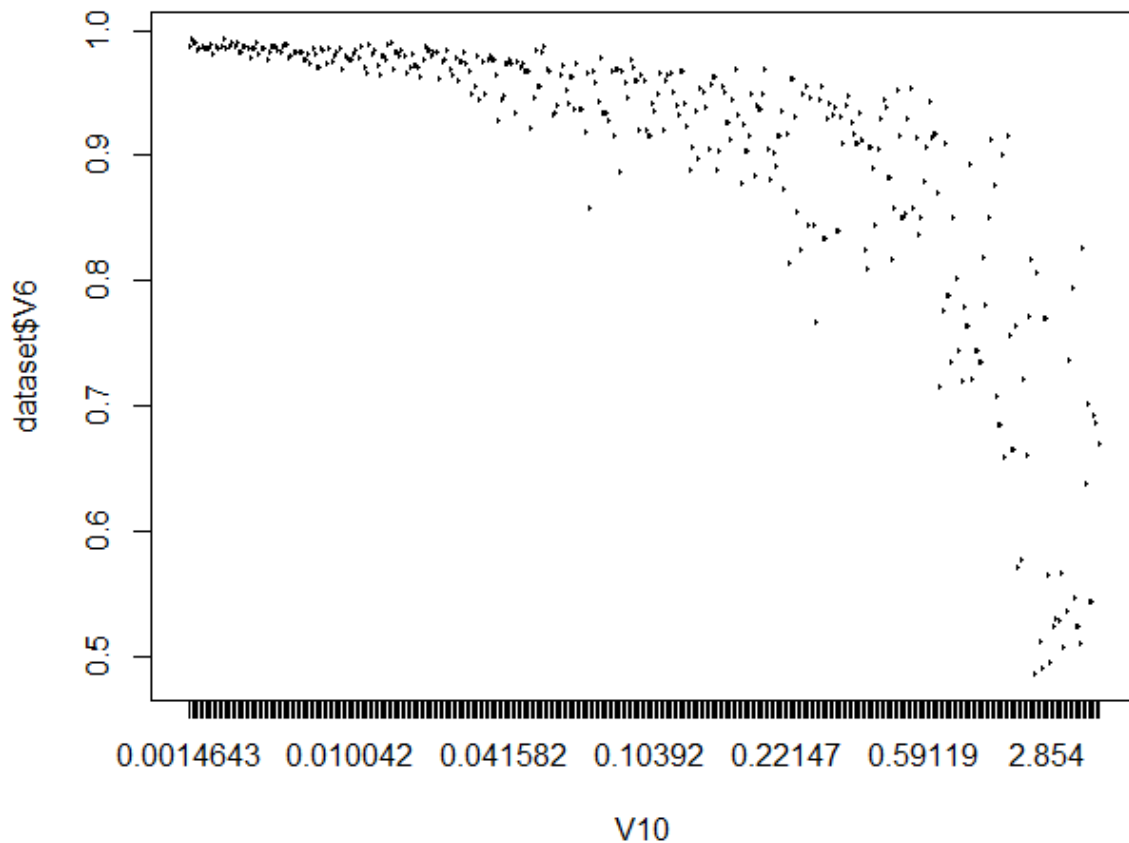
O grupo **QCIM** apresentou folhas com valores muito baixos de Profundidade máxima de recuo e também de Lobedness.

O grupo **QBEP** possui tanto folhas com valores baixos de Profundidade máxima de recuo e de Lobedness quanto valores altos.

O grupo **CCMJ** apresentou folhas com valores médios de Profundidade máxima de recuo e de Lobedness.

O grupo **PAAH** apresentou ter a médias de valores mais altos para Profundidade máxima de recuo e de Lobedness.

V6- Solidez e V10- Lobedness :



(Figura 6: Boxplot V10 e V6)

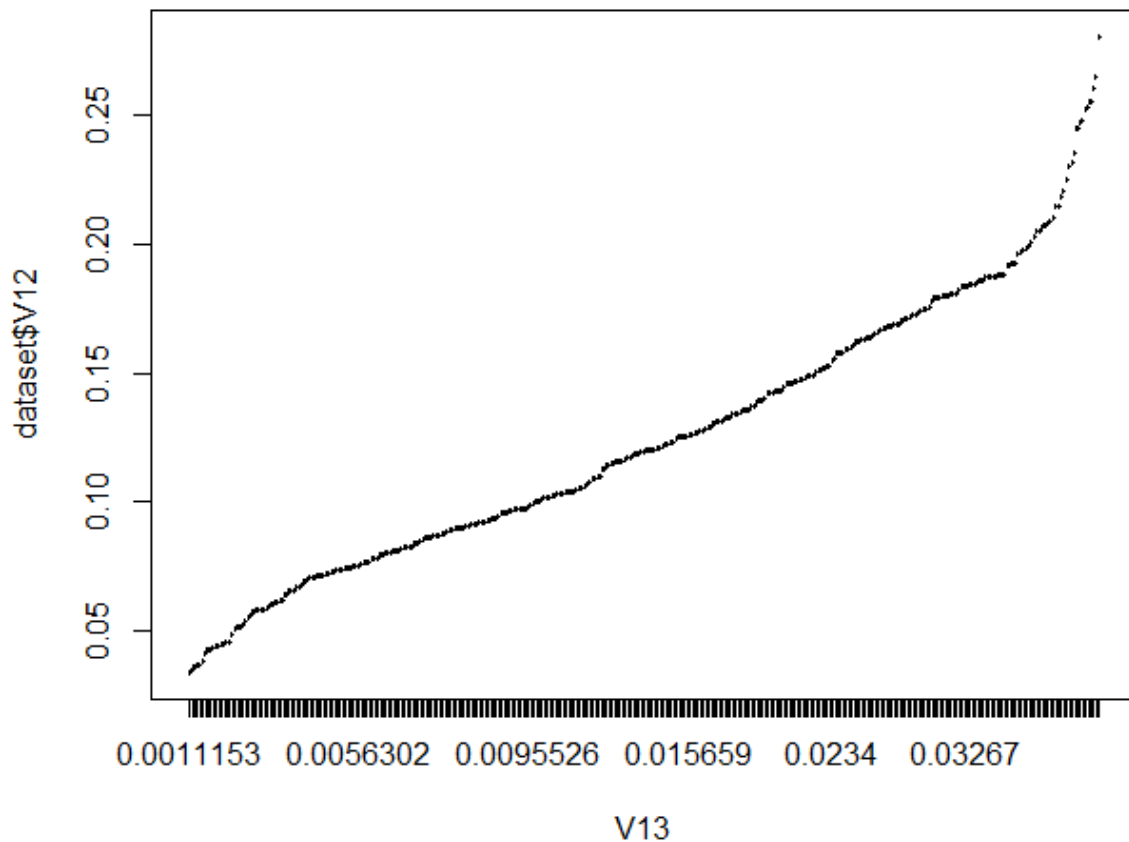
Essa se assemelha muito com as anteriores, para valores iniciais de V10 temos classificações do grupo **PAAH** e com sua progressão até a metade do gráfico temos a alternância para entre os grupos **QBEP** e **CCMJ** sendo o principal diferencial entre elas o valor do de V6, quanto maior V6 maior predominância da **QBEP** e quanto menor B6 maior predominância de **CCMJ**. Depois disto, temos vários grupos com se destacando para os mesmos valores de V10 e V6.

