Evaluation

Lesson Objectives

- After completing this lesson, you should be able to:
  - Evaluate binary classification algorithms using area under the ROC curve
  - Evaluate multiclass classification algorithms using several metrics
  - Evaluate regression algorithms using several metrics
  - Evaluate logistic and linear regression algorithms using summaries

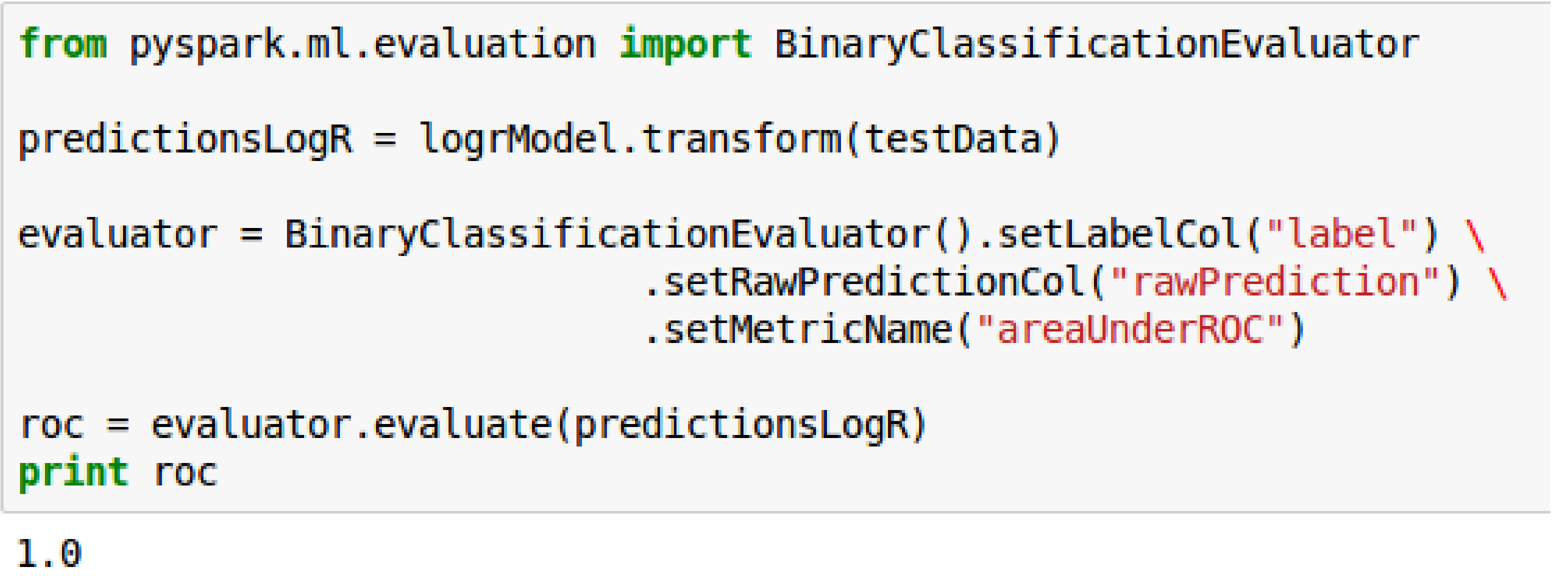
Evaluators

- Computes metrics from predictions
- Available Evaluators
    - BinaryClassificationEvaluator
    - MultiClassClassificationEvaluator
    - RegressionEvaluator

BinaryClassificationEvaluator

- Evaluator for binary classification
- Expects two input columns: rawPrediction and label
- Supported metric: areaUnderROC

