Relatório Mineração de Dados

Aluno: Lucas Andrei Moraes da Silva

Base de dados: Horse Colic Data Set

Link: https://archive.ics.uci.edu/ml/datasets/Horse+Colic

Dados: Aspectos patológicos de cavalos que apresentam cólicas.

Número de Amostras: 300

Número de Atributos: 28

Tipos de atributos: numéricos e discretos.

Valores ausente: Sim

Objetivo: ser capaz de, por meio dos dados fornecidos em relação a saúde do cavalo com cólicas, definir se o cavalo viveu, morreu ou foi eutanasiado.

Github: https://github.com/lucasamsilva/Data-Mining

Obs.: Todos os comandos devem ser executados a partir do diretório raiz do projeto.

1 – Seleção e pré-processamento de dados:

Passos:

1. Nomeação de todos atributos.
2. Especificação dos atributos que serão utilizados.
3. Identificação de atributos numéricos e discretos.
4. Especificação de atributos numéricos e discretos.
5. Identificação de dados faltantes.
6. Preenchimento de dados faltante de acordo com sua categoria. Obs.: Numéricos (Média); Discretos (Moda).
7. Geração de um arquivo com os dados ausentes preenchidos.

Arquivo: **PreProcessamento/Limpeza.py**

Comando: **python PreProcessamento/Limpeza.py**

Resultado: Foi gerado um novo arquivo chamado horse-colic-clean.data dentro do diretório Dataset, esse novo arquivo não contém valores faltantes e será utilizado nas próximas etapas no lugar do arquivo original do dataset.

2 – Normalização e redução de dados:

Passos:

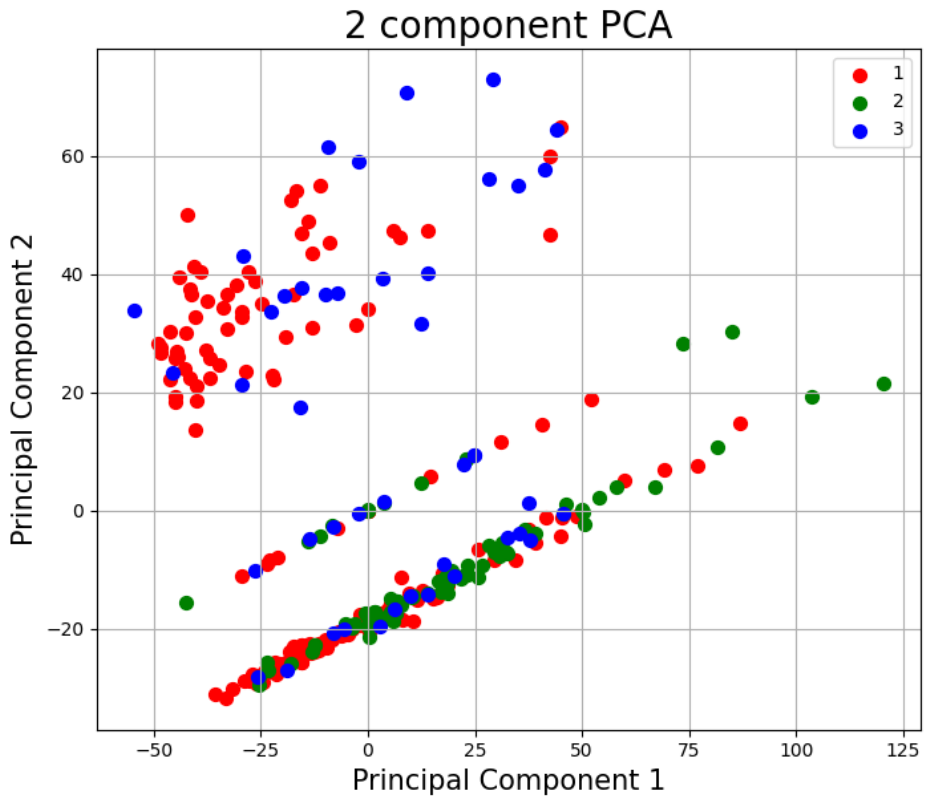
1. Redução do dataset por meio do Z-Score.
2. Criação de um novo arquivo chamado horse-colic-clean-normalized.data dentro do diretório Dataset
3. Utilização da biblioteca sklearn para realizar a análise de componente principal (PCA).
4. Impressão das componentes principais e realização do plot do gráficos com as duas principais componentes.

