Code Documentation for StatsLibrary Project

By Rachel Hussmann

Overview

The purpose of the StatsLibrary project was to create classes and methods that calculate statistical operations that we learned in class. This project is a continuation from project 1 and has five classes, each that manage a separate subset of statistical operations.

How It Works

Factorial

The Factorial class is responsible for containing the methods that calculate factorials. This class has two methods:

- factorial
    - Parameters: int n – The number to find the factorial of
    - Functionality – Calculates the number equal to 1 * 2 * 3 * . . . * n-1 * n (notes usually as n!)
    - Returns: The number equal to the factorial of n as an integer
- factorial
    - Parameters: BigInteger – The number to find the factorial of
    - Functionality – Calculates the number equal to 1 * 2 * 3 * . . . * n-1 * n (notes usually as n!)
    - Returns: The number equal to the factorial of n as a BigInteger object

PoissonDistribution

The PoissonDistribution class is responsible for containing the methods that calculate statistical numbers relating to the Poisson Probability Distribution including the probability, lambda and standard deviation. This class has nine methods:

- poissonDistribution
    - Parameters: int n – The number of units, int k – The total number of events, int scalar – Scalar value for lambda, int y – The number of events that occur
    - Functionality – Calculates the probability of an event happening based on a Poisson Probability distribution
    - Returns: double – The probability of the event occurring
- poissonDistribution

    - Parameters: double lambda – The mean or variance of the Poisson distribution, int y – The number of events that occur
    - Functionality – Calculates the probability of an event happening based on a Poisson Probability distribution
    - Returns: double – The probability of the event occurring
- poissonDistribution
    - Parameters: BigInteger n – The number of units, BigInteger k – The total number of events, BigInteger scalar – Scalar value for lambda, BigInteger y – The number of events that occur
    - Functionality – Calculates the probability of an event happening based on a Poisson Probability distribution
    - Returns: BigDecimal – The probability of the event occurring
- poissonDistribution
    - Parameters: BigDecimal lambda – The mean or variance of the Poisson distribution, BigInteger y – The number of events that occur
    - Functionality – Calculates the probability of an event happening based on a Poisson Probability distribution
    - Returns: BigDecimal – The probability of the event occurring
- lambda
    - Parameters: int n – The number of units, int k – The total number of events, int scalar – Scalar value for lambda
    - Functionality – Finds the value of lambda for a poisson distribution
    - Returns: double – The value of lambda
- lambda
    - Parameters: BigInteger n – The number of units, BigInteger k – The total number of events, BigInteger scalar – Scalar value for lambda
    - Functionality – Finds the value of lambda for a poisson distribution
    - Returns: BigDecimal – The value of lambda
- standardDeviation
    - Parameter: double lambda – The mean or variance of the distribution
    - Functionality – Finds the standard deviation of the distribution
    - Returns: double – The standard deviation of the distribution
- standardDeviation
    - Parameter: BigDecimal lambda – The mean or variance of the distribution
    - Functionality – Finds the standard deviation of the distribution
    - Returns: BigDecimal – The standard deviation of the distribution
- testOutput
    - Parameters: None
    - Functionality – Tests the methods that have been developed in the class
    - Returns: Nothing, but prints statements

TchebysheffsTheorem

The TchebysheffsTheorem class is responsible for containing the methods that calculate and test Tchebysheff's theorem. The class has three methods:

- tchebysheffsTheorem
    - Parameters: int lowerBound – The lower value in the range of data, int upperBound – The higher value in the range of data, int SD – The standard deviation of the data, int mean – The mean (or expected value) of the data
    - Functionality – Calculates the percentage of data that is within the two bounds
    - Returns: double – The percentage of data that is within the two bounds
- tchebysheffsTheorem
    - Parameters: double lowerBound – The lower value in the range of data, double upperBound – The higher value in the range of data, double SD – The standard deviation of the data, double mean – The mean (or expected value) of the data
    - Functionality – Calculates the percentage of data that is within the two bounds
    - Returns: double – The percentage of data that is within the two bounds
- testOutput
    - Parameters: None
    - Functionality – Prints out test statements for the class methods
    - Returns: Nothing, but prints out statements

