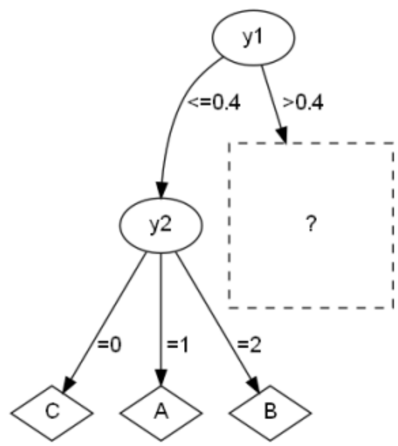
Aprendizagem 2023
Homework I – Group 78
(ist1102633, ist1102741)

Part I: Pen and paper

Considere a árvore de decisão parcial proveniente do DataSet D. D é descrito por 4 variáveis de input- uma numérica (y_1) com valores [0,1] e 3 categóricas (y_2, y_3 e y_4) - e uma variável de output com 3 classes.

D	y_1	y_2	y_3	y_4	y_{out}
$\mathbf{x}_1$	0.24	1	1	0	A
$\mathbf{x}_2$	0.06	2	0	0	B
$\mathbf{x}_3$	0.04	0	0	0	B
$\mathbf{x}_4$	0.36	0	2	1	C
$\mathbf{x}_5$	0.32	0	0	2	C
$\mathbf{x}_6$	0.68	2	2	1	A
$\mathbf{x}_7$	0.9	0	1	2	A
$\mathbf{x}_8$	0.76	2	2	0	A
$\mathbf{x}_9$	0.46	1	1	1	B
$\mathbf{x}_{10}$	0.62	0	0	1	B
$\mathbf{x}_{11}$	0.44	1	2	2	C
$\mathbf{x}_{12}$	0.52	0	2	0	C



1. De forma a construir as árvores de decisão, necessitamos de conhecer o Information Gain das várias variáveis, em cada nó, de forma a privilegiar aquelas que tenham Information Gain superior. Este é calculado por:

$$IG(y_i) = H(z) - H(z|y_i) \tag{1}$$

em que

$$H(z) = \sum_{i=0}^{n} -P(z_i) \cdot log_2(P(z_i)) \tag{2}$$

$$H(z|y_i) = \sum_{i=0}^{X_i} P(X_i) \cdot H(z|X_i) \tag{3}$$

Como tal, para completar o lado direito da árvore sugerida no enunciado ($y_1 > 0.4$), é necessário calcular o Information Gain para as variáveis y_2, y_3 e y_4, apenas nos acontecimentos x_i que tenham $y_1 > 0.4$, para, com isso, escolhermos aquela que será a variável do nó seguinte.

$$H(z) = -P(A) \cdot \log_2(P(A)) + -P(B) \cdot \log_2(P(B)) + -P(C) \cdot \log_2(P(C)) = 1.557$$

$$
\begin{aligned}
H(z|y_2) &= P(y_2=0)\cdot H(z|y_2=0) + P(y_2=1)\cdot H(z|y_2=1) + P(y_2=2)\cdot H(z|y_2=2) = \\
&= P(y_2=0)\Big[-P(A|y_2=0)\cdot\log_2\big(P(A|y_2=0)\big) - P(B|y_2=0)\cdot\log_2\big(P(B|y_2=0)\big) \\
&\qquad -P(C|y_2=0)\cdot\log_2\big(P(C|y_2=0)\big)\Big] \\
&+P(y_2=1)\Big[-P(A|y_2=1)\cdot\log_2\big(P(A|y_2=1)\big) - P(B|y_2=1)\cdot\log_2\big(P(B|y_2=1)\big) \\
&\qquad -P(C|y_2=1)\cdot\log_2\big(P(C|y_2=1)\big)\Big] \\
&+P(y_2=2)\Big[-P(A|y_2=2)\cdot\log_2\big(P(A|y_2=2)\big) - P(B|y_2=2)\cdot\log_2\big(P(B|y_2=2)\big) \\
&\qquad -P(C|y_2=2)\cdot\log_2\big(P(C|y_2=2)\big)\Big] =
\end{aligned}
$$

$$
= \frac{3}{7}\cdot\left[-\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right] + \frac{2}{7}\cdot\left[-\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right] + \frac{2}{7}\cdot\left[-1\log_2(1)\right] = 0.965
$$

$$IG(y_2) = H(z) - H(z|y_2) = 0.592$$

Para y_3 e y_4, foram feitos os mesmos cálculos.