BinaryClassificationEvaluator



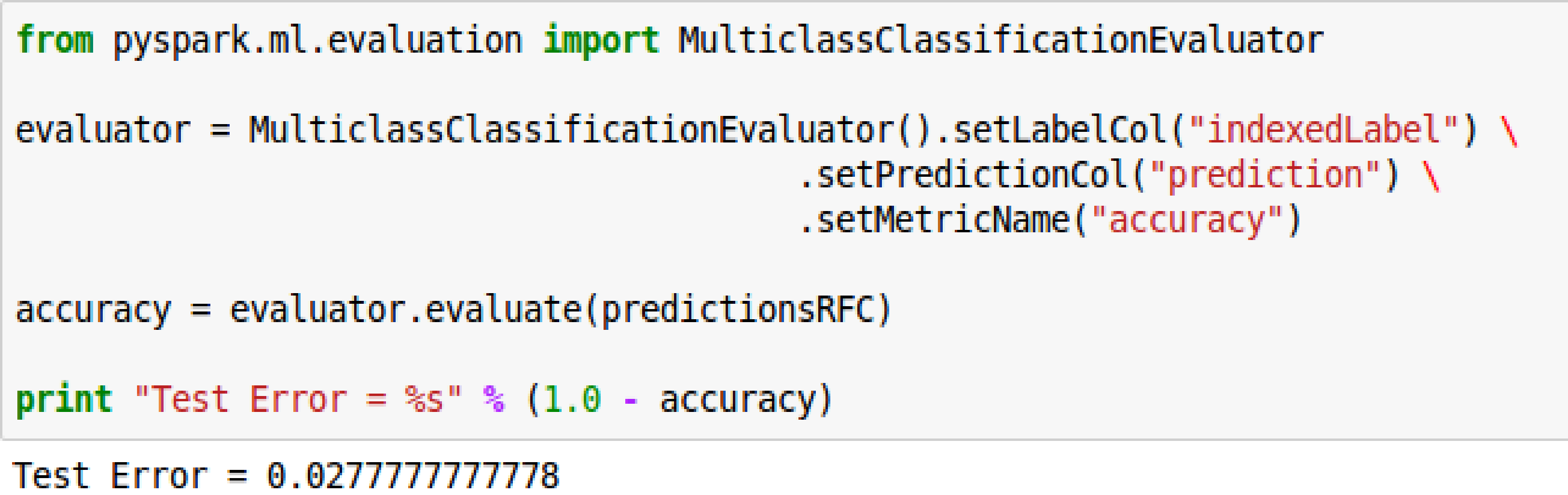
```
from pyspark.ml.evaluation import BinaryClassificationEvaluator

predictionsLogR = logrModel.transform(testData)

evaluator = BinaryClassificationEvaluator().setLabelCol("label") \
                            .setRawPredictionCol("rawPrediction") \
                            .setMetricName("areaUnderROC")

roc = evaluator.evaluate(predictionsLogR)
print roc
```

```
1.0
```

MulticlassClassificationEvaluator

- valuator for multiclass classification
- Expects two input columns: prediction and label
- Supported metrics:
  - F1 (default)
  - Precision
  - Recall
  - weightedPrecision
  - weightedRecall