Análise:

obs: Lembrando que Lobedness se trata da curvatura da borda da folha.

Os grupos apresentaram os dados conforme o esperado com as Análises anteriores, como a média nos valores do grupo **PAAH** tanto na solidez quanto no Lobedness. Os demais grupos apresentaram dados semelhantes o que não impactou em uma mudança no conceito de classificação.

V12- Contraste Médio e V13- Suavidade:



(Figura 7: Boxplot V13 e V12)

Neste gráfico podemos ver um início de gráfico com o grupo **IPB**, com a progressão de v12 e V13 é possível identificar diversas classificações de grupos diferentes, sendo elas um pouco do grupo **QBEP** que logo é substituído pelo grupo **ACEE**. Em seguida temos algumas sequências de classificações para os grupos **QCIM** e **STBA**. Depois temos uma grande taxa de classificação do grupo **CCMJ** que é substituído pelos grupos **STBA** e **PAAH**.

Análise:

O grupo **IPB** apresentou folhas com os menores valores baixos para o Contraste médio e Suavidade.

O grupo **QBEP** apresentou dados de folhas com baixos valores de Contraste médio e Suavidade.

O grupo **ACEE** possui plantas com valores baixo e médio de Contraste médio e Suavidade.

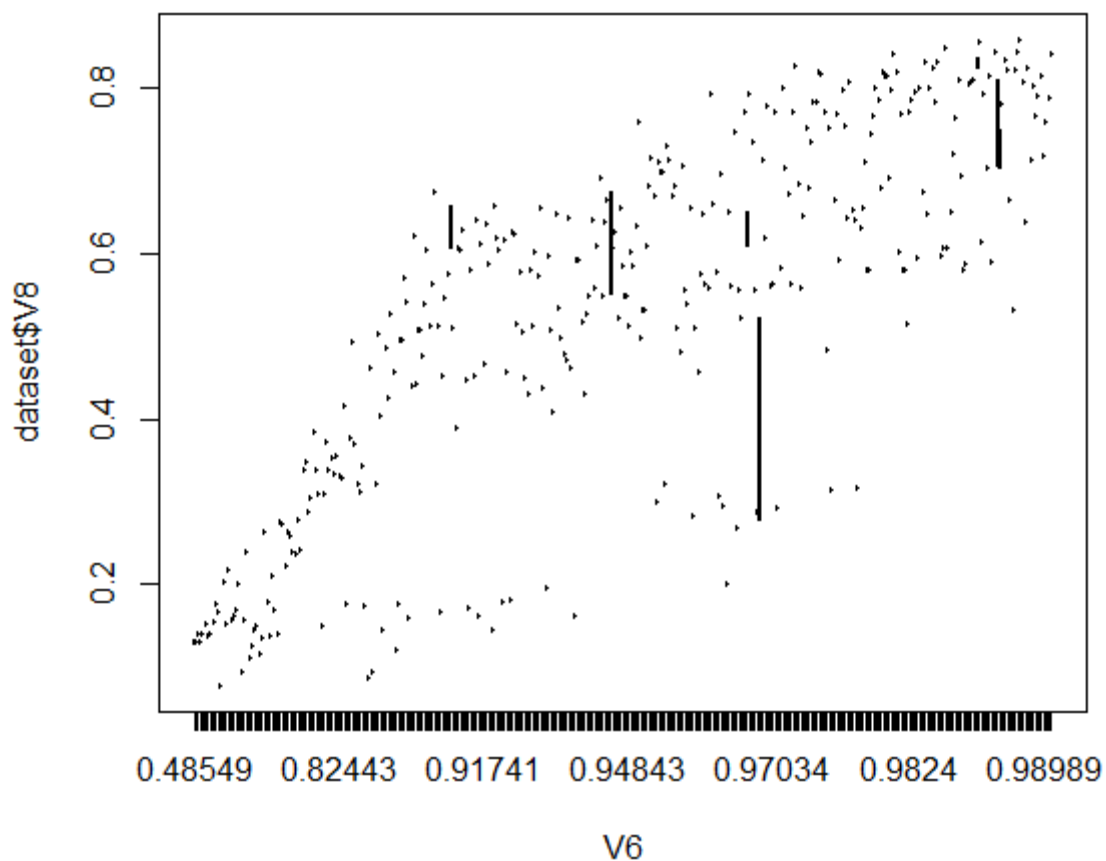
O grupo [QCIM](#) apresentou ter plantas com valores médios e altos de Contraste médio e Suavidade.

O grupo [STBA](#) apresentou possuir a maioria de suas folhas com médios valores de Contraste médio e Suavidade e algumas folhas com valores altos.

O grupo [PAAH](#) apresentou possuir algumas folhas com valores de Contraste médio e Suavidade baixo e sua maioria com valores altos.

O grupo [CCMJ](#) apresentou possuir folhas com valores altos de Contraste médio e Suavidade.

