Теория Вероятности.
Практическая работа №5.

Группа: P32131
Студент: Смирнов Виктор Игоревич
Вариант: 16

2023

1 Исходный код

1.1 Модуль для вычисления основных статистических показателей

```python
'''
Statistics functions.
'''

from typing import (
    Collection,
    Iterable,
    List,
    NamedTuple,
    Callable,
)
from numbers import Number
from functools import reduce
from math import sqrt, log, ceil, factorial, exp

Probability = Number

Function = Callable[[Number], Number]


class Interval(NamedTuple):
    begin: Number
    end: Number

    @property
    def length(self) -> Number:
        return self.end - self.begin

    @property
    def middle(self) -> Number:
        return self.begin + self.length / 2

    def __str__(self) -> str:
        return f'({self.begin}, {self.end})'


class Bin(NamedTuple):
    interval: Interval
    count: int


class Histogram:
    def __init__(self, bins: List[Bin]):
        self.bin_list = bins

    @property
    def bins(self) -> List[Number]:
        return [self.bin_list[0].interval.begin] + \
            list(map(lambda b: b.interval.end, self.bin_list))

    @property
    def counts(self) -> List[int]:
        return list(map(lambda b: b.count, self.bin_list))


def max(numbers: Collection[Number]) -> Number:
    return reduce(
        lambda a, b: a if a > b else b,
        numbers,
        float('-inf'),
    )


def min(numbers: Collection[Number]) -> Number:
    return reduce(
        lambda a, b: a if a < b else b,
        numbers,
        float('inf'),
    )
```

```python
70
71
72  def scope(numbers: Collection[Number]) -> Interval:
73      return Interval(min(numbers), max(numbers))
74
75
76  def amplitude(numbers: Collection[Number]) -> Number:
77      return scope(numbers).length
78
79
80  def mean(numbers: Collection[Number]) -> Number:
81      return reduce(
82          lambda a, b: a + b,
83          numbers,
84          0,
85      ) / len(numbers)
86
87
88  def variance(numbers: Collection[Number], fixed=False) -> Number:
89      m = mean(numbers)
90      return reduce(
91          lambda a, b: a + (b - m) ** 2,
92          numbers,
93          0,
94      ) / (len(numbers) - (1 if fixed else 0))
95
96
97  def std(numbers: Collection[Number], fixed=False) -> Number:
98      return sqrt(variance(numbers, fixed))
99
100
101 def distinct(numbers: Collection[Number]) -> Collection[Number]:
102     return set(numbers)
103
104
105 def empirical_distribution_function(numbers: Collection[Number]) -> Function:
106     return lambda t: len(list(filter(lambda n: n <= t, numbers))) / len(numbers)
107
108
109 def partition(scope: Interval, n: int) -> Iterable[Interval]:
110     step = scope.length / n
111     current = scope.begin + step
112     while current <= scope.end:
113         yield Interval(current - step, current)
114         current += step
115
116
117 def tabulate(scope: Interval, n: int) -> Iterable[Number]:
118     return map(lambda interval: interval.middle, partition(scope, n))
119
120
121 def sturges_step(numbers: Collection[Number]) -> Number:
122     return (max(numbers) - min(numbers)) / (1 + log(len(numbers)))
123
124
125 def histogram(numbers: Collection[Number], step) -> Histogram:
126     F = empirical_distribution_function(numbers)
127     scope = Interval(min(numbers) - step / 2, max(numbers) + 0.00000001)
128     bins = ceil(scope.length / step)
129     def generate():
130         for interval in partition(scope, bins):
131             count = (F(interval.end) - F(interval.begin)) * len(numbers)
132             yield Bin(interval, count)
133     return Histogram(list(generate()))
```
Листинг 1: Модуль для вычисления основных статистических показателей

1.2 Модуль для отрисовки графиков

```python
1  '''
2  Plotting functions.
3  '''
4
```

```
5  import matplotlib.pyplot as plt
6  from typing import NamedTuple, Collection, Callable, List
7  from numbers import Number
8  from stati import Histogram
9
10 Function = Callable[[Number], Number]
11
12
13 class Point(NamedTuple):
14     x: Number
15     y: Number
16
17
18 class Plot:
19     def __init__(self, title: str = ""):
20         self.figure, self.axes = plt.subplots()
21         self.axes.set_title(title)
22
23     def points(self, points: Collection[Point]):
24         self.axes.plot(
25             list(map(lambda p: p.x, points)),
26             list(map(lambda p: p.y, points))
27         )
28
29     def function(self, x: Collection[Number], f: Function):
30         self.points(list(map(lambda x: Point(x, f(x)), x)))
31
32     def histogram(self, histogram: Histogram):
33         bins = histogram.bins
34         counts = histogram.counts
35         self.axes.hist(bins[:-1], bins, weights=counts)
36
37     def show(self):
38         plt.ion()
39         plt.show()
```

