Модели и первичный анализ данных

Лекция 2

09.20.2020

- Статистические показатели на основе сор тированных (ранжированных) данных называются порядковыми статистиками.
- Элементарная мера — это размах, т. е. разница между самым большим и самым малым числом.
- Минимальные и максимальные значения как таковые полезно знать, поскольку они помогают идентифицировать выбросы, но размах чрезвычайно чувствителен к выбросам и не очень полезен в качестве общей меры дисперсности в данных.

В формальном плане процентиль — это средневзвешенное значение:

$$\text{Процентиль } (P) = (1 - w)\, x_{(j)} + w x_{(j+1)}$$

Пример: оценки вариабельности населения штатов

Таблица 1.3. *Несколько строк из кадра* `data.frame` *с данными о численности населения и уровне убийств по каждому штату*

№	Штат	Население	Уровень убийств
1	Alabama	4 779 736	5,7
2	Alaska	710 231	5,6
3	Arizona	6 392 017	4,7
4	Arkansas	2 915 918	5,6
5	California	37 253 956	4,4
6	Colorado	5 029 196	2,8
7	Connecticut	3 574 097	2,4
8	Delaware	897 934	5,8

sd 6848235
IQR 4847308
mad 3849870

Используя встроенные функции для стандартного отклонения (sd), межквартильного размаха (IQR) и медианного абсолютного отклонения из медианы (mad), можно вычислить оценки вариабельности данных о населении штатов

Команды Python

Основные статистические характеристики

amin(a[, axis, out, keepdims, initial])
Минимальное значение элементов массива. Параметр axis позволяет указывать оси, вдоль которых необходим поиск минимальных значений.
amax(a[, axis, out, keepdims, initial])
Максимальное значение в массиве. Параметр axis позволяет указывать оси, вдоль которых необходим поиск максимальных значений.
nanmin(a[, axis, out, keepdims])
Минимальное значение массива или минимальное значение вдоль указанной оси. Элементы с значением *np.nan* игнорируются.
nanmax(a[, axis, out, keepdims])
Максимальное значение массива или максимальное значение вдоль указанной оси. Элементы с значением *np.nan* игнорируются.
numpy.ptp(a[, axis, out, keepdims])
Возвращает диапазон значений ([*max* - *min*]) массива или указанной оси массива.
numpy.percentile(a, q[, axis, out, overwrite_input, interpolation, keepdims])
Вычисляет q-й процентиль (перцентиль) значений элементов массива или элементов вдоль указанной оси.
numpy.nanpercentile(a, q[, axis, out, overwrite_input, interpolation, keepdims])
Вычисление q-го процентиля (перцентиля) значений вдоль указанной оси массива. Элементы с значением *np.nan* игнорируются.
numpy.quantile(a, q[, axis, out, overwrite_input, interpolation, keepdims])
Вычисление q-го процентиля (перцентиля) значений вдоль указанной оси массива. Элементы с значением *np.nan* игнорируются.
numpy.nanquantile(a, q[, axis, out, overwrite_input, interpolation, keepdims])
Вычисление q-го процентиля (перцентиля) значений вдоль указанной оси массива. Элементы с значением *np.nan* игнорируются.

numpy.amin

`numpy.amin(a, axis=None, out=None, keepdims=<no value>, initial=<no value>)`

Функция **amin()** возвращает минимальное значение элементов массива. Параметр axis позволяет указывать оси, вдоль которых необходим поиск минимальных значений.

Является эквивалентной функции np.min().

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *целое число или кортеж целых чисел, необязательный параметр.*

Указывает ось или оси по которым выполняется поиск (доступно в NumPy с версии 1.7.0). По умолчанию axis = None, что соответствует поиску в массиве a так, словно он сжат до одной оси.

out - *массив NumPy, необязательный параметр.*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

keepdims - *True или False, необязательный параметр.*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел.

initial - *число, необязательный параметр.*

Число, которое возвращается, если не найдется меньшее значение. Этот параметр должен обязательно присутствовать, если в вашем коде возможна передача данной функции пустого массива или пустого среза. Доступно в NumPy с версии 1.15.0.

Возвращает:

результат - *число или массив NumPy*

Если параметр axis не указан, то будет возвращено одно число - элемент с минимальным значением в исходном массиве. Если в параметре axis указана одна ось, то будет возвращен массив, содержащий минимальные элементы вдоль указанной оси с формой a.ndim - 1. Если количество указанных осей равно *d*, то будет возвращен массив с формой a.ndim - d.

numpy.nanmin

`numpy.nanmin(a, axis=None, out=None, keepdims=<no value>)`

Функция **nanmin()** возвращает мнимальное значение элементов массива, игнорируя значения np.nan. Параметр axis позволяет указывать оси, вдоль которых необходим поиск минимальных значений.

