Nome: Leonardo Santos Miranda Mestrado

Relatório Final - Trabalho Prático 3

Classificação das diversas classes de animais de um zoológico

Base de dados e contextualização

A base de dados escolhida foi encontrada no Kaggle, denominada como Car Features and MSRP (https://www.kaggle.com/uciml/zoo-animal-classification). A base de dados está no formato CSV e possui 101 animais diferentes descritos por 16 atributos. O dataset foi fornecido pelo website UCI Machine Learning, pelo criador Richard Forsyth. Dentre os atributos, temos o nome do animal, se o animal tem cabelo, se o animal tem penas, se o animal é aquático, se o animal é predador, etc. Os animais são classificados em 7 classes distintas:

1. Mamífero
2. Passáro
3. Réptil
4. Peixe
5. Anfíbio
6. Inseto
7. Invertebrado

A base de dados foi escolhida por possuir uma documentação clara e atributos de fácil entendimento para o treinamento de dados nos algoritmos de classificação, como o Multilayer Perceptron, por exemplo.

Objetivo

Treinar um modelo de classificação probabilístico através da base de dados citada na seção anterior. A princípio, o objetivo principal é utilizar o algoritmo de classificação *Multilayer Perceptron* para o treinamento e validação do modelo, observando as acurácias obtidas para cada uma das 7 classes e tentando melhorar através de ajustes nos parâmetros do algoritmo. A base de dados também será treinada e avaliada no algoritmo de Árvore de Decisão comparando e discutindo os resultados obtidos com o algoritmo *Multilayer Perceptron*.

Motivação

Utilizar algoritmos de classificação em cima de uma base de dados que possui multiclasses é uma experiência relativamente nova para mim, visto que as minhas experiências passadas utilizando estes algoritmos sempre foram para classificação binária. Realizar experimentos e avaliar os resultados nessa base de dados heterogênea (possui 17 features e 7 classes possíveis) será de grande ajuda para o entendimento do comportamento dos algoritmos de classificação quando se trata de uma base de dados ampla.

Trabalhos Relacionados

Dentre os trabalhos relacionados com a base de dados, alguns notebooks na página do Kaggle (https://www.kaggle.com/uciml/zoo-animal-classification/notebooks) foram importantes para um maior entendimento dos dados, dentre eles o trabalho que mais se destacou, sendo relevante para este trabalho prático foi o:

- Zoo Animal Classification, do usuário Pranav Kumar (https://www.kaggle.com/pranavkumar623/zoo-animal-classification)

Neste trabalho foi realizada uma breve análise dos dados para um maior entendimento da base de dados antes de realizar o treinamento, utilizando as bibliotecas do Python:

Figura 1 - Leitura da base de dados

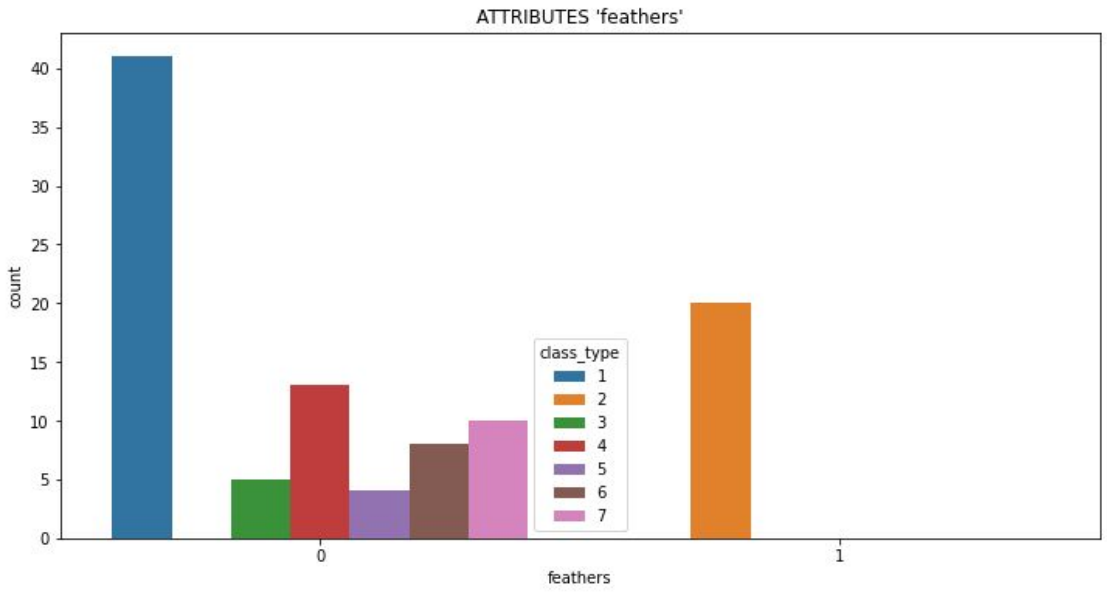
```
import numpy as np
import pandas as pd


df = pd.read_csv('../input/zoo-animal-classification/zoo.csv')
```

Figura 2 - Informações relevantes sobre cada atributo

```
In [6]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 18 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   animal_name  101 non-null    object
 1   hair         101 non-null    int64
 2   feathers     101 non-null    int64
 3   eggs         101 non-null    int64
 4   milk         101 non-null    int64
 5   airborne     101 non-null    int64
 6   aquatic      101 non-null    int64
 7   predator     101 non-null    int64
 8   toothed      101 non-null    int64
 9   backbone     101 non-null    int64
 10  breathes     101 non-null    int64
 11  venomous     101 non-null    int64
 12  fins         101 non-null    int64
 13  legs         101 non-null    int64
 14  tail         101 non-null    int64
 15  domestic     101 non-null    int64
 16  catsize      101 non-null    int64
 17  class_type   101 non-null    int64
dtypes: int64(17), object(1)
memory usage: 14.3+ KB
```

Figura 3 - Plotagem da contagem de cada atributo marcados como “1 (true)” para cada classe da base de dados (i.e. feathers)



Após a breve exploração da base de dados, o ator do notebook realizou o treinamento dividindo 80% dos dados para o conjunto de treinamento e 20% dos dados

para o conjuntos de testes e aplicou o algoritmo de classificação Gaussiana de Naive Bayes, obtendo uma acurácia de aproximadamente 90%.

Figura 4 - Divisão da base de dados, aplicação do algoritmo de classificação e resultados obtidos implementação do algoritmo

Metodologia

A metodologia seguida foi inspirada no CRISP-DM e ilustrada por fluxos de trabalhos realizados na plataforma Lemonade.

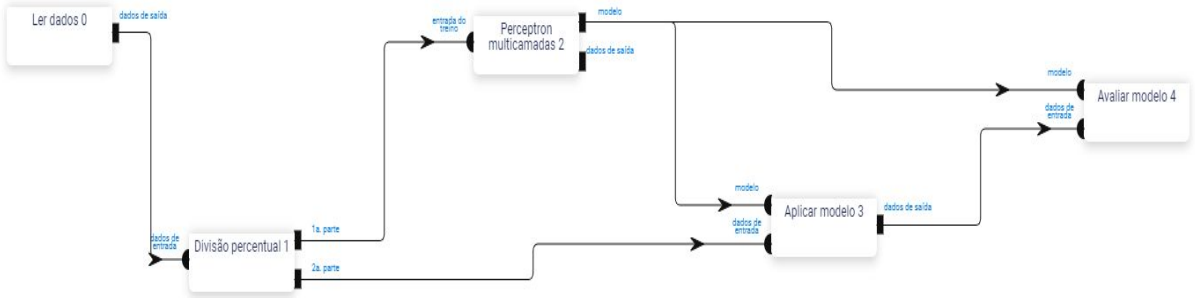
Resultados experimentais e análise

Nesta seção haverá imagens contendo ilustrações dos resultados obtidos através dos fluxos de trabalhos realizados na plataforma Lemonade, após a utilização dos algoritmos de classificação descritos na seção de objetivo.

