

Sayeed Ahmed – SEC01 (NUID 002191535)

Big Data System Engineering with Scala

Spring 2023

Spark Assignment No. 2



Github: <https://github.com/sayeedahmed01/CSYE7200/tree/Spring2022/Spark-Assignments/>

Implementation:

1) Loading datasets and imports:

```
1 //Imports
2 import org.apache.spark.sql.SparkSession
3 import org.apache.spark.sql.functions._
4 import org.apache.spark.ml.classification.RandomForestClassifier
5 import org.apache.spark.ml.feature.{StringIndexer, OneHotEncoder, VectorAssembler}
6 import org.apache.spark.ml.Pipeline
7 import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
import org.apache.spark.ml.classification.RandomForestClassifier
import org.apache.spark.ml.feature.{StringIndexer, OneHotEncoder, VectorAssembler}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

Command took 0.44 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:59:57 AM on My Cluster

Cmd 2

1 val spark = SparkSession.builder().appName("TitanicEDA").getOrCreate()
2 val trainData = spark.read.option("header", "true").option("inferSchema", "true").csv("/FileStore/shared_uploads/ahmed.say@northeastern.edu/train.csv")
3 val testData = spark.read.option("header", "true").option("inferSchema", "true").csv("/FileStore/shared_uploads/ahmed.say@northeastern.edu/test-2.csv")

> (4) Spark Jobs

> trainData: org.apache.spark.sql.DataFrame = [PassengerId: integer, Survived: integer ... 10 more fields]
> testData: org.apache.spark.sql.DataFrame = [PassengerId: integer, Pclass: integer ... 9 more fields]

spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@63a76fd0
trainData: org.apache.spark.sql.DataFrame = [PassengerId: int, Survived: int ... 10 more fields]
testData: org.apache.spark.sql.DataFrame = [PassengerId: int, Pclass: int ... 9 more fields]

Command took 3.64 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster
```

2) Exploratory Data Analysis:

```
1 // view the schema of the DataFrame
2 trainData.printSchema()

root
 |-- PassengerId: integer (nullable = true)
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- SibSp: integer (nullable = true)
 |-- Parch: integer (nullable = true)
 |-- Ticket: string (nullable = true)
 |-- Fare: double (nullable = true)
 |-- Cabin: string (nullable = true)
 |-- Embarked: string (nullable = true)

Command took 0.34 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster
```

a.

```
1 // get summary statistics for numerical columns
2 trainData.describe().show()

> (2) Spark Jobs

[summary] PassengerId| Survived| Pclass| Name| Sex| Age| SibSp| Parch|
Ticket| Fare|Cabin|Embarked|
-----+-----+-----+-----+-----+-----+-----+-----+-----+
| count|      891|      891|      891|      891|  891|    714|      891|      891|
891|      204|      889|      891|      891|    null|    null|    null|    null|
| mean|    446.0| 0.3838383838383838| 2.388641975368642|    null|    null| 29.69911764705882| 0.5230678563411896| 0.38159371492704824| 260318.
54916792738| 32.2042079685746|    null|    null|    null|    null| 14.526497332334035| 1.1027434322934315| 0.8060572211299488| 471609.
| stddev| 257.3538420152301| 0.48659245426485753| 0.8369712409770491|    null|    null| 14.526497332334035| 1.1027434322934315| 0.8060572211299488| 471609.
26868834975| 49.69342859718089|    null|    null|    null|    null| 14.526497332334035| 1.1027434322934315| 0.8060572211299488| 471609.
| min|      1|      0|      C|      0|      1|  1|  0|      0|      0|
110152|      0.0| A10|      C|      0|      1|  1|  0|      0|      0|
| max|      891|      1|      3|van Melkebeke, Mr...| male|  80.0|      8|      6|
WE/P 5735| 512.3292| T|      S|

Command took 2.27 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster
```

b.

Cmd 6

```
1 // check for missing values
2 val null_counts = trainData.select(trainData.columns.map(c => count(when(col(c).isNull, c)).alias(c)): _*)
3 null_counts.show()
```

▶ (2) Spark Jobs

▶ null_counts: org.apache.spark.sql.DataFrame = [PassengerId: long, Survived: long ... 10 more fields]

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	0	0	0	177	0	0	0	0	687	2

null_counts: org.apache.spark.sql.DataFrame = [PassengerId: bigint, Survived: bigint ... 10 more fields]

Command took 2.02 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster

C.

Cmd 7

```
1 // check the number of distinct values in each column
2 trainData.select(trainData.columns.map(c => countDistinct(col(c)).alias(c)): _*).show()
```

▶ (3) Spark Jobs

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
891	2	3	891	2	88	7	7	681	248	147	3

Command took 1.96 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster

d.