MulticlassClassificationEvaluator



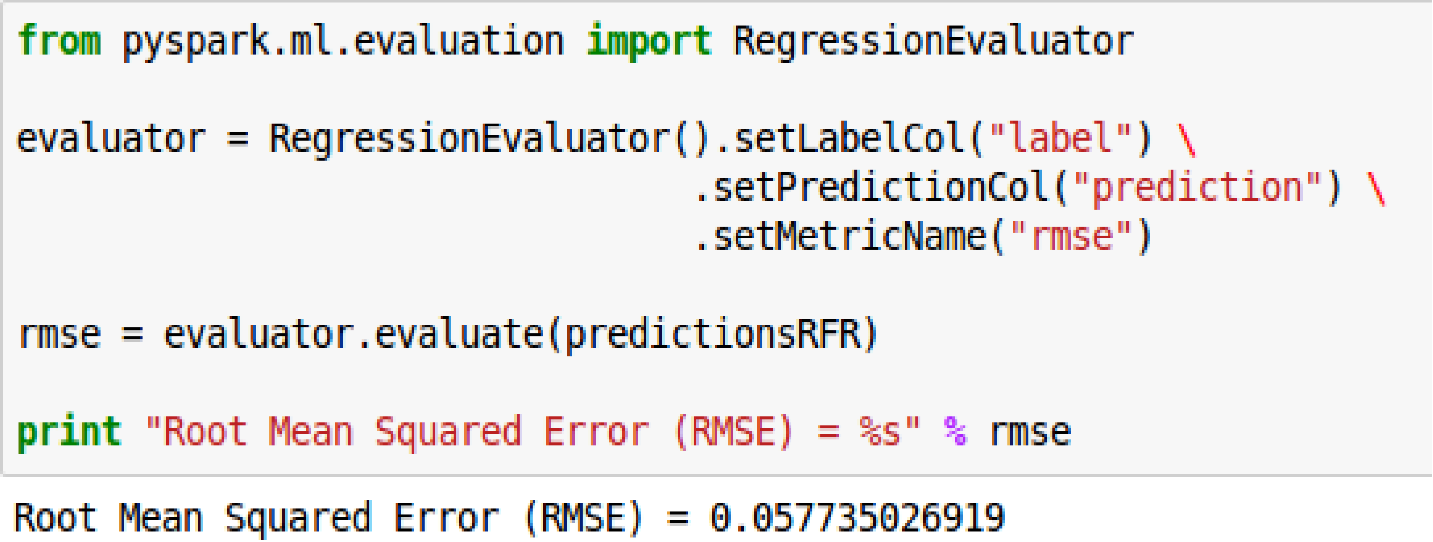
```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

evaluator = MulticlassClassificationEvaluator().setLabelCol("indexedLabel") \
                                        .setPredictionCol("prediction") \
                                        .setMetricName("accuracy")

accuracy = evaluator.evaluate(predictionsRFC)

print "Test Error = %s" % (1.0 - accuracy)
```

```
Test Error = 0.0277777777778
```

RegressionEvaluator

- Evaluator for regression
- Expects two input columns: prediction and label
- Supported metrics:
  - rmse: root mean squared error (default)
  - mse: mean squared error
  - r2: R2, the coefficient of determination
  - mae: mean absolute error

A Simple Regression Evaluator



```
from pyspark.ml.evaluation import RegressionEvaluator

evaluator = RegressionEvaluator().setLabelCol("label") \
                                .setPredictionCol("prediction") \
                                .setMetricName("rmse")

rmse = evaluator.evaluate(predictionsRFR)

print "Root Mean Squared Error (RMSE) = %s" % rmse
```

```
Root Mean Squared Error (RMSE) = 0.057735026919
```

