ПРАКТИЧЕСКАЯ ЧАСТЬ 2 МОДУЛЯ КУРСА «МАТЕМАТИЧЕСКАЯ СТАТИСТИКА»

Повторение

1. Основные термины

- Случайная величина
- Событие
- Вероятность
- Распределение вероятностей
- Функция вероятности
- Кумулятивная функция распределения
- Выборка
- Математическое ожидание (мат. ожидание, среднее значение)
- Медиана
- Мода
- Дисперсия

2. Предельные теоремы

- Закон больших чисел (ЗБЧ)
- Центральная предельная теорема (ЦПТ)
- Теорема Муавра-Лапласа (ТМЛ)

3. Методы статистики

- Статистика
- Классификация методов статистки: параметрические, непараметрические
- Метод моментов
- Метод максимального правдоподобия

Краткое описание модуля

- Повторение классификаций распределений: дискретные, абсолютно непрерывные
- Изучение основных дискретных распределений
- Изучение основных абсолютно непрерывных распределений
- Знакомство с модулем scipy для статистики

Дискретные распределения

1. *Равномерное дискретное распределение*

Задача. В корзине 9 красных шариков и один белый. Наугад вынимают по одному шарику до тех пор, пока шар не окажется белым. Красные шарики в корзину не возвращаются. Пусть случайная величина ξ – число красных шариков, покинувших корзину. Найти закон распределения ξ, математическое ожидание, дисперсию и среднеквадратическое отклонение ξ, и вероятности следующих событий: белый шарик извлечен за 5 попыток; не менее чем за 3 попытки; число попыток от 7 до 10 включительно.

Решение.

$p_1 = P(\xi=0)$ = P(сразу вынули белый шар) = 1/10 = 0.1

$p_2 = P(\xi=1)$ = P(сначала вынули красный шар, потом белый) = 9/10 * 1/9 = 0.1

$p_3 = P(\xi=2)$ = P(сначала вынули два красных шара, потом белый) = 9/10 * 8/9 * 1/8 = 0.1

и т.д.

Таким образом, $p_{k+1} = P(\xi=k) = p = 0.1$ → случайная величина ξ принимает целочисленные значения от 0 до 9 с одинаковой вероятностью p = 0.1 → ξ имеет дискретное равномерное распределение с n = 10

Ряд распределения случайной величины ξ:

ξ	0	1	2	…	k	…	9
p	0.1	0.1	0.1	…	0.1	…	0.1

$M\xi = (0+9)/2 = 4.5$ (мат. ожидание)

$D\xi = (100 – 1)/12 = 8.25$ (дисперсия)

$\sigma\xi = 8.25^{0.5} \approx 2.9$ (среднеквадратическое отклонение)

$P(\xi=4) = 0.1$ (вероятность, что белый шарик извлечен за 5 попыток)

$P(\xi\geq2) = 1 – P(\xi<2) = 1 – P(\xi=0) – P(\xi=1) = 1 – 0.2 = 0.8$ (вероятность, что белый шарик извлечен не менее чем за 3 попытки)

$P(6\leq\xi\leq9) = P(\xi=6) + P(\xi=7) + P(\xi=8) + P(\xi=9) = 0.4$ (вероятность, что белый шарик извлечен число попыток от 7 до 10 включительно)

2. *Биномиальное распределение*

Задача 1. Из *m=200* аккумуляторов за год хранения *l=13* выходит из строя. Наудачу выбирают *n=12* аккумуляторов. Определить вероятность того, что среди них *k=4* неисправных. Определить вероятность того, что среди выбранных аккумуляторов неисправно от 1 до 3 включительно.

Решение.

Имеем схему Бернулли с параметрами *p=l/m=0.065* (вероятность того, что аккумулятор выйдет из строя), *n=12* (число испытаний), *k=4* (число «успехов», неисправных аккумуляторов).

Будем использовать формулу Бернулли (вероятность того, что в *n* испытаниях событие произойдет *k* раз):

$$\frac{n!}{k!(n-k)!}p^k q^{n-k}$$

$P_{12}(4) = 12! / (4! * 8!) * 0.065^4 * 0.935^8 \approx 0.00516$

Вероятность того, что неисправно от 1 до 3 включительно означает, что неисправно 1, 2 или 3; тогда:

$P(1\leq\xi\leq3) = P_{12}(1) + P_{12}(2) + P_{12}(3) \approx 0.5478$

Однако эту задачу можно решить, используя Python

```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n, p = 13, 0.065
x = np.arange(0, n, 1)
our_pmf = binom.pmf(x, n-1, p)
s = {}
for i in range(n):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
plt.figure(figsize=(8,5))
x = np.arange(0, n, 0.01)
```

```
plt.scatter(x, binom.cdf(x, n, p))
plt.title('CDF')
plt.grid()
plt.show()
```