$$IG(y_3) = H(z) - H(z|y_3) = 0.700$$

$$IG(y_4) = H(z) - H(z|y_4) = 0.592$$

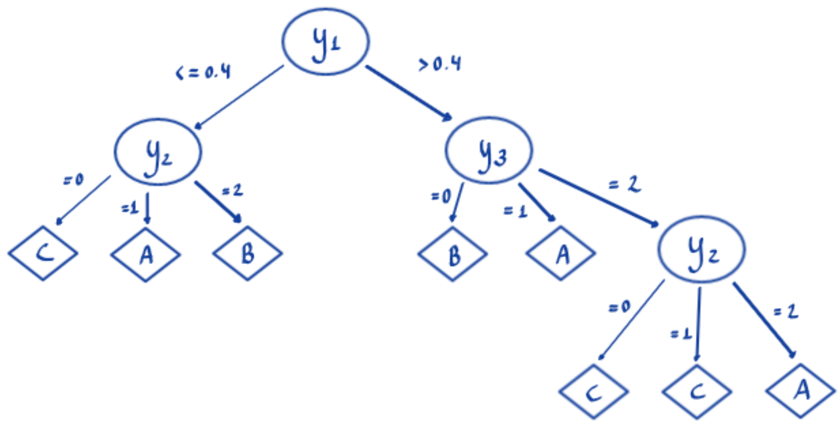
Concluímos assim que a variável com o maior ganho de informação é y_3, pelo que é a seguinte a ser colocada no lado direito da árvore. De seguida, calculámos o ganho de informação para as variáveis y_2 e y_4 (de forma igual ao feito anteriormente), para os valores de $y_3 = 2$, dado que estes são os únicos valores com mais de 4 observações.

$$H(z) = -P(A)\cdot log_2(P(A)) - P(C)\cdot log_2(P(C)) = -\frac{1}{2}\cdot log_2\left(\frac{1}{2}\right) - \frac{1}{2}\cdot log_2\left(\frac{1}{2}\right) = 1$$

$$IG(y_2) = H(z) - H(z|y_2) = 1 - 0 = 1$$

$$IG(y_4) = H(z) - H(z|y_4) = 1 - \frac{1}{2} = \frac{1}{2}$$

Como neste caso IG(y_2) > IG(y_4), a variável colocada no nó seguinte é y_2, dado que tem o maior ganho de informação entre as duas opções. Uma vez que não existem mais de 4 observações para nenhum dos três casos ($y_2 = 0$, $y_2 = 1$ e $y_2 = 2$), é impossível de dividir qualquer um dos nós em mais ramos, pelo que a árvore termina aqui. A árvore de decisão final obtida foi a seguinte:



2. A partir da tabela de dados do enunciado, obtém-se o vetor de output real:

$$z = [\mathrm{A\ B\ B\ C\ C\ A\ A\ A\ B\ B\ C\ C}]$$

Seguindo a árvore de decisão encontrada na alínea anterior, obtém-se o vetor de output previsto:

$$\hat{z} = [\mathrm{A\ B\ C\ C\ C\ A\ A\ A\ A\ B\ C\ C}]$$

Tendo os vetores de output previsto e real é possível de construir a matriz de confusão:

		real		
		A	B	C
previsto	A	4	1	0
	B	0	2	0
	C	0	1	4

3. A medida de F-measure ou F_β é dada pela expressão

$$F_{\beta_i} = \frac{1}{\alpha\frac{1}{P_i} + (1-\alpha)\frac{1}{R_i}} \tag{4}$$

em que $\alpha = \frac{1}{1+\beta^2}$, P_i é a precisão e R_i é a sensibilidade, para a classe $i \in \{A, B, C\}$.