- **Algoritmo de classificação 1 - *Multilayer Perceptron*:**

Utilizando os blocos da plataforma Lemonade, o seguinte fluxo de trabalho foi definido:

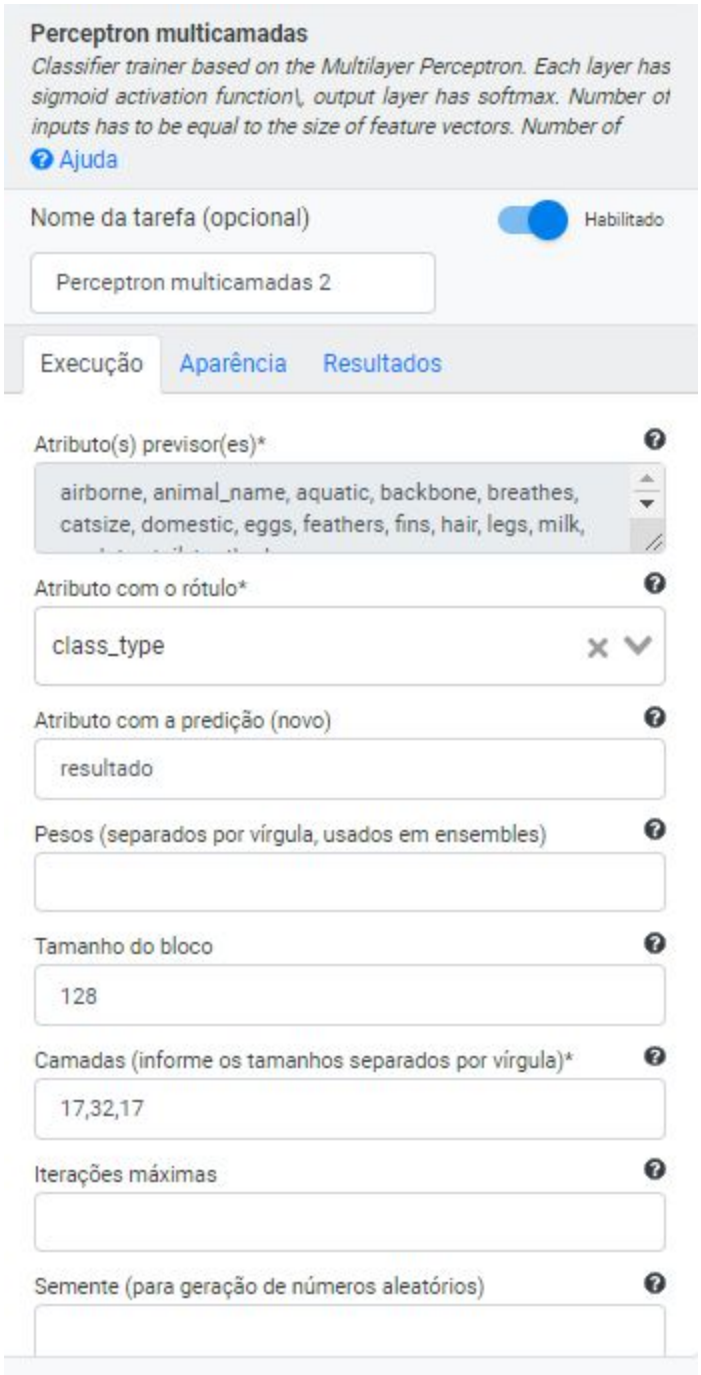
Figura 5 - Fluxo de trabalho para o algoritmo *Multilayer Perceptron*, no Lemonade.



No primeiro bloco (Ler dados) foi realizado a leitura da base de dados em formato tabular (.csv), logo em seguida o próximo bloco (Divisão Percentual) realizou a divisão do conjunto de dados em duas partes, sendo a primeira parte o conjunto de treino e a segunda parte o conjunto de testes. Assim como no notebook citado na seção de trabalhos relacionados, a mesma estratégia para a divisão foi aplicada: 80% dos dados para a 1° parte (treino) e os 20% restantes para a 2° parte (teste).

Seguindo o fluxo, o conjunto de treino é aplicado no terceiro bloco (Perceptron multicamadas), que implementa o algoritmo de classificação *Multilayer Perceptron* com os seguintes parâmetros:

Figura 6 - Parâmetros utilizados na execução do algoritmo *Multilayer Perceptron*, no Lemonade



Em “Atributo(s) Previsor(es)” todas os atributos da base de dados que definem um animal foram selecionadas e em “Atributo com o rótulo” apenas o atributo que define a classe do animal (class_type) foi selecionado para a realização da classificação (gabarito). Nas camadas, 17 camadas de entrada e saída (mesma quantidade de atributos da base de dados) e na camada escondida, diversos valores foram testados

(24,32,51,64) mas o que melhor obteve resultados foi com um total de 32 camadas escondidas.

