UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
ELE0606 - TÓPICOS ESPECIAIS EM INTELIGENCIA ARTIFICIAL

Docente:
José Alfredo Ferreira Costa

Discente:
Douglas Wilian Lima Silva
Semestre 2023.2 - Turma 01

Árvore de decisão

Natal - RN
Setembro de 2023

Sumário

1 Apresentação

As árvores de decisão são uma técnica amplamente utilizada em aprendizado de máquina e análise de dados que desempenham um papel fundamental na tomada de decisões automatizadas e na extração de informações valiosas a partir de conjuntos de dados complexos. Essa abordagem é uma das ferramentas mais versáteis e interpretáveis disponíveis para resolver uma variedade de problemas em diferentes domínios, desde classificação até regressão e até mesmo tarefas de pré-processamento de dados.

Tendo em vista o apresentado, o objetivo deste trabalho é a aplicação de árvores de decisão para analisar a base de dados wine disponível na Scikit-Learn e classificar os dados, de forma semelhante ao realizado utilizando o algoritmo dos vizinhos mais próximos. Além disso, também será aplicada a mesma metodologia para a base de dados Heart-Disease, com o intuito de classificação dos pacientes estudados.

A ideia é justamente tomar decisões baseadas em pesos, para que, ao fim seja possível a classificação correta dos dados estudados. A figura 1 apresenta um exemplo didático de aplicação de árvores de decisão.

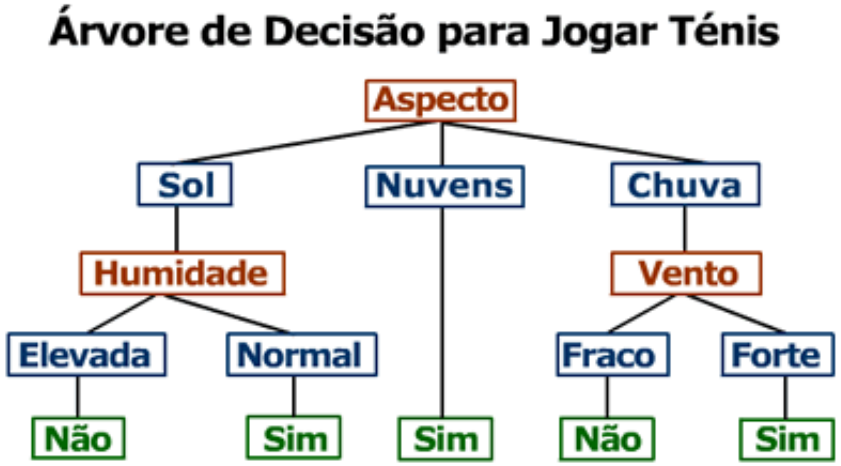

Figura 1: Árvore didática.

2 Desenvolvimento

As árvores de decisão são uma técnica de aprendizado de máquina supervisionado utilizada para resolver problemas de classificação e regressão. A ideia principal por trás das árvores de decisão é criar um modelo que possa tomar decisões ou prever valores alvo com base em uma série de perguntas sobre os atributos (características) dos dados de entrada.

A estrutura de uma árvore de decisão é semelhante a uma árvore real, com um tronco (nó raiz) que se ramifica em galhos (nós intermediários) e, finalmente, em folhas (nós folha). Cada nó intermediário representa uma decisão com base em um atributo específico e possui ramificações que levam a outros nós intermediários ou folhas, dependendo da resposta à pergunta sobre o atributo.