O valor de F1 é calculado através da F-measure com $\alpha = \frac{1}{2}$

Em primeiro lugar, fizemos os cálculos para A. A precisão é dada pelo quociente entre os valores que foram previstos como A e são realmente A, e a soma destes últimos com os valores que tinham sido previstos como A e na realidade são B ou C.

$$P(A) = \frac{4}{4+1} = \frac{4}{5}$$

A sensibilidade é dada pelo quociente entre os valores previstos como A que são realmente A, e a soma destes últimos com os valores que são realmente A e foram previstos como B ou C.

$$R(A) = \frac{4}{4+0} = 1$$

Assim, o valor de F1 obtido para A foi:

$$F1(A) = \frac{1}{\frac{1}{2}\cdot\frac{5}{4} + \frac{1}{2}\cdot 1} = \frac{8}{9}$$

De forma análoga efetuámos os cálculos de F1 para B e C, obtendo os seguintes resultados.

$$P(B) = \frac{2}{2+0} = 1 \qquad R(B) = \frac{2}{4} = \frac{1}{2} \qquad F1(B) = \frac{2}{3}$$

$$P(C) = \frac{4}{4+1} = \frac{4}{5} \qquad R(C) = \frac{4}{4+0} = 1 \qquad F1(C) = \frac{8}{9}$$

Comparando os valores de F1 das três classes, concluímos que a que tem o menor valor é B.

4. Em primeiro lugar, classificámos as variáveis y_1 e y_2 de acordo com o seu rank de Spearman.

y_1	rank y_1	y_2	rank y_2
0.24	3	1	8
0.06	2	2	11
0.04	1	0	3.5
0.36	5	0	3.5
0.32	4	0	3.5
0.68	10	2	11
0.9	12	0	3.5
0.76	11	2	11
0.46	7	1	8
0.62	9	0	3.5
0.44	6	1	8
0.52	8	0	3.5

Aplicando o coeficiente de correlação de Pearson a rank y_1 e rank y_2, pela expressão (5) obtém-se o valor de correlação de $r_S = 0.0797$.

$$r_S = \frac{cov(y_1, y_2)}{\sqrt{var(y_1)}\sqrt{var(y2)}} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(y_{1i} - \overline{y_1})(y_{2i} - \overline{y_2})}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_{1i} - \overline{y_1})} \cdot \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_{2i} - \overline{y_2})}} \tag{5}$$

De notar que, embora na fórmula acima seja utilizada y_1 e y_2, nos cálculos efetuados foi utilizado os valores de rank para y_1 e y_2.

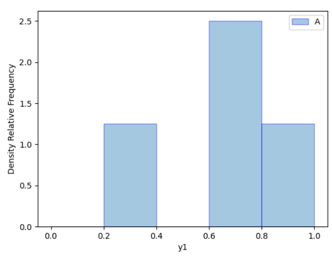
Como o coeficiente de correlação entre y_1 e y_2 (r_S) é muito próximo de zero, concluimos que as variáveis não estão correlacionadas.

5. Para construir os histogramas de y_1 condicionados às classes, com 5 bins igualmente espaçados entre [0, 1], é necessário construir um histograma para cada classe do output, em que os limites dos bins são 0, 0.2, 0.4, 0.6, 0.8, 1. De seguida, para cada um dos histogramas contamos o número de ocorrências em cada bin e construímos o histograma de frequência absoluta, em função do valor (intervalo de valores), do y_1. Por fim, para transformar os histogramas de frequência absoluta em frequência relativa é necessário "normalizar"os valores de frequência de cada bin (altura de cada bin) com a expressão

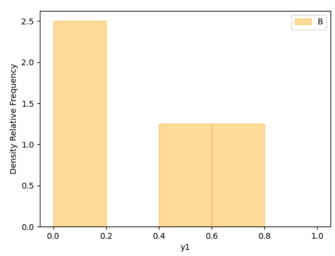
$$h = \frac{C}{N \cdot l}$$

com C o nº de contagens em cada bin, N o nº de contagens totais e l a largura do bin.