Arquivos: **PreProcessamento/Normalizacao.py, PreProcessamento/Reducao.py**

Comando: **python PreProcessamento/Normalizacao.py && python PreProcessamento/Reducao.py**

Resultado: A primeira imagem demonstra a variância por componente, é possível notar que as duas componentes principais representam mais de 75% da variação total, isso demonstra uma boa relação entre os dados do dataset. Na segunda imagem é possível observar o resultado da plotagem utilizando as duas componentes principais.

```
Explained variance per component:
[0.4790427790858315, 0.360269398774071, 0.1060364732950811, 0.045541671293296025, 0.0018769350659
985462, 0.001354854920594242, 0.0009589325475500258, 0.0006861334608155692, 0.0006042041698051516
, 0.0005457317614732829, 0.0004679318257336 1364, 0.0004328295509507 7515, 0.0003489735563654091, 0
.0002813007946801 4255, 0.0002722318785521 4065, 0.0002469748761886 7723, 0.0002172699903711 8707, 0.
0001990379739512733, 0.0001525981364090 6828, 0.0001481904546092 4785, 0.0001166148669423 4183, 8.57
6400028074572e-05, 6.886312297313807e-05, 4.430459747588282e-05]
```



3 – Análise descritiva de dados - Visualização:

Passos:

1. Seleção dos dados que serão utilizados para realizar a distribuição de frequência.
2. Plotagem dos gráficos.

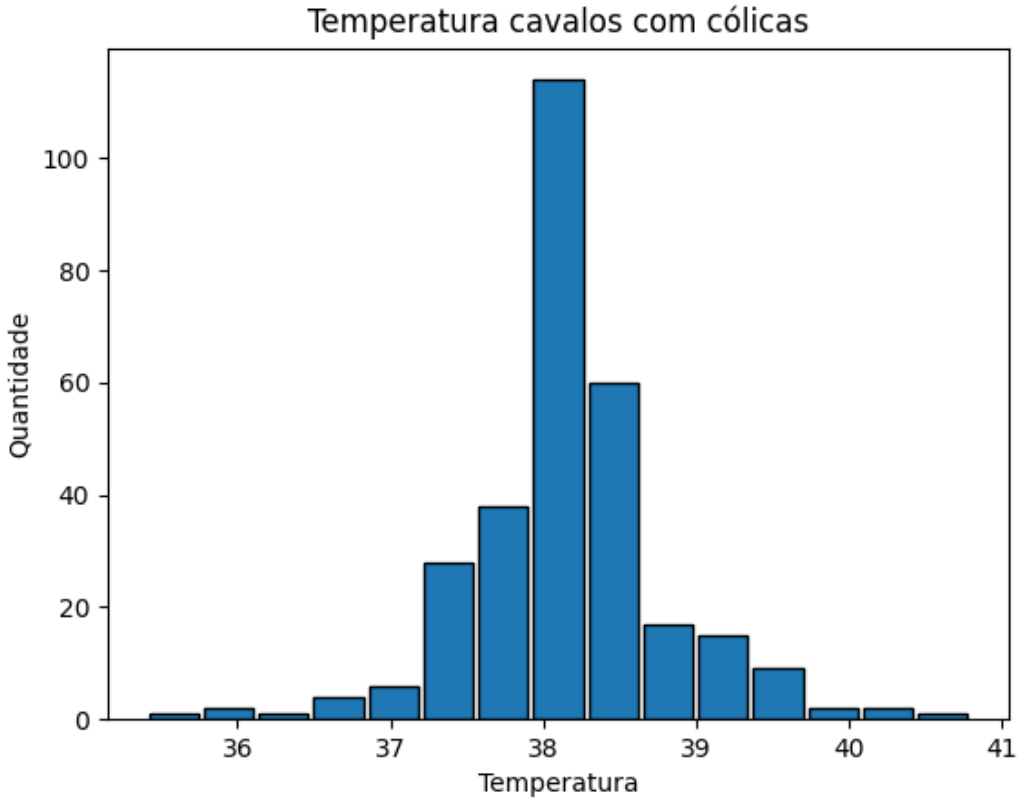
Arquivos: **Visualizacao/Graficos.py**

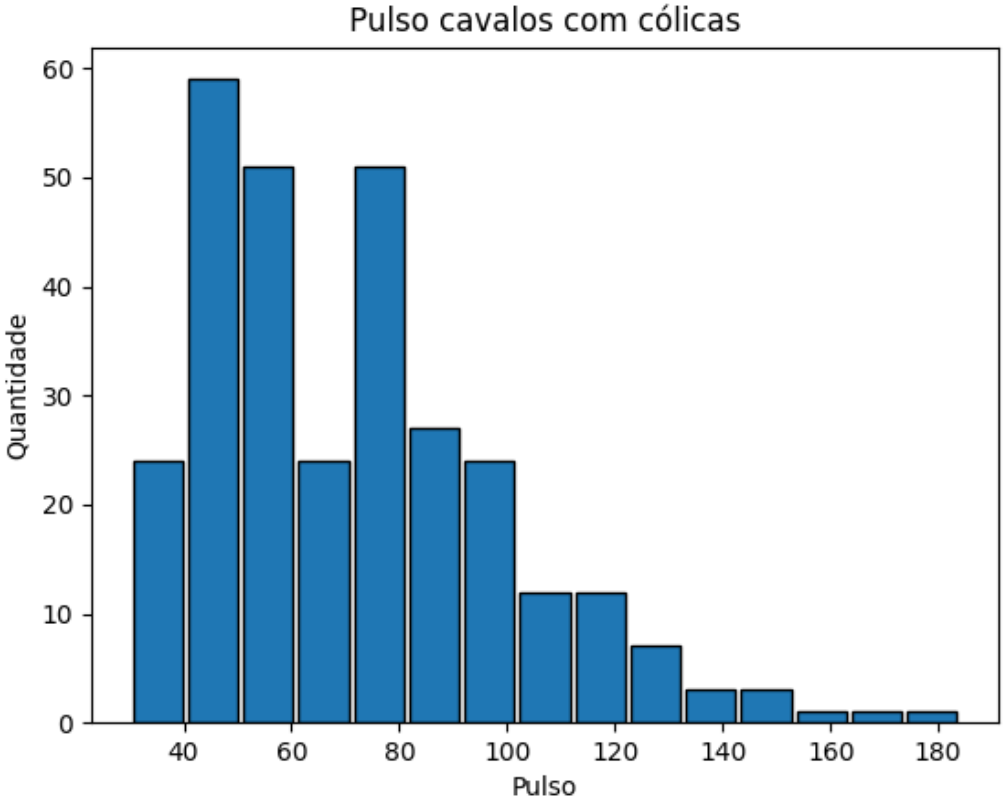
Comando: **python Visualizacao/Graficos.py**

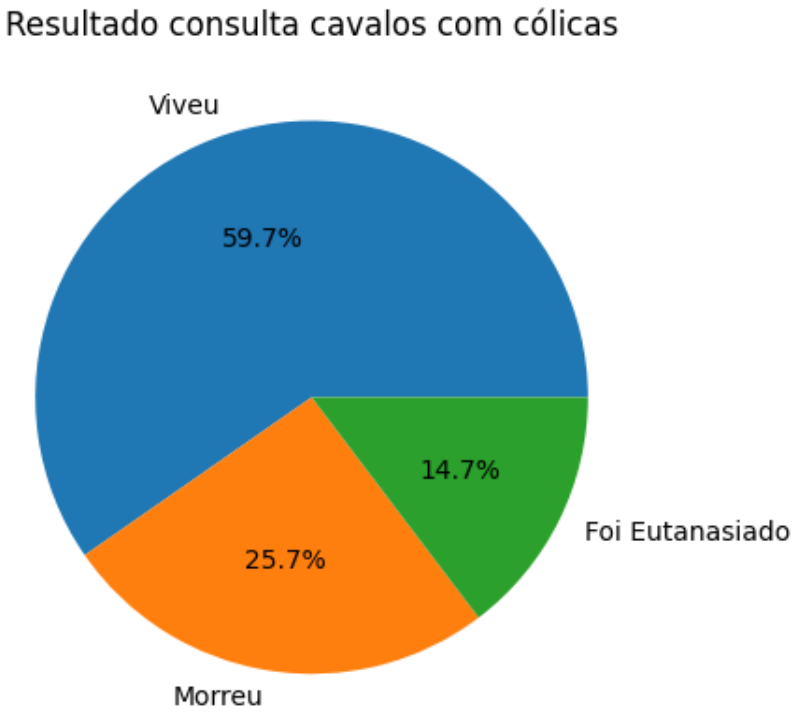
Resultado: O primeiro gráfico apresenta as temperaturas dos cavalos com cólicas, a maior parte dos cavalos analisados apresentam temperaturas entre 37ºC e 39ºC, sendo uma faixa normal para cavalos.

O segundo gráfico apresenta o pulso dos cavalos com cólicas, com uma parte dos dados distribuídos na faixa entre 40 e 60 batimentos, valores normais para cavalos, no entanto existe um número considerável de cavalos com batimentos entre 60 e 100, valores considerados anormais para esses animais.

O terceiro gráfico representa se o cavalo com cólica viveu, morreu ou foi eutanasiado, com quase 60% dos animais sobrevivendo, 25,7% morrendo e 14,7% sendo eutanasiados.


Temperatura cavalos com cólicas
Quantidade
0
20
40
60
80
100
36
37
38
39
40
41
Temperatura


Pulso cavalos com cólicas
Quantidade
0
10
20
30
40
50
60
40
60
80
100
120
140
160
180
Pulso



4 – Análise descritiva de dados – Medidas:

Passos:

1. Seleção dos valores de pulso do cavalo.
2. Calculo dos valores das medidas de tendência central para o pulso: media, moda, mediana e ponto médio.
3. Calculo dos valores das medidas de dispersão para o pulso: amplitude, desvio padrão, variância e coeficiente de variação.
4. Calculo dos valores de medidas de posição relativa para o pulso: boxplot.