Если весь массив или некоторый его срез, по которому ведется поиск минимального элемента состоит только лишь из одних элементов np.nan, то запускается предупреждение *RuntimeWarning*, после чего будет возвращено значение np.nan.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *целое число или кортеж целых чисел, необязательный параметр.*

Указывает ось или оси по которым выполняется поиск (доступно в NumPy с версии 1.7.0). По умолчанию axis = None, что соответствует поиску в массиве a так, словно он сжат до одной оси.

out - *массив NumPy, необязательный параметр.*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле. Доступно в NumPy с версии 1.8.0.

keepdims - *True или False, необязательный параметр.*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.8.0.

Возвращает:

результат - *число или массив NumPy*

Если параметр axis не указан, то будет возвращено одно число - элемент с минимальным значением в исходном массиве. Если в параметре axis указана одна ось, то будет возвращен массив, содержащий минимальные элементы вдоль указанной оси с формой a.ndim - 1. Если количество указанных осей равно *d*, то будет возвращен массив с формой a.ndim - d.

numpy.percentile

```
numpy.percentile(a, q, axis=None, out=None, overwrite_input=False, interpolation='linear', keepdims=False)
```

Функция **percentile()** вычисляет q-й процентиль (перцентиль) значений элементов массива или элементов вдоль указанной оси.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

q - *целое положительное число, массив NumPy или подобный массиву объект.*

Процентиль или последовательность процентилей. Допустимые значения находятся в интервале *[0, 100]*..

axis - *целое число или кортеж целых чисел, необязательный параметр.*

Указывает ось или оси по которым выполняется вычисление (доступно в NumPy с версии 1.9.0). По умолчанию `axis = None`, что соответствует вычислению процентиля так, словно `a` сжат до одной оси.

out - *массив NumPy, необязательный параметр.*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

overwrite_input - *True или False, необязательный параметр.*

Значение *True* позволяет использовать входной массив `a` для промежуточных вычислей, что позволяет сэкономит память но приводит к потере данных.

interpolation - *{'linear', 'lower', 'higher', 'midpoint', 'nearest'}, необязательный параметр.*

Если требуемый процентиль находится между двумя значениями элементов входного массива, то данный параметр позволяет задать метод интерполяции для расчета конечного значения. Обозначим два элемента, между которыми находится значение процентиля, как *i* и *j*, причем *i* < *j*, тогда методы интерполяции будут задаваться следующими правилами:

- *'linear'* - *i + (j - i) * fract*, где *frac* является дробной частью процентиля.
- *'lower'* - возвращается *i*.
- *'higher'* - возвращается *j*.
- *'midpoint'* - к *p* значение.
- *'nearest'* - среднее значение между *i* и *j*.

keepdims - *True или False, необязательный параметр.*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Если параметр `q` - это одно число, то результатом так же будет одно число. Если параметр `q` - это массив из нескольких процентилей, то будет возвращен массив аналогичной длинны. Если входной массив имеет несколько осей, указано несколько процентилей и указана ось (или указано несколько осей) в параметре `axis`, то будет возвращен массив, по первой оси которого расположены значения (массивы значения) для каждого указанного в `q` процентиля.

numpy.quantile

```
numpy.quantile(a, q, axis=None, out=None, overwrite_input=False, interpolation='linear', keepdims=False)
```

Функция **quantile()** вычисляет q-й квантиль значений элементов массива или элементов вдоль указанной оси.

Единственное отличие квантиля от процентиля - диапазон значений параметра q принимает значения в интервале *[0, 1]*.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

q - *вещественное число, массив NumPy или подобный массиву объект.*

Квантиль или последовательность квантилей. Допустимые значения находятся в интервале *[0, 1].*.

axis - *целое число или кортеж целых чисел, необязательный параметр.*

Указывает ось или оси по которым выполняется вычисление. По умолчанию axis = None, что соответствует вычислению квантиля так, словно a сжат до одной оси.

out - *массив NumPy, необязательный параметр.*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

overwrite_input - *True или False, необязательный параметр.*

Значение *True* позволяет использовать входной массив a для промежуточных вычислей, что позволяет сэкономит память но приводит к потере данных.

interpolation - *{'linear', 'lower', 'higher', 'midpoint', 'nearest'}, необязательный параметр.*

Если требуемый квантиль находится между двумя значениями элементов входного массива, то данный параметр позволяет задать метод интерполяции для расчета конечного значения. Обозначим два элемента, между которыми находится значение квантиля, как *i* и *j*, причем *i* < *j*, тогда методы интерполяции будут задаваться следующими правилами:

- *'linear'* - *i + (j - i) * q*.
- *'lower'* - возвращается *i*.
- *'higher'* - возвращается *j*.
- *'midpoint'* - к *p* значение.
- *'nearest'* - среднее значение между *i* и *j*.

keepdims - *True или False, необязательный параметр.*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Если параметр q - это одно число, то результатом так же будет одно число. Если параметр q - это массив из нескольких квантилей, то будет возвращен массив аналогичной длинны. Если входной массив имеет несколько осей, указано несколько квантилей и указана ось (или указано несколько осей) в параметре axis, то будет возвращен массив, по первой оси которого расположены значения (массивы значения) для каждого указанного в q квантиля.

numpy.median

`numpy.median(a, axis=None, out=None, overwrite_input=False, keepdims=False)`

Функция **median()** вычисляет медиану элементов массива.

Медиана - это такое значение, что ровно половина элементов массива окажется меньше него, а друга больше.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *число, кортеж целых чисел или None (необязательный параметр).*

Позволяет задать ось или несколько осей вдоль которых вычисляются медианы. По умолчанию установлено значение *None*, что соответствует вычислению медианы всех элементов массива, так словно он сжат до одной оси.

out - *массив NumPy (необязательный параметр).*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

overwrite_input - *True или False (необязательный параметр).*

Значение *True* позволяет использовать входной массив a для промежуточных вычислей, что позволяет сэкономит память но приводит к потере данных. Если установлен в *True* и `a` не является массивом NumPy, то возникнет ошибка.

keepdims - *True или False (необязательный параметр).*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Медиану значений элементов массива в виде числа для одномерного массива, или многомерного если параметр `axis = None`. Для многомерного массива возвращает массив NumPy если в параметре `axis` оси указаны.

numpy.average

`numpy.average(a, axis=None, weights=None, returned=False)`

Функция **average()** вычисляет среднее арифметическое взвешенное значений элементов массива.

Средневзвешенное чисел a1,…,ana1,…,an, с весами w1,…,wnw1,…,wn записывается как:

$$\bar{a} = \frac{\sum_{i=1}^{n} w_i \cdot a_i}{\sum_{i=1}^{n} w_i}$$

Сумма весов в знаменателе *не может* быть равна 0, но *некоторые веса* могут быть равны 0. Если сумма весов все же оказалась равна 0, то возникает ошибка *ZeroDivisionError*.

В случае одинаковых весов, мы получим обычное среднее арифметическое, которое можно вычислить функцией numpy.mean().

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *число, кортеж целых чисел или None (необязательный параметр).*

Позволяет задать ось или несколько осей вдоль которых вычисляются медианы. По умолчанию установлено значение *None*, что соответствует вычислению медианы всех элементов массива, так словно он сжат до одной оси.

weights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив весов каждого элемента в a. Данный массив может быть одномерным, но в этом случае, его длинна должна быть равна длине указанной оси исходного массива, или, он может иметь туже форму, что и входной массив. Если weights = None то все веса предполагаются равными 1. В случае несовпадения размеров weights = None и a или невозможности выполнить транслирование возникает ошибка *TypeError*.

returned - *True или False (необязательный параметр).*

Если установлено значение *True*, то возвращается кортеж в котором первый элемент - средневзвешенное, второй - сумма весов. По умолчанию установлено значение *False*.

Возвращает:

результат - *массив NumPy или число*

Средневзвешенное значений элементов массива в виде числа для одномерного массива, или многомерного если параметр axis = None. Для многомерного массива возвращает массив NumPy если в параметре axis оси указаны.

numpy.mean

```
numpy.mean(a, axis=None, dtype=None, out=None, keepdims=<no value>)
```

Функция **mean()** вычисляет среднее арифметическое значений элементов массива.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *число, кортеж целых чисел или None (необязательный параметр).*

Позволяет задать ось или несколько осей вдоль которых вычисляется среднее арифметическое. По умолчанию установлено значение *None*, что соответствует вычислению среднего арифметического всех элементов массива, так словно он сжат до одной оси.

dtype - *тип данных NumPy (необязательный параметр).*

По умолчанию равен *None*, что означает использование для вычисления среднего арифметического целых чисел типа *float64*, а для чисел с плавающей точкой тип данных будет совпадать с типом данных входного массива.

out - *массив NumPy (необязательный параметр).*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

keepdims - *True или False (необязательный параметр).*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Среднее арифметическое значений элементов массива в виде числа для одномерного массива, или многомерного если параметр `axis = None`. Для многомерного массива возвращает массив NumPy если в параметре `axis` оси указаны.

numpy.std

```
numpy.std(a, axis=None, dtype=None, out=None, ddof=0, keepdims=<no value>)
```

Функция **std()** вычисляет среднеквадратичное (стандартное) отклонение значений элементов массива.