UniformDistribution

The UniformDistribution class is responsible for containing the methods that calculate statistical numbers relating to the Uniform Probability Distribution including the probability, expected value, variance and standard deviation. This class has nine methods:

- uniformDistribution
    - Parameters: int a – The lower constant of the uniform distribution, int b – The higher constant of the uniform distribution, int c – The lower value of the probability question, int d – The higher value of the probability question
    - Functionality – Calculates the probability of an event happening based on a Uniform Probability distribution
    - Returns: double – The probability of the event occurring
- uniformDistribution
    - Parameters: BigInteger a – The lower constant of the uniform distribution, BigInteger b – The higher constant of the uniform distribution, BigInteger c – The lower value of the probability question, BigInteger d – The higher value of the probability question

    - Functionality – Calculates the probability of an event happening based on a Uniform Probability distribution
    - Returns: BigDecimal – The probability of the event occurring
- expectedValue
    - Parameters: int a – The lower constant of the uniform distribution, int b – The higher constant of the uniform distribution
    - Functionality – Calculates the expected value of the uniform distribution
    - Returns: double– The expected value of the uniform distribution
- expectedValue
    - Parameters: BigInteger a – The lower constant of the uniform distribution, BigInteger b – The higher constant of the uniform distribution
    - Functionality – Calculates the expected value of the uniform distribution
    - Returns: BigDecimal – The expected value of the uniform distribution
- variance
    - Parameters: int a – The lower constant of the uniform distribution, int b – The higher constant of the uniform distribution
    - Functionality – Calculates the variance of the uniform distribution
    - Returns: double – The variance of the uniform distribution
- variance
    - Parameters: BigInteger a – The lower constant of the uniform distribution, BigInteger b – The higher constant of the uniform distribution
    - Functionality – Calculates the variance of the uniform distribution
    - Returns: BigDecimal – The variance of the uniform distribution
- standardDeviation
    - Parameter: double variance – The variance of the uniform distribution
    - Functionality – Finds the standard deviation of the distribution
    - Returns: double – The standard deviation of the distribution
- standardDeviation
    - Parameter: BigDecimal variance – The variance of the uniform distribution
    - Functionality – Finds the standard deviation of the distribution
    - Returns: BigDecimal – The standard deviation of the distribution
- testOutput
    - Parameters: None
    - Functionality – Tests the methods that have been developed in the class
    - Returns: Nothing, but prints statements

StatsLibraryTester

The StatsLibraryTester class contains the main method. It is responsible for running and testing the methods for each of the classes created.

Output

Screenshots

```
Poisson Distribution using lambda = 2 and y = 4: 0.0902235221577418
Standard deviation of the distribution: 1.4142135623730951
Poisson Distribution using lambda = 2, y = 4, BigInteger and BigDecimal: 0.091
Standard deviation of distribution using BigDecimal: 1.414214

Poisson Distribution using k = 80, n = 60 and y = 0: 0.2635971381157268
Poisson Distribution using k = 80, n = 60, y = 0 and BigInteger: 0.264

Using Tchebysheff's Theorem (Integers): 0.75

Using Tchebysheff's Theorem (Double): 0.7500000000000004

Uniform distribution using a = 20, b = 25, c = 20, d = 22: 0.4
Expected value using a = 20 and b = 25: 22.5
Variance using a = 20 and b = 25: 2.0833333333333335
Standard deviation using a = 20 and b = 25: 1.4433756729740645

Now using BigInteger and BigDecimal objects
Uniform distribution using a = 20, b = 25, c = 20, d = 22: 0.4000
Expected value using a = 20 and b = 25: 22.5000
Variance using a = 20 and b = 25: 2.0834
Standard deviation using a = 20 and b = 25: 1.443399
```