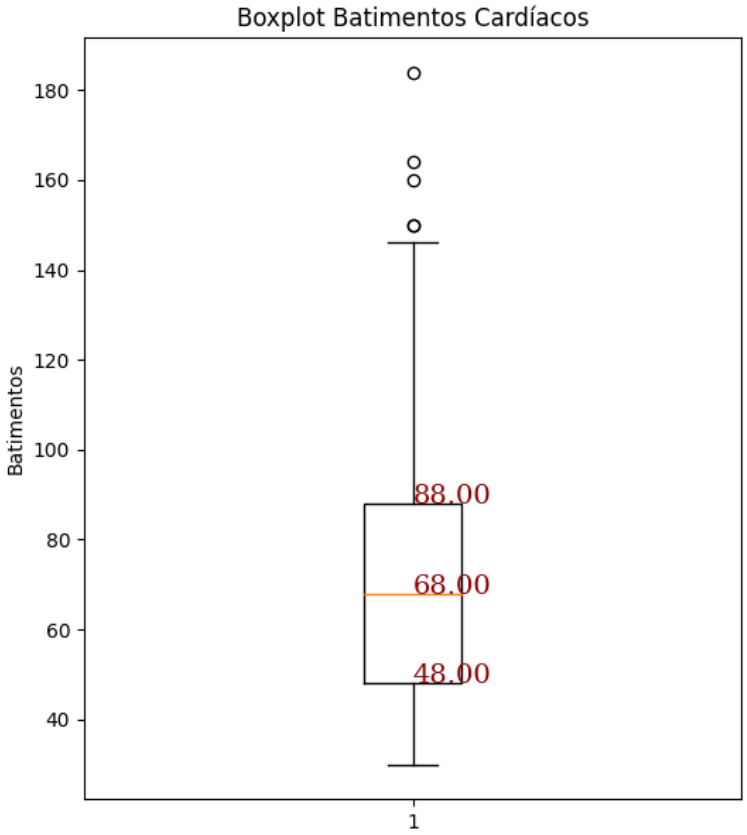
Arquivos: **Visualizacao/ValoresPulso.py**

Comando: **python Visualizacao/ValoresPulso.py**

Resultado: Na primeira imagem é possível observar os resultados para as medidas de tendência central e dispersão para o pulso dos animais. Enquanto na segunda imagem é possível o boxplot referente aos dados de pulso dos cavalos.

```
Tendência Central de batimentos cardíacos cavalos com cólicas
Média = 71.912
Moda = 48.0
Mediana = 68.0
Ponto Médio = 107.0

Medidas de dispersão de batimentos cardíacos cavalos com cólicas
Amplitude = 154.0
Desvio Padrão = 27.411671285543072
Variância = 751.3997226666667
Coeficiente de Variação = 38.12%
```



5 – Análise de grupos:

Passos:

1. Definição do número de grupos pelo parâmetro “k”.
2. Extração das 2 principais componentes.
3. Calculo do K-Means por meio de função própria e função da biblioteca sklearn.
4. Demonstração da pontuação de silhueta para o número de grupos “k”.

Arquivos: **Clustering/Kmeans.py**

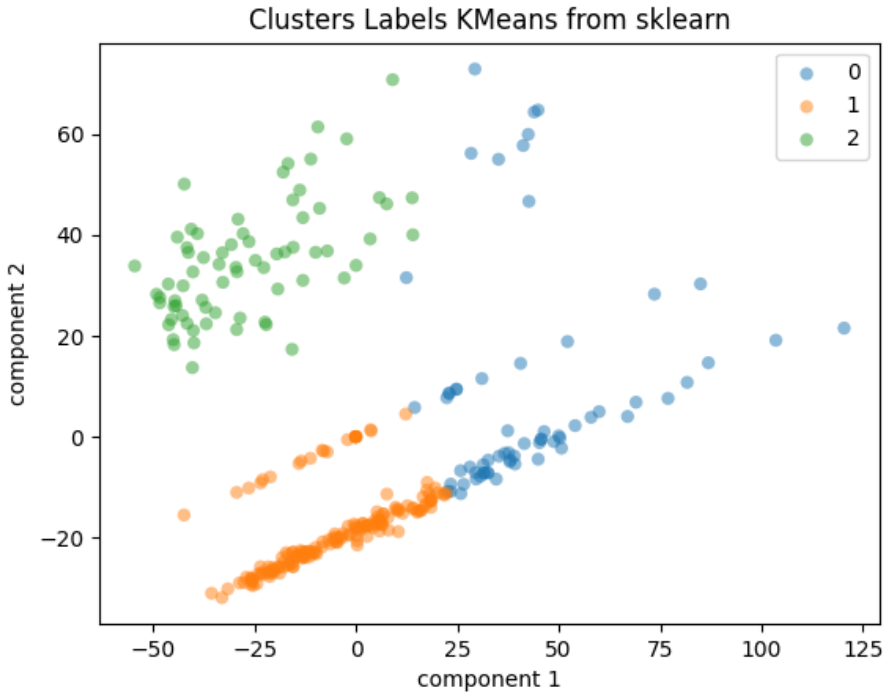
Comando: **python Clustering/Kmeans.py**

Resultado: o algoritmo foi executado utilizando 2, 3, 4 e 5 números de grupos.