Após o treino os resultados são aplicados no modelo (Aplicar modelo) juntamente com o conjunto de testes e o modelo é então avaliado (Avaliar modelo) no último bloco do fluxo, obtendo os seguintes resultados:

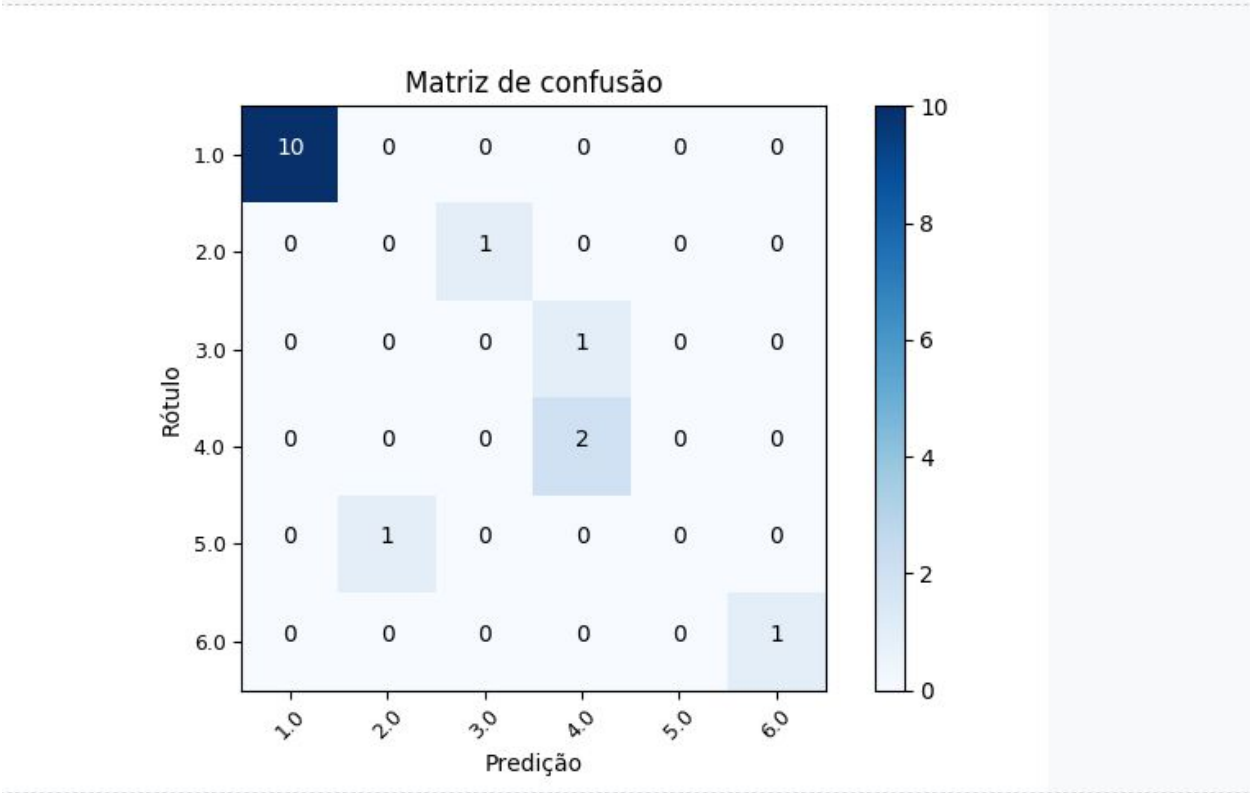
Figura 7 - Acurácia e precisão obtido através do treino do modelo de dados e avaliação no conjunto de testes

Resultado da avaliação

Métrica	Valor
Medida F (F1)	0.7875
Precisão ponderada	0.7708333333333334
Revocação ponderada	0.8125
Acurácia	0.8125

O modelo obteve uma acurácia em torno de 81%, um bom resultado aparentemente. Podemos entrar em mais detalhes e ver em qual classe o modelo mais acertou/errou através da matriz de confusão:

Figura 8 - Matriz de confusão gerada através do resultado que o modelo obteve no conjunto de testes

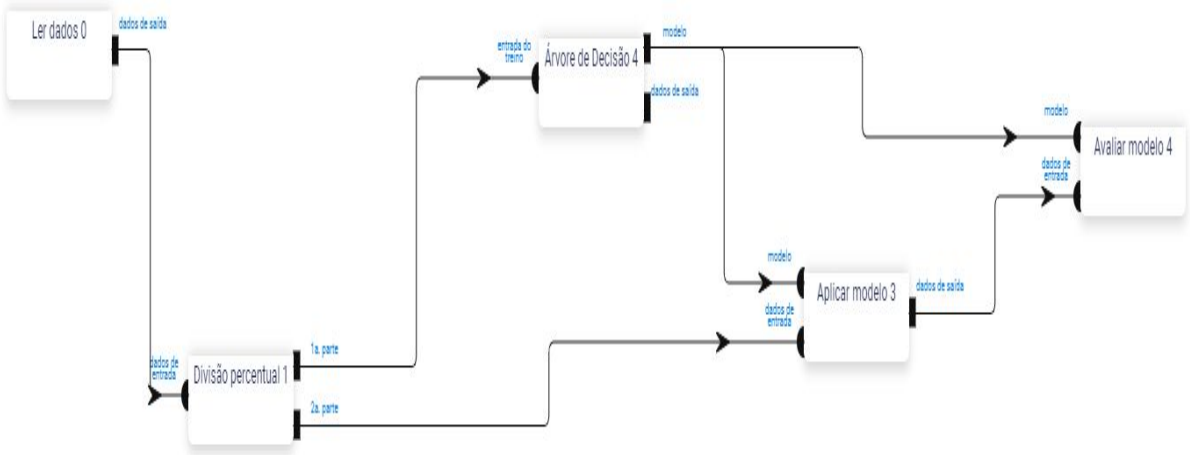


Para a classe 1 (Mamífero) podemos dizer que o nosso modelo foi perfeito! Acertou todas as previsões. Mas, em comparação às outras, podemos ver que o modelo não se saiu tão bem: não acertou nada para a classe 2,3 e 5. Analisando desta maneira, podemos ver que o modelo obteve a acurácia de 80% acertando praticamente apenas uma classe: A de mamíferos. De qualquer forma, foi um resultado satisfatório visto que a base de dados ao todo tinha apenas 101 registros de animais, uma quantidade pequena para a rede neural de fato aprender todas as classes distintas do zoológico.

- **<u>Algoritmo de classificação 2 - *Árvore de Decisão*</u>:**

De forma semelhante, o fluxo de trabalho para o algoritmo de árvore de decisão não teve muitas mudanças: apenas o bloco que implementa o algoritmo foi alterado:

Figura 9 - Fluxo de trabalho para o algoritmo de Árvore de Decisão



Executando-o podemos visualizar a árvore gerada e a importância dos atributos:

Figura 10 - Visualização da árvore de decisão gerada para o modelo

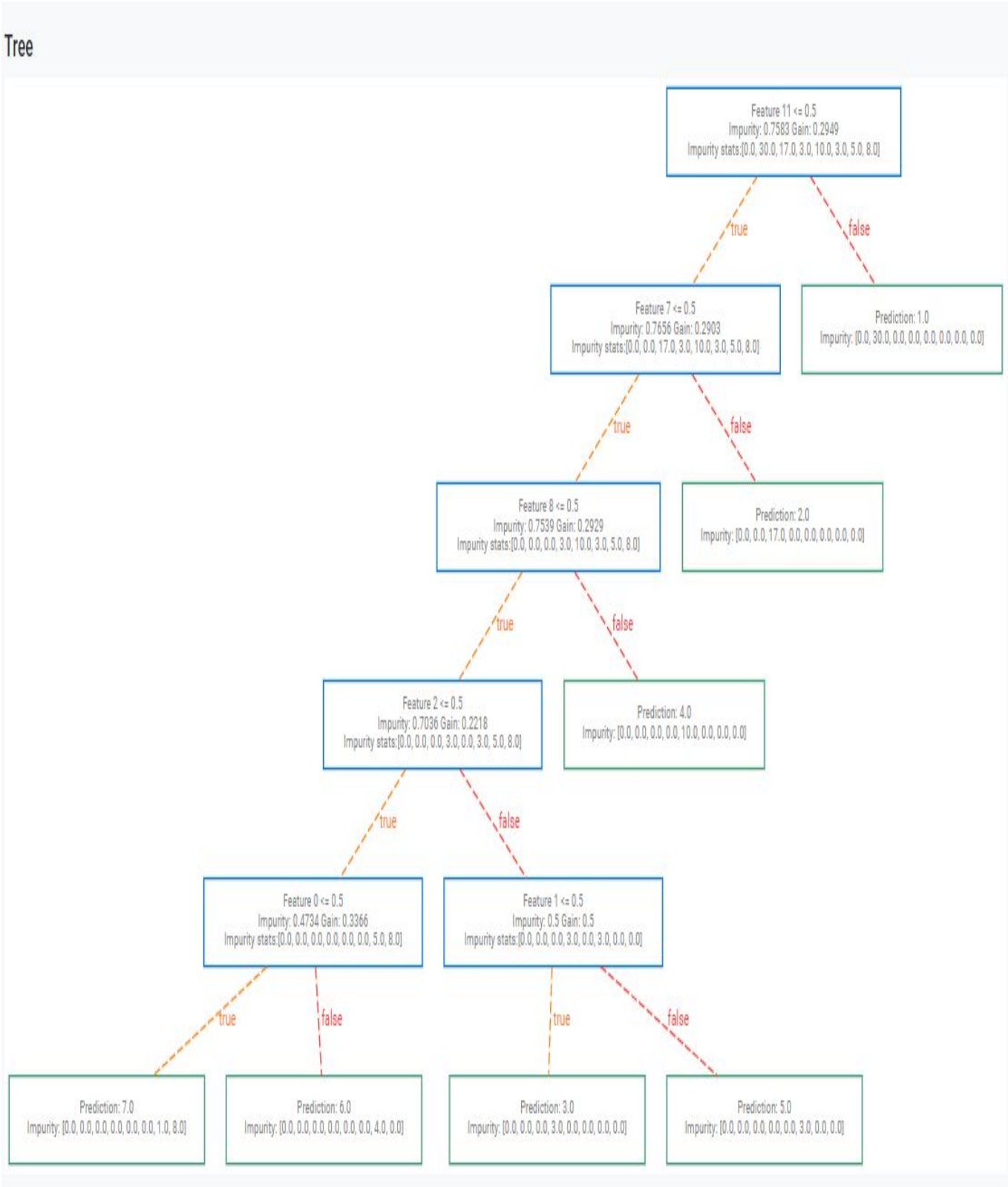
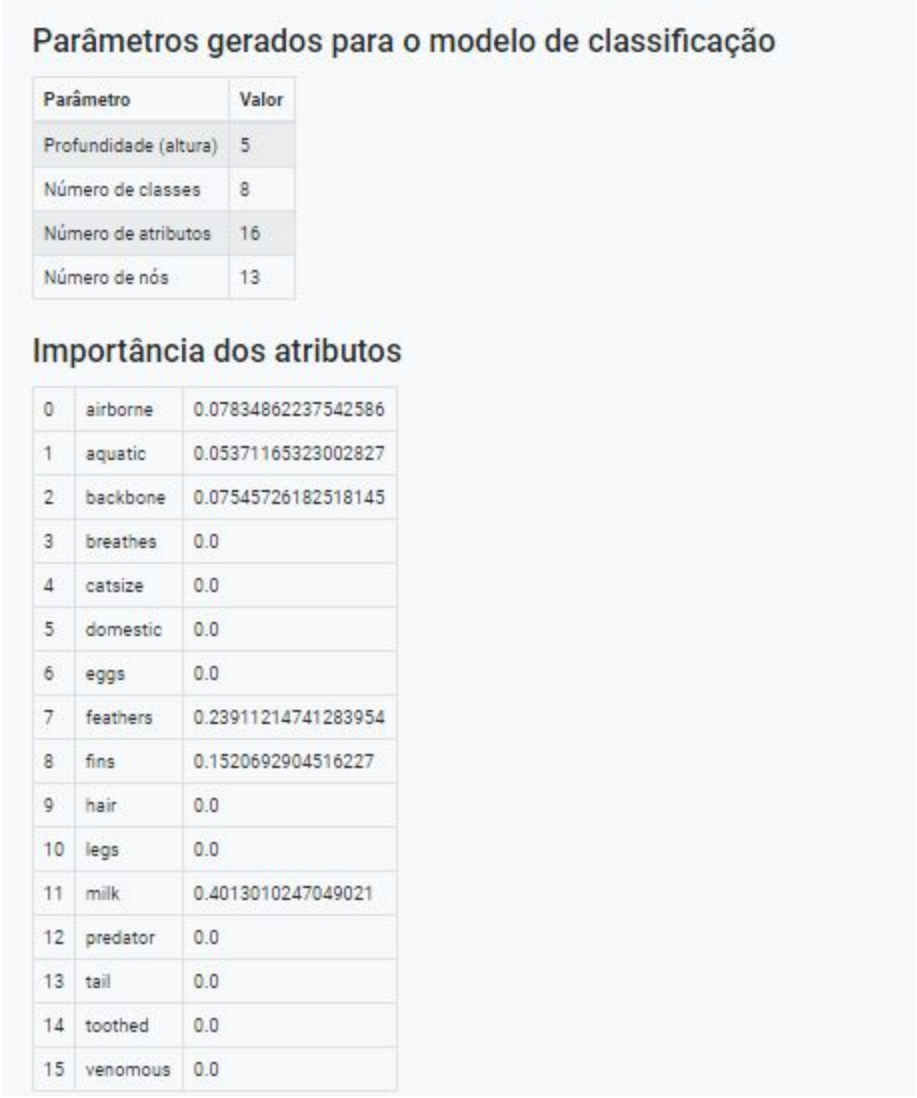