Задача 2. Монету подбрасывают 10 раз. Случайная величина ξ – число появления герба в 10 бросаниях монеты. Для случайной величины ξ найти массовую функцию вероятности (PMF) и кумулятивную функцию распределения (CDF), вычислить математическое ожидание и дисперсию.

```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n, p = 11, 0.5
x = np.arange(0, n, 1)
our_pmf = binom.pmf(x, n-1, p)
s = {}
for i in range(n):
    s[i] = round(our_pmf[i], 3)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
plt.figure(figsize=(8,5))
x = np.arange(0, n, 0.01)
plt.scatter(x, binom.cdf(x, n, p))
plt.title('CDF')
plt.grid()
plt.show()
mean, var, skew, kurt = binom.stats(n, p, moments='mvsk')
print(f'Математическое ожидание = {mean}, дисперсия = {var}')
```

3. *Геометрическое распределение*

Задача 1. В корзине 9 красных шариков и один белый. Наугад вынимают по одному шарику до тех пор, пока шар не окажется белым. Красные шарики возвращаются в корзину. Пусть случайная величина ξ – это число извлеченных красных шариков, а η – общее число испытаний. Найти законы распределения и вероятностные характеристики случайных величин и вероятности следующих событий: белый шарик извлекли за 5 попыток; не менее чем за 3 попытки.

Решение.

Случайная величина ξ равна числу “неудач” до первого “успеха”, под “неудачей” понимается извлечение красного шара, под “успехом” – белого. Вероятность извлечь белый шар не зависит от номера попытки, поскольку красные шары возвращаются в корзину, то есть ξ ~ Geom(0.1), где p = 0.1 – вероятность появления белого шарика:

$P(\xi=k) = 0.1 * 0.9^k$.

Тогда:

$M\xi = 9$ (мат. ожидание)

$D\xi = 90$ (дисперсия)

$\sigma\xi \approx 9.5$ (среднеквадратическое отклонение)

Случайная величина η равна общему числу испытаний, которое совпадает с номером первого “успеха”. То есть η ~ Geom(0.1). В свою очередь вероятность появления белого шарика вычисляется по формуле:

$P(\eta=r) = 0.1 * 0.9^{r-1}$

Тогда:

$M\eta = 10$ (мат. ожидание)

$D\eta = 90$ (дисперсия)

$\sigma\eta \approx 9.5$ (среднеквадратическое отклонение)

Вероятность извлечь белый шарик с 5 попытки: $P(\eta=5) = 0.1*0.9^4 = 0.06561$

Вероятность извлечь белый шарик не менее чем за 3 попытки: $P(\eta\geq3) = 1 – P(\eta<3) = 1 – P(\eta=1) – P(\eta=2) = 1 – 0.1 – 0.1*0.9 = 0.81$

Для случайной величины η можно построить модель в Python

```
import numpy as np
```

```
from scipy.stats import geom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n = 20
p = 0.1
x = np.arange(0, n, 1)
our_pmf = geom.pmf(x, p)
s = {}
for i in range(n):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
mean, var, skew, kurt = geom.stats(p, moments='mvsk')
print(f'Математическое ожидание = {mean}, дисперсия = {var}')
```

Задача 2. Студент едет в другой конец города, стоя в переполненном автобусе. Он пытается занять свободное место. Будем считать, что эти попытки не зависят друг от друга, и вероятность успеть к свободному месту постоянна и равна 0.05. Пусть случайная величина ξ – общее число попыток сесть (последняя – удачная). Найдите закон распределения ξ, ее вероятностные характеристики и вероятности следующих событий: студенту повезет на 7-й раз; число неудачных попыток не превышает 2; номер успешной попытки начинается с 5.

Решение.

```
import numpy as np
from scipy.stats import geom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n = 40
p = 0.05
x = np.arange(0, n, 1)
our_pmf = geom.pmf(x, p)
s = {}
for i in range(n):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
mean, var, skew, kurt = geom.stats(p, moments='mvsk')
print(f'Математическое ожидание = {mean}, дисперсия = {round(var, 1)}')
print(f'Вероятность, что повезет на 7 раз = {s[7]}')
print(f'Вероятность, что число неудач не больше 2 = {s[1] + s[2]+ s[3]}')
print(f'Вероятность, что успех будет после 4 попыток = {1 - s[1] - s[2] - s[3] - s[4]}')
```

4. Отрицательное биномиальное распределение (распределение Паскаля)

Задача 1. Спутник сканирует заданную акваторию океана за 4 оборота Земли. Если на каком-либо витке из-за различных помех происходит искажение текущего результата, то оно обнаруживается, и сканирование, выполненное на этом витке, повторяется заново. Найти вероятность того, что всё сканирование будет завершено не более чем за 10 витков, если вероятность искажения результата на одном витке составляет 0.2.

