Instrukcja do zadania 2

Celem ćwiczenia jest napisanie testu pokerowego. Program ma za zadanie na podstawie testu chi-kwadrat powiedzieć, czy wpisane slowo/zdanie zawiera przypadkowe litery.

Poker test

Hipoteza zerowa H0 – ilości 0 i 1 są sobie równe

Operujemy na binarnym ciągu wpisanego tekstu o długości s. Wybieramy liczbę m, jak najmniejszą, ale większą od 3, która dzieli s. k to ilość sekwencji wynikająca z zależności k = s/m.

Obrazując:

101 000 110 110 101 111 010 001 000 111 – tekst zamieniony na postać binarną

s = 30
m = 3
k = 10

Następnie liczymy ile mamy wszystkich kombinacji 0 i 1 w “trójce”. Jest to 2^m czyli w tym przypadku 2^3 = 8

Obrazując:

zero jedynek (0x”1”, czyli 000) – 1 kombinacja
jedna jedynka (1x”1”, czyli 100, 010, 001) – 3 kombinacje
dwie jedynki (2x”1”, czyli 110, 011, 101) – 3 kombinacje
trzy jedynki (3x”1”, czyli 111) – 1 kombinacja

Celem jest aby wyznaczyć ile razy w ciągu binarnym mieliśmy doczynienia z każdym z tych czterech przypadków. Czyli potrzebujemy ile razy wystąpiły trójki, które miały np dwie jedynki. (Określamy częstotliwość). Po wykonaniu tego otrzymujemy hashmape częstotliwości.

Obrazując:

{0=3, 1=7, 2=12, 3=1}
zero jedynek – 3 razy
jedna jedynka – 7 razy
dwie jedynki – 12 razy
trzy jedynki – 1 raz

Dla każdego przypadku potrzebujemy określić wartość oczekiwaną. Jest ona określona prawdopodobieństwem wystąpienia każdego przypadku.

Wartość oczekiwana w poker teście dana jest wzorem:

(kombinacja m po 0) * 1/2^m * k – dla 0
(kombinacja m po 1) * 1/2^m * k – dla 1
itd aż kolokwialnie mowiąc “dolna” liczba osiągnie m-1
Uwaga: kombinacja oznacza symbol Newtona

Przypominając z prezentacji wartość chi2 można wyliczyć jako:

chi2 = sum([freq(i) – expectedVal(i)]^2/expectedVal(i) [1]

Podsumowując program działa następująco:

1) Zamiana wpisanego słowa na ciąg binarny. (metoda *strToBinary(String s)*)
2) Wyliczenie m – dzielnika powstałego ciągu binarnego. (metoda *checkBlockSize(Integer m, String strBinary))*
3) Sprawdzenie częstotliwości pojawiania się konkretnych “trójek” (metody: *getSamplesFrequencyMap(String s, Integer m) oraz occurrences(String str, HashMap<Long, Integer> combFreq))*
4) Wyliczenie wartości oczekiwanych (metody: *expectedValues(Integer m, Integer k), combinations(int N, int R) I factorialGenerator(int givenNumberToReturnFactorial)*)
5) Użycie częstotliwości oraz wartości oczekiwanych I wyliczenie współczynnika chi2.(metoda: *chiSquared(HashMap<Long, Integer> combFreq, ArrayList<Double> expectedValues)*)
6) Porównanie uzyskanego wyniku z wartością tablicową. (metoda: *freedomLevels(Double pokerResult, Integer m))*
7) Wypisanie czy ciąg zdał lub nie zdał testu randomowości. (metoda: *printResults(Boolean test)*)

Waszym zadaniem jest:

uzupełnić funcje chiSquared oraz expectedValues. Dla chiSquared jest to implementacja wzoru na chi kwadrat [1], a do expectedValues pętla, która zwraca ArrayList<Double> expectedValues.

Przydatna strona w razie problemów:
https://towardsdatascience.com/poker-test-pattern-based-random-number-detection-in-python-ec2dba4955bc