V6- Solidez e V8- Fator Isoperimétrico:



(Figura 8: Boxplot V6 e V8)

Neste gráfico temos alguns grupos que possuem valores semelhantes entre V8 e V6, o que necessita de uma análise mais criteriosa. Para valores iniciais tanto para V6 quanto para V8 temos classificação para o grupo [PAAH](#) até 0.5 de V6 e 0.4 de V8. Entre 0.6 e 0.8 de V8 e 0.2 a 0.4 de V6 temos classificações para os grupos [CCMJ](#) e [IPB](#). Sendo valores os valores altos de V6 voltado para o grupo [IPB](#) e os baixos para o [CCMJ](#). Mantendo o valor de V6 mas analisando o intervalo de V8

entre 0.8 a 0.95 temos outras classificações para 2 grupos, o QBEP e ACEE. Sendo os valores baixo de V6 voltados para o grupo ACEE e o alto para o QBEP.

Análise:

O grupo PAAH apresentou possuir folhas valores médios de solidez e baixos de Fator Isoperimétrico.

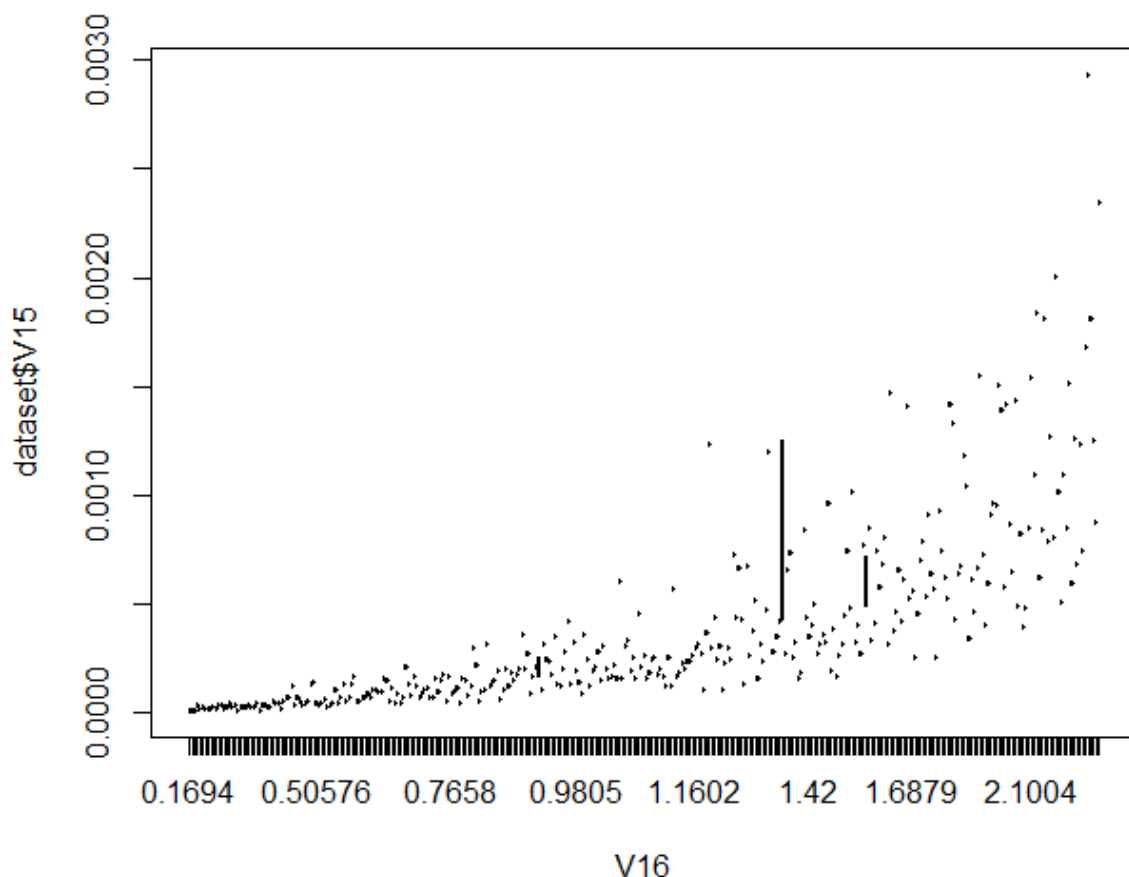
O grupo CCMJ apresentou possuir folhas com valores médios de solidez e Fator Isoperimétrico.

O grupo IPB apresentou possuir folhas alguns valores médios de solidez mas altos valores de Fator Isoperimétrico e outros altos.

O grupo ACEE apresentou possuir algumas folhas com baixos valores de solidez e Fator Isoperimétrico e outras altas, sendo a maioria de suas folhas com valores altos.

O grupo QBEP apresentou ter algumas folhas com baixos valores de solidez e Fator Isoperimétrico, enquanto a maioria com valores altos de ambas características.

V15- Uniformidade e V16 - Entropy:



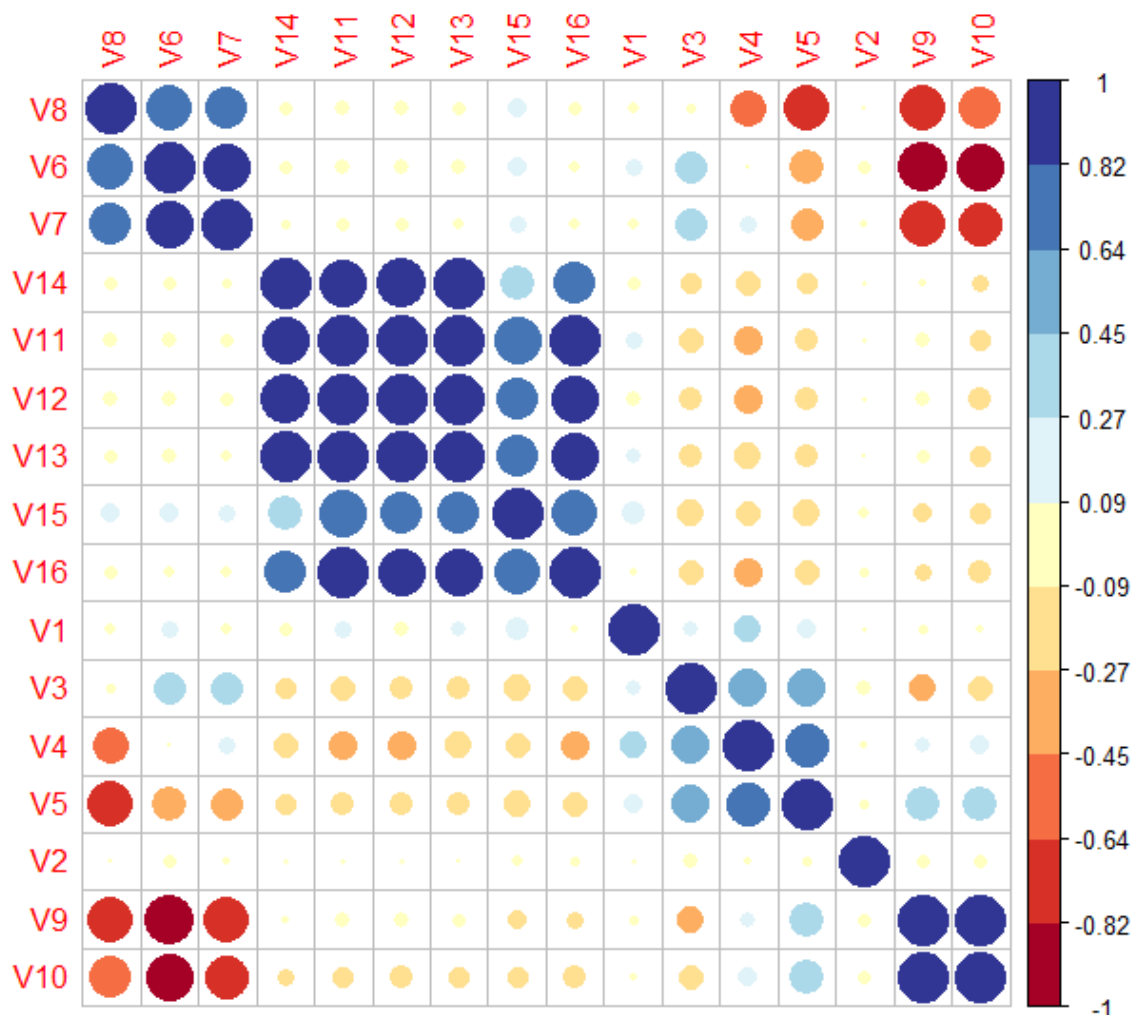
(Figura 9 Boxplot V16 e V15)

Após as features V13 os gráficos ficam mais complexos e maiores, com índice muito alto e várias classes. No exemplo V15 e V16 temos a presença das classes vistas anteriormente, porém dessa vez temos elas misturadas com as demais, com quase nenhuma predominância fixa de alguma.

Análise:

Devido muitos grupos possuírem dados semelhantes neste gráfico, utilizaremos ele apenas como escopo sobre quais características a maioria dos grupos têm em comum.

Ferramenta corrplot:



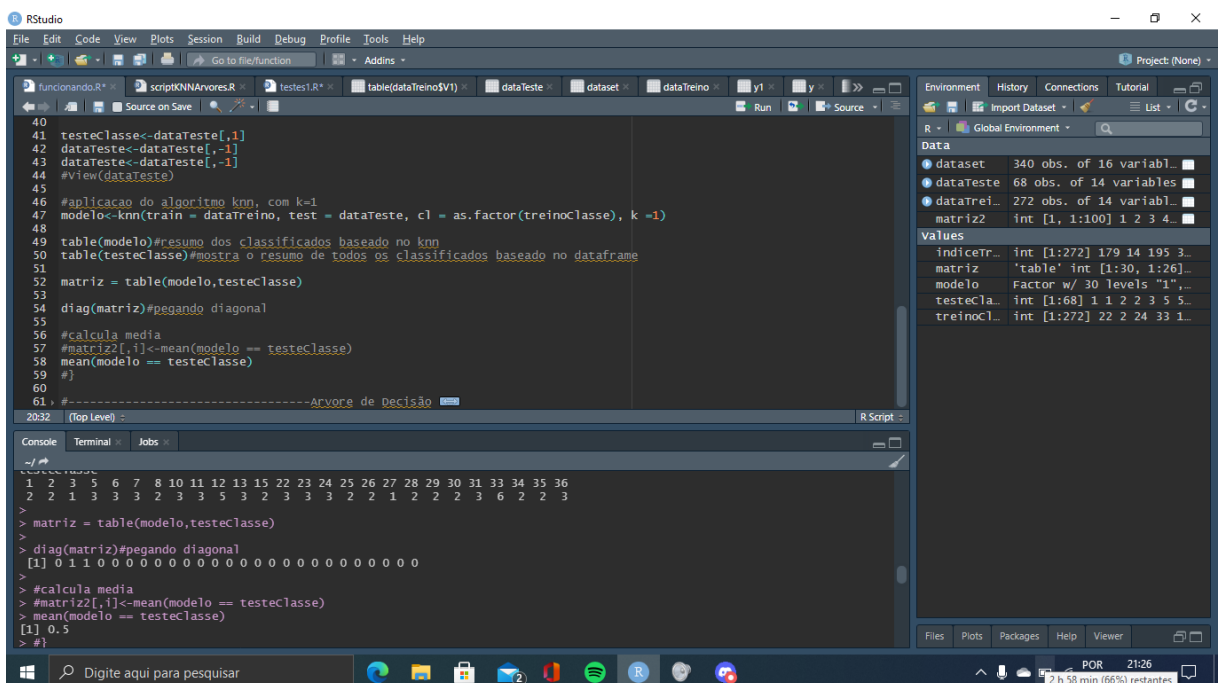
(Figura 10: Corrplot)

Esta foi uma das ferramentas que descobrimos no processo do experimento. Ela utiliza uma matriz de correlação e realiza uma análise entre features e mostra quais possuem características mais diferentes e quão a análise entre elas podem ser importantes para classificar um dado.

Análise KNN:

O algoritmo KNN, também conhecido como algoritmo do vizinho mais próximo, é um dos métodos de classificação, no que diz respeito à aprendizagem supervisionada. Seu funcionamento consiste basicamente em um sistema de coordenadas de n-dimensões, onde o conjunto de dados de treinamento são plotados, sendo n o número de features do dataset a ser classificado. Dessa forma, todo dado a ser testado, só precisa calcular quem são seus K vizinhos mais próximos do conjunto de treino.

