databricksDecision Tree

```
path = "/FileStore/tables/"
firis = path + "iris.csv"
iris = spark.read.csv( firis, header=True, inferSchema=True )
iris.describe().show()
```

```
+-------+------------------+-------------------+------------------+---------------
--+---------+
|summary|      Sepal.Length|        Sepal.Width|      Petal.Length|       Petal.Wid
th|  Species|
+-------+------------------+-------------------+------------------+---------------
--+---------+
|  count|               150|                150|               150|              1
50|      150|
|   mean| 5.843333333333335|  3.057333333333334|3.7580000000000027| 1.1993333333333
34|     null|
| stddev|0.8280661279778637|0.43586628493669793|1.7652982332594662|0.76223766896034
67|     null|
|    min|               4.3|                2.0|               1.0|              0
.1|   setosa|
|    max|               7.9|                4.4|               6.9|              2
.5|virginica|
+-------+------------------+-------------------+------------------+---------------
--+---------+
```

```
# Labels
iris.show(2, False)
iris.groupby("Species").count().show()
```

```
+---------+--------+---------+--------+-------+
|sepLength|sepWidth|petLength|petWidth|Species|
+---------+--------+---------+--------+-------+
|5.1      |3.5     |1.4      |0.2     |setosa |
|4.9      |3.0     |1.4      |0.2     |setosa |
+---------+--------+---------+--------+-------+
only showing top 2 rows
```

```
+----------+-----+
|   Species|count|
+----------+-----+
| virginica|   50|
|versicolor|   50|
|    setosa|   50|
+----------+-----+
```

```
iris = iris.withColumnRenamed("Sepal.Length", "sepLength")\
           .withColumnRenamed("Sepal.Width", "sepWidth")\
           .withColumnRenamed("Petal.Length", "petLength")\
           .withColumnRenamed("Petal.Width", "petWidth")


# Assigning the numerical labels (Transformer)
from pyspark.ml.feature import StringIndexer

stindexer1 = StringIndexer( inputCol="Species", outputCol="SpeciesInd" )
iris2 = stindexer1.fit( iris ).transform( iris )



# feature correlation with label
from pyspark.sql import functions as fun

cols = iris2.drop("Species", "SpeciesInd").columns
exprs = [ fun.corr( x, "SpeciesInd" ) for x in cols ]
iris2.agg( *exprs ).show()


+--------------------------+-------------------------+--------------------------
+--------------------------+
|corr(sepLength, SpeciesInd)|corr(sepWidth, SpeciesInd)|corr(petLength, SpeciesInd)
|corr(petWidth, SpeciesInd)|
+--------------------------+-------------------------+--------------------------
+--------------------------+
|       -0.46003915650023736|        0.6183715308237437|        -0.6492418307641745
|       -0.5803770334306263|
+--------------------------+-------------------------+--------------------------
+--------------------------+


# combining the features in a vector
from pyspark.ml.feature import VectorAssembler
vecassem1 = VectorAssembler( inputCols=[ "sepLength", "sepWidth", "petLength",
"petWidth" ], outputCol="features" )
iris3 = vecassem1.transform( iris2 )
iris3.show(3, False)

+---------+--------+---------+--------+-------+----------+-----------------+
|sepLength|sepWidth|petLength|petWidth|Species|SpeciesInd|features         |
+---------+--------+---------+--------+-------+----------+-----------------+
|5.1      |3.5     |1.4      |0.2     |setosa |2.0       |[5.1,3.5,1.4,0.2]|
|4.9      |3.0     |1.4      |0.2     |setosa |2.0       |[4.9,3.0,1.4,0.2]|
```

```
|4.7      |3.2     |1.3      |0.2     |setosa |2.0       |[4.7,3.2,1.3,0.2]|
+---------+--------+---------+--------+-------+----------+-----------------+
only showing top 3 rows


# Fitting
from pyspark.ml.classification import DecisionTreeClassifier
dtc1 = DecisionTreeClassifier( featuresCol="features", labelCol="SpeciesInd",
predictionCol="pred_species", maxDepth=2 )
iris_train, iris_test = iris3.randomSplit( [.7, .3] )
dtc1Fit = dtc1.fit( iris_train )
iris_predict1 = dtc1Fit.transform( iris_test )
iris_predict1.drop("features", "rawPrediction").show(10, False)

+---------+--------+---------+--------+----------+----------+----------------------
----------------------+------------+
|sepLength|sepWidth|petLength|petWidth|Species   |SpeciesInd|probability
|pred_species|
+---------+--------+---------+--------+----------+----------+----------------------
----------------------+------------+
|4.8      |3.0     |1.4      |0.1     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|4.8      |3.1     |1.6      |0.2     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|4.8      |3.4     |1.6      |0.2     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|4.9      |2.4     |3.3      |1.0     |versicolor|0.0       |[0.9210526315789473,0.