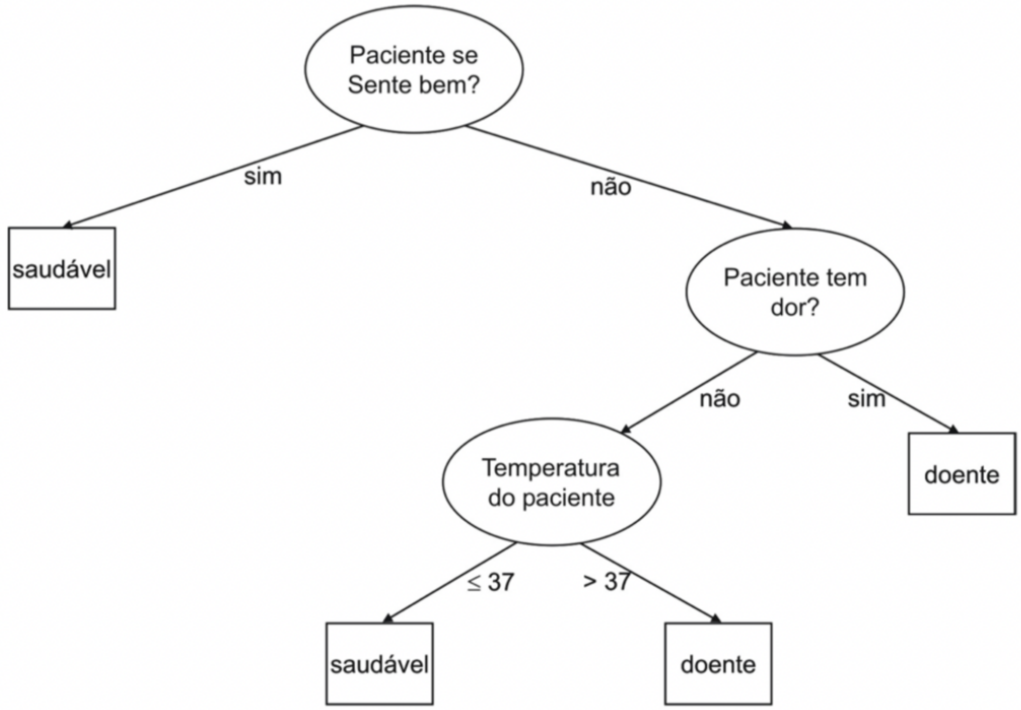

Figura 2: Exemplo.

Como citado anteriormente, o desenvolvimento do algoritmo será separado em dois momentos: a aplicação do método na base de dados wine, e na base de dados heart-disease. Em ambos os casos a ideia é classificar os dados. No primeiro caso, podemos classificar os vinhos entre os tipos 1, 2 e 3. E no segundo caso, a classificação dos pacientes como doentes ou não.

2.1 Base de dados Wine

Para a base de dados wine, a implementação se deu de forma semelhante ao realizado com o KNN, com a importação e processamento dos dados.

```
wine = load_wine()
df = pd.DataFrame(data = wine['data'], columns = wine['feature_names'])
df['target'] = wine['target']

df.head()
```

Listing 1: Importação dos dados.

De forma semelhante, foram separados os dados de treinamento e teste, utilizando a biblioteca NumPy para a "randomização"dos dados.

```
indices = np.random.permutation(df.shape[0])
div = int(0.4*len(indices))
desen_id , test_id = indices[:div], indices[div:]

cj_desen, cj_test = df.loc[desen_id,:], df.loc[test_id,:]

xd = cj_desen.drop('target', axis =1)
yd = cj_desen.target

xt = cj_test.drop('target', axis=1)
yt = cj_test.target

display(xd.head())
display(yd.head())
```

Listing 2: Randomização dos dados.

Por fim, para realizar a classificação dos dados, foi utilizada a biblioteca tree da Scikit-Learn. Ela fornece os métodos necessários para treinamento e predição dos dados.

```
from sklearn import tree
classi = tree.DecisionTreeClassifier(random_state=42)
classi = classi.fit(xd, yd)
prev = classi.predict(xt)
ac = accuracy_score(yt, prev)
cf = confusion_matrix(yt, prev)

plt.figure(figsize=(8, 6))
sns.heatmap(cf, annot=True, fmt='d', cmap='Blues', annot_kws={"size":
    14})
plt.xlabel('Previsoes')
plt.ylabel('Valores Verdadeiros')
plt.title('Matriz de Confusao')
plt.gca().set_xticklabels(["Classe 1", "Classe 2", "Classe 3"])
```

```
plt.gca().set_yticklabels(["Classe 1", "Classe 2", "Classe 3"])
plt.show()

fig, ax = plt.subplots(figsize = (13,11))
tree.plot_tree(classi)
plt.show()

ac = ac*100
print(f"A acuraria do processo foi {ac:.4f} %!")
```

Listing 3: Randomização dos dados.

Com essa implementação, foi possível a obtenção dos resultados através da visualização da matriz de confusão, da árvore de decisão do problema e da precisão (acurácia obtida). Os resultados serão apresentados na seção seguinte.

2.2 Base de dados Heart-Disease

A forma de implementação da árvore é exatamente igual a base de dados anterior. No entanto, para conseguir acesso à base de dados heart-disease, foi necessário retirar o arquivo csv, upado previamente ao drive. Com essa pequena modificação, foi possível o acesso aos dados desejados.

```
from google.colab import drive
drive.mount('/content/drive')

caminho_arquivo = "/content/drive/My Drive/HD/heart.csv"

heart = pd.read_csv(caminho_arquivo)
heart.head()
```

Listing 4: Randomização dos dados.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Figura 3: Dados obtidos.

Usando a função info(), podemos observar a configuraçãod dos dados.

```
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

Figura 4: Informação dos dados.

Os demais passos foram realizados de forma semelhante à base wine. Fazendo com que os resultados pudessem ser obtidos. O código completo pode ser visualizado no link disponível nas referências.

3 Resultados

3.1 Resultados Wine

Através da implementação apresentada, foi possível a plotagem da matriz de confusão referente aos resultados da simulação. Observa-se o número de casos previstos incorretamente durante o teste, já que, idealmente, todos os valores deveriam se encontrar na diagonal.

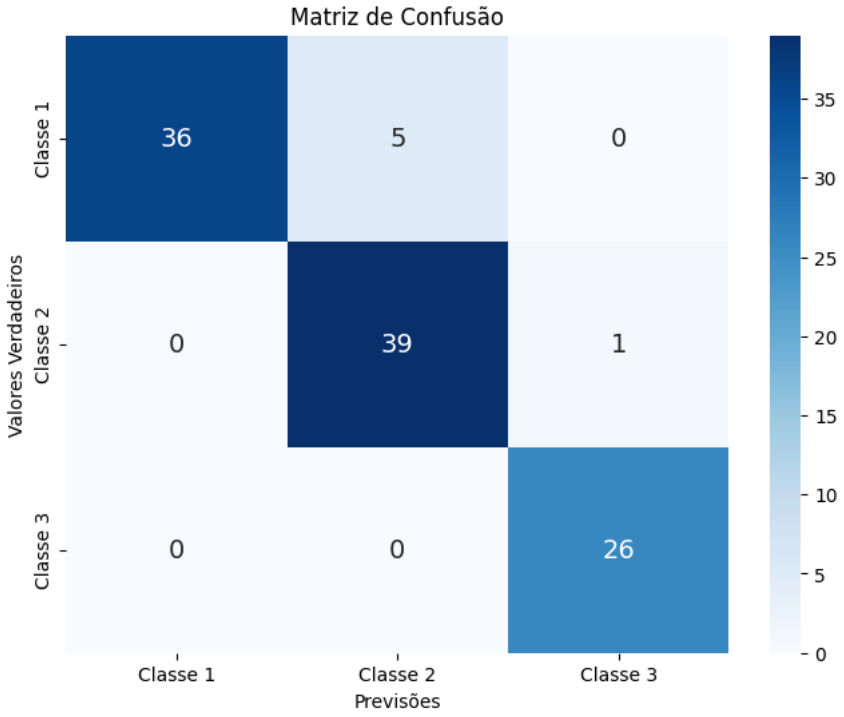

Figura 5: Matriz de confusão.

Com esse resultado, foi possível obter uma acurácia de 94,39%, mostrando uma boa eficiência de aprendizagem.

Por fim, foi plotada a árvore de decisão que modela o sistema, podendo ser visualizada na figura abaixo.

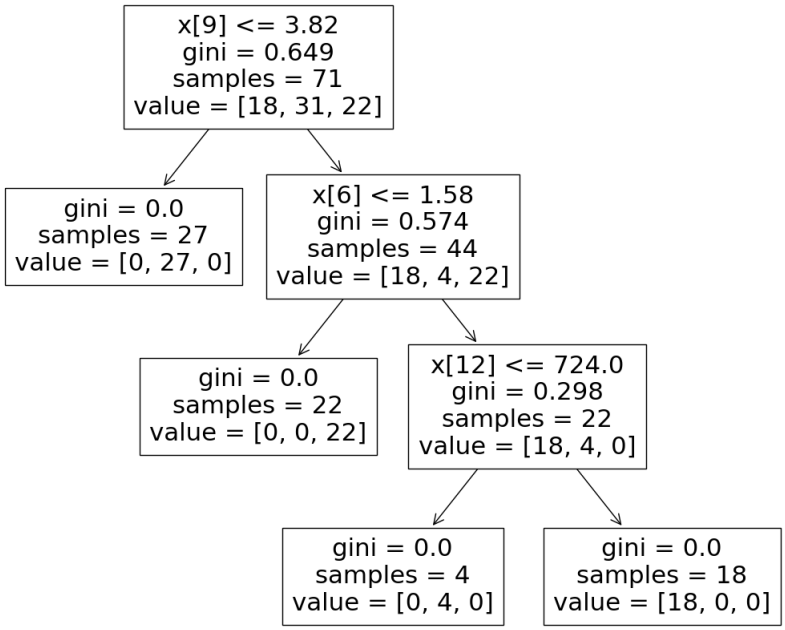

Figura 6: Árvore do problema.

3.1.1 Comparação com o KNN

Observamos que o algoritmo de árvore de decisão permite uma melhor eficiência em relação ao KNN. Em primeiro lugar, O KNN é um algoritmo de aprendizado de máquina baseado em instância que toma decisões com base na proximidade dos pontos de dados no espaço de características. Ele não cria explicitamente um modelo, mas em vez disso, consulta o conjunto de treinamento para encontrar os k vizinhos mais próximos de um novo ponto de dados para fazer previsões enquanto que, a árvore de decisão é um algoritmo de aprendizado supervisionado que cria uma estrutura de árvore para representar decisões e resultados. Ele divide o espaço de características em regiões distintas com base em regras de decisão aprendidas a partir dos dados de treinamento.