Листинг 2: Модуль для отрисовки графиков

1.3 Скрипт для обработки данных

```
1  '''
2  Probability Theory Assignment 1.
3  Basics.
4  Variant 16.
5  Smirnov @vityaman Victor 2023.
6  (1, 16) - (1, 20), (2, 16) - (2, 20)
7  '''
8
9  from stati import *
10 from ploti import *
11
12 print('Probability Theory Assignment 1 by Smirnov Victor')
13
14 numbers = [
15     -0.45,  0.52, -1.63, -0.42, -1.18,
16      1.42,  0.66, -1.70,  0.17,  0.14,
17      0.83, -0.48, -1.35,  0.31,  0.59,
18      0.73,  0.00,  1.59,  0.17, -0.45
19 ]
20
21 print('Input data:')
22 print(numbers)
23
24 numbers = sorted(numbers)
25 print('Sorted data:')
26 print(numbers)
27
28 distinct_numbers = sorted(distinct(numbers))
29 print('Distinct data:')
30 print(distinct_numbers)
31
32 print(f'Data size:          {len(numbers)}')
33 print(f'Distinct data size: {len(distinct_numbers)}')
```

```
34
35 print(f'Max:                {max(numbers)}')
36 print(f'Min:                {min(numbers)}')
37 print(f'Amplitude:          {amplitude(numbers)}')
38 print(f'Mean:               {mean(numbers)}')
39 print(f'Variance:           {variance(numbers)}')
40 print(f'Standart deviation: {std(numbers)}')
41
42 F = empirical_distribution_function(numbers)
43 plot = Plot('Empirical Distribution Function')
44 plot.function(tabulate(scope(numbers), 1000), F)
45 plot.show()
46
47 hist = histogram(numbers, sturges_step(numbers))
48 plot = Plot('Histogram')
49 plot.histogram(hist)
50 plot.points([Point(bin.interval.middle, bin.count) for bin in hist.bin_list])
51 plot.show()
52
53 input('Press any button to exit...')
```
Листинг 3: Скрипт для обработки данных

2 Результаты работы программы

2.1 Вывод в терминал

```
1 $ python3 lab5.py
2 Probability Theory Assignment 1 by Smirnov Victor
3 Input data:
4 [-0.45, 0.52, -1.63, -0.42, -1.18, 1.42, 0.66, -1.7, 0.17, 0.14, 0.83, -0.48, -1.35,
      0.31, 0.59, 0.73, 0.0, 1.59, 0.17, -0.45]
5 Sorted data:
6 [-1.7, -1.63, -1.35, -1.18, -0.48, -0.45, -0.45, -0.42, 0.0, 0.14, 0.17, 0.17, 0.31,
      0.52, 0.59, 0.66, 0.73, 0.83, 1.42, 1.59]
7 Distinct data:
8 [-1.7, -1.63, -1.35, -1.18, -0.48, -0.45, -0.42, 0.0, 0.14, 0.17, 0.31, 0.52, 0.59,
      0.66, 0.73, 0.83, 1.42, 1.59]
9 Data size:         20
10 Distinct data size: 18
11 Max:               1.59
12 Min:               -1.7
13 Amplitude:         3.29
14 Mean:              -0.026500000000000044
15 Variance:          0.82767275
16 Standart deviation: 0.9097652169653443
```
Листинг 4: Результаты вывода программы
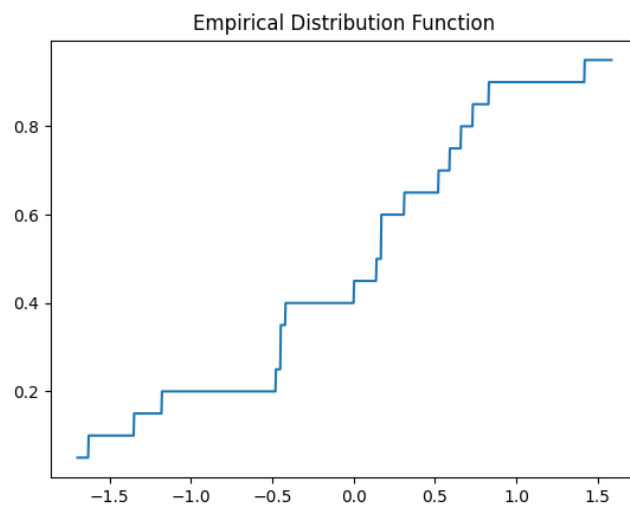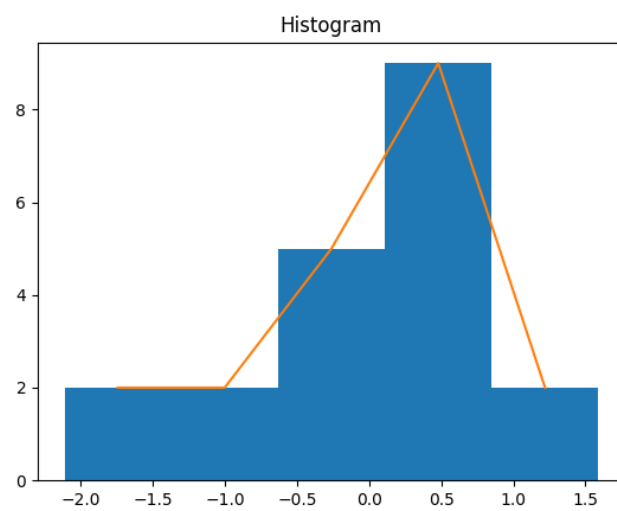
2.2 Построенные графики

Рис. 1: График функции распределения

Рис. 2: График гистограммы распределения