No experimento o valor de K utilizado foi 1, devido ao fato de o número de amostras de cada folha ser baixo, como por exemplo a *Alnus sp* e *Crataegus monogyna* que possuíam apenas 8 amostras no total.



The screenshot shows the RStudio interface with a script editor, console, and environment pane. The script editor contains R code for KNN classification. The console shows the execution of the code, including the creation of a confusion matrix and the calculation of accuracy. The environment pane shows the objects created during the execution.

```
40
41 testeClasse<-dataTeste[,1]
42 dataTeste<-dataTeste[,-1]
43 dataTeste<-dataTeste[,-1]
44 #view(dataTeste)
45
46 #aplicacao do algoritmo knn, com k=1
47 modelo<-knn(train = dataTreino, test = dataTeste, cl = as.factor(treinoClasse), k =1)
48
49 table(modelo)#resumo dos classificados baseado no knn
50 table(testeClasse)#mostra o resumo de todos os classificados baseado no dataframe
51
52 matriz = table(modelo,testeClasse)
53
54 diag(matriz)#pegando diagonal
55
56 #calcula media
57 #matriz[,1]<-mean(modelo == testeClasse)
58 mean(modelo == testeClasse)
59 #}
60
61 #-----Arvore de Decisao
```

Console output:

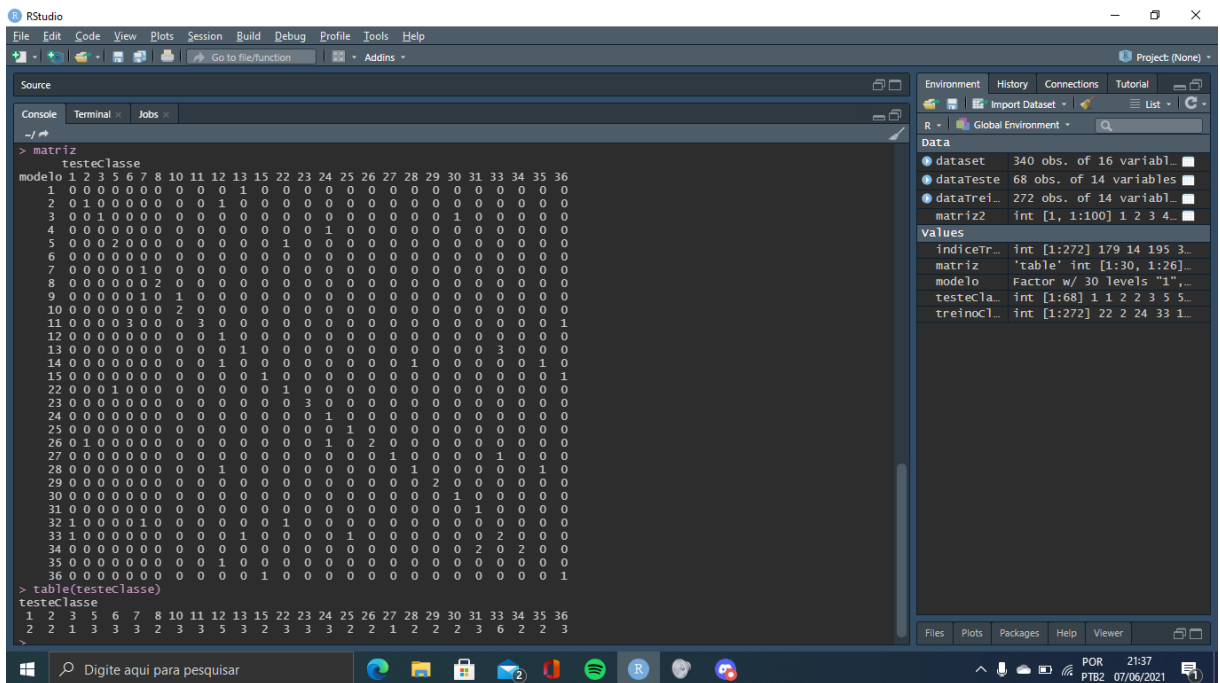
```
1 2 3 5 6 7 8 10 11 12 13 15 22 23 24 25 26 27 28 29 30 31 33 34 35 36
2 2 1 3 3 3 3 2 3 3 3 3 3 3 3 2 2 1 2 2 2 2 3 6 2 2 3
> matriz = table(modelo, testeClasse)
> diag(matriz)#pegando diagonal
[1] 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
>
> #calcula media
> #matriz[,1]<-mean(modelo == testeClasse)
> mean(modelo == testeClasse)
[1] 0.5
> #}
```

Environment pane:

Object	Class	Attributes
dataset	data.frame	340 obs. of 16 variables
dataTeste	data.frame	68 obs. of 14 variables
dataTreino	data.frame	272 obs. of 14 variables
matriz2	matrix	int [1, 1:100] 1 2 3 4...
indiceTr	int	[1:272] 179 14 195 3...
matriz	'table' int	[1:30, 1:26]
modelo	Factor w/ 30 levels	"1"...
testeCla	int	[1:68] 1 1 2 2 3 5...
treinoCl	int	[1:272] 22 2 24 33 1...