Среднеквадратичное отклонение чисел a1,…,ana1,…,an записывается как::

$$S = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(a_i - \bar{a})^2}$$

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *число, кортеж целых чисел или None (необязательный параметр).*

Позволяет задать ось или несколько осей вдоль которых вычисляется среднеквадратичное отклонение. По умолчанию установлено значение *None*, что соответствует вычислению среднеквадратичного всех элементов массива, так словно он сжат до одной оси.

dtype - *тип данных NumPy (необязательный параметр).*

По умолчанию равен *None*, что означает использование для вычисления среднеквадратичного целых чисел типа *float64*, а для чисел с плавающей точкой тип данных будет совпадать с типом данных входного массива.

out - *массив NumPy (необязательный параметр).*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

ddof - *вещественное число (необязательный параметр).*

Параметр *ddof* - дельта степени свободы. Обычно, принято (и по умолчанию установлено) считать ddof = 0, но в разных источниках (например ГОСТ) требуется что бы параметр *ddof* имел значение отличное от 0, например 1 или 1,5.

keepdims - *True или False (необязательный параметр).*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Среднеквадратичное (стандартное) отклонение значений элементов массива в виде числа для одномерного массива, или многомерного если параметр axis = None. Для многомерного массива возвращает массив NumPy если в параметре axis оси указаны.

numpy.var

`numpy.var(a, axis=None, dtype=None, out=None, ddof=0, keepdims=<no value>)`

Функция **var()** вычисляет дисперсию значений элементов массива. Дисперсия случайной величины *A* записывается как:

$$D[X] = M\left[(X - M[X])^2\right]$$

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные.

axis - *число, кортеж целых чисел или None (необязательный параметр).*

Позволяет задать ось или несколько осей вдоль которых вычисляется дисперсия. По умолчанию установлено значение *None*, что соответствует вычислению дисперсии всех элементов массива, так словно он сжат до одной оси.

dtype - *тип данных NumPy (необязательный параметр).*

По умолчанию равен *None*, что означает использование для вычисления дисперсии целых чисел типа *float64*, а для чисел с плавающей точкой тип данных будет совпадать с типом данных входного массива.

out - *массив NumPy (необязательный параметр).*

Массив в который можно поместить результат функции. Данный массив должен соответствовать форме и типу данных результирующего массива функции (зачастую, тип данных может быть преобразован автоматически). Указание данного параметра, позволяет избежать лишней операции присваивания тем самым немного ускоряя работу вашего кода. Полезный параметр если вы очень часто обращаетесь к функции в цикле.

ddof - *вещественное число (необязательный параметр).*

Параметр *ddof* - дельта степени свободы. Обычно, принято (и по умолчанию установлено) считать `ddof = 0`, но в разных источниках (например ГОСТ) требуется что бы параметр *ddof* имел значение отличное от 0, например 1 или 1,5.

keepdims - *True или False (необязательный параметр).*

Если данный параметр указан как *True*, то результат работы функции по указанным осям будет способен к транслированию по исходному массиву, т.е. результат функции оформляется в массив с количеством осей исходного массива. Если параметр установлен в значение *False*, то результатом работы функции будет либо число, либо одномерный массив чисел. Доступно в NumPy с версии 1.9.0.

Возвращает:

результат - *массив NumPy или число*

Дисперсию значений элементов массива в виде числа для одномерного массива, или многомерного если параметр `axis = None`. Для многомерного массива возвращает массив NumPy если в параметре `axis` оси указаны.

Корреляции

corrcoef(x[, y, rowvar, bias, ddof])
Коэфициент корреляции Пирсона.
correlate(a, v[, mode])
Взаимнокорреляционная функция двух одномерных последовательностей.

---

cov(m[, y, rowvar, bias, ddof, fweights, …])
Ковариационная матрица.