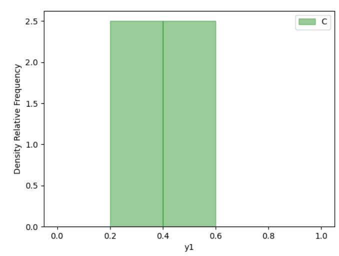
Os histogramas finais obtidos são:



((a)) Histograma y_1 condicionado à Classe A



((b)) Histograma y_1 condicionado à Classe B



((c)) Histograma y_1 condicionado à Classe C

Para encontrar as "root split"começou-se por sobrepor os três histogramas de y_1 condicionados às 3 classes de y_{output} (A, B e C).

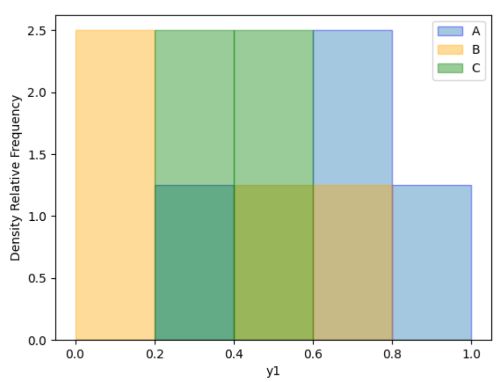


Figura 2: Histogramas de y_1 condicionados às 3 classes.

Observando os 3 histogramas sobrepostos (figura 2), é possível encontrar as "root split", pontos estes que separam os locais em que a classe condicional predominante muda. Desta forma, para:

$$
\begin{aligned}
y_1 < 0.2 &\implies B \\
0.2 \leq y_1 < 0.6 &\implies C \\
y_1 \geq 0.6 &\implies A
\end{aligned}
$$

Part II: Programming

1. As variáveis com maior e menor F-statistic são *degree_spondylolisthesis* e *pelvic_radius*, respetivamente. Estas variáveis são, também, as que têm menor e maior p-value, respetivamente. Assim, a variável com o maior poder discriminativo é *degree_spondylolisthesis* e a variável com o menor poder discriminativo é *pelvic_radius*.

   As classes do DataSet são Hernia, Spondylolisthesis e Normal. É assim necessário desenhar os gráficos das funções de densidade de probabilidade das variáveis anteriormente ditas condicionadas às classes. Estas funções apresentam-se nos seguintes gráficos:

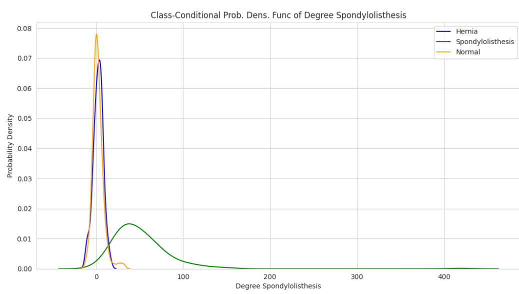


Figura 3: Função de densidade de probabilidade do *Degree Spondylolisthesis*

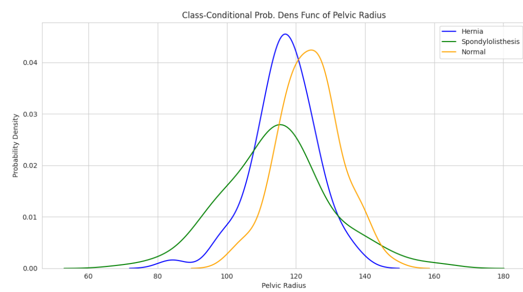


Figura 4: Função de densidade de probabilidade do *Pelvic Radius*

```python
import pandas as pd
from scipy.io.arff import loadarff
from sklearn.feature_selection import f_classif
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy import stats
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree

# Reading the file
data = loadarff('column_diagnosis.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')

X = df.drop('class', axis=1) #variables
y = df['class'] #target

fimportance = f_classif(X, y)

fstat = fimportance[0]  #f-statistic
pval = fimportance[1]   #p-value

print('features', X.columns.values)
print('scores', fimportance[0])
print('pvalues', fimportance[1])