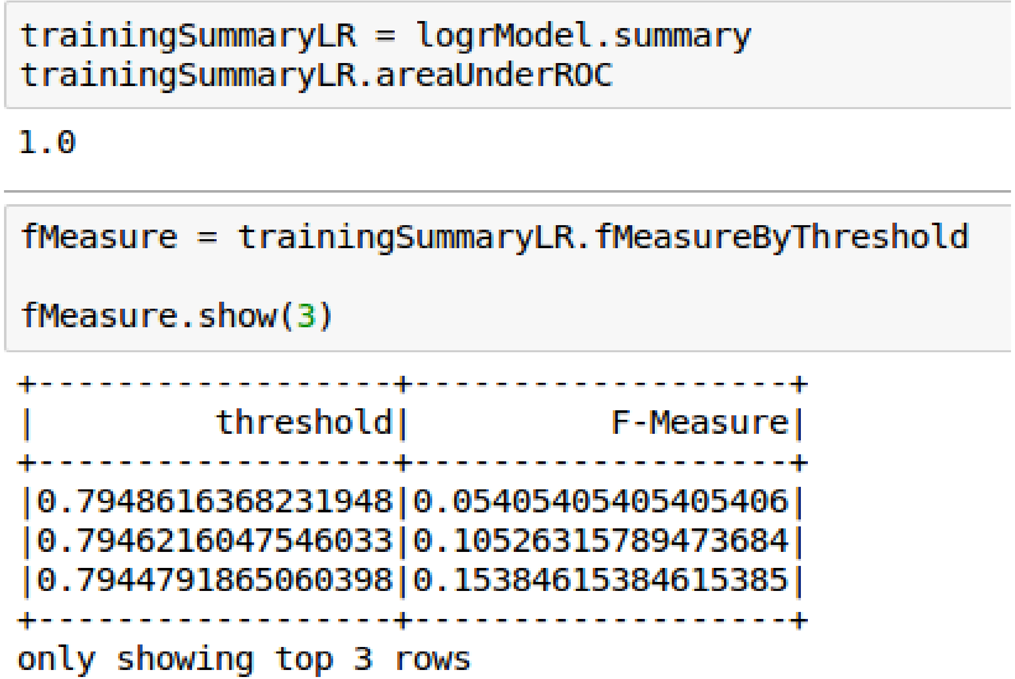
LogisticRegressionSummary

- LogisticRegressionSummary accessible through summary attribute of a LogisticRegressionModel
- Summarizes the model over the training set

BinaryLogisticRegressionSummary

- Supported metrics:
  - areaUnderROC: area under the receiver operating characteristic (ROC) curve
  - fMeasureByThreshold: dataframe with two fields (threshold, F-Measure) curve with beta = 1
  - pr: precision-recall curve, dataframe containing two fields recall, precision with (0.0, 1.0) prepended to it
  - precisionByThreshold: dataframe with two fields (threshold, precision) curve
  - recallByThreshold: dataframe with two fields (threshold, recall) curve
  - roc: receiver operating characteristic (ROC) curve, dataframe having two fields (FPR, TPR) with (0.0, 0.0) prepended and (1.0, 1.0) appended to it

A Simple Logistic Regression Summary (1)



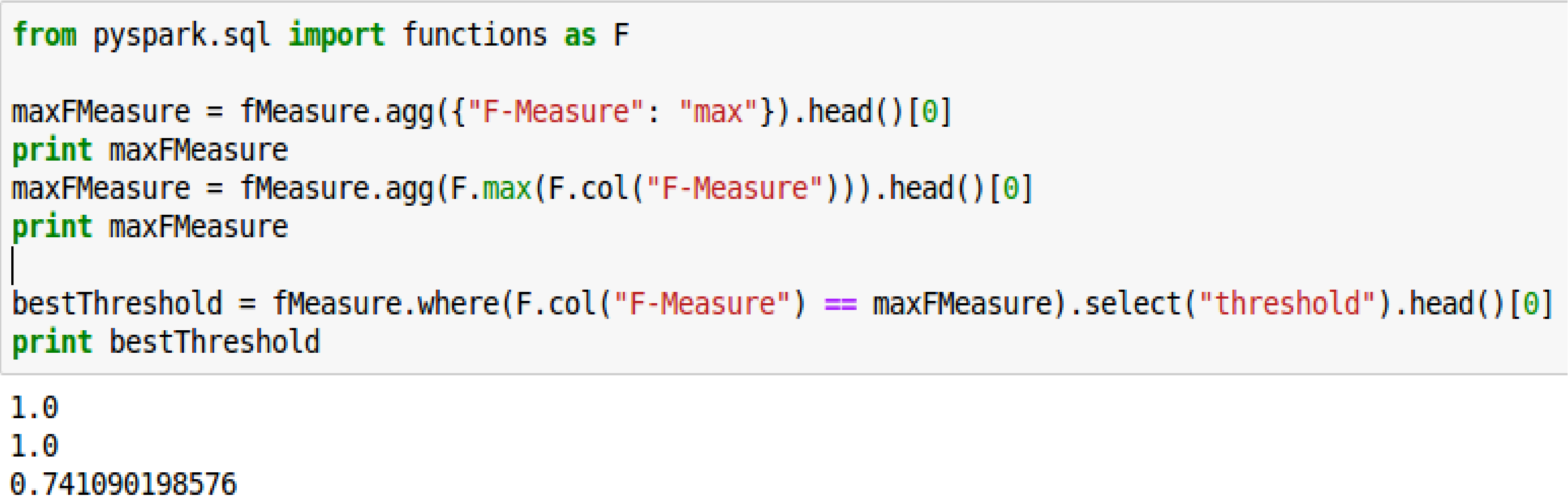
```
trainingSummaryLR = logrModel.summary
trainingSummaryLR.areaUnderROC
```

```
1.0
```

```
fMeasure = trainingSummaryLR.fMeasureByThreshold

fMeasure.show(3)
```

```
+------------------+-------------------+
|         threshold|          F-Measure|
+------------------+-------------------+
|0.7948616368231948|0.05405405405405406|
|0.7946216047546033|0.10526315789473684|
|0.7944791865060398|0.15384615384615385|
+------------------+-------------------+
only showing top 3 rows
```