Гистограммы

histogram(a[, bins, range, normed, weights, …])
Вычисление гистограммы набора данных.
histogram2d(x, y[, bins, range, normed, weights])
Вычисление двумерной гистограммы двух наборов данных.
histogramdd(sample[, bins, range, normed, …])
Вычисление N-мерной гистограммы N-го количества наборов данных.
bincount(x[, weights, minlength])
Количество вхождений значений в массиве.
numpy.histogram_bin_edges(a, bins=10, range=None, weights=None)
Значения для прямоугольников гистограммы.
digitize(x, bins[, right])
Вычисление индексов числовых интервалов массива *bins* в которые входит каждое последующее значение элемента массива *x*.

numpy.corrcoef

`numpy.corrcoef(x, y=None, rowvar=True, bias=<no value>, ddof=<no value>)`

Функция **corrcoef()** вычисляет коэфициент корреляции Пирсона (линейный коэфициент корреляции).
Данный коэфициент вычисляется по формуле:

$$R_{XY} = \frac{C_{XY}}{\sigma_X \sigma_Y} = \frac{\sum(X - \bar{X})(Y - \bar{Y})}{\sqrt{\sum(X - \bar{X})^2 \sum(Y - \bar{Y})^2}}$$

Где C_{XY} - ковариационная матрица, а $\overline{X} = \frac{1}{n}\sum_{t=1}^{n} X_t$ и $\overline{Y} = \frac{1}{n}\sum_{t=1}^{n} Y_t$ это средние значения выборок.

Параметры:

x - *массив NumPy или подобный массиву объект.*

Входные данные в виде одномерного или двумерного массива, содержащего несколько выборок (случайных величин). Каждая строка в этом массиве представляет собой отдельную выборку (случайную переменную), а столбец отдельное наблюдение в каждой выборке.

y - *массив NumPy или подобный массиву объект (необязательный параметр).*

Дополнительный набор выборок, который имеет ту же форму что и *m*.

rowvar - *True или False (необязательный параметр).*

Если rowvar = True (по умолчанию это так), то каждой строке соответствует определенная выборка, а столбцу определенное наблюдение из этой выборки. Если параметр установлен в значение *False*, выполняется транспонирование массива, т.е. каждый столбец начинает соответсвовать выборкам, а каждая строка соответствующим наблюдениям из этих выборок.

bias - *<no value>.*

Начиная с версии 1.10.0 считается устаревшим и не используется.

ddof - *<no value>.*

Начиная с версии 1.10.0 считается устаревшим и не используется.

Возвращает:

результат - *массив NumPy или число*

Матрица корреляционных коэфициентов для указанной выборки наблюдений.

numpy.correlate

```
numpy.correlate(a, v, mode='valid')
```

Функция **correlate()** вычисляет значения взаимнокорреляционной функции (кросс-корреляцию) двух одномерных последовательностей.

Данная функция часто используется для поиска некоторой короткой последовательности в другой длинной последовательности. Для двух дискретных рядов *a* и *v* взаимная корреляция определяется по формуле:

$$C_{av} = \sum_{n=0}^{n} a_{n+k} \cdot v_n^*$$

Где *k* - это сдвиг между последовательностями относительно друг друга, а V∗V∗ означает комплексное сопряжение VV.

Параметры:

a, v - *массивы NumPy или подобные массивам объекты.*

Входные одномерные последовательности. В случае необходимости, данные последовательности могут дополняться нулями.

mode - *{'full', 'valid', 'same'} (необязательный параметр).*

Определяет режим вычисления линейной свертки:

- **'full'** - В данном режиме функция вычисляет свертку в каждой точке перекрытия, а длина выходного массива будет равна *(N + M - 1,)*. На краях свертки массивы могут перекрываться не полностью, что вызывает граничные эффекты;
- **'valid'** - Режим установленный по умолчанию. Возвращает массив длинны max(N, M), при этом так же наблюдаются граничные эффекты;
- **'same'** - Свертка вычисляетя только в точках с перекрытием сигналов, поэтому граничные эффекты не возникают. Длина результирующего массива определяется как max(N,M) - min(N, M) + 1.

Возвращает:

результат - *массив NumPy или число*

Кросс-корреляция двух одномерных функций *a* и *v*.

numpy.cov

```
numpy.cov(m, y=None, rowvar=True, bias=False, ddof=None, fweights=None, aweights=None)
```

Функция **cov()** вычисляет ковариационную матрицу.

Ковариация двух выборок (двух случайных величин) - это мера их линейной зависимости, которая определяется следующим образом:

$$cov(X,Y)=M[(X-MX)(Y-MY)]cov(X,Y)=M[(X-MX)(Y-MY)]$$

Где MM - математическое ожидание.