Решение.

В данном случае случайная величина – вероятность k неудачных испытаний вплоть до r-ого успеха (включая этот успех). Поэтому используем распределение Паскаля (а не биномиальное распределение, где случайная величина – это вероятность r успехов в n испытаниях).

Неискажённый результат – «успех», p = 0.8. Тогда числу r будет соответствовать число успешно завершенных витков с вероятность успеха p, то есть 4 оборота. При этом мы должны добиться этого не более, чем за n = 10 витков. Тогда искомая вероятность будет $P(\xi \leq 10-4)$. Используем формулу для $P(\xi=k)$:

$$\frac{(k+r-1)!}{k!(r-1)!}p^r q^k$$

Используем Python

```
import numpy as np
from scipy.stats import nbinom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n, p = 4, 0.8
x = np.arange(0, 12, 1)
our_pmf = nbinom.pmf(x, n, p)
s = {}
for i in range(12):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
print(f'Вероятность, что будет не более 6 неудач (то есть из 10 испытаний 4 хорошие) = {sum(our_pmf[0:7])}')
```

Задача 2. Для медицинского исследования последствий от COVID-19 набирается группа из 20 испытуемых. Каждый человек, с которым исследователь проводит собеседование, имеет 60%-ный шанс на участие в исследовании. Какова вероятность того, что исследователю придется опросить более 40 человек?

Решение.

«Успех» - участие в исследовании, p = 0.6. Тогда числу r будет соответствовать число успешно набранных людей с вероятность успеха p, то есть 20. При этом нам придется опросить более 40 человек. Тогда искомая вероятность будет 1 - $P(\xi \leq 40\text{-}20)$.

```
import numpy as np
from scipy.stats import nbinom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n, p = 20, 0.6
x = np.arange(0, 30, 1)
our_pmf = nbinom.pmf(x, n, p)
s = {}
for i in range(30):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
print(f'Вероятность, что будет более 20 неудач (то есть опросим более 40 человек при нужных 20) = {1 - sum(our_pmf[0:21])}')
```

5. *Гипергеометрическое распределение*

Задача 1. Организация пытается создать группу из 8 человек, обладающих определенными знаниями о производственном процессе. В организации работают 30 сотрудников, обладающих необходимыми знаниями, причем 10 из них работают в конструкторском бюро. Какова вероятность того, что в группу попадут один или два сотрудника из конструкторского бюро, если членов группы выбирают случайно? Объем генеральной совокупности этой задаче N = 30, объем выборки n = 8, а количество успехов D = 10.

Решение.

Используем формулу для гипергеометрического распределения $P(\xi=k)$:

$$\frac{C_D^k C_{N-D}^{n-k}}{C_N^n}$$

В нашем случае k = 1 или k=2, то есть мы ищем $P(\xi=1)+P(\xi=2)$.

Воспользуемся Python.

```
import numpy as np
from scipy.stats import hypergeom
```

```
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
[N, D, n] = [30, 10, 8]
rv = hypergeom(N, D, n)
x = np.arange(0, D+1)
our_pmf = rv.pmf(x)
s = {}
for i in range(D+1):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
print(f'Вероятность, что попадут один или два сотрудника из КБ = {sum(our_pmf[1:3])}')
```

Задача 2. Для исследования в стае из 50 редких птиц окольцевали 20 особей. Через некоторое время отловили 10 птиц. Для случайной величины ξ – числа окольцованных птиц среди отловленных, найти массовую функцию вероятности (PMF) и кумулятивную функцию распределения (CDF), с использованием Matplotlib нарисовать их графики, вычислить мат. ожидание и дисперсию.

```
import numpy as np
from scipy.stats import hypergeom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
[N, D, n] = [50, 20, 10]
rv = hypergeom(N, D, n)
x = np.arange(0, D+1)
our_pmf = rv.pmf(x)
s = {}
for i in range(D+1):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
x = np.arange(0, n+1)
plt.scatter(x, rv.cdf(x))
plt.title('CDF')
plt.grid()
plt.show()
mean, var, skew, kurt = hypergeom.stats(N, D, n, moments='mvsk')
print(f'Математическое ожидание = {mean}, дисперсия = {var}')
```

6. Распределение Пуассона

Задача 1. Студент делает в среднем 1 ошибку на 5 страниц конспекта. Пусть ξ – число ошибок, которое студент сделает в 24-листовой тетради. Найти закон распределения ξ, математическое ожидание, дисперсию, среднеквадратическое отклонение ξ и вероятности следующих событий: в конспекте будет 5 ошибок; не меньше 5 ошибок; от 7 до 10 ошибок включительно.