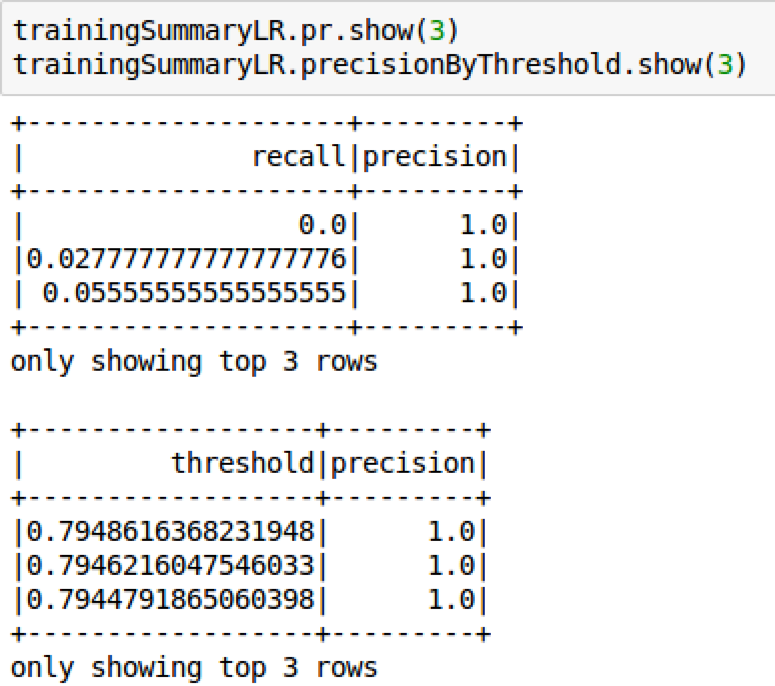
A Simple Logistic Regression Summary (2)



```
from pyspark.sql import functions as F

maxFMeasure = fMeasure.agg({"F-Measure": "max"}).head()[0]
print maxFMeasure
maxFMeasure = fMeasure.agg(F.max(F.col("F-Measure"))).head()[0]
print maxFMeasure

bestThreshold = fMeasure.where(F.col("F-Measure") == maxFMeasure).select("threshold").head()[0]
print bestThreshold
```

```
1.0
1.0
0.741090198576
```