Если мы рассмотрим *n*-мерную выборку $X=[x_1,x_2,...,x_n]^TX=[x1,x2,...,xn]T$, то элемент ковариационной матрицы $C_{ij}Cij$ для данной выборки будет представлять собой ковариацию двух соответствующих выборок $C_{ij}=cov(x_i,x_j)Cij=cov(xi,xj)$. Диагональные элементы данной матрицы $C_{ii}Cii$ будут представлять собой дисперсии соответствующей выборки x_ixi.

Параметры:

m - *массив NumPy или подобный массиву объект.*

Входные данные в виде одномерного или двумерного массива, содержащего несколько выборок (случайных величин). Каждая строка в этом массиве представляет собой отдельную выборку (случайную переменную), а столбец отдельное наблюдение в каждой выборке.

y - *массив NumPy или подобный массиву объект (необязательный параметр).* Дополнительный набор выборок, который имеет ту же форму что и *m*.

rowvar - *True или False (необязательный параметр).*

Если `rowvar = True` (по умолчанию это так), то каждой строке соответствует определенная выборка, а столбцу определенное наблюдение из этой выборки. Если параметр установлен в значение *False*, выполняется транспонирование массива, т.е. каждый столбец начинает соответсвовать выборкам, а каждая строка соответствующим наблюдениям из этих выборок.

bias - *True или False (необязательный параметр).*

Определяет используемую нормализацию. *False* - *n - 1* нормализация (установлено по умолчанию). *True* - *n* нормализация, где *n* - это количество данных в выборке. Данный параметр может быть переопределен с помощью параметра `ddof` в NumPy начиная с версии 1.5.0.

ddof - *вещественное число (необязательный параметр).*

Параметр *ddof* - дельта степени свободы. Обычно, принято (и по умолчанию установлено) считать `ddof = 0`, но в разных источниках (например ГОСТ) требуется что бы параметр *ddof* имел значение отличное от 0, например 1 или 1,5.

Если установлен в отличное от *None* значение, то это приводит к переопределению параметра `bias`. Так, например, если `ddof = 1`, то будет возвращена несмещенная оценка, даже несмотря на установленные параметры `fweights` и `aweights`. А если `ddof = 0`, то это приведет к вычислению простого среднего арифметического.

fweights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Одномерный массив целых чисел - частотных весов, указывающих количество повторения каждого вектора наблюдений. По умолчанию установлен в значение *None*. Доступно в NumPy с версии 1.10.0.

aweights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Одномерный массив весовых коэфициентов, указывающих "важность" отдельных наблюдений. Если `ddof = 0`, то данный массив позволяет определить вероятности отдельных векторов. По умолчанию установлен в значение *None*. Доступно в NumPy с версии 1.10.0.

Возвращает: результат - *массив NumPy* Ковариационная матрица, указанной выборки наблюдений.

numpy.histogram

`numpy.histogram(a, bins=10, range=None, normed=None, weights=None, density=None)`

Функция **histogram()** вычисляет гистограмму набора данных.

В данном случае речь идет об одномерной гистограме, которая позволяет ответить на вопрос о количестве вхождений значений элементов массива в определенные числовые интервалы. Например, у нас есть последовательность чисел [1, 2, 1, 4, 1, 3] гистограмма этих чисел для двух интервалов [1, 3) и [3, 4] будет равна [4, 2].

Данная функция часто используется в статистике при анализе случайных рядов, так как позволяет проанализировать распределение плотности вероятности значений, что может помоч в установлении вида функции вероятности.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные. Многомерные массивы сжимаются до одной оси.

bins - *целое число, последовательность целых чисел или строка (необязательный параметр).*

Если указано целое число, то оно определяет количество интервалов равной ширины всех ячеек (по умолчанию 10 ячеек). Если указана последовательность целых чисел, то границы интервалов определяют соседние числа в данной последовательности.

В NumPy начиная с версии 1.11.0 в качестве данного параметра можно указывать строку с названием метода расчета оптимальной ширины ячеек. Подробнее см. в histogram_bin_edges().

range - *(float_1, float_2) (необязательный параметр).*

Определяет минимальное и максимальное значение ширины ячеек, при этом значения выходящие за пределы диапазона игнорируются. Если range = None то границы определяются интервалом (a.min(), a.max()). Должно выполняться условие float_1 < float_2.