Figura 11 - Importância de cada atributo para a árvore gerada



Parâmetros gerados para o modelo de classificação

Parâmetro	Valor
Profundidade (altura)	5
Número de classes	8
Número de atributos	16
Número de nós	13

Importância dos atributos

0	airborne	0.07834862237542586
1	aquatic	0.05371165323002827
2	backbone	0.07545726182518145
3	breathes	0.0
4	catsize	0.0
5	domestic	0.0
6	eggs	0.0
7	feathers	0.23911214741283954
8	fins	0.1520692904516227
9	hair	0.0
10	legs	0.0
11	milk	0.4013010247049021
12	predator	0.0
13	tail	0.0
14	toothed	0.0
15	venomous	0.0

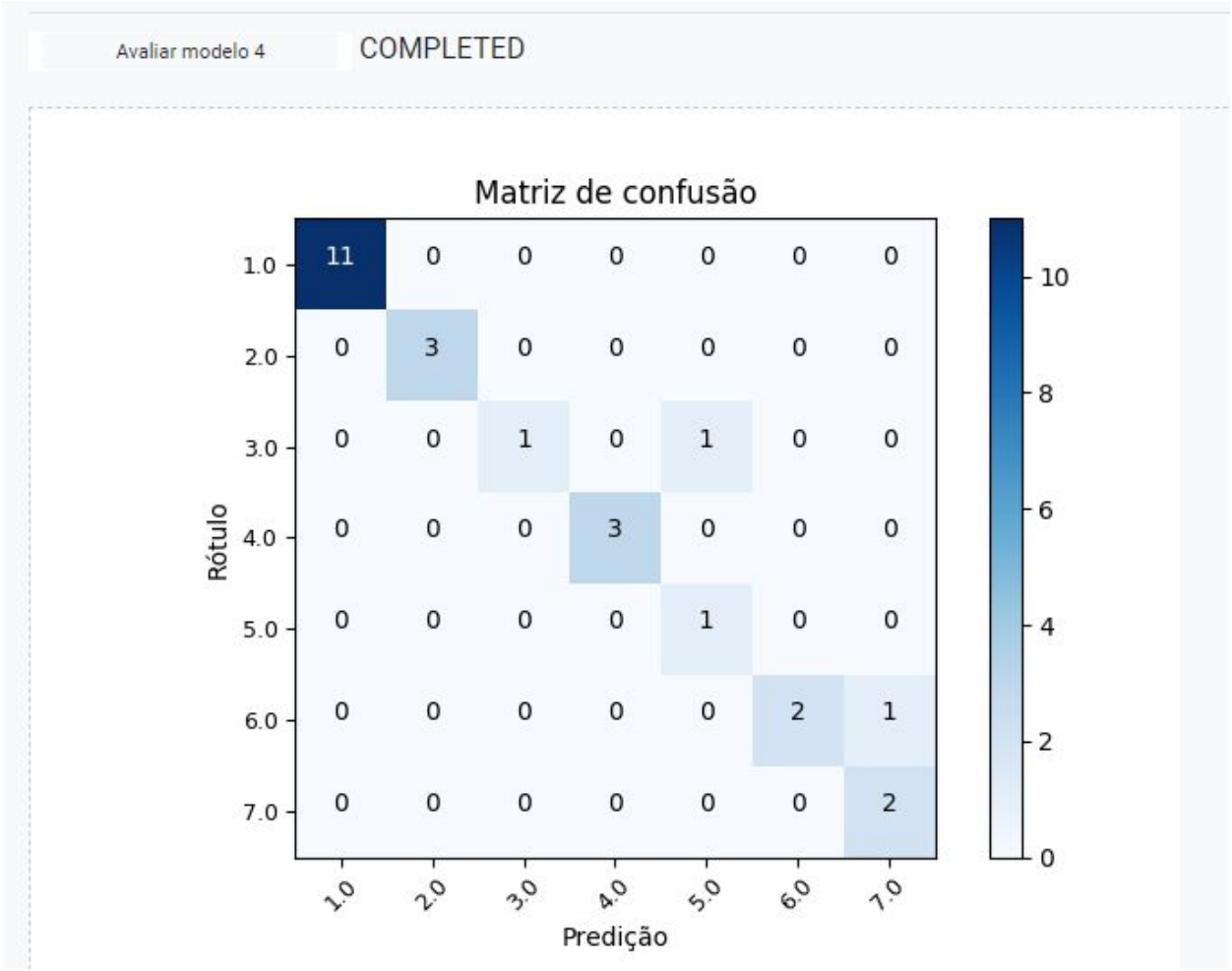
Analisando a importância de cada atributo, podemos ver que há atributos que possui nenhuma relevância, isto porque não são atributo determinantes para a classificação da classe, segundo o algoritmo. Seguindo, os resultados que o modelo previu para o conjunto de testes foram melhores e mais convincentes quando comparado aos resultados obtidos pelo algoritmo de *Multilayer Perceptron*:

Figura 12 - Resultado obtido no conjunto de testes para o algoritmo de árvore de decisão

Resultado da avaliação

Métrica	Valor
Medida F (F1)	0.9199999999999999
Precisão ponderada	0.9533333333333334
Revocação ponderada	0.9199999999999999
Acurácia	0.92

Figura 13 - Matriz de confusão obtida no conjunto de testes para o algoritmo de árvore de decisão



Analisando a matriz de confusão, podemos visualizar que a classe de mamíferos ainda é a mais acertada pelo modelo, mas a diferença é que dessa vez, todas as classes obtiveram pelo menos 1 acerto. Uma melhoria e um resultado mais confiável em relação ao algoritmo anterior.

Conclusões e perspectivas

O desenvolvimento deste trabalho me fez entender de forma prática como os parâmetros são importantes para um bom resultado e como os algoritmos se comportam em cima de uma base de dados multiclasse. Foi observado que o algoritmo de *Multilayer Perceptron* não obteve resultados ótimos, acredito pelo fato de ele precisar de uma base de dados com um maior número de registros. De outro lado, o algoritmo de Árvore de Decisão obteve ótimos resultados, por mais que a base de dados tivesse apenas 101 registros diferentes, ele obteve pelo menos um acerto para cada classe. Também foi observado que a classe de mamíferos foi a mais acertada em ambos os algoritmos. Espero que os resultados que obtive através das diversas execuções realizadas no Lemonade sejam satisfatórios e tenham relevância para este trabalho prático. Vou anexar juntamente com este pdf uma pasta com todas as imagens dos resultados obtidos e fluxos de execuções, caso sejam difíceis de visualizar.