МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ»
ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра алгоритмической математики

ОТЧЁТ

по практической работе №4
по дисциплине «Статистический анализ»
Тема: Элементы корреляционного анализа.
Проверка статистической гипотезы о равенстве коэффициента корреляции нулю

Студент гр. 9372 _____ Иванов Р. С.

Преподаватель _____ Сучков А. И.

Санкт-Петербург
2021

Цель работы

Освоение основных понятий, связанных с корреляционной зависимостью между случайными величинами, статистическими гипотезами и проверкой их «справедливости».

Основные теоретические положения

Определение 1. Статистической называют зависимость, при которой изменение одной из величин влечёт изменение распределения другой. Если при этом изменение одной величины приводит к изменению среднего значения другой, то статистическую зависимость называют корреляционной.

Корреляционная таблица

где $m_(x_i)$ – частота появления варианты x_i,
$m_(y_j)$ – частота появления варианты j
$m_(x_i y)$ – частота появления варианты x_i при заданном значении y,
$m_(y_j x)$ – частота появления варианты y_j при заданном значении x,
n – объём выборки.

Проверка гипотезы о наличии линейной зависимости

Пусть некоторая двумерная генеральная совокупность распределена нормально и из неё извлечена выборка объёма n , для которой найден выборочный коэффициент корреляции

$$r_{\text{в}} = \frac{\overline{xy} - \overline{x} \cdot \overline{y}}{\sigma_X \cdot \sigma_Y}$$

Тогда проверяют нулевую гипотезу H_0 : «$\text{r}_\text{г} = 0$ − линейная зависимость в генеральной совокупности отсутствует» при выдвигаемой альтернативной гипотезе H_1 : «$\text{r}_\text{г} \neq 0$ − линейная зависимость присутствует».

Алгоритм

1. Вычислить статистику $T_{\text{набл}} = \dfrac{r_{\text{в}} \cdot \sqrt{n-2}}{\sqrt{1 - r_{\text{в}}^2}}$,

2. Определить критическое значение распределения Стьюдента (табл. приложения 4) $t_{\text{кр}} = t(\alpha; k = n - 2)$. Критической областью при этом является двусторонняя область) $D = (-\infty; -t_{\text{кр}}) \cup (t_{\text{кр}}; +\infty)$

3. Сделать вывод:
   $T_{\text{набл}} \in D \rightarrow$ гипотезу H_0 отвергают, т. е. имеется линейная зависимость;
   $T_{\text{набл}} \notin D \rightarrow$ гипотезу H_0 принимают, т. е. линейной зависимости нет;

Постановка задачи

Из заданной генеральной совокупности сформировать выборку по второму признаку. Провести статистическую обработку второй выборки в объеме практических работ №1 и №2, с целью определения точечных статистических оценок параметров распределения исследуемого принака (математического ожидания, дисперсии, среднеквадратичного отклонения, асимметрии и эксцесса). Для системы двух случайных величин X (первый признак) и Y(второй признак) сформировать двумерную выборку и найти статистическую оценку коэффициента корреляции, построить доверительный интервал для коэффициента корреляции и осуществить проверку статистической гипотезы о равенстве коэффициента корреляции нулю. Полученные результаты содержательно проинтерпретировать.

Выполнение работы

1.

Проведена статистическая обработка выборки в объёме работ №1 и №2. По результатам были составлены таблицы:

X	Y
[0, 8200)	[0, 27023)
[8200, 16400)	[27023, 54046)
[16400, 24600)	[54046, 81069)
[24600, 32800)	[81069, 108092)
[32800, 41000)	[108092, 135115)
[41000, 49200)	[135115, 162138)
[49200, 57400)	[162138, 189161)
[57400, 65600)	[189161, 216184)
[65600, 73800)	[216184, 243207)

Таблица 1 – Интервалы

	X	Y
Мат ожидание	17112	50225
Дисперсия	141091232	1565675261
СКО	11878	39569
Ассиметрия	1.2775	1.7157
Эксцесс	917.69	1139
Мода	6938	33259
Медиана	15200	39029
Коэф. вариации	1.37	1.24

Таблица 2 – Вычисления

2.Двумерный интервальный ряд

Построен двумерный интервальный ряд. Результат представлен в виде таблицы:

		X								
Y		[0, 8200)	[8200, 16400)	[16400, 24600)	[24600, 32800)	[32800, 41000)	[41000, 49200)	[49200, 57400)	[57400, 65600)	[65600, 73800)
	[0, 27023)	2	3	16	7	2	2	2	0	1
	[27023, 54046)	16	9	8	10	1	0	0	0	0
	[54046, 81069)	5	7	0	2	0	0	0	0	0
	[81069, 108092)	6	6	1	0	0	0	0	0	0
	[108092, 135115)	3	1	0	0	0	0	0	0	0
	[135115, 162138)	1	0	0	0	0	0	0	0	0
	[162138, 189161)	1	1	1	0	0	0	0	0	0
	[189161, 216184)	0	0	0	0	0	0	0	0	0
	[216184, 243207)	0	1	0	0	0	0	0	0	0

Таблица 3 – Двумерный интервальный ряд

3. Кореляционная таблица

По полученному двумерному интервальному вариационному ряду была построена корреляционная таблица.

		X									m
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	
Y	y_1	2	3	16	7	2	2	2	0	1	35
	y_2	16	9	8	10	1	0	0	0	0	44
	y_3	5	7	0	2	0	0	0	0	0	14
	y_4	6	6	1	0	0	0	0	0	0	13
	y_5	3	1	0	0	0	0	0	0	0	4
	y_6	1	0	0	0	0	0	0	0	0	1
	y_7	1	1	1	0	0	0	0	0	0	3
	y_8	0	0	0	0	0	0	0	0	0	0
	y_9	0	1	0	0	0	0	0	0	0	1
m		34	28	26	19	3	2	2	0	1	115

Таблица 4 – Корреляционная таблица

По полученной таблице можно сделать вывод от том, что распределение выборки сильно отличается от нормального распределения, в ней присутствуют слишком удалённые элементы, которые было бы целесообразнее исключить на этапе работы №1.

4.

По корреляционной таблице вычислена статистическая оценка корреляционного момента(ковариация) и коэффициэнт корреляции Пирсона.

$$K^*_{\mathrm{XY}} = 353695795$$
$$r^{\text{Пиросона}}_{\mathrm{XY}} = 0.7525$$

5.

Получен коэффициент коррелиции.

$$r_{\text{в}} = -0.52$$

Значение коэффициэнта отрицательно, значит можно сделать вывод о том, что при увеличении одной переменной вторая убывает и наоборот.

6.

Доверительные интервалы для коэффициента корреляции.

$$r_{\text{в}} \in (-0.66; -0.35) \text{ при } \gamma = 95\%$$
$$r_{\text{в}} \in (-0.72; -0.23) \text{ при } \gamma = 99\%$$

7. Гипотеза о нормальности распределения

Были найдены все необходимые для проверки гипотезы значения

$$T_{\text{набл}} = -6.47$$
$$t_{\text{крит}} = 1.98$$
$$D \in (-\infty; -1.98) \cup (1.98; \infty)$$

$T_{\text{набл}}$ попало в критическую область D, поэтому H_0 отвергается, в исходных данных имеется линейная зависимость

Выводы

В процессе работы мы познакомились корреляционной зависимостью. Научились содержательно интерпретировать полученные значения. Выдвинули статистическую гипотезу и получили её опровержение.

ПРИЛОЖЕНИЕ А

```
import numpy as np
import matplotlib.pyplot as plt
import math
import csv
# import scipy
import random


# import sklearn
# from openpyxl import Workbook

def ReadS():
    with open('Price-Mileage.csv') as csv_file:  # Читаем выборку из файла 0 работы
        s = []
        spam_reader = csv.reader(csv_file, quotechar='|')
        for row in spam_reader:
            x, y = row[0].split(';')
            if y.isdigit() and x.isdigit():
                s.append([int(x), int(y)])
    return s


def find_Xs(s):
    x = 0
    for i in range(n):
        x += s[i]
    return x / len(s)


```

```
def find_Dx(s, x):
    d = 0
    for i in range(n):
        d += (s[i] - x) ** 2
    return d / n


def First_Practice(s):
    s.sort()  # Используя встроенную функцию сортировки получаем ранжированный ряд

    R = s[len(sample) - 1] - s[0]  # Размах
    print("R =", R)

    k = round(1 + math.log2(len(s)))  # Число интерваловФормула( Стёрджеса)
    print("k = ", k)

    h = round(R / k)  # Длина интервала
    print("h =", h)

    k += 1  # Иначе интервалы не покроют выборку

    x0 = s[0] - h / 2  # Начало первого частичного интервала
    if x0 < 0:
        x0 = 0

    print("x0 =", x0)

    interval = []
    x = x0

    for i in range(k):
```

```
        interval.append([(x, x + h), 0, 0])
        x += h

    # Получаем интервальный ряд
    for i in s:
        for j in range(k):
            if interval[j][0][0] < i <= interval[j][0][1]:
                interval[j][1] += 1
                break

    for i in interval:
        i[2] = i[1] / len(sample)

    print(interval)

    middle_int = []
    accum_freq = []
    accum_afreq = []
    accum_freq.append(0)
    accum_afreq.append(0)
    a = 0
    b = 0

    # Вычисляем серидины интервалов и их накопленные частоты
    for i in range(len(interval)):
        a = a + interval[i][2]
        b = b + interval[i][1]
        middle_int.append(interval[i][0][0] + h / 2)
        accum_afreq.append(a)
        accum_freq.append(b)