Cmd 8

```
1 // Count the number of passengers in each category
2 trainData.groupBy("sex").count().show()
3 trainData.groupBy("pclass").count().show()
4 trainData.groupBy("embarked").count().show()
```

▶ (6) Spark Jobs

sex	count
female	314
male	577

pclass	count
1	216
3	491
2	184

embarked	count
Q	77
null	2
C	168

Command took 2.48 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster

e.

Cmd 9

```

Cmd 9
1 // Create a pivot table to show the survival rate by sex and class
2 val pivotDF = trainData.filter(col("survived").isNotNull).groupBy("sex").pivot("pclass").agg(avg("survived"))
3 pivotDF.show()

(7) Spark Jobs
pivotDF: org.apache.spark.sql.DataFrame = [sex: string, 1: double ... 2 more fields]
+-----+-----+-----+
| sex|          1|          2|          3|
+-----+-----+-----+
|female| 0.9680851063829787|0.9210526315789473|          0.5|
| male|0.36885245901639346|0.1574074074074074|0.13544668587896252|
+-----+-----+-----+

pivotDF: org.apache.spark.sql.DataFrame = [sex: string, 1: double ... 2 more fields]
Command took 2.53 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:52:27 AM on My Cluster

```

f.

3) Feature Engineering:

```

1 // Drop irrelevant columns
2 val trainData2 = trainData.drop("PassengerId", "Ticket", "Name", "Cabin")
3 val testData2 = testData.drop("PassengerId", "Ticket", "Name", "Cabin")
4
5 // Create a new feature "FamilySize"
6 val trainData3 = trainData2.withColumn("FamilySize", col("SibSp") + col("Parch") + 1)
7 val testData3 = testData2.withColumn("FamilySize", col("SibSp") + col("Parch") + 1)
8
9 // Create a new feature "IsAlone"
10 val trainData4 = trainData3.withColumn("IsAlone", when(col("FamilySize") === 1, 1).otherwise(0))
11 val testData4 = testData3.withColumn("IsAlone", when(col("FamilySize") === 1, 1).otherwise(0))

trainData2: org.apache.spark.sql.DataFrame = [Survived: integer, Pclass: integer ... 6 more fields]
testData2: org.apache.spark.sql.DataFrame = [Pclass: integer, Sex: string ... 5 more fields]
trainData3: org.apache.spark.sql.DataFrame = [Survived: integer, Pclass: integer ... 7 more fields]
testData3: org.apache.spark.sql.DataFrame = [Pclass: integer, Sex: string ... 6 more fields]
trainData4: org.apache.spark.sql.DataFrame = [Survived: integer, Pclass: integer ... 8 more fields]
testData4: org.apache.spark.sql.DataFrame = [Pclass: integer, Sex: string ... 7 more fields]

trainData2: org.apache.spark.sql.DataFrame = [Survived: int, Pclass: int ... 6 more fields]
testData2: org.apache.spark.sql.DataFrame = [Pclass: int, Sex: string ... 5 more fields]
trainData3: org.apache.spark.sql.DataFrame = [Survived: int, Pclass: int ... 7 more fields]
testData3: org.apache.spark.sql.DataFrame = [Pclass: int, Sex: string ... 6 more fields]
trainData4: org.apache.spark.sql.DataFrame = [Survived: int, Pclass: int ... 8 more fields]
testData4: org.apache.spark.sql.DataFrame = [Pclass: int, Sex: string ... 7 more fields]

Command took 0.58 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:15:35 PM on My Cluster

```

a.

```

Cmd 12
1 trainData4.show(5)
2 testData4.show(5)

(2) Spark Jobs

+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|Survived|Pclass| Sex| Age|SibSp|Parch| Fare|Embarked|FamilySize|IsAlone|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|      0|      3| male|22.0|      1|      0| 7.25|      S|          2|      0|
|      1|      1|female|38.0|      1|      0|71.2833|      C|          2|      0|
|      1|      3|female|26.0|      0|      0| 7.925|      S|          1|      1|
|      1|      1|female|35.0|      1|      0| 53.1|      S|          2|      0|
|      0|      3| male|35.0|      0|      0| 8.05|      S|          1|      1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|Pclass| Sex| Age|SibSp|Parch| Fare|Embarked|FamilySize|IsAlone|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|      3| male|34.5|      0|      0| 7.8292|      Q|          1|      1|
|      3|female|47.0|      1|      0|  7.0|      S|          2|      0|
|      2| male|62.0|      0|      0| 9.6875|      Q|          1|      1|
|      3| male|27.0|      0|      0| 8.6625|      S|          1|      1|
|      3|female|22.0|      1|      1|12.2875|      S|          3|      0|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

Command took 1.11 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:16:21 PM on My Cluster

```

b.