Além disso, o desempenho do KNN é altamente sensível ao valor de k escolhido. Um valor de k pequeno pode tornar o modelo muito sensível ao ruído nos dados, enquanto um valor grande pode fazer com que o modelo seja muito simplista. A árvore de decisão também possui hiperparâmetros, como a profundidade da árvore, que podem influenciar o desempenho. O ajuste correto desses hiperparâmetros pode aumentar a precisão do modelo.

3.2 Resultados Heart-Disease

Da mesma maneira do algoritmo apresentado, a matriz de confusão do algoritmo foi apresentada.

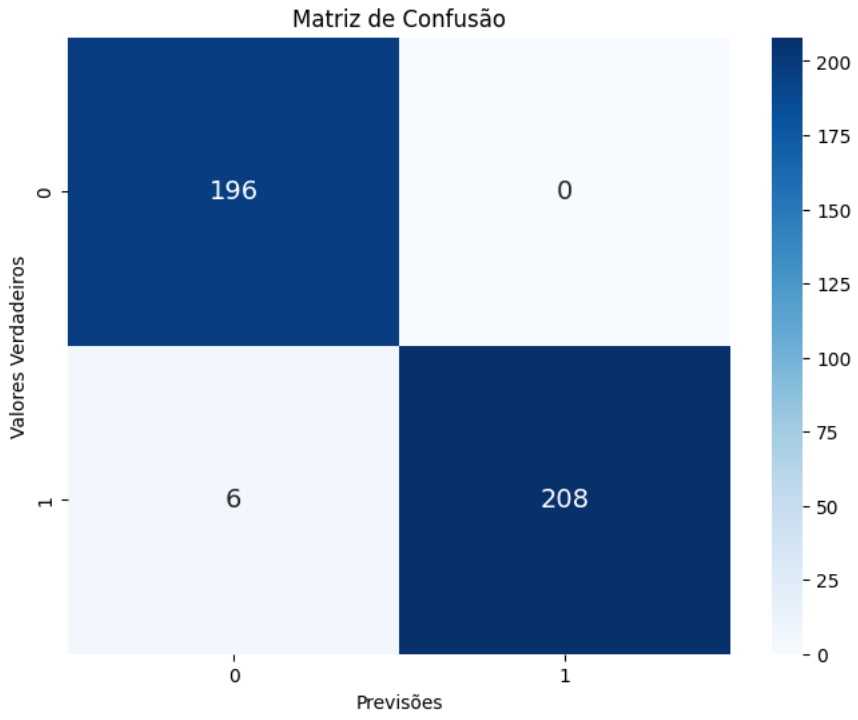

Figura 7: Matriz de confusão.

Nesse caso, como observado, temos apenas dois valores de comparação, já que o banco de dados possui apenas dois parâmetros de classificação: doente ou não.

Com essa implementação, a acurácia do modelo foi de 98,53%.

E por fim, a árvore de decisão que modela o sistema, pode ser visualizada abaixo.

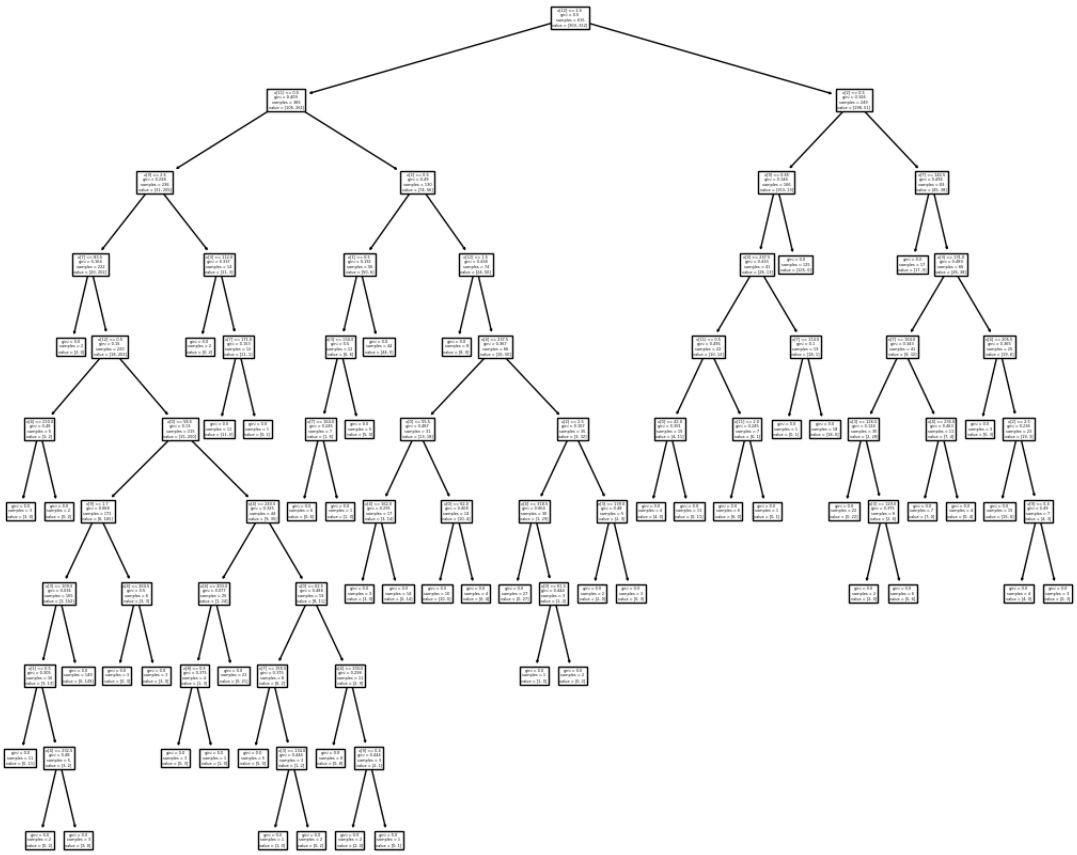

Figura 8: Árvore de decisão.

4 Considerações finais

Com base no apresentado, foi possível a obtenção de êxito na aplicação do algoritmo de árvore de decisão e a comparação com os resultados obtidos anteriormente com a aplicação KNN. Toda implementação foi utilizada através da aplicação das funções disponíveis na Sklearn, Numpy, pandas e tree.

5 Referencial Teórico

[1] Árvores de Decisão. Disponível em: <http://web.tecnico.ulisboa.pt/ana.freitas/bioinformatics.ath.cx/bioinformatics.ath.cx/indexf23d.html?id=199>.

[2] LINK DO CÓDIGO COMPLETO - GOOGLE COLAB.