```

```
  C = middle_int[int(k / 2)]  # Число C из теории, h было вычислено ранее при построении интервального ряда
  con_var = []  # Условные варианты

  for i in range(k):
      con_var.append([int((middle_int[i] - C) / h), interval[i][1], interval[i][2]])

  SEM = []  # Выборочные начальные моменты до 4 порядка
  CEM = []  # Выборочные центральные моменты до 4 порядка

  # Вычисляем условные эмп момент 1 порядка
  for i in range(4):
      SEM.append(0)
      for j in range(k):
          SEM[i] += (con_var[j][0] ** (i + 1)) * con_var[j][2]

  CEM.append(0)  # Центральный момент 1 порядка выборочное( среднее для усл вариант)
  CEM.append(SEM[1] - SEM[0] ** 2)  # 2 порядка выборочная( дисперсия для усл вариант)
  CEM.append(SEM[2] - 3 * SEM[1] * SEM[0] + 2 * SEM[0] ** 3)  # 3 порядка
  CEM.append(SEM[3] - 4 * SEM[0] * SEM[3] + 6 * (SEM[0] ** 2) * SEM[3] - 3 * SEM[0] ** 4)  # 4 порядка

  Xs = 0  # Выборочное среднее по обычной формуле
  CXs = 0  # Выборочное среднее через моменты условных вариант
  Ds = 0  # Дисперсия по обычной формуле
```

```
115   CXs = 0  # Дисперсия через моменты условных
вариант
116
117   for i in range(k):
118       Xs += middle_int[i] * interval[i][2]
119
120   for i in range(k):
121       Ds += (middle_int[i] - Xs) ** 2 * interval[i
][2]
122
123   CXs = SEM[0] * h + C  # Из формулы метода
упрощённых вычислений
124   CDs = CEM[1] * (h ** 2)  # Из формулы метода
упрощённых вычислений
125   ds = math.sqrt(Ds)  # Выборочное СКО
126   cds = math.sqrt(CEM[1])  # Выборочное СКО для усл
вариант
127
128   Asym = CEM[2] / (cds ** 3)  # Коэффициэнт
ассиметрии для условных вариант как( я понимаю это
и есть стат оценка?)
129   Excess = CEM[3] / (cds ** 4)  # Коэффициент
эксцесса для условных вариант
130
131   sDs = Ds * len(sample) / (len(sample) - 1)  #
Исправленная выборочная дисперсия
132   so = math.sqrt(sDs)  # Исправленное выборочное СКО
133
134   Mod = 0  # Частость модального интервала
135   i0 = 0  # Его номер
136   Mod0 = 0  # Истинное значение моды
137
138   for i in range(k):
139       if interval[i][1] > Mod:
```

```
            Mod = interval[i][1]
            i0 = i

    Mod0 = int(interval[i0][0][0] + h * ((interval[i0][2] - interval[i0 - 1][2]) / ((interval[i0][2] - interval[i0 - 1][2]) + (interval[i0][2] - interval[i0 + 1][2]))))

    i0 = 0  # Номер медианного интервала
    Med0 = 0  # Истинное значение медианы

    for i in range(k):
        if accum_afreq[i] > 0.5:
            i0 = i
            break

    px = 0  # Число нужное для линейной интерполяции медианы
    for i in range(i0 - 1):
        px += 1 / interval[i][2]

    # Med0 = interval[i0][0][1] + (h/interval[i0][2])*(0.5 - h*px)
    # При выполнении линейной интерполяции для медианы получилось неадекватно большое отрицательное число
    # Я не знаю с чем это связано поэтому найду её просто как серидину ранжированной ряда

    Med0 = s[int(len(s) / 2)]

    CV = Xs / ds  # Коэффициент вариации

