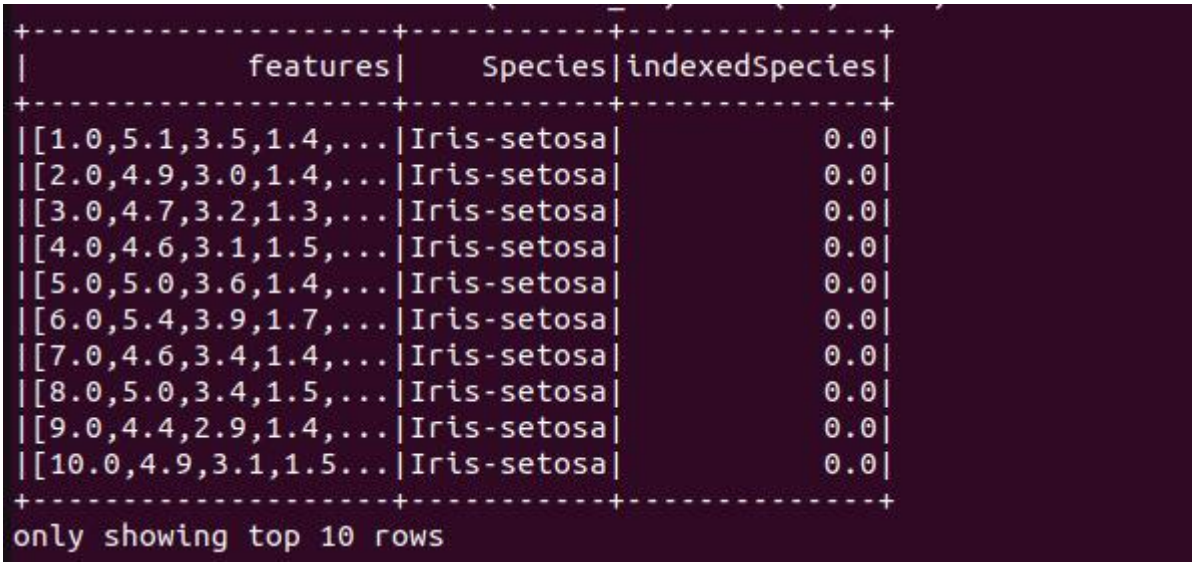
```
+---+-------------+------------+-------------+------------+-----------+
| Id|SepalLengthCm|SepalWidthCm|PetalLengthCm|PetalWidthCm|    Species|
+---+-------------+------------+-------------+------------+-----------+
|  1|          5.1|         3.5|          1.4|         0.2|Iris-setosa|
|  2|          4.9|         3.0|          1.4|         0.2|Iris-setosa|
|  3|          4.7|         3.2|          1.3|         0.2|Iris-setosa|
|  4|          4.6|         3.1|          1.5|         0.2|Iris-setosa|
|  5|          5.0|         3.6|          1.4|         0.2|Iris-setosa|
|  6|          5.4|         3.9|          1.7|         0.4|Iris-setosa|
|  7|          4.6|         3.4|          1.4|         0.3|Iris-setosa|
|  8|          5.0|         3.4|          1.5|         0.2|Iris-setosa|
|  9|          4.4|         2.9|          1.4|         0.2|Iris-setosa|
| 10|          4.9|         3.1|          1.5|         0.1|Iris-setosa|
| 11|          5.4|         3.7|          1.5|         0.2|Iris-setosa|
| 12|          4.8|         3.4|          1.6|         0.2|Iris-setosa|
| 13|          4.8|         3.0|          1.4|         0.1|Iris-setosa|
| 14|          4.3|         3.0|          1.1|         0.1|Iris-setosa|
| 15|          5.8|         4.0|          1.2|         0.2|Iris-setosa|
| 16|          5.7|         4.4|          1.5|         0.4|Iris-setosa|
| 17|          5.4|         3.9|          1.3|         0.4|Iris-setosa|
| 18|          5.1|         3.5|          1.4|         0.3|Iris-setosa|
| 19|          5.7|         3.8|          1.7|         0.3|Iris-setosa|
| 20|          5.1|         3.8|          1.5|         0.3|Iris-setosa|
+---+-------------+------------+-------------+------------+-----------+
only showing top 20 rows

>>> df.groupBy('Species').count().show()
+---------------+-----+
|        Species|count|
+---------------+-----+
| Iris-virginica|   50|
|    Iris-setosa|   50|
|Iris-versicolor|   50|
+---------------+-----+

>>> df.printSchema()
root
 |-- Id: integer (nullable = true)
 |-- SepalLengthCm: double (nullable = true)
 |-- SepalWidthCm: double (nullable = true)
 |-- PetalLengthCm: double (nullable = true)
 |-- PetalWidthCm: double (nullable = true)
 |-- Species: string (nullable = true)
```



```
+--------------------+-----------+--------------+
|            features|    Species|indexedSpecies|
+--------------------+-----------+--------------+
|[1.0,5.1,3.5,1.4,...|Iris-setosa|           0.0|
|[2.0,4.9,3.0,1.4,...|Iris-setosa|           0.0|
|[3.0,4.7,3.2,1.3,...|Iris-setosa|           0.0|
|[4.0,4.6,3.1,1.5,...|Iris-setosa|           0.0|
|[5.0,5.0,3.6,1.4,...|Iris-setosa|           0.0|
|[6.0,5.4,3.9,1.7,...|Iris-setosa|           0.0|
|[7.0,4.6,3.4,1.4,...|Iris-setosa|           0.0|
|[8.0,5.0,3.4,1.5,...|Iris-setosa|           0.0|
|[9.0,4.4,2.9,1.4,...|Iris-setosa|           0.0|
|[10.0,4.9,3.1,1.5...|Iris-setosa|           0.0|
+--------------------+-----------+--------------+
only showing top 10 rows
```

```
>>> tmodel = tpipeline.fit(train_df)
>>> tpredictions = tmodel.transform(test_df)
>>> tpredictions.select("features","Species","predictedLabel").show(5)
+--------------------+-----------+--------------+
|            features|    Species|predictedLabel|
+--------------------+-----------+--------------+
|[6.0,5.4,3.9,1.7,...|Iris-setosa|   Iris-setosa|
|[12.0,4.8,3.4,1.6...|Iris-setosa|   Iris-setosa|
|[14.0,4.3,3.0,1.1...|Iris-setosa|   Iris-setosa|
|[16.0,5.7,4.4,1.5...|Iris-setosa|   Iris-setosa|
|[20.0,5.1,3.8,1.5...|Iris-setosa|   Iris-setosa|
+--------------------+-----------+--------------+
only showing top 5 rows

>>>
>>> rmodel = rpipeline.fit(train_df)
>>> rpredictions = rmodel.transform(test_df)
>>> rpredictions.select("features","Species","predictedLabel").show(5)
+--------------------+-----------+--------------+
|            features|    Species|predictedLabel|
+--------------------+-----------+--------------+
|[6.0,5.4,3.9,1.7,...|Iris-setosa|   Iris-setosa|
|[12.0,4.8,3.4,1.6...|Iris-setosa|   Iris-setosa|
|[14.0,4.3,3.0,1.1...|Iris-setosa|   Iris-setosa|
|[16.0,5.7,4.4,1.5...|Iris-setosa|   Iris-setosa|
|[20.0,5.1,3.8,1.5...|Iris-setosa|   Iris-setosa|
+--------------------+-----------+--------------+
only showing top 5 rows

>>>
>>> nmodel = npipeline.fit(train_df)
>>> npredictions = nmodel.transform(test_df)
>>> tpredictions.select("features","Species","predictedLabel").show(5)
+--------------------+-----------+--------------+
|            features|    Species|predictedLabel|
+--------------------+-----------+--------------+
|[6.0,5.4,3.9,1.7,...|Iris-setosa|   Iris-setosa|
|[12.0,4.8,3.4,1.6...|Iris-setosa|   Iris-setosa|
|[14.0,4.3,3.0,1.1...|Iris-setosa|   Iris-setosa|
|[16.0,5.7,4.4,1.5...|Iris-setosa|   Iris-setosa|
|[20.0,5.1,3.8,1.5...|Iris-setosa|   Iris-setosa|
+--------------------+-----------+--------------+
only showing top 5 rows
```