Если в параметре bins указана строка с методом расчета ширины, то значения в range повлияют на эти вычисления. В этом случае по прежнему будет вычисляться оптимальная ширина ячеек, но только на основе тех данных, которые находятся в пределах указанного интервала. В любом случае, количество ячеек будет заполнять весь интервал, даже в тех участках, которые не содержат никаких данных.

normed - *True или False (необязательный параметр).*

Считается устаревшим начиная с версии 1.6.0.

weights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив весовых коэфициентов той же формы что и a. Каждое значение в a добавляет величину к соответствующей ячейке в соответствии с указанным весом (который по умолчанию считается равным 1). Если параметр density = True, то все коэфициенты в weights нормализуются так, что интеграл плотности по диапазону range будет равен 1.

density - *True или False (необязательный параметр).*

Если density = True, то результатом вычислений окажется функция (точнее ее значения) плотности вероятности, а интеграл по диапазону значений range будет равен 1. Однако, интеграл не будет равен 1 если ширина ячеек не равна 1. Не путайте данную *функцию плотности вероятности* с *функцией вероятности*.

Возвращает: **hist** - *массив NumPy* Значения гистограммы. **bin_edges** - *массив NumPy* Массив с границами каждой ячейки. len(bin_edges) = len(hist) + 1.

numpy.histogram2d

`numpy.histogram2d(x, y, bins=10, range=None, normed=None, weights=None, density=None)`

Функция **histogram2d()** вычисляет двумерную гистограмму двух наборов данных.

Наборы x и y должны быть одинаковой длинны, и по сути выступают в роли единого набора координат точек на плоскости. А функция **histogram2d()**, по сути, просто разбивает плоскость на прямоугольные области и занимается подсчетом вхождений точек в каждую из них.

Параметры:

x - *массив NumPy или подобный массиву объект.* Входные данные в виде одномерного массива.

y - *массив NumPy или подобный массиву объект.* Входные данные в виде одномерного массива, той же длинны что и x.

bins - *целое число, последовательность целых чисел или строка (необязательный параметр), список из двух целых чисел или список из двух массивов или список из одного целого числа и массива (необязательный параметр).*

- Если указано одно число, то оно определяет одинаковое количество интервалов одинаковой ширины сразу по двум осям, т.е. nx = ny = bins;
- Если указан массив то его значения задают идентичные границы интервалов сразу по двум осям x_edges = y_edges = bins. В данном случае можно задать длины интервалов разной ширины;
- Если bins = [int_1, int_2], то nx = int_1, а ny = int_2;
- Если bins = [array_1, array_2], то x_edges = array_1 а y_edges = array_2, т.е. плоскость может разбиваться на прямоугольные ячейки произвольного размера;
- Если bins = [int, array] или bins = [array, int], то одна ось разбивается на интервалы произвольной ширины а другая на интервалы равной ширины.

range - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив с размером *(2, 2)* вида [[xmin, xmax], [ymin, ymax]], который определяет допустимые крайние значения по двум осям. Все значения из массивов x и y, которые не входят в данный интервал будут считаться недопустимыми (считаться выбросами) и не учитываться при расчете гистограммы. Данный параметр может не указываться, если границы заданы явно в параметре bins.

density - *True или False (необязательный параметр).*

Если density = True, то результатом вычислений окажется функция (точнее ее значения) плотности вероятности - bin_count / sample_count / bin_area, а интеграл по диапазону значений range будет равен 1. Однако, интеграл не будет равен 1 если ширина ячеек не равна 1. Не путайте данную *функцию плотности вероятности* с *функцией вероятности*. Если density = False, товозвращается обычное количество вхождений значений в каждую ячейку.

normed - *True или False (необязательный параметр).*

Является псевдонимом для density и считается устаревшим начиная с версии 1.6.0.

weights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив весовых коэфициентов той же длины, что и x. Каждое значение в weights добавляет соответствующий вес к паре соответствующих элементов из x и y. Если параметр density = True, то все коэфициенты в weights нормализуются так, что интеграл плотности по диапазону range будет равен 1.

Возвращает:

hist - *массив NumPy* Значения двумерной гистограммы для двух наборов данных x и y. **x_edges** - *массив NumPy* Массив с границами каждой ячейки по первому измерению len(x_edges) = len(x) + 1. **y_edges** - *массив NumPy* Массив с границами каждой ячейки по второму измерению. len(y_edges) = len(y) + 1.

numpy.bincount

`numpy.bincount(x, weights=None, minlength=0)`

Функция **bincount()** возвращает количество вхождений значений в массиве.

Данная функция выполняет подсчет только целых положительных чисел. Если на вход подан массив с числами типа *float* или *complex*, то будет вызвано исключение *TypeError*.

Параметры:

x - *массив NumPy или подобный массиву объект.*

Одномерная последовательность целых положительных чисел. В случае недопустимых значений параметра вызывается исключение *ValueError*.

weights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив, который имеет ту же длину что и x, и, позволяет задать весовые коэфициенты каждого его элемента.

minlength - *целое положительное число или 0 (необязательный параметр).*

Позволяет задать минимальное значение длины выходного массива. Если указано недопустимое значение, то будет вызвано исключение *ValueError*.