```

```
165     return (interval, middle_int, Asym, Excess, Mod0,
    Med0, CV)
166
167
168 sample = ReadS()
169
170 n = len(sample)
171 k = math.ceil(math.sqrt(n / 2))
172
173 Xs = find_Xs([sample[i][0] for i in range(n)])  #
    Выброчное среднее по x
174 Ys = find_Xs([sample[i][1] for i in range(n)])  # По y
175
176 Dsx = find_Dx([sample[i][0] for i in range(n)], Xs)  #
     Дисперсия по x
177 Dsy = find_Dx([sample[i][1] for i in range(n)], Ys)  #
     По y
178
179 sx = math.sqrt(find_Dx([sample[i][0] for i in range(n)
    ], Xs))  # КОС по x
180 sy = math.sqrt(find_Dx([sample[i][1] for i in range(n)
    ], Ys))  # СКО по y
181
182 intervalX, middle_intX, AsymX, ExcessX, Mod0X, Med0X,
    CVX = First_Practice([sample[i][0] for i in range(n)
    ])
183 intervalY, middle_intY, AsymY, ExcessY, Mod0Y, Med0Y,
    CVY = First_Practice([sample[i][1] for i in range(n)
    ])
184
185 print(f'Xs{Xs}, \nDsx{Dsx} \nsx{sx} \nAsymX {AsymX}, \
    nExcessX {ExcessX}, \nMod0X {Mod0X}, \nMed0X {Med0X
    }, \nCVX {CVX}')
```

```
186 print(f'Ys{Ys}, \nDsx{Dsy} \nsx{sy}\nAsymY {AsymY}, \
    nExcessY {ExcessY}, \nMod0X {Mod0Y}, \nMed0X {Med0Y
    }, \nCVX {CVY}')
187 corr = [[0 for i in range(10)] for i in range(10)]
188 i = 0
189 j = 0
190 l = 0
191
192 while i < len(sample):
193     while j != 9:
194         if intervalX[j][0][0] < sample[i][0] <
    intervalX[j][0][1]:
195             break
196         j += 1
197     while l != 9:
198         if intervalY[l][0][0] < sample[i][1] <
    intervalY[l][0][1]:
199             break
200         l += 1
201     corr[l][j] += 1
202     j = 0
203     l = 0
204     i += 1
205
206 for i in range(10):
207     corr[9][i] = sum(map(lambda x: x[i], corr))
208
209 for i in range(10):
210     corr[i][9] = sum(corr[i])
211
212 print('\n'.join(map(str, corr)))
213
214 Cov = 0
215 for i in range(9):
```

```
216     for j in range(9):
217         Cov += middle_intX[j] * middle_intX[i] - Xs *
      Ys
218
219 Cov /= len(sample)       # Ковариация
220 print("Cov = " + str(Cov))
221
222 Cov /= sx*sy             # Исправленная ковариация
223 print("Cov = " + str(Cov))
224
225 DsX = 0      # межгрупповое КОС св X
226 XY = 0       # межгруповое серднее выборочное
227 for i in range(k):
228     Xyi = 0
229     for j in range(k):
230         XY += corr[i][j]*middle_intX[j]*middle_intY[i]
231         Xyi += corr[i][j]*middle_intX[j]
232     if corr[i][9] != 0:
233         Xyi /= corr[i][9]
234     DsX += ((Xyi-Xs)**2)*corr[i][9]
235
236 XY /= n
237 DsX = math.sqrt(DsX/n)
238 print("Dsx = " + str(Dsx))
239
240 r = (XY - Xs*Ys)/(sx*sy)     # Коэффициэнт корреляции
241 print("r = " + str(r))
242
243 # 95%
244 z = 0.5 * math.log((1 + r) / (1 - r))
245 se = 1 / ((115) ** 0.5)
246 z_l = z - 2.306 * se
247 z_u = z + 2.306 * se
248 z_l = math.tanh(z_l)
```

```
z_u = math.tanh(z_u)
print(f'Доверительный интервал для 95% [{z_l};{z_u}]')

# 99%
z = 0.5 * math.log((1 + r) / (1 - r))
se = 1 / ((99) ** 0.5)
z_l = z -3.355 * se
z_u = z + 3.355 * se
z_l = math.tanh(z_l)
z_u = math.tanh(z_u)
print(f'Доверительный интервал для 99% [{z_l};{z_u}]')

T_emp = (r*math.sqrt(n-2))/math.sqrt(1-r**2)
T_crit = 1.98
print(f'T_emp = {T_emp}')
if abs(T_emp) < T_crit:
    print("H0 верна")
else:
    print("H0 неверна")
```

Листинг 1 – Исходный код программы