```

```
# Create a dataframe with the F-statistic and p-value for each variable
fstat_df = pd.DataFrame({'F-statistic': fstat, 'p-value': pval})
fstat_df.index = X.columns
fstat_df = fstat_df.sort_values(by=['F-statistic'], ascending=False)
print(fstat_df)

plt.figure(figsize=(10, 6))
sns.distplot(df[df['class'] == 'Hernia']['pelvic_radius'], label='Hernia'
    , hist=False)
sns.distplot(df[df['class'] == 'Normal']['pelvic_radius'], label='Normal'
    , hist=False)
sns.distplot(df[df['class'] == 'Spondylolisthesis']['pelvic_radius'],
    label='Spondylolisthesis', hist=False)
plt.xlabel('pelvic_radius')
plt.ylabel('class-conditional probability density')
plt.title('Class-conditional probability density of pelvic_radius')
plt.legend()
plt.show()


plt.figure(figsize=(10, 6))
sns.distplot(df[df['class'] == 'Hernia']['degree_spondylolisthesis'],
    label='Hernia', hist=False)
sns.distplot(df[df['class'] == 'Normal']['degree_spondylolisthesis'],
    label='Normal', hist=False)
sns.distplot(df[df['class'] == 'Spondylolisthesis']['
    degree_spondylolisthesis'], label='Spondylolisthesis', hist=False)
plt.xlabel('degree_spondylolisthesis')
plt.ylabel('class-conditional probability density')
plt.title('Class-conditional probability density of
    degree_spondylolisthesis')
plt.legend()
plt.show()
```

2. Primeiro fizemos o split treino/teste com 70% para treino e 30% para teste. Queremos obter a accuracy para árvores com profundidade limite de 1 a 10, pelo que utilizámos um ciclo criando uma árvore de decisão em cada iteração, com a profundidade limite correspondente.

   Inicialmente treinámos a árvore com os dados de treino, obtendo os valores previstos para as target variables. Utilizando o output previsto e real de treino e de teste, calculámos as Accuracies para cada profundidade limite. Por fim, desenhamos um único gráfico com a Accuracy de treino e teste para cada profundidade limite.

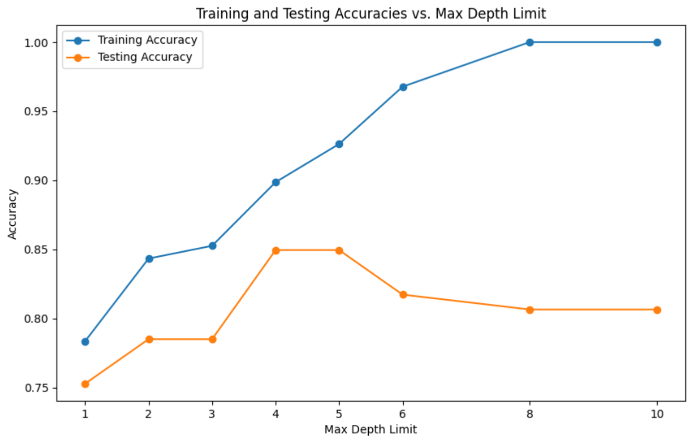


Figura 5: Gráfico da Accuracy de treino e teste em função da Depth Limit.

```python
random_seed = 0

train_accuracies = []
test_accuracies = []

depth_limits = [1, 2, 3, 4, 5, 6, 8, 10]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
    random_state=random_seed, stratify=y)

for depth_limit in depth_limits:
    clf = DecisionTreeClassifier(max_depth=depth_limit, random_state=
    random_seed)
    clf.fit(X_train, y_train)

    y_pred_test = clf.predict(X_test)
    y_pred_train = clf.predict(X_train)

    train_accuracy = metrics.accuracy_score(y_pred_train, y_train)
    train_accuracies.append(train_accuracy)

    test_accuracy = metrics.accuracy_score(y_pred_test, y_test)
    test_accuracies.append(test_accuracy)

plt.figure(figsize=(10, 6))
plt.plot(depth_limits, train_accuracies, marker='o', label='Training
    Accuracy')
plt.plot(depth_limits, test_accuracies, marker='o', label='Testing
    Accuracy')
plt.xlabel('Max Depth Limit')
plt.ylabel('Accuracy')
plt.title('Training and Testing Accuracies vs. Max Depth Limit')
plt.xticks(depth_limits)
plt.legend()
plt.show()
```

3. Observando o gráfico da figura 5, verifica-se um crescimento da accuracy das observações de treino à medida que o limite de profundidade aumenta. A accuracy de treino atinge o valor de 1 para valores de limite de profundidade de 8 a 10. Contudo, para as observações de teste é visível um crescimento da accuracy até aos valores de depth limit intermédios (4 e 5), havendo um decréscimo progressivo desta após estes valores.

