Summary Statistics, Correlations, and Random Data

Lesson Objectives

- After completing this lesson, you should be able to:
  - Compute column summary statistics
  - Compute pairwise correlations between series/columns
  - Generate random data from different distributions

Summary Statistics

- Column summary statistics for an instance of `RDD[Vector]` are available through the `colStats()` function in `Statistics`
- It returns an instance of `MultivariateStatisticalSummary`, which contains column-wise results for:
  - min, max
  - mean, variance
  - numNonzeros
  - normL1, normL2
- `count` returns the total count of elements

Example

```
from pyspark.mllib.stat import Statistics, MultivariateStatisticalSummary
```

```
observations = sc.parallelize([Vectors.dense(1.0,2.0),
                               Vectors.dense(4.0,5.0),
                               Vectors.dense(7.0,8.0)])
```

```
summary = Statistics.colStats(observations)
```

```
summary.mean()
```

```
array([ 4.,  5.])
```

Summary Statistics Example

```
summary.variance()
```

```
array([ 9.,  9.])
```

```
summary.numNonzeros()
```

```
array([ 3.,  3.])
```

```
summary.normL1()
```

```
array([ 12.,  15.])
```

```
summary.normL2()
```

```
array([ 8.1240384 ,  9.64365076])
```

Correlations

- Pairwise correlations among series is available through the `corr()` function in `Statistics`
- Correlation methods supported:
  - Pearson (default)
  - Spearman (used for rank variables)
- Inputs supported:
  - two `RDD[Double]`s, returning a single Double value
  - an `RDD[Vector]`, returning a correlation Matrix

Pearson Correlation

```
X = sc.parallelize([2.0, 9.0, -7.0])
Y = sc.parallelize([1.0, 3.0, 5.0])
```

```
Statistics.corr(X, Y, "pearson")
```

```
-0.5610408535732834
```

```
data = sc.parallelize([Vectors.dense(2.0,9.0,-7.0),
                       Vectors.dense(1.0,-3.0,5.0),
                       Vectors.dense(4.0,0.0,-5.0)])
```

```
Statistics.corr(data, method="pearson")
```

```
array([[ 1.        ,  0.05241424, -0.64490202],
       [ 0.05241424,  1.        , -0.79701677],
       [-0.64490202, -0.79701677,  1.        ]])
```

Random Data Generation

- RandomRDDs generate either random double RDDs or vector RDDs
- Supported distributions:
  - uniform, normal, lognormal, poisson, exponential, and gamma
- Useful for randomized algorithms, prototyping and performance testing

Simple Example

```
from pyspark.mllib.random import RandomRDDs
```

```
million = RandomRDDs.poissonRDD(sc, mean=1.0, size=1000000L, numPartitions=10)
```

```
million.mean()
```

```
0.9998949999999995
```

```
million.variance()
```

```
0.9989229889750021
```

Simple Vector Example

```
data = RandomRDDs.normalVectorRDD(sc, numRows=10000L, numCols=3, numPartitions=10)
```

```
stats = Statistics.colStats(data)
```

```
stats.mean()
```

```
array([-0.01571908,  0.00353506, -0.00245943])
```

```
data.take(5)
```

```
[array([-0.32289664,  0.31055341, -0.620877  ]),
 array([-1.43262361, -1.23162754,  0.41518041]),
 array([-0.05028347, -0.48672274,  0.27745594]),
 array([ 0.04270499, -1.00523163, -0.6207748 ]),
 array([-1.17200235, -1.53537526, -0.03264326])]
```

Available Distributions

- `exponentialRDD`
- `gammaRDD`
- `logNormalRDD`
- `normalRDD`
- `poissonRDD`
- `uniformRDD`

- `exponentialVectorRDD`
- `gammaVectorRDD`
- `logNormalVectorRDD`
- `normalVectorRDD`
- `poissonVectorRDD`
- `uniformVectorRDD`

Lesson Summary

- Having completed this lesson, you should be able to:
  - Compute column summary statistics
  - Compute pairwise correlations between series/columns
  - Generate random data from different distributions