K = 2

Pontuação de Silhueta = 0.515

K = 3

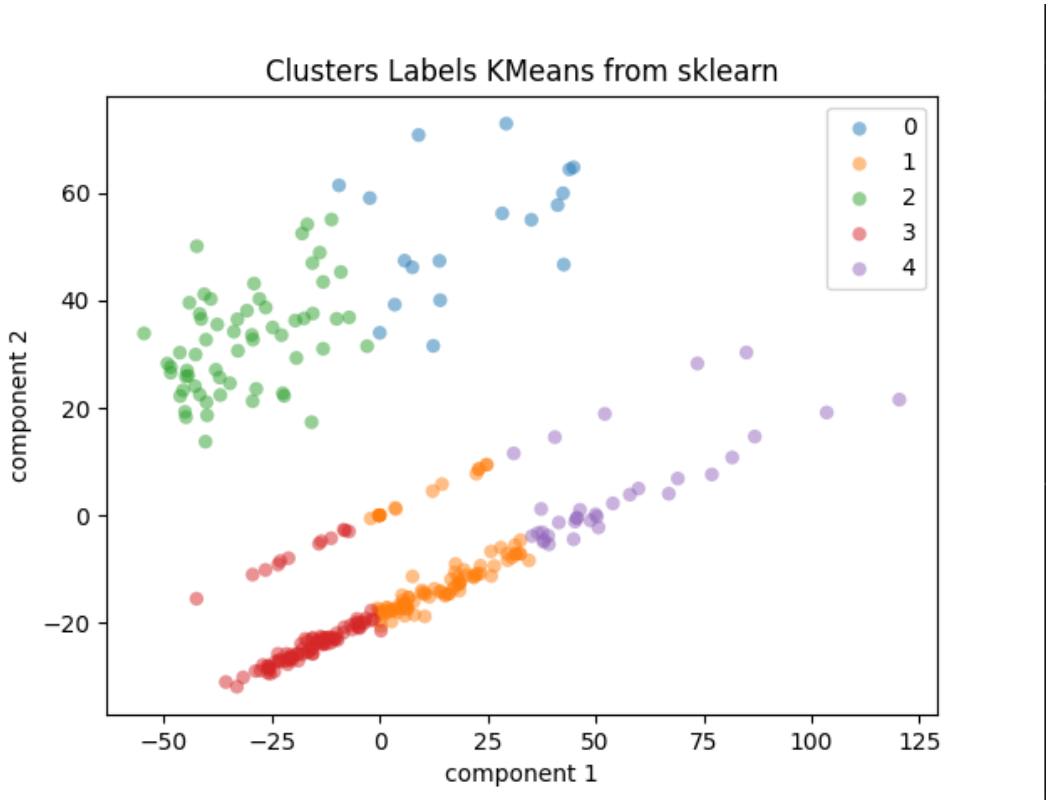


Pontuação de Silhueta = 0.543

K = 4

Pontuação de Silhueta = 0.512

K = 5



Pontuação de Silhueta = 0.499

Foi possível obter a melhor pontuação de silhueta quando utilizado 3 grupos, o que era esperado, pois os cavalos podem ser separados exatamente em 3 grupos, viveu, morreu ou foi eutanasiado. No entanto é possível perceber que as pontuações não são muito diferentes e que existem pontos de intersecção entre os grupos, talvez removendo algumas das informações possa fazer com que sejam obtidos resultados melhores.

5 – Classificação – KNN:

Passos:

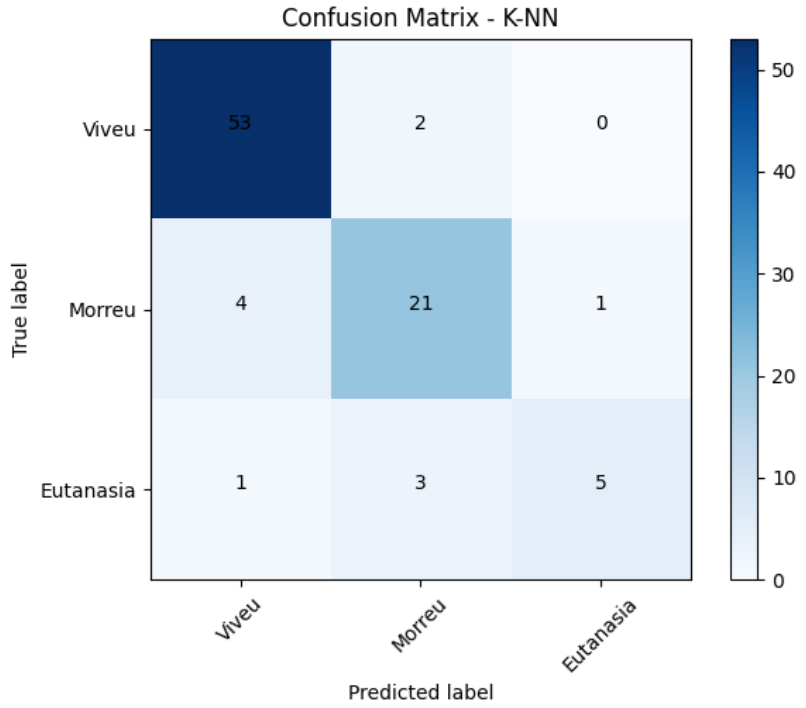
1. Divisão do dataset entre dados para treino e dados para teste
2. Normalização por meio de Z-Score
3. Execução do KNN sem utilização de biblioteca externa
4. Cálculo da acurácia, F1 score e matriz de confusão
5. Execução do KNN com utilização da biblioteca sk-learn
6. Cálculo da acurácia, F1 score e matriz de confusão
7. Cross-validation para o KNN utilizando a biblioteca sk-learn com k = 10

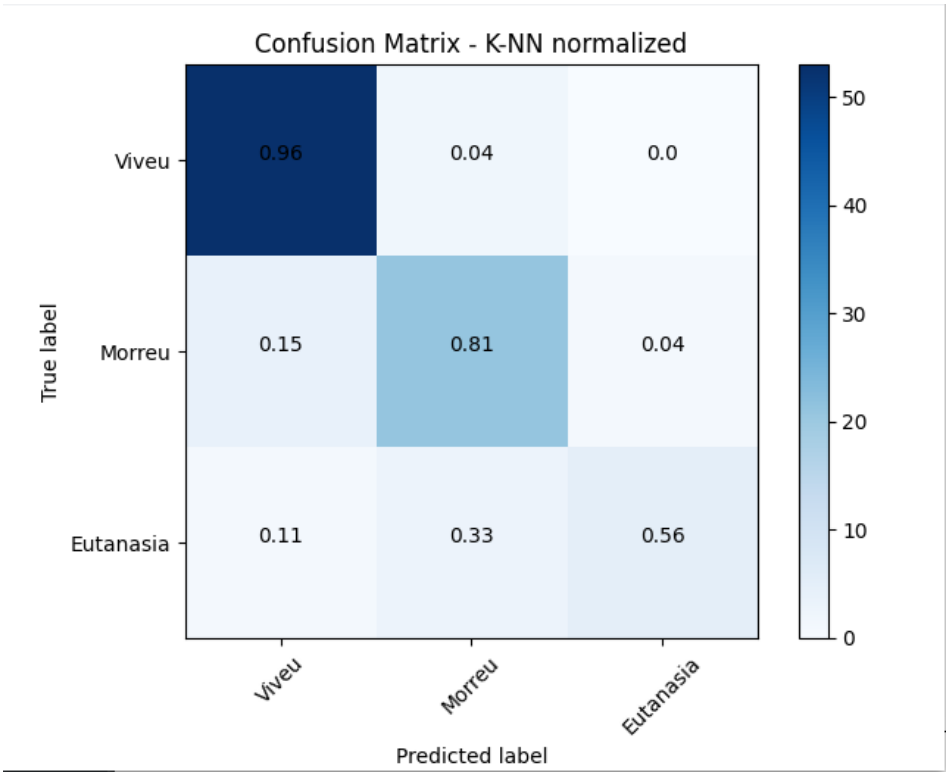
8. Plotagem das matrizes de confusão

Arquivos: **Classificação/knn.py**

Comando: **python Classificação/knn.py**

Resultado: Para execução sem biblioteca externa foram obtidos os seguintes resultados:

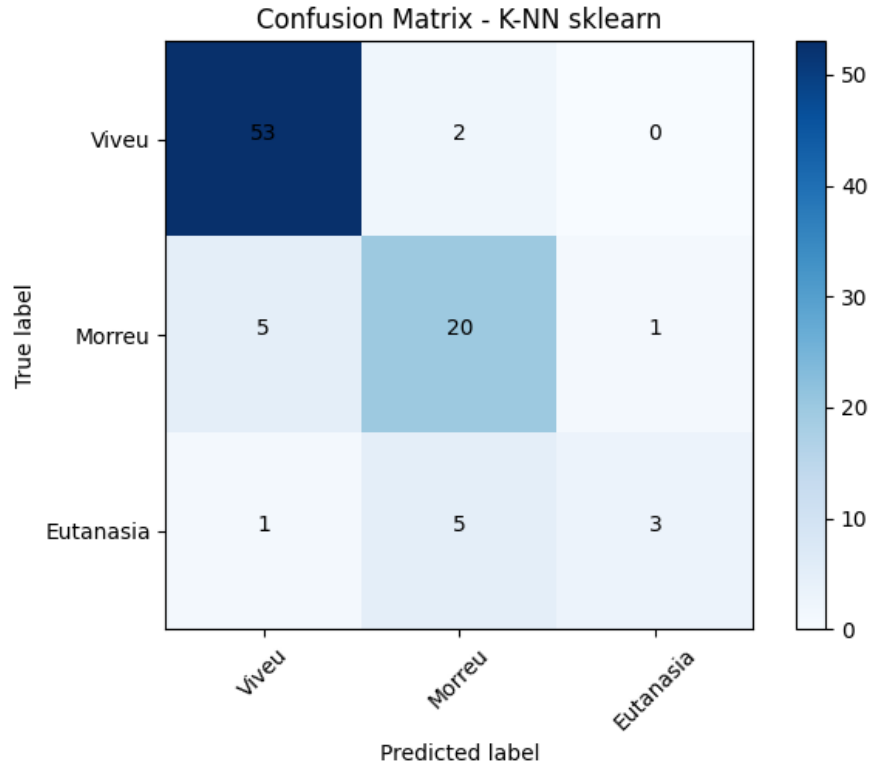


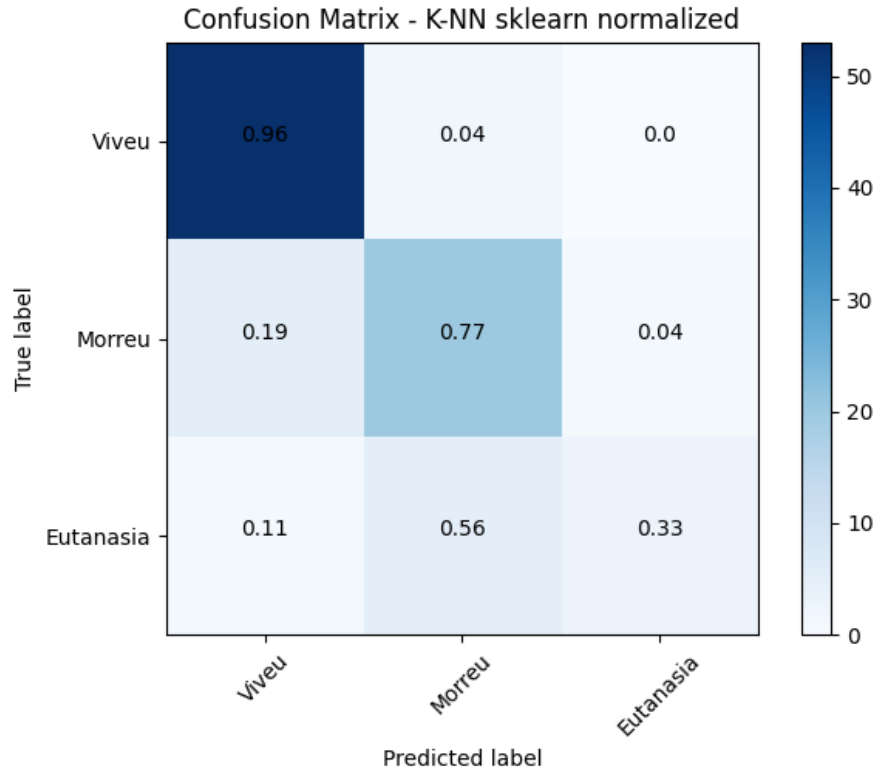


Acurácia = 87,78%

F1 Score = 0,80%

Utilizando a função fornecida pela biblioteca sk-learn foram obtidos os seguintes resultados:





Acurácia = 84,44%

F1 Score = 0,72%

Acurácia média Cross-Validation = 64,66%

6 – Classificação – SVM:

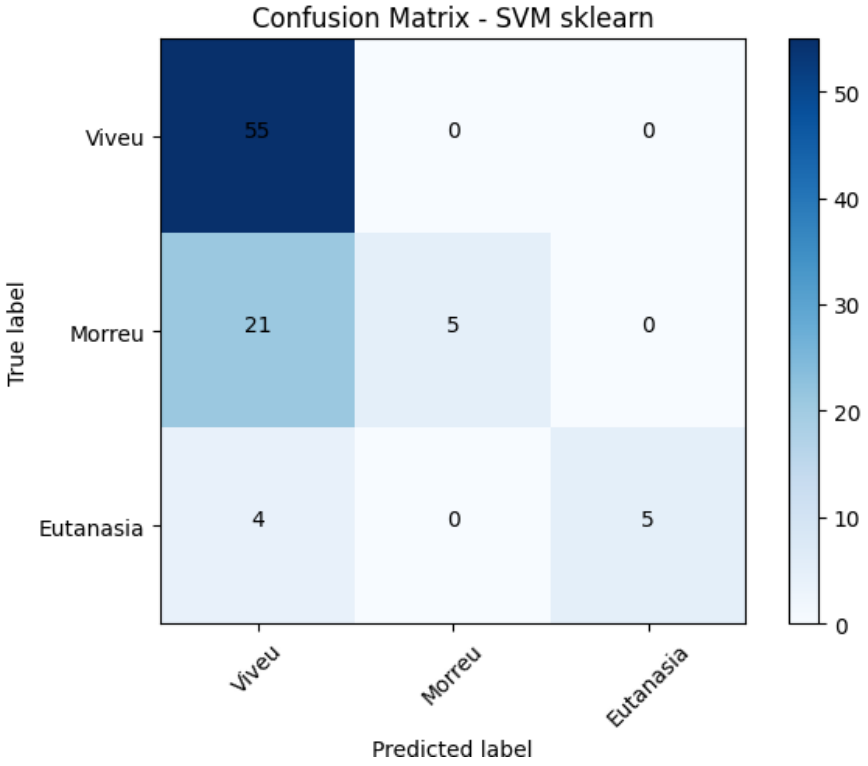
Passos:

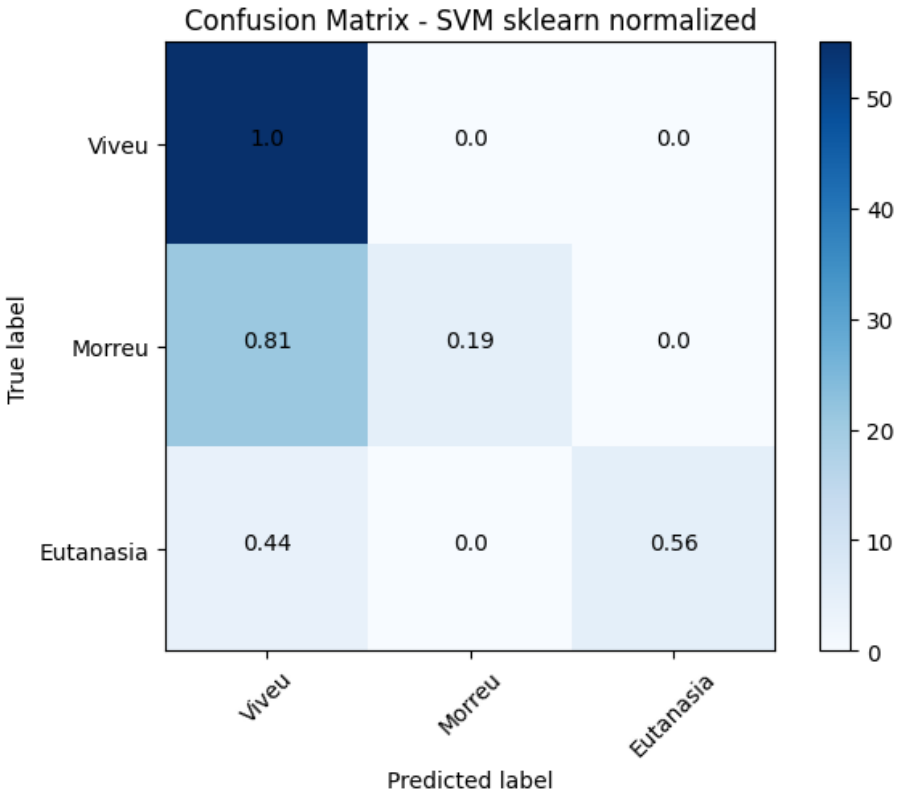
1. Divisão do dataset entre dados para treino e dados para teste
2. Normalização por meio de Z-Score
3. Execução do SVM com utilização da biblioteca sk-learn
4. Cálculo da acurácia, F1 score e matriz de confusão
5. Cross-validation para o SVM utilizando a biblioteca sk-learn com k = 10
6. Plotagem das matrizes de confusão

Arquivos: **Classificação/svm.py**

Comando: **python Classificação/svm.py**

Resultados: Utilizando a função disponibilizada pela biblioteca sk-learn foram obtidos os seguintes resultados para a técnica SVM:





Acurácia = 72,22%

F1 Score = 0,62%

Acurácia média Cross-Validation = 65%