07894736842105263,0.0]|0.0         |
|4.9      |3.1     |1.5      |0.1     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|5.0      |3.3     |1.4      |0.2     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|5.1      |2.5     |3.0      |1.1     |versicolor|0.0       |[0.9210526315789473,0.
07894736842105263,0.0]|0.0         |
|5.1      |3.8     |1.5      |0.3     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
|5.2      |2.7     |3.9      |1.4     |versicolor|0.0       |[0.9210526315789473,0.
07894736842105263,0.0]|0.0         |
|5.2      |3.5     |1.5      |0.2     |setosa    |2.0       |[0.0,0.0,1.0]
|2.0         |
+---------+--------+---------+--------+----------+----------+----------------------
----------------------+------------+
only showing top 10 rows
```

```
# Evaluation
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
eval1 = MulticlassClassificationEvaluator(predictionCol="pred_species",
labelCol="SpeciesInd",metricName="accuracy")
print( eval1.evaluate(iris_predict1) )

iris_predict1.groupBy("SpeciesInd","pred_species").count().show()
dtc1Fit.transform( iris3 ).groupBy("SpeciesInd","pred_species").count().show()

0.9210526315789473
+----------+------------+-----+
|SpeciesInd|pred_species|count|
+----------+------------+-----+
|       1.0|         1.0|   10|
|       0.0|         1.0|    1|
|       2.0|         2.0|   11|
|       1.0|         0.0|    2|
|       0.0|         0.0|   14|
+----------+------------+-----+

+----------+------------+-----+
|SpeciesInd|pred_species|count|
+----------+------------+-----+
|       1.0|         1.0|   45|
|       0.0|         1.0|    1|
|       2.0|         2.0|   50|
|       1.0|         0.0|    5|
|       0.0|         0.0|   49|
+----------+------------+-----+


# Feature importance
dtc1Fit.featureImportances

Out[60]: SparseVector(4, {2: 0.5524, 3: 0.4476})

+-----------------------------------------------+-----+
|probability                                    |count|
+-----------------------------------------------+-----+
|[0.0,0.0,1.0]                                  |11   |
|[0.9210526315789473,0.07894736842105263,0.0]|16   |
|[0.0,1.0,0.0]                                  |11   |
+-----------------------------------------------+-----+

+-----------------------------------------------+--------------+-------------+--------
------+-------------+
|probability                                    |min(petLength)|min(petWidth)|max(petL
```

```
ength)|max(petWidth)|
+-----------------------------------------+-------------+-------------+--------
------+-------------+
|[0.0,0.0,1.0]                            |1.2          |0.1          |1.6
|0.3          |
|[0.9210526315789473,0.07894736842105263,0.0]|3.0          |1.0          |5.8
|1.6          |
|[0.0,1.0,0.0]                            |4.8          |1.8          |6.6
|2.4          |
+-----------------------------------------+-------------+-------------+--------
------+-------------+
```