Решение.

Случайная величина ξ является пуассоновской с параметром λ , т. к. равна числу ошибок, возникающих с постоянной интенсивностью независимо одна от другой.

Вычислим интенсивность появления ошибок в 24-листовой тетради, если в среднем на 5 страницах допускается одна ошибка, а в тетради 48 страниц: λ = 9.6.

Закон распределения P(ξ=k):

$$\frac{\lambda^k}{k!}e^{-\lambda}$$

Вероятностные характеристики ξ:

Mξ = 9.6 (мат. ожидание)

Dξ = 9.6 (дисперсия)

$\sigma\xi = 9.6^{0.5} \approx 3.1$ (среднеквадратическое отклонение)

$P(\xi=5) \approx 0.046$ (вероятность, что в конспекте 5 ошибок)

$P(\xi\geq5) = 1 - P(\xi<5) = 1 - P(\xi=0) - P(\xi=1) - P(\xi=2) - P(\xi=3) - P(\xi=4) \approx 0.9622$ (вероятность, что не меньше 5 ошибок)

$P(7\leq\xi\leq10) = P(\xi=7) + P(\xi=8) + P(\xi=9) + P(\xi=10) \approx 0.4755$ (вероятность, что от 7 до 10 ошибок включительно)

Та же задача в Python:

```
import numpy as np
from scipy.stats import poisson
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
mu = 9.6
x = np.arange(0, 25, 1)
our_pmf = poisson.pmf(x, mu)
s = {}
for i in range(25):
   s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
print(f'Вероятность, что в конспекте 5 ошибок = {s[5]}')
print(f'Вероятность, что в конспекте не меньше 5 ошибок = {1 - s[0] - s[1] - s[2] - s[3] - s[4]}')
print(f'Вероятность, что в конспекте от 7 до 10 ошибок включительно = {s[7] + s[8] + s[9] + s[10]}')
```

Задача 2. На концерте популярного исполнителя аппаратура ломается с вероятностью 3%, и приходится петь вживую. Пусть ξ – число концертов, на которых аппаратура вышла из строя. На год запланировано ровно 100 концертов. Найти вероятность того, что аппаратура выйдет из строя не менее чем на четырех концертах.

Решение.

В задаче рассматривается схема Бернулли с малой вероятностью “успеха” $p = 0.03$, большим числом испытаний $n = 100$ и числом “успехов” $k << n$, где под “успехом” понимается отказ аппаратуры. При заданных условиях вычислить биномиальные коэффициенты и возвести величину $q =1-p$ в большую степень достаточно сложно. В таком случае справедлива теорема:

Если вероятность “успеха” p в каждом испытании постоянна и мала, а число независимых испытаний n достаточно велико, то вероятность наступления “успеха” ровно k раз, k << n , приближенно вычисляется по формуле Пуассона с параметром $\lambda = np$. Поэтому распределение Пуассона иногда называют распределением редких событий.

Тогда в нашем случае $\lambda = 3$.

Вероятность отказа аппаратуры не меньше, чем на 4-х концертах:

$P(\xi\geq4) = 1 - P(\xi<4) = 1 - P(\xi=0) - P(\xi=1) - P(\xi=2) - P(\xi=3) \approx 0.353$

Сравним ответ при решении через биномиальное распределение и через Пуассона в Python

```
import numpy as np
from scipy.stats import poisson
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
mu = 3
x = np.arange(0, 12, 1)
our_pmf = poisson.pmf(x, mu)
s = {}
for i in range(12):
   s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x, our_pmf)
plt.title('PMF')
plt.grid()
plt.show()
y = 1 - s[0] - s[1] - s[2] - s[3]
print(f'Вероятность отказа аппаратуры не меньше, чем на 4-х концертах по Пуассону = {round(y, 4)}')
```

```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
n, p = 100, 0.03
x = np.arange(0, n, 1)
our_pmf = binom.pmf(x, n-1, p)
s = {}
for i in range(12):
    s[i] = round(our_pmf[i], 5)
print(s)
plt.scatter(x[:12], our_pmf[:12])
plt.title('PMF')
plt.grid()
plt.show()
z = 1 - s[0] - s[1] - s[2] - s[3]
print(f'Вероятность отказа аппаратуры не меньше, чем на 4-х концертах по Биномиальному = {round(z, 4)}')

print(abs(1-y/z))
```

В итоге получили относительную погрешность 2%. То есть теорема действительно неплохо работает.