   Para valores de depth limit pequenos(< 4), é expectável que a accuracy tanto das observações de treino como das de teste aumente, dado que uma maior profundidade da árvore implica a abrangência de um maior número de cenários possíveis. Contudo, para os valores de depth limit superiores (>5), a accuracy do treino continua a aumentar, enquanto a accuracy de teste diminui. Isto acontece devido ao facto da árvore entrar em overfitting, dado que esta se adapta muito aos dados treinados, perdendo a capacidade de generalização para eventuais acontecimentos que não sejam cobertos pelo treino.

4. (a) Fizémos um plot de uma árvore com um mínimo de 20 indivíduos por folha, para evitar overfitting. Treinámos a árvore utilizando todos os dados como treino. De seguida, fizémos um plot da árvore.

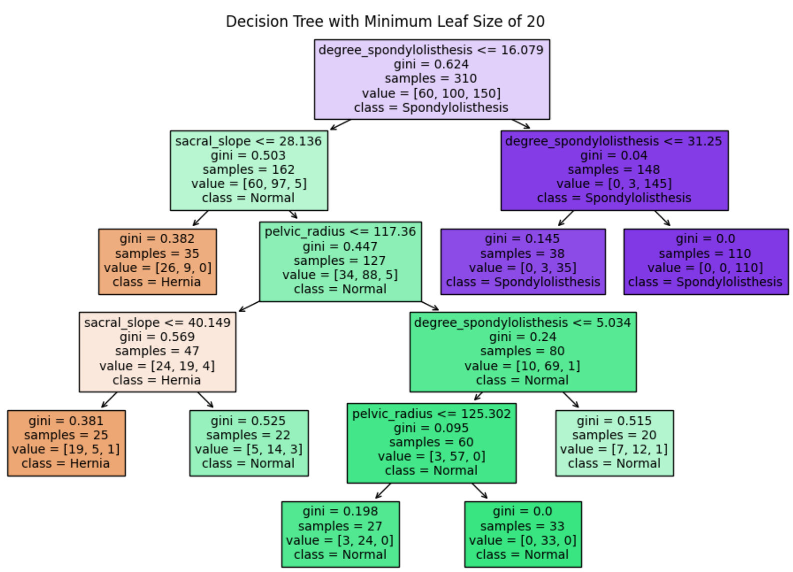


Figura 6: Árvore de Decisão

```
    #create a decision tree
    clf = DecisionTreeClassifier(min_samples_leaf=20,
random_state=random_seed)
    clf.fit(X, y)

```

```
        plt.figure(figsize=(12, 8))
        plot_tree(clf, filled=True, feature_names=df.columns,
    class_names=np.unique(y).astype(str), fontsize=10)
        plt.title("Decision Tree with Minimum Leaf Size of 20")
        plt.show()

```

(b) Dos indivíduos observados (pacientes) há dois grupos que foram diagnosticados com a classe Hernia, pelo que, para caracterizar as condições de ter Hernia, é necessário identificar as associações das diversas features consideradas à classe final (Hernia), para cada um dos grupos.

Para o primeiro grupo, a condição de ser diagnosticado com Hernia é ter a feature *degree_spondylolisthesis* menor ou igual que 16.079 e, de seguida, a feature *sacral_slope* menor ou igual que 28.136. Para o segundo grupo ser diagnosticado com Hernia, embora também seja necessário que a feature *degree_spondylolisthesis* seja menor ou igual a 16.079, a feature *sacral_slope* tem de ser, neste caso, maior que 28.136. Quando o *sacral_slope* é maior que 28.136 a árvore direciona para a feature *pelvic_radius*. Para a classe final ser Hernia, é necessário que esta feature seja menor ou igual que 117.36 e, finalmente, é ainda necessário que a feature *sacral_slope* seja menor ou igual que 40.149.