Training=65% and testing=35%

IF we looked to the Evaluation of each algorithm we will find that Random forest accuracy is the highest accuracy =1, and accuracy of decision tree= naive bayes=0.9811

```
>>> #Evaluate Tree
>>> tevaluator = MulticlassClassificationEvaluator(
...     labelCol="indexedSpecies", predictionCol="prediction", metricName="accuracy")
>>> accuracy = tevaluator.evaluate(tpredictions)
>>> print(accuracy)
0.9811320754716981
>>> print("tTest Error = %g" % (1.0 - accuracy))
tTest Error = 0.0188679
>>> dtModel = tmodel.stages[-2]
>>> print(dtModel)
DecisionTreeClassificationModel: uid=DecisionTreeClassifier_69e29fe00843, depth=3, numNodes=7, numClasses=3, numFeatures=5
>>>
>>>
>>> #evaluate Random forest
>>> revaluator = MulticlassClassificationEvaluator(
...     labelCol="indexedSpecies", predictionCol="prediction", metricName="accuracy")
>>> accuracy = revaluator.evaluate(rpredictions)
>>> print(accuracy)
1.0
>>> print("rTest Error = %g" % (1.0 - accuracy))
rTest Error = 0
>>> rfModel = rmodel.stages[-2]
>>> print(rfModel)
RandomForestClassificationModel: uid=RandomForestClassifier_fb9797819204, numTrees=20, numClasses=3, numFeatures=5
>>>
>>>
>>> #evaluate Naive Bayes
>>> nevaluator = MulticlassClassificationEvaluator(
...     labelCol="indexedSpecies", predictionCol="prediction", metricName="accuracy")
>>> accuracy = nevaluator.evaluate(npredictions)
>>> print(accuracy)
0.9811320754716981
>>> print("nTest Error = %g" % (1.0 - accuracy))
nTest Error = 0.0188679
>>> nbModel = nmodel.stages[-2]
>>> print(nbModel)
DecisionTreeClassificationModel: uid=DecisionTreeClassifier_35bf4c3b0efb, depth=3, numNodes=7, numClasses=3, numFeatures=5
```

When we change number of training and testing samples the accuracy also changes