7. Произвольное дискретное распределение

Задача. На переэкзаменовку по теории вероятностей явились 3 студента. Вероятность того, что студент сдаст экзамен, равна 0,8. Найдите ряд распределения случайной величины ξ числа студентов, сдавших экзамен, найдите M(ξ), D(ξ).

Решение.

Введем независимые события:

A1 = (Первый студент сдал экзамен)

A2 = (Второй студент сдал экзамен)

A3 = (Третий студент сдал экзамен)

Вероятности известны: P(A1) = P(A2) = P(A3) = 0.8. Соответственно, вероятности того, что событие не произойдет, равны: P(¬A1) = P(¬A2) = P(¬A3) = 0.2.

p_0 = P(ξ=0) = P(ни один не сдал экзамен) = P(¬A1) * P(¬A2) * P(¬A3) = 0.008

p_1 = P(ξ=1) = P(один из трех сдал) = P(A1) * P(¬A2) * P(¬A3) + P(¬A1) * P(A2) * P(¬A3) + P(¬A1) * P(¬A2) * P(A3) = 0.096

p_2 = P(ξ=2) = P(двое из трех сдали) = P(A1) * P(A2) * P(¬A3) + P(¬A1) * P(A2) * P(A3) + P(A1) * P(¬A2) * P(A3) = 0.384

p_3 = P(ξ=3) = P(все трое сдали) = P(A1) * P(A2) * P(A3) = 0.512

Ряд распределения случайной величины ξ:

ξ	0	1	2	3
p	0.008	0.096	0.384	0.512

$M\xi = 0*p_0 + 1*p_1 + 2*p_2 + 3*p_3 = 2.4$

$D\xi = 0^2*p_0 + 1^2*p_1 + 2^2*p_2 + 3^2*p_3 – 2.4^2 = 0.48$

```
import numpy as np
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
s = {0: 0.008,
     1: 0.096,
     2: 0.384,
     3: 0.512}
plt.scatter(s.keys(), s.values())
plt.title('PMF')
plt.grid()
plt.show()
our_cdf = [0, 0, 0, 0, 0]
our_pmf = [0.008, 0.096, 0.384, 0.512]
```

```
for i in range (1, len(our_pmf)+1):
    our_cdf[i] = our_pmf[i-1] + our_cdf[i-1]
x = np.arange(-0.5, 3.5, 0.001)
F = np.zeros(len(x))
for i in range(len(x)):
    if x[i]<=0: F[i] = our_cdf[0]
    if 0<x[i]<=1: F[i] = our_cdf[1]
    if 1<x[i]<=2: F[i] = our_cdf[2]
    if 2<x[i]<=3: F[i] = our_cdf[3]
    if 3<x[i]: F[i] = our_cdf[4]
plt.scatter(x, F)
plt.title('CDF')
plt.grid()
plt.show()
M = 0
D = 0
for i in range(len(our_pmf)):
    M = M+our_pmf[i]*i
    D = D+our_pmf[i]*i*i
D = D - M*M
print(f'Математическое ожидание = {M}, дисперсия = {round(D, 2)}')
```

8. *Работа с дискретными распределениями в pandas*

```
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.pyplot as plt
S = 10000
x = range(0, S, 1)
r1 = stats.randint.rvs(low = 15, high = 30, size=S)
r2 = stats.binom.rvs(n=20, p=0.3, size=S)
r3 = stats.geom.rvs(p = 0.4, size=S)
r4 = stats.nbinom.rvs(n=25, p=0.6, size=S)
r5 = stats.hypergeom.rvs(M=300, n=100, N=150, size=S)
r6 = stats.poisson.rvs(mu=0.8, size=S)

df = pd.DataFrame()
df = pd.DataFrame()
df['randint'] = r1
df['binom'] = r2
df['geom'] = r3
df['nbinom'] = r4
df['hypergeom'] = r5
df['poisson'] = r6

df.hist(bins=15)

for i in df.columns:
    df[i].hist(bins=15)

sl = {}
for i in df.columns:
    sl[i] = (df[i].mean(), df[i].var())
sl
```