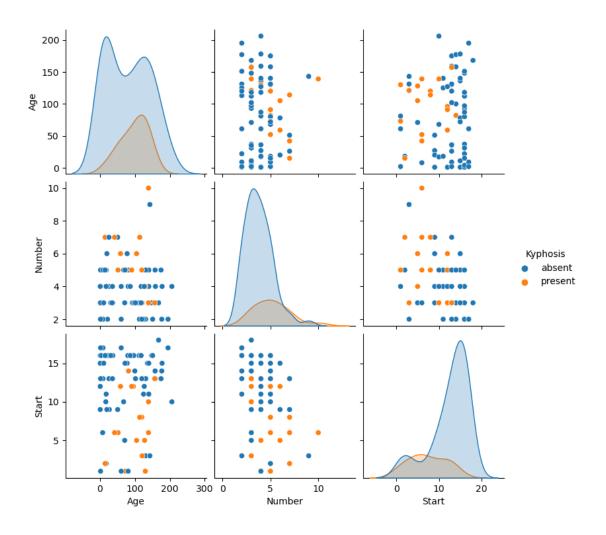
Arvore e Floresta

April 29, 2025

1 Árvores de decisão e florestas aleatórias em Python

```
[2]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
     %matplotlib inline
[3]: df=pd.read_csv('kyphosis.csv')
[4]: df.head()
[4]:
       Kyphosis
                 Age
                      Number
                              Start
     0
         absent
                  71
                           3
                                  5
         absent
                158
                           3
                                  14
     1
                128
                           4
                                  5
     2 present
                   2
     3
         absent
                           5
                                  1
         absent
                           4
                                  15
                   1
[5]: df.info()
    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 81 entries, 0 to 80
    Data columns (total 4 columns):
         Column
                   Non-Null Count Dtype
     0
         Kyphosis 81 non-null
                                    object
                                    int64
     1
         Age
                   81 non-null
     2
                   81 non-null
         Number
                                    int64
         Start
                   81 non-null
                                    int64
    dtypes: int64(3), object(1)
    memory usage: 2.7+ KB
[6]: sns.pairplot(df, hue='Kyphosis')
```

[6]: <seaborn.axisgrid.PairGrid at 0x7fda7d3f0750>



2 Divisão do dataset em treino e teste

```
[7]: from sklearn.model_selection import train_test_split
     df.head()
 [8]:
 [8]:
        Kyphosis
                   Age
                        Number
                                 Start
          absent
                              3
      0
                    71
      1
          absent
                   158
                              3
                                    14
      2
         present
                   128
                              4
                                     5
      3
          absent
                     2
                              5
                                     1
          absent
                     1
                                    15
 [9]: x=df.drop('Kyphosis', axis=1)
[10]: y=df['Kyphosis']
```

```
[11]: X_train, X_test, y_train, y_test = train_test_split(x, y,
                                                           test_size=0.3,
                                                           random_state=42)
[21]: X_train.head()
[21]:
          Age Number
                      Start
      62
                    4
          81
                           1
      42
          143
                    9
                           3
         140
                          15
      54
                    4
      16
          78
                    6
                          15
      39
                    5
           91
                          12
 [1]: from sklearn.tree import DecisionTreeClassifier
[12]: dtree= DecisionTreeClassifier()
[13]: dtree.fit(X_train, y_train)
[13]: DecisionTreeClassifier()
[14]: predicao=dtree.predict(X_test)
[15]: from sklearn.metrics import classification_report, confusion_matrix
[16]: print(classification_report(y_test, predicao))
                   precision
                                recall f1-score
                                                    support
                        0.76
                                   0.84
                                             0.80
           absent
                                                         19
                                   0.17
          present
                        0.25
                                             0.20
                                                          6
         accuracy
                                             0.68
                                                         25
                                  0.50
                                             0.50
        macro avg
                        0.51
                                                         25
     weighted avg
                        0.64
                                   0.68
                                             0.66
                                                         25
[17]: print(confusion_matrix(y_test, predicao))
     [[16 3]
      [5 1]]
[19]: from IPython.display import Image
      #from sklearn.externals.six import StringIO
      from io import StringIO
      from sklearn.tree import export_graphviz
      import pydot
      features = list(df.columns[1:])
```

features [19]: ['Age', 'Number', 'Start'] [20]: dot_data=StringIO() export_graphviz(dtree, out_file=dot_data, feature_names= features, filled=True,_u →rounded=True) graph=pydot.graph_from_dot_data(dot_data.getvalue()) Image(graph[0].create_png()) [20]: Start <= 8.5 gini = 0.316 samples = 56 value = [45, 11] False True Start <= 4.0 gini = 0.48 samples = 15 value = [6, 9] Age <= 89.0 gini = 0.093 samples = 41 value = [39, 2] Age <= 130.5 gini = 0.469 samples = 8 value = [5, 3] Number <= 3.5 gini = 0.245 samples = 7 value = [1, 6] Age <= 94.0 gini = 0.188 samples = 19 value = [17, 2] samples = 22 value = [22, 0]Start <= 1.5 gini = 0.5 samples = 6 Start <= 13.5 gini = 0.105 samples = 18 value = [17, 1] gini = 0.0 samples = 6 value = [0, 6] gini = 0.0 samples = 2 value = [2, 0] gini = 0.0 samples = 1 value = [1, 0] gini = 0.0 samples = 1 value = [0, 1] value = [3, 3] Age <= 143.5 gini = 0.278 samples = 6 value = [5, 1] gini = 0.375 samples = 4 value = [3, 1] samples = 2 value = [0, 2] samples = 12 value = [12, 0]

3 Floresta Aleatória

gini = 0.0 samples = 1 value = [0, 1]

gini = 0.0 samples = 3 value = [3, 0]

```
[21]: from sklearn.ensemble import RandomForestClassifier
    rfc=RandomForestClassifier(n_estimators=100)
    rfc.fit(X_train,y_train)

[21]: RandomForestClassifier()

[22]: rfc_pred=rfc.predict(X_test)

[23]: print(confusion_matrix(y_test, rfc_pred))
```

Age <= 166.0 gini = 0.5 samples = 2

value = [1, 1]

gini = 0.0 samples = 1 value = [1, 0]

gini = 0.0 samples = 4 value = [4, 0]

```
[[19 0]
[5 1]]
```

[24]:	<pre>print(classification_report(y_test,</pre>	rfc_pred))

	precision	recall	f1-score	support
absent	0.79	1.00	0.88	19
present	1.00	0.17	0.29	6
accuracy			0.80	25
macro avg	0.90	0.58	0.58	25
weighted avg	0.84	0.80	0.74	25

Bagging model

```
[49]: from sklearn.ensemble import RandomForestClassifier
      from sklearn.model_selection import cross_val_score
      # Define the model
      random_forest_model = RandomForestClassifier()
      # Fit the random search object to the data
      random_forest_model.fit(X_train, y_train)
```

[49]: RandomForestClassifier()

```
[50]: # Make predictions
      y_pred = random_forest_model.predict(X_test)
```

[51]: print(confusion_matrix(y_test, y_pred))

[[19 0] [5 1]]

[31]: print(classification_report(y_test, y_pred))

	precision	recall	f1-score	support
absent	0.83	1.00	0.91	20
present	1.00	0.20	0.33	5
accuracy			0.84	25
macro avg	0.92	0.60	0.62	25
weighted avg	0.87	0.84	0.79	25

[]: