Project 2 Option-1 Part 4

P4

Spark ML and MLlib provide a rich set of machine learning libraries, e.g. Random Forest Classifier(link) and Decision Tree Classifier (link).

You are expected to redo P3 with Random Forest Classifier and Decision Tree Classifier by using the Apache Spark ML/MLlib.

Decision Tree Classifier

Code

Initialization: To implement a Decision Tree Classifier in Spark, it needs to initialize the session.

```
if __name__ == "__main__":
    spark = SparkSession\
        .builder\
        .appName("AdultCensusDecisionTree")\
        .getOrCreate()
```

Load Data:

There are errors when calling the feature names, so we make sure there are no white space before or after the name. Also, we remove the "." in the features to avoid any unnesscary error.

```
18  #load data
19  dataset = spark.read.format("csv").load("/decision_tree_classification/input/census_clean.csv",
20                                          header="true", inferSchema="true",
21                                          ignoreLeadingWhiteSpace='true',
22                                          ignoreTrailingWhiteSpace='true')
23  #remove whitespace, replace feature name with a '.' to '' and convert to a dataframe
24  data_list = []
25  for col in dataset.columns:
26      new_name = col.strip()
27      new_name = "".join(new_name.split())
28      new_name = new_name.replace('.','')
29      data_list.append(new_name)
30  print(data_list)
31  data = dataset.toDF(*data_list)
32  #data.show()
```

We need to vectorize the features and change binary label to index of 0 and 1.

```
34  #add a label column using feature income
35  index = StringIndexer(inputCol = 'income', outputCol = 'label')
36  data_all =index.fit(data).transform(data)
37
38  #vectorize features
39  assembler = VectorAssembler(inputCols= ['age', 'workclass', 'maritalstatus',
40                                          'educationnum', 'occupation', 'relationship',
41                                          'race', 'sex', 'capitalgain', 'capitalloss',
42                                          'hoursperweek', 'nativecountry'],
43                                          outputCol="features")
44  feature = assembler.transform(data_all)
45  #feature.show()
```

Train the Model:

Before training the model, we need to split the dataset randomly into training and testing set to 80% and 20% accordingly. First fit the model using the training set, then predict it with the testing set. Lastly, print out the testing accuracy and testing error.

```
47      #Split data to train and test set
48      train, test = feature.randomSplit((0.8, 0.2), seed=0)
49
50      # Train a DecisionTree model.
51      decision_tree = DecisionTreeClassifier(labelCol="label", featuresCol="features")
52
53      model = decision_tree.fit(train)
54
55      # Make predictions.
56      predictions = model.transform(test)
57
58      # Select (prediction, true label) and compute test error, accuracy
59      evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
60      accuracy = evaluator.evaluate(predictions)
61      print("Testing accuracy: ", accuracy)
62      print("Test Error = %g " % (1.0 - accuracy))
```

***Find full example code in the zip folder.*

Result

Testing Accuracy: 0.8448, 84.48%

Test Error: 0.1552, 15.52%

```
2021-05-13 08:47:03 INFO  DAGScheduler:54 - ResultStage 21 (countByValue at MulticlassMetrics.scala:42) finished in 0.070 s
2021-05-13 08:47:03 INFO  DAGScheduler:54 - Job 12 finished: countByValue at MulticlassMetrics.scala:42, took 0.736133 s
Testing accuracy:  0.844773142762
Test Error = 0.155227
2021-05-13 08:47:03 INFO  SparkContext:54 - Invoking stop() from shutdown hook
2021-05-13 08:47:03 INFO  AbstractConnector:318 - Stopped Spark@74a13b68{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
2021-05-13 08:47:03 INFO  SparkUI:54 - Stopped Spark web UI at http://node-1.c.noted-field-305223.internal:4040
```

Test.sh

```
root@node-1:/spark-examples/test-mllib/python/decision_tree_classification# cat test.sh
#!/bin/bash
source ../../../env.sh
/usr/local/hadoop/bin/hdfs dfs -rm -r /decision_tree_classification/input/
/usr/local/hadoop/bin/hdfs dfs -mkdir -p /decision_tree_classification/input/
/usr/local/hadoop/bin/hdfs dfs -copyFromLocal ../../../test-data/census_clean.csv /decision_tree_classification/input/
/usr/local/spark/bin/spark-submit --master=spark://$SPARK_MASTER:7077 ./Q4.py
```

