Untitled2

March 28, 2020

[1]:
```
import os
os.environ['PYSPARK_PYTHON'] = '/usr/bin/python3'

from pyspark.sql import SparkSession

sc = SparkSession \
    .builder \
    .master('spark://172.18.0.10:7077') \
    .config('spark.executor.memory', '512mb') \
    .getOrCreate()
```

[2]:
```
dfTreinamento = sc.read \
    .option('delimiter', ',') \
    .option('header', 'true') \
    .option('inferschema', 'true') \
    .csv('hdfs://172.18.0.12:9000/treinamento.csv')
```

[3]:
```
dfTreinamento.printSchema()
```

```
root
 |-- hora: double (nullable = true)
 |-- minuto: double (nullable = true)
 |-- temp_minima: double (nullable = true)
 |-- temp_maxima: double (nullable = true)
 |-- latitude_media: double (nullable = true)
 |-- longitude_media: double (nullable = true)
 |-- Classe: string (nullable = true)
```

[5]:
```
from pyspark.ml.feature import StringIndexer, VectorAssembler

datasetTreinamento = StringIndexer(inputCol='Classe', outputCol='label')\
    .fit(dfTreinamento).transform(dfTreinamento)

features = ['hora', 'minuto', 'temp_minima', 'temp_maxima',\
            'latitude_media', 'longitude_media']

datasetTreinamento = VectorAssembler(inputCols=features, outputCol='features')\
    .transform(datasetTreinamento)

datasetTreinamento.printSchema()
```

```
root
 |-- hora: double (nullable = true)
 |-- minuto: double (nullable = true)
 |-- temp_minima: double (nullable = true)
 |-- temp_maxima: double (nullable = true)
 |-- latitude_media: double (nullable = true)
 |-- longitude_media: double (nullable = true)
 |-- Classe: string (nullable = true)
 |-- label: double (nullable = false)
 |-- features: vector (nullable = true)
```

[6]:
```
datasetTreinamento = datasetTreinamento.select('label', 'features')
datasetTreinamento.printSchema()
```

```
root
 |-- label: double (nullable = false)
 |-- features: vector (nullable = true)
```

[7]:
```
datasetTreinamento.take(5)
```

[7]:
```
[Row(label=0.0, features=DenseVector([11.3128, 30.1692, -1.859, 27.495, 36.17,
139.2302])),
 Row(label=3.0, features=DenseVector([11.2923, 29.6388, 8.543, 36.178, 31.3491,
73.5096])),
 Row(label=0.0, features=DenseVector([11.6003, 29.6428, -1.861, 27.695, 36.1716,
139.2294])),
 Row(label=3.0, features=DenseVector([11.4624, 30.1573, 9.777, 36.078, 31.3516,
73.5104])),
 Row(label=0.0, features=DenseVector([11.7355, 29.9684, -1.662, 27.695, 36.1695,
139.2302]))]
```

[8]:
```
(treinamento, test) = datasetTreinamento.randomSplit([0.7, 0.3])
```

[9]:
```
from pyspark.ml.classification import DecisionTreeClassifier

arvore = DecisionTreeClassifier(labelCol='label', featuresCol='features')
modeloArvore = arvore.fit(treinamento)
```

[10]:
```
modeloArvore.toDebugString
```

[10]:
```
'DecisionTreeClassificationModel (uid=DecisionTreeClassifier_00841bfcaac6) of
depth 5 with 47 nodes\n  If (feature 2 <= 7.6989985)\n   If (feature 2 <=
-5.145999)\n    If (feature 4 <= 39.379509)\n     If (feature 2 <= -9.2125015)\n
If (feature 3 <= 25.8695015)\n       Predict: 2.0\n      Else (feature 3 >
25.8695015)\n       Predict: 0.0\n     Else (feature 2 > -9.2125015)\n      If
(feature 3 <= 24.5055005)\n       Predict: 2.0\n      Else (feature 3 >
24.5055005)\n       Predict: 0.0\n    Else (feature 4 > 39.379509)\n     If
(feature 3 <= 27.8375035)\n      Predict: 2.0\n     Else (feature 3 >
27.8375035)\n      If (feature 4 <= 44.2012625)\n       Predict: 0.0\n      Else
(feature 4 > 44.2012625)\n       Predict: 2.0\n   Else (feature 2 > -5.145999)\n
If (feature 3 <= 32.618998000000005)\n     If (feature 4 <= 50.6295055)\n
Predict: 0.0\n     Else (feature 4 > 50.6295055)\n      If (feature 2 <=
-3.3505000000000003)\n       Predict: 2.0\n      Else (feature 2 >
-3.3505000000000003)\n       Predict: 0.0\n    Else (feature 3 >
32.618998000000005)\n     If (feature 4 <= 34.5599785)\n      Predict: 3.0\n
Else (feature 4 > 34.5599785)\n      Predict: 0.0\n  Else (feature 2 >
7.6989985)\n   If (feature 2 <= 12.522498500000001)\n    If (feature 3 <=
36.0694815)\n     If (feature 5 <= 30.450263)\n      If (feature 4 <=
31.350878)\n       Predict: 3.0\n      Else (feature 4 > 31.350878)\n
Predict: 0.0\n     Else (feature 5 > 30.450263)\n      Predict: 3.0\n    Else
(feature 3 > 36.0694815)\n     If (feature 2 <= 10.1969955)\n      If (feature 5
<= 72.52138)\n       Predict: 1.0\n      Else (feature 5 > 72.52138)\n
Predict: 3.0\n     Else (feature 2 > 10.1969955)\n      Predict: 1.0\n   Else
(feature 2 > 12.522498500000001)\n    If (feature 3 <= 28.3340035)\n     If
(feature 3 <= 23.8875015)\n      Predict: 0.0\n     Else (feature 3 >
23.8875015)\n      If (feature 2 <= 22.598998)\n       Predict: 3.0\n      Else
(feature 2 > 22.598998)\n       Predict: 1.0\n    Else (feature 3 >
28.3340035)\n     If (feature 2 <= 20.3584985)\n      If (feature 3 <=
30.687501)\n       Predict: 3.0\n      Else (feature 3 > 30.687501)\n
Predict: 1.0\n     Else (feature 2 > 20.3584985)\n      Predict: 1.0\n'
```

[11]:
```
test.printSchema()
```

```
root
 |-- label: double (nullable = false)
 |-- features: vector (nullable = true)
```

[12]:
```
resultadoTest = modeloArvore.transform(test)
```

[13]:
```
resultadoTest.printSchema()
```

```
root
 |-- label: double (nullable = false)
 |-- features: vector (nullable = true)
 |-- rawPrediction: vector (nullable = true)
 |-- probability: vector (nullable = true)
 |-- prediction: double (nullable = false)
```

```
[14]: resultadoTest.take(5)
```

```
[14]: [Row(label=0.0, features=DenseVector([10.9795, 29.4047, 0.405, 32.699, 28.1306,
      115.4512]), rawPrediction=DenseVector([29.0, 0.0, 0.0, 101.0]),
      probability=DenseVector([0.2231, 0.0, 0.0, 0.7769]), prediction=3.0),
       Row(label=0.0, features=DenseVector([11.0114, 29.7909, -1.78, 27.495, 36.1698,
      139.2295]), rawPrediction=DenseVector([7343.0, 0.0, 21.0, 9.0]),
      probability=DenseVector([0.9959, 0.0, 0.0028, 0.0012]), prediction=0.0),
       Row(label=0.0, features=DenseVector([11.0172, 29.8351, 6.516, 26.486, 29.7407,
      79.3792]), rawPrediction=DenseVector([7343.0, 0.0, 21.0, 9.0]),
      probability=DenseVector([0.9959, 0.0, 0.0028, 0.0012]), prediction=0.0),
       Row(label=0.0, features=DenseVector([11.0191, 29.9951, 16.809, 23.256, 7.2302,
      36.49]), rawPrediction=DenseVector([71.0, 0.0, 0.0, 17.0]),
      probability=DenseVector([0.8068, 0.0, 0.0, 0.1932]), prediction=0.0),
       Row(label=0.0, features=DenseVector([11.0526, 29.4198, -1.763, 27.395, 36.1701,
      139.2298]), rawPrediction=DenseVector([7343.0, 0.0, 21.0, 9.0]),
      probability=DenseVector([0.9959, 0.0, 0.0028, 0.0012]), prediction=0.0)]
```

```
[16]: from pyspark.ml.evaluation import MulticlassClassificationEvaluator

      evaluator = MulticlassClassificationEvaluator(labelCol='label',\
                                                    predictionCol='prediction',\
                                                     metricName='accuracy')
      acc = evaluator.evaluate(resultadoTest)
      acc
```

```
[16]: 0.94656150954522
```

```
[ ]:
```