(Figura 11: Código KNN)

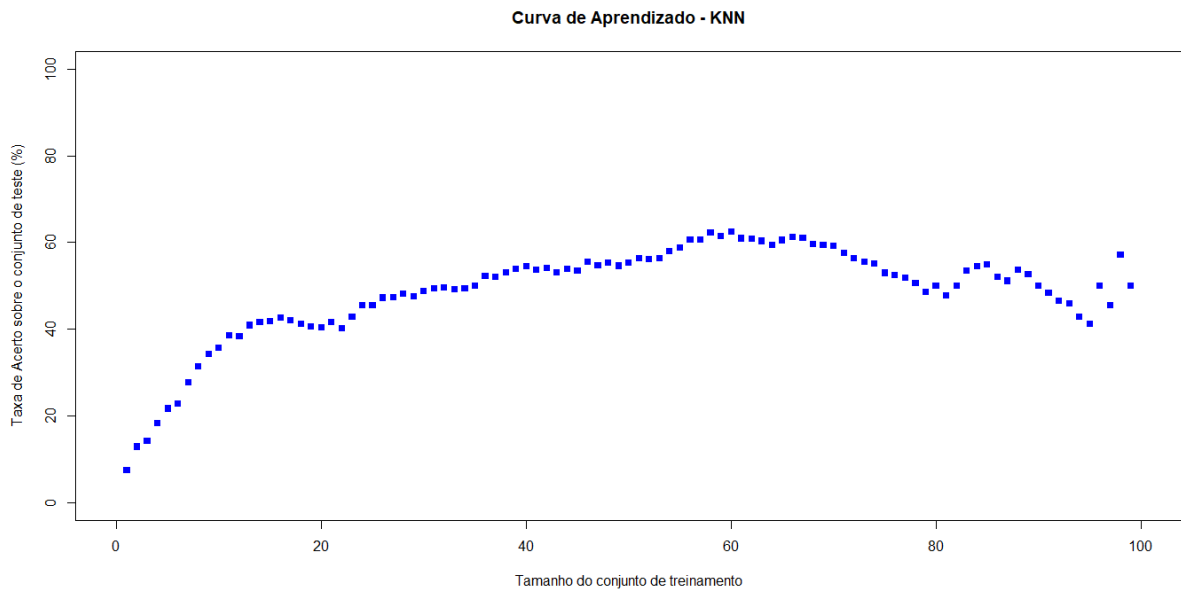
Para a implementação do KNN utilizamos como base as aulas da professora e fizemos proveito do código apresentado e das explicações de funcionalidades. Não foi necessário fazer alterações complexas no código em relação ao código base. A principal mudança foi na maneira de se calcular a acurácia da classificação, devido ao fato de que, no momento de se realizar a separação entre conjunto de treino e conjunto de testes, algumas classes tinham 100% de suas amostras alocadas no conjunto de treino. Com isso, obtém-se uma matriz de confusão não quadrada (figura 12), resultando em uma soma equivocada da diagonal da matriz. Para resolver tal problema, utilizou-se a função *mean()*. Dessa forma, foi possível obter qual foi a acurácia da classificação, que para k=1, alocando 80% do dataset para o conjunto de treino, obtém-se 50% de acerto.



(Figura 12: Matriz de Confusão -KNN)

É perceptível pela matriz de confusão, que as colunas 4, 9, 14 e 32 das folhas *Alnus sp*, *Betula pubescens*, *Castanea sativa* e *Acca sellowiana* estão ausentes, pois todas as amostras dessas classes, foram alocadas no conjunto de treino. Porém, houveram dados que, ainda assim, foram classificados como se pertencessem a essas classes. As únicas classes em que se teve 100% de seus elementos classificados corretamente, e que não houve classificações errôneas, foram as de número 8, 23 e 29, *Nerium oleande*, *Erodium sp.* e *Buxus sempervirens* respectivamente. Já as classes 1, 6 e 35, *Quercus suber*, *Crataegus monogyna* e *Magnolia grandiflora*, tiveram todas as suas amostras classificadas de maneira equivocada. As classes restantes oscilaram, algumas classificando elementos a mais, outras a menos.

Visando conhecer como a taxa de acerto respondia diante de diferentes razões conjunto de treino/conjunto total, estabeleceu-se o gráfico da figura 13, onde o eixo x é a porção do dataset que é alocado para conjunto de treino, e o eixo y a taxa de acerto que o conjunto de teste tinha ao ser submetido pelo classificador desenvolvido. A taxa de acerto é calculada pela soma da diagonal da matriz de confusão, dividida pelo número de elementos do conjunto de teste. A obtenção de todos os valores do gráfico, foi por meio de um loop de 1 a 100, onde se variava a proporção do conjunto de treino, com os valores da taxa de acerto sendo salvos em um vetor.

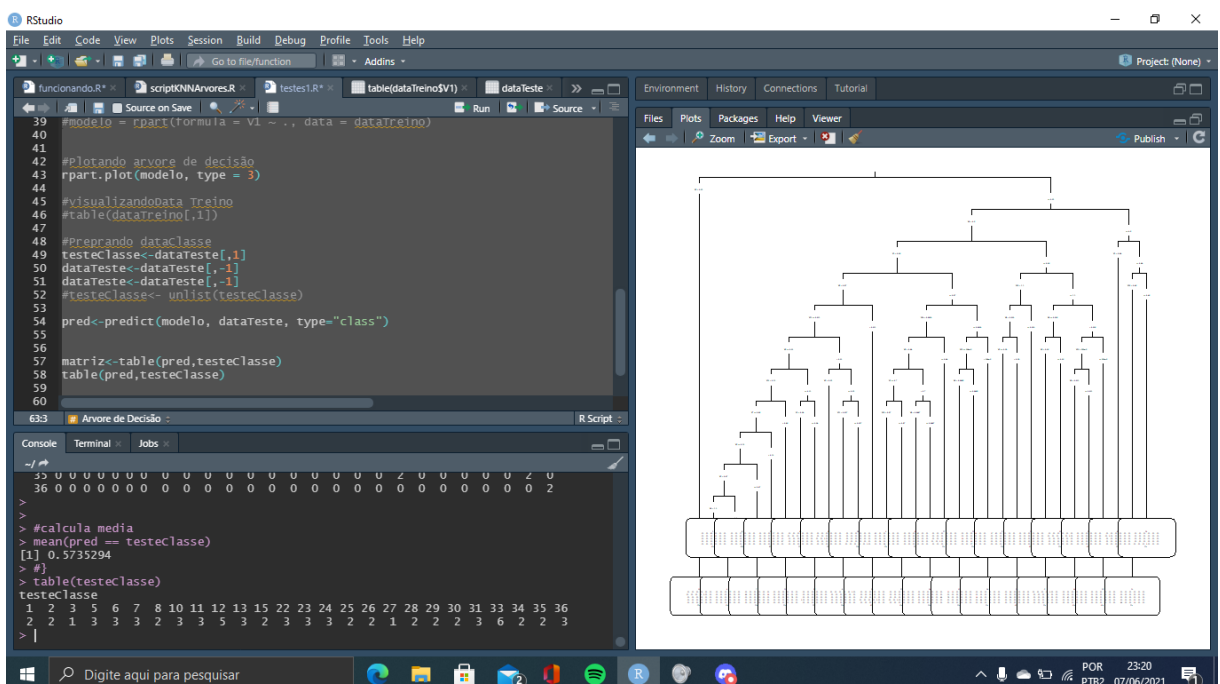


(Figura 13: Curva de aprendizado - KNN)

Por meio do gráfico acima, foi possível conhecer qual a melhor taxa de acerto, com base no tamanho do conjunto de treinamento. Usando $k=1$, e seed = 123, a maior acurácia encontrada foi de 62,5%, isso quando o conjunto de treinamento foi de 60%.

Análise Árvore de Decisão:

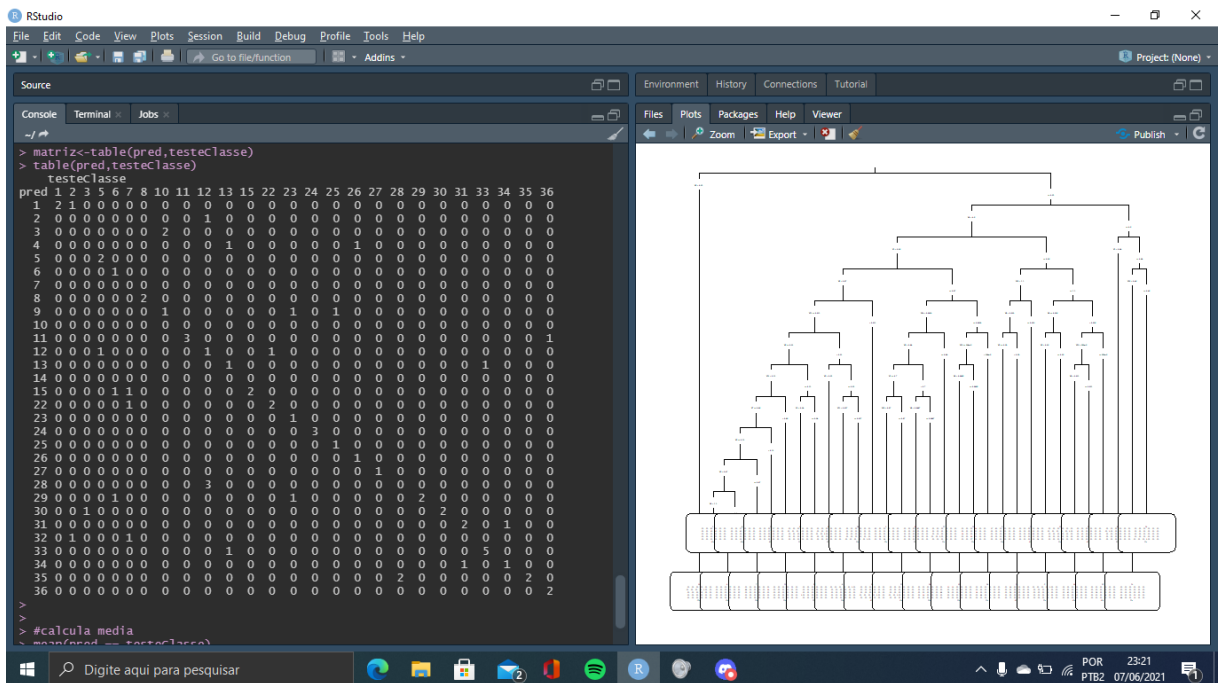
O algoritmo de árvore de decisão, também usado para realizar classificações, funciona da seguinte forma: o algoritmo realiza uma busca pelo conjunto de treino, a fim de dividi-lo em subconjuntos. Essa divisão usa as features do dataset para ocorrer, sejam elas numéricas ou booleanas, além disso é importante escolher o atributo que vá o mais longe possível na tentativa de fornecer uma classificação exata do conjunto de treino, minimizando ao máximo a profundidade da árvore. Esse é um processo recursivo, e ocorre até que haja uma árvore com caminhos distintos para cada classe.



(Figura 14: Árvore de Decisão Gerada)

Devido a ilegibilidade da árvore de decisão por meio da imagem, um arquivo contendo a árvore de decisão gerada será anexado junto ao relatório.

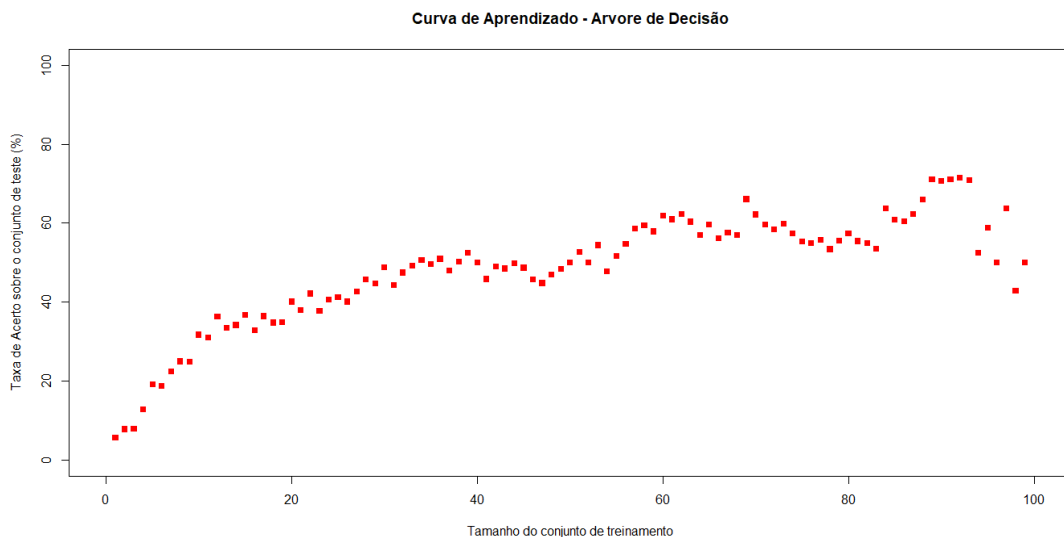
Assim como no código do KNN, a única mudança necessária foi a respeito do cálculo da acurácia, que foi realizado pela função *mean()*. Usando 80% do dataset como conjunto de treino, e com um valor de seed =123, a taxa de acerto obtida foi de 57,35%.