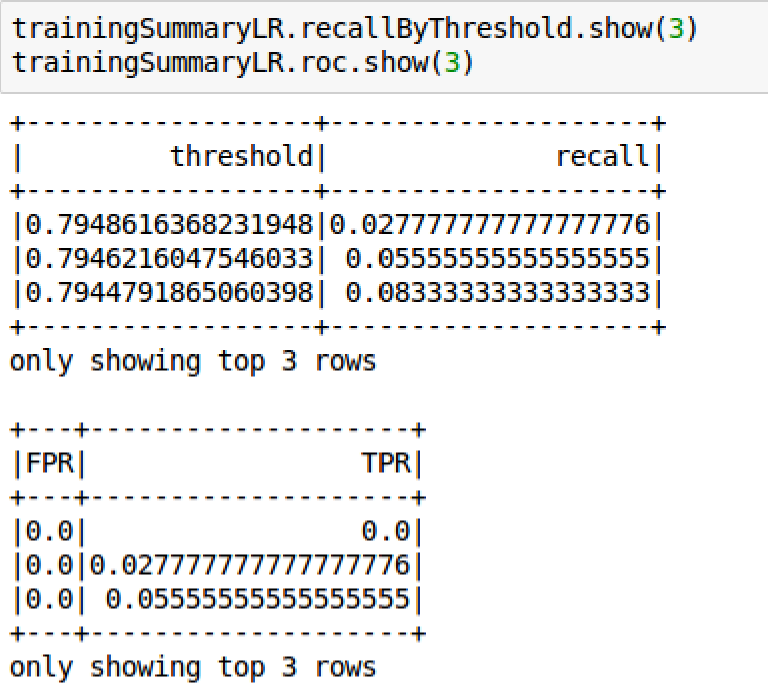
A Simple Logistic Regression Summary (3)



```
trainingSummaryLR.pr.show(3)
trainingSummaryLR.precisionByThreshold.show(3)
```

```
+--------------------+---------+
|              recall|precision|
+--------------------+---------+
|                 0.0|      1.0|
|0.027777777777777776|      1.0|
| 0.05555555555555555|      1.0|
+--------------------+---------+
only showing top 3 rows

+------------------+---------+
|         threshold|precision|
+------------------+---------+
|0.7948616368231948|      1.0|
|0.7946216047546033|      1.0|
|0.7944791865060398|      1.0|
+------------------+---------+
only showing top 3 rows
```

A Simple Logistic Regression Summary (4)



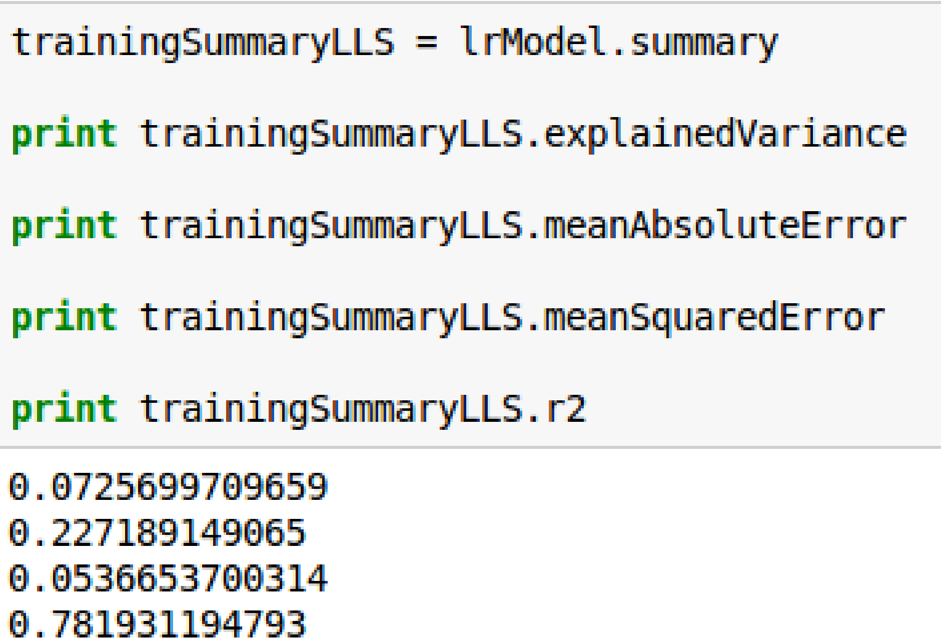
```
trainingSummaryLR.recallByThreshold.show(3)
trainingSummaryLR.roc.show(3)
```

```
+------------------+--------------------+
|         threshold|              recall|
+------------------+--------------------+
|0.7948616368231948|0.027777777777777776|
|0.7946216047546033| 0.05555555555555555|
|0.7944791865060398| 0.08333333333333333|
+------------------+--------------------+
only showing top 3 rows

+---+--------------------+
|FPR|                 TPR|
+---+--------------------+
|0.0|                 0.0|
|0.0|0.027777777777777776|
|0.0| 0.05555555555555555|
+---+--------------------+
only showing top 3 rows
```