Random Forest

Random Forest is performed in the decision_tree_classification path

Code

Initialization: To implement a Random Forest Classifier in Spark, it needs to initialize the session.

```
if __name__ == "__main__":
    spark = SparkSession\
        .builder\
        .appName("AdultCensusRandomForest")\
        .getOrCreate()
```

Load Data:

There are errors when calling the feature names, so we make sure there are no white space before or after the name. Also, we remove the "." in the features to avoid any unnesscary error.

```
18  #load data
19  dataset = spark.read.format("csv").load("/decision_tree_classification/input/census_clean.csv",
20                                          header="true", inferSchema="true",
21                                          ignoreLeadingWhiteSpace='true',
22                                          ignoreTrailingWhiteSpace='true')
23  #remove whitespace, replace feature name with a '.' to '' and convert to a dataframe
24  data_list = []
25  for col in dataset.columns:
26      new_name = col.strip()
27      new_name = "".join(new_name.split())
28      new_name = new_name.replace('.','')
29      data_list.append(new_name)
30  print(data_list)
31  data = dataset.toDF(*data_list)
32  #data.show()
```

We need to vectorize the features and change binary label to index of 0 and 1.

```
34  #add a label column using feature income
35  index = StringIndexer(inputCol = 'income', outputCol = 'label')
36  data_all =index.fit(data).transform(data)
37
38  #vectorize features
39  assembler = VectorAssembler(inputCols= ['age', 'workclass', 'maritalstatus',
40                                          'educationnum', 'occupation', 'relationship',
41                                          'race', 'sex', 'capitalgain', 'capitalloss',
42                                          'hoursperweek', 'nativecountry'],
43                                          outputCol="features")
44  feature = assembler.transform(data_all)
45  #feature.show()
```

Train the Model:

Before training the model, we need to split the dataset randomly into training and testing set to 80% and 20% accordingly. First fit the model using the training set, then predict it with the testing set. Lastly, print out the testing accuracy and testing error.

```
47      #Split data to train and test set
48      train, test = feature.randomSplit((0.8, 0.2), seed=0)
49
50      # Train a DecisionTree model.
51      decision_tree = RandomForestClassifier(labelCol="label", featuresCol="features", numTrees=10)
52
53      model = decision_tree.fit(train)
54
55      # Make predictions.
56      predictions = model.transform(test)
57
58      # Select (prediction, true label) and compute test error, accuracy
59      evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
60      accuracy = evaluator.evaluate(predictions)
61      print("Testing accuracy: ", accuracy)
62      print("Test Error = %g " % (1.0 - accuracy))
63
```

*****Find full example code in the zip folder.*

Result

Testing Accuracy: 0.8391, 84.91%

Test Error: 0.1608, 16.08%

```
2021-05-13 09:50:34 INFO  TaskSchedulerImpl:54 - Removed TaskSet 22.0, whose tasks have all completed, from pool
2021-05-13 09:50:34 INFO  DAGScheduler:54 - ResultStage 22 (countByValue at MulticlassMetrics.scala:42) finished in 0.064 s
2021-05-13 09:50:34 INFO  DAGScheduler:54 - Job 13 finished: countByValue at MulticlassMetrics.scala:42, took 0.529069 s
Testing accuracy:  0.839122486289
Test Error = 0.160878
2021-05-13 09:50:34 INFO  SparkContext:54 - Invoking stop() from shutdown hook
2021-05-13 09:50:34 INFO  AbstractConnector:318 - Stopped Spark@3398195e{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
```

Test.sh

```
#!/bin/bash
source ../../../env.sh
/usr/local/hadoop/bin/hdfs dfs -rm -r /decision_tree_classification/input/
/usr/local/hadoop/bin/hdfs dfs -mkdir -p /decision_tree_classification/input/
/usr/local/hadoop/bin/hdfs dfs -copyFromLocal ../../../test-data/census_clean.csv /decision_tree_classification/input/
/usr/local/spark/bin/spark-submit --master=spark://$SPARK_MASTER:7077 ./Q4_random.py
root@node-1:/spark-examples/test-mllib/python/decision_tree_classification#
```

Summary

	Accuracy
Logistic Regression	81.99%
Decision Tree	84.48%
Random Forest	84.91%

Overall, random forest classifier gives the best accuracy with 84.91%.