4) Prediction:

```
1 // Convert the string columns to numerical indices
2 val sexIndexer = new StringIndexer().setInputCol("Sex").setOutputCol("SexIndex").setHandleInvalid("skip")
3 val embarkedIndexer = new StringIndexer().setInputCol("Embarked").setOutputCol("EmbarkedIndex").setHandleInvalid("skip")
4
5 // Convert the numerical indices to binary vectors
6 val sexEncoder = new OneHotEncoder().setInputCol("SexIndex").setOutputCol("SexVec")
7 val embarkedEncoder = new OneHotEncoder().setInputCol("EmbarkedIndex").setOutputCol("EmbarkedVec")
8
9 // Vectorize the features
10 val assembler = new VectorAssembler()
11   .setInputCols(Array("Pclass", "SexVec", "Age", "SibSp", "Parch", "Fare", "EmbarkedVec", "FamilySize", "IsAlone"))
12   .setOutputCol("features")
13   .setHandleInvalid("skip")
```

sexIndexer: org.apache.spark.ml.feature.StringIndexer = strIdx_df497a020a70
embarkedIndexer: org.apache.spark.ml.feature.StringIndexer = strIdx_373ff8a64bde
sexEncoder: org.apache.spark.ml.feature.OneHotEncoder = oneHotEncoder_da9434b204b1
embarkedEncoder: org.apache.spark.ml.feature.OneHotEncoder = oneHotEncoder_e04dcbbb970a
assembler: org.apache.spark.ml.feature.VectorAssembler = VectorAssembler: uid=vecAssembler_2536348786b8, handleInvalid=skip, numInputCols=9

Command took 0.36 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:16:21 PM on My Cluster

a.

```
1 // Create a pipeline
2 val pipeline = new Pipeline().setStages(Array(sexIndexer, embarkedIndexer, sexEncoder, embarkedEncoder, assembler))
3
4 // Fit the pipeline to the training data
5 val pipelineModel = pipeline.fit(trainData4)
```

► (4) Spark Jobs

pipeline: org.apache.spark.ml.Pipeline = pipeline_3d0ab8fe066b
pipelineModel: org.apache.spark.ml.PipelineModel = pipeline_3d0ab8fe066b

Command took 1.26 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:16:21 PM on My Cluster

b.

```
1 // Transform the training and test data
2 val trainDataFinal = pipelineModel.transform(trainData4).select("features", "Survived")
3 val testDataFinal = pipelineModel.transform(testData4).select("features")
4
5 // Split the data into training and testing sets
6 val Array(trainingData, testingData) = trainDataFinal.randomSplit(Array(0.7, 0.3))
7
8 // Create the Random Forest Classifier model
9 val rf = new RandomForestClassifier().setLabelCol("Survived").setFeaturesCol("features")
10
11 // Train the model
12 val model = rf.fit(trainingData)
13
14 // Make predictions on the testing set
15 val predictions = model.transform(testingData)
16
17 // Evaluate the model using MulticlassClassificationEvaluator
18 val evaluator = new MulticlassClassificationEvaluator().setLabelCol("Survived").setPredictionCol("prediction").setMetricName("accuracy")
19 val accuracy = evaluator.evaluate(predictions)
```

► (11) Spark Jobs

trainDataFinal: org.apache.spark.sql.DataFrame = [features: udt, Survived: integer]
testDataFinal: org.apache.spark.sql.DataFrame = [features: udt]
trainingData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [features: udt, Survived: integer]
testingData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [features: udt, Survived: integer]
predictions: org.apache.spark.sql.DataFrame = [features: udt, Survived: integer ... 3 more fields]
trainDataFinal: org.apache.spark.sql.DataFrame = [features: vector, Survived: int]
testDataFinal: org.apache.spark.sql.DataFrame = [features: vector]
trainingData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [features: vector, Survived: int]
testingData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [features: vector, Survived: int]
rf: org.apache.spark.ml.classification.RandomForestClassifier = rfc_ed3679acf7e0
model: org.apache.spark.ml.classification.RandomForestClassificationModel = RandomForestClassificationModel: uid=rfc_ed3679acf7e0, numTrees=20, numClasses=2, numFeatures=10
predictions: org.apache.spark.sql.DataFrame = [features: vector, Survived: int ... 3 more fields]
evaluator: org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator = MulticlassClassificationEvaluator: uid=mcEval_49257563977, metricName=accuracy, metricLabel=0.0, beta=1.0, eps=1.0E-15
accuracy: Double = 0.8285714285714286

c.

Command took 0.24 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:16:21 PM on My Cluster

5) Result:

Cmd 17

```
1 println(s"Accuracy: ${accuracy*100}%")
```

Accuracy: 82.85714285714286%

Command took 0.24 seconds -- by ahmed.say@northeastern.edu at 4/6/2023, 12:16:21 PM on My Cluster

Cmd 18