LinearRegressionSummary

- Accessible through summary attribute of a LinearRegressionModel
- Summarizes the model over the training set
- Supported metrics:
  - explainedVariance: explained variance regression score
  - meanAbsoluteError: mean absolute error (L1-norm loss)
  - meanSquaredError: mean squared error (quadractic loss)
  - r2: R2, the coefficient of determination
  - residuals: residuals (label - predicted value)
  - rootMeanSquaredError: root mean squared error

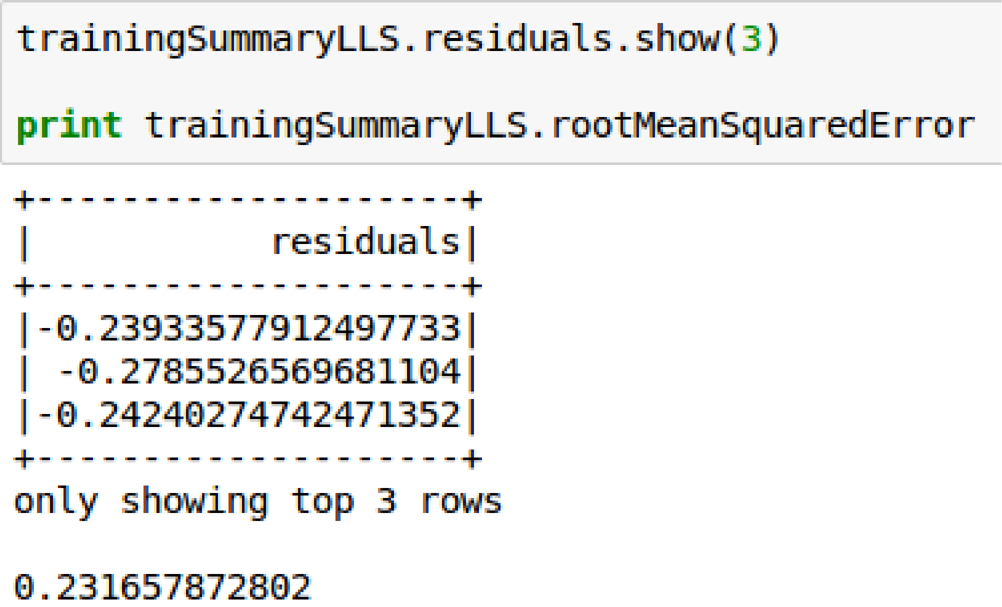
A Simple Linear Regression Summary (1)



```
trainingSummaryLLS = lrModel.summary

print trainingSummaryLLS.explainedVariance

print trainingSummaryLLS.meanAbsoluteError

print trainingSummaryLLS.meanSquaredError

print trainingSummaryLLS.r2
```

```
0.0725699709659
0.227189149065
0.0536653700314
0.781931194793
```

A Simple Linear Regression Summary (2)



```
trainingSummaryLLS.residuals.show(3)

print trainingSummaryLLS.rootMeanSquaredError
```

```
+--------------------+
|           residuals|
+--------------------+
|-0.23933577912497733|
| -0.2785526569681104|
|-0.24240274742471352|
+--------------------+
only showing top 3 rows

0.231657872802
```

Lesson Summary

- Having completed this lesson, you should be able to:
  - Evaluate binary classification algorithms using area under the ROC curve
  - Evaluate multiclass classification algorithms using several metrics
  - Evaluate regression algorithms using several metrics
  - Evaluate logistic and linear regression algorithms using summaries