**StatsLibrary Documentation**

**Author:** Arnab Das Utsa

**Class: StatsLibrary** This class provides statistical calculations for:

* Poisson Distribution
* Uniform Distribution
* Chebyshev's Theorem It also computes expected values and variances where applicable.

**Constructors:**

public StatsLibraryEnding() {}

* Default constructor. Initializes an empty object. Useful if you want to create an instance without any immediate parameters.

public StatsLibraryEnding(String input) {}

* Constructor with a string parameter. Prepared for future expansions where input data might be passed as a string.

public StatsLibraryEnding(double input) {}

* Constructor with a double parameter. Similarly allows flexibility if initial calculations need a numeric input.

**Methods:**

public BigInteger findFactorial(int n) {

BigInteger factorial = new BigInteger(Integer.toString(n));

BigInteger zero = new BigInteger("0");

BigInteger one = new BigInteger("1");

if (factorial.equals(zero)) {

factorial = one;

} else {

for (int i = n; i > 1; i--) {

BigInteger index = new BigInteger(Integer.toString(i));

BigInteger oneLessThanIndex = index.subtract(one);

factorial = factorial.multiply(oneLessThanIndex);

}

}

return factorial;

}

* This method computes factorial by repeatedly multiplying descending integers. BigInteger is used to handle very large numbers that int or long cannot store. It checks if n = 0 separately because 0! = 1.

public double findPoisson(double l, int y) {

double e = Math.E;

double factorialOfy = findFactorial(y).doubleValue();

double numerator = (Math.exp(-l) \* Math.pow(l, y));

double poisson = numerator / factorialOfy;

return poisson;

}

* Calculates the probability of exactly y events occurring in a fixed interval, given the average rate l. It uses the standard Poisson formula. It first calculates and and divides by .

public double findExpectedPoisson(double l) {

return l;

}

* In Poisson distribution, the expected value (mean) is simply the rate parameter .

public double findVariancePoisson(double l) {

return l;

}

* Similarly, the variance of a Poisson distribution is also , so the method just returns it.

public double findChebyshev(double k) {

double chebyshev = 1 - (1 / Math.pow(k, 2));

return chebyshev;

}

* Applies Chebyshev's inequality. This method computes the minimum probability that a value falls within k standard deviations of the mean. As k increases, the probability approaches 1.

public double findUniform(double theta1, double theta2) {

double uniform = 1 / (theta2 - theta1);

return uniform;

}

* For a uniform distribution, the density is constant between the two bounds. This method calculates that constant value.

public double findExpectedUniform(double theta1, double theta2) {

double expected = (theta1 + theta2) / 2;

return expected;

}

* The expected value (mean) of a uniform distribution is simply the midpoint between and .

public double findVarianceUniform(double theta1, double theta2) {

double variance = Math.pow((theta2 - theta1), 2) / 12;

return variance;

}

* Calculates the variance, which measures how spread out the values are in a uniform distribution. It uses the standard variance formula for uniform distributions.

**StatsLibraryEndingTester.java Documentation**

**Class: StatsLibraryEndingTester** This class tests all the methods implemented in the StatsLibraryEnding class by calling each method and printing the results.

**Main Method:**

* public static void main(String[] args) {
* StatsLibraryEnding test = new StatsLibraryEnding();
* double testerResults1 = test.findPoisson(2.3, 4);
* double testerResults2 = test.findExpectedPoisson(2.3);
* double testerResults3 = test.findVariancePoisson(2.3);
* double testerResults4 = test.findChebyshev(2.7);
* double testerResults5 = test.findUniform(0.2, 0.7);
* double testerResults6 = test.findExpectedUniform(0.2, 0.7);
* double testerResults7 = test.findVarianceUniform(0.2, 0.7);
* System.out.println("This is the calculation of the Poisson Distribution: " + testerResults1);
* System.out.println("This is the calculation of the Expected of the Poisson Distribution: " + testerResults2);
* System.out.println("This is the calculation of the Variance the Poisson Distribution: " + testerResults3);
* System.out.println("This is the calculation of Chebyshev's Theorem: " + testerResults4);
* System.out.println("This is the calculation of the Uniform Distribution: " + testerResults5);
* System.out.println("This is the calculation of the Expected of the Uniform Distribution: " + testerResults6);
* System.out.println("This is the calculation of the Variance the Uniform Distribution: " + testerResults7);

}

* This method acts like a small tester program. It creates an object of the StatsLibraryEnding class and runs each method with example inputs.
* It prints the results clearly labeled, so a human reading the console output understands which value belongs to which statistical calculation.
* Helpful for verifying that the functions work as intended before deploying or integrating them elsewhere.