Возвращает:

результат - *массив NumPy*

Массив положительных чисел, указывающих на количество вхождений значений его индекса в исходный массив. Длина выходного массива равна np.amax(x) + 1 Если в параметре weights, указаные вещественные веса, то выходной массив будет так же состоять из вещественных чисел.

numpy.histogram_bin_edges

`numpy.histogram_bin_edges(a, bins=10, range=None, weights=None)`

Функция **histogram_bin_edges()** вычисляет границы интервалов для ячеек гистограммы.

Параметры:

a - *массив NumPy или подобный массиву объект.*

Входные данные. Многомерные массивы сжимаются до одной оси.

bins - *целое число, последовательность целых чисел или строка (необязательный параметр).*

Если указано целое число, то оно определяет количество интервалов равной ширины всех ячеек (по умолчанию 10 ячеек). Если указана последовательность целых чисел, то границы интервалов определяют соседние числа в данной последовательности.

В NumPy начиная с версии 1.11.0 в качестве данного параметра можно указывать строку с названием метода расчета оптимальной ширины ячеек.

- **'auto'** - максимальные значения *'sturges'* и *'fd'*. Обеспечивает наилучшую производительность;
- **'fd'** - метод оценки Фридмана-Диакониса, который учитывает размер данных и их изменчивость. Устойчив к выбросам и считается наиболее надежным;
- **'doane'** - улучшенная версия *'sturges'*, которая лучше всего подходит для данных с ненормальным распределением значений;
- **'scott'** - метод, который учитывает изменчивость и размер данных, но является менее надежным;
- **'rice'** - учитывает только размер данных и обычно переоценивает количество необходимых ячеек;
- **'sturges'** - *R*-метод, который учитывает только размер данных. Оптимален только для данных с гаусовым распределением значений. Для больших негаусовых данных недооценивается необходимое количество ячеек;
- **'sqrt'** - метод на основе квадратного корня от размера данных, самый быстрый и простой метод, но может подойти не для всех данных.

range - *(float_1, float_2) (необязательный параметр).*

Определяет минимальное и максимальное значение ширины ячеек, при этом значения выходящие за пределы диапазона игнорируются. Если `range = None` то границы определяются интервалом `(a.min(), a.max())`. Должно выполняться условие `float_1 < float_2`.

Если в параметре `bins` указана строка с методом расчета ширины, то значения в `range` повлияют на эти вычисления. В этом случае по прежнему будет вычисляться оптимальная ширина ячеек, но только на основе тех данных, которые находятся в пределах указанного интервала. В любом случае, количество ячеек будет заполнять весь интервал, даже в тех участках, которые не содержат никаких данных.

weights - *массив NumPy или подобный массиву объект (необязательный параметр).*

Массив весовых коэфициентов той же формы что и `a`. Каждое значение в `a` добавляет величину к соответствующей ячейке в соответствии с указанным весом (который по умолчанию считается равным 1). В настоящее время данный параметр не используется при вычислении, но планируется его введение в будущих версиях.

Возвращает:

результат - *массив NumPy*

Массив с границами ячеек для подсчета одномерной гистограммы исходных данных.

numpy.digitize

`numpy.digitize(x, bins, right=False)`

Функция **digitize()** возвращает индексы числовых интервалов в которые входит каждое значение элементов массива.

Параметры:

x - *массив NumPy или подобный массиву объект.*

Входные данные. Многомерные массивы сжимаются до одной оси.

bins - *массив NumPy или подобный массиву объект.*

Одномерный массив, соседние значения которого задают границы полуоткрытых интервалов. Значения должны быть возрастающими. Если значения массива x выходят за границы интервалов, то в зависимости от ситуации будет возвращен 0 или len(x).

right - *True или False (необязательный параметр).*

Указанные в параметре bins интервалы являются полуоткрытыми, а параметр right позволяет указать какой его край является включенным в него. right = False - не включает правое значение и включает левое; right = True - наоборот. Однако поведение для убывающих и возрастающих интервалов может отличаться:

- если возрастающие интервалы и если right = False, то bins[i-1] <= x < bins[i];
- если возрастающие интервалы и если right = True, то bins[i-1] < x <= bins[i];
- если убывающие интервалы и если right = False, то bins[i-1] > x >= bins[i];
- если убывающие интервалы и если right = True, то bins[i-1] >= x > bins[i].

Возвращает:

результат - *массив NumPy*

Массив индексов интервалов той же формы что и x.