(Figura 15: Matriz de Confusão)

Assim como no KNN, a matriz de confusão, não possui as colunas 4, 9, 14 e 32 do dataset. Todavia, também houveram dados que, ainda assim, foram classificados como se pertencessem a essas classes, com exceção da classe 14, *Castanea sativa*, que de maneira precisa, não teve nenhum elemento classificado. As únicas classes em que se teve 100% de seus elementos classificados corretamente, e que não houve classificações equivocadas, foram as de número 8 e 27, *Nerium Oleande* e *Ilex Perado ssp. Azorica*. Já as classes 2,3,7,10 e 28 tiveram todas as suas amostras classificadas de maneira equivocada. As classes restantes oscilaram, algumas classificando elementos a mais, outras a menos.

Também foi possível estabelecer um gráfico taxa de acerto x conjunto de treino, que permitiu visualizar como a acurácia da classificação respondia de acordo com o tamanho do conjunto de treino. O tamanho do conjunto de treino que obteve maior taxa de acerto, foi 92% do dataset, com uma acurácia de 71,42% na classificação.



(Figura 15:Curva de Aprendizado)

3. CONCLUSÃO

O desenvolvimento desse projeto, permitiu aos integrantes conhecer de maneira mais profunda a linguagem R e algumas de suas ferramentas, além é claro, aprimorar os conhecimentos sobre os métodos de classificação KNN e Árvore de decisão, incorporando um conhecimento teórico ao realizar as implementações e análise dos testes. Foi possível notar pelos resultados, e também pelos gráficos de curva de aprendizado, que o algoritmo de árvore de decisão possui um desempenho em média melhor que o KNN.

Também ficou evidente algumas limitações desses algoritmos. No KNN por exemplo, classes que não possuíam suas amostras bem agrupadas, ou que compartilhavam os mesmos espaços com outras classes, o que foi possível de perceber pelos plots, tendem a darem resultados piores. Já na árvore de decisão, se os argumentos do dataset não forem capazes de traçar um padrão, fazendo distinção entre as diferentes classes, provavelmente os resultados obtidos não serão bons, além da possibilidade de se obter uma árvore de decisão ultra ajustada ao conjunto de teste, o que impossibilita a classificação com êxito de outras amostras.

4. APÊNDICE

```
#1) Realizar a leitura do dataset em um dataframe.
dataset = read.csv("C:/Users/joaoh/Desktop/Projeto 2 Inteligência
Artificial/leaf.csv", header = FALSE)
View(dataset)

#matriz2 <- matrix (data = 1:100, nrow = 1, ncol = 100)
#names(dataset) <- c("age", "sex", "cp", "trestbps", "chol", "fbs",
"restecg",
#
"thalach", "exang", "oldpeak", "slope", "ca",
"thal", "num")

#plot(dataset$V3, dataset$V4, col = as.factor(dataset$V2))
plot(dataset, col = as.factor(dataset$V1))
plot(dataset)

boxplot(data = dataset, dataset$V1~V2)
boxplot(data = dataset, dataset$V7~V2)

#-----matriz correlcaoooooo
library(RColorBrewer)
library(corrplot)
matriz_cor <- cor(dataset)
matriz_cor

library(corrplot)
#matriz de correlação com cores
corrplot(matriz_cor, type="full", order="hclust",
col=brewer.pal(n=11, name="RdYlBu"))

#matriz de correlação com cores
corrplot(matriz_cor, type="full", order="hclust",
tl.col = "black", tl.srt = 45)

#----- Separando conjunto de treino e conjunto de teste ###
library(class)

#for (i in 1:100){
set.seed(123)
indiceTreino<-sample(1:nrow(dataset),0.8*nrow(dataset))

#separando dataset de treino com 80% do dataset original
dataTreino<-dataset[indiceTreino,]
#View(dataTreino)
```

```

#separando 20% restante do anterior
dataTeste<-dataset[-indiceTreino,]
#View(dataTeste)

#View(table(dataTreino$V1))
#View(table(dataTeste$V1))

#-----Para KNN
-----
#separando coluna de classificado
treinoClasse<-dataTreino[,1]
dataTreino<-dataTreino[,-1]
dataTreino<-dataTreino[,-1]
#View(dataTreino)

testeClasse<-dataTeste[,1]
dataTeste<-dataTeste[,-1]
dataTeste<-dataTeste[,-1]
#View(dataTeste)

#aplicacao do algoritmo knn, com k=1
modelo<-knn(train = dataTreino, test = dataTeste, cl =
as.factor(treinoClasse), k =1)

table(modelo)#resumo dos classificados baseado no knn
table(testeClasse)#mostra o resumo de todos os classificados
baseado no dataframe

matriz = table(modelo,testeClasse)

#diag(matriz)#pegando diagonal

#calcula media
#acerto = sum(diag(matriz))
#total = sum(matriz)
#acerto/total

mean(modelo == testeClasse)
#matriz2[,i]<-mean(modelo == testeClasse)

#}

#-----Arvore de Decisão
-----
library(rpart)

```

```
library(rpart.plot)

modelo<-rpart(V1~V3+V4+V5+V6+V7+V8+V9+V10+V11+V12+V13+V14+V15+V16,
dataTreino, method = "class", control = rpart.control(minsplit =
1))
#modelo = rpart(formula = V1 ~ ., data = dataTreino)

#Plotando arvore de decisao
rpart.plot(modelo, type = 3)

#visualizando Data Treino
#table(dataTreino[,1])

#Preparando dataClasse
testeClasse<-dataTeste[,1]
dataTeste<-dataTeste[,-1]
dataTeste<-dataTeste[,-1]
#testeClasse<- unlist(testeClasse)

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

#matriz de confusao
matriz<-table(pred,testeClasse)
table(pred,testeClasse)

diag(matriz)

#calcula media
#acerto = sum(diag(matriz))
#total = sum(matriz)
#acerto/total

mean(pred == testeClasse)
```