

Python для анализа данных

Лекция 3.

План

- Генеральная совокупность, выборка, репрезентативность. Частотные таблицы и их визуализация.
- Группировка данных в pandas
- Базовая визуализация в pandas
- Описательные статистики: меры центральной тенденции и разброса
- Выбросы

Генеральная совокупность и выборка



Генеральная совокупность — это совокупность всех объектов, которые представляют интерес в конкретном исследовании.

Выборка — это группа объектов, отобранных из генеральной совокупности для исследования.

Репрезентативность

Репрезентативность —
соответствие
характеристик выборки
характеристикам
генеральной совокупности.



Частотные распределения

Пусть у нас есть 25 школьников, которые написали контрольную по программированию и получили разные оценки от 2 до 5:

3,2,3,3,5,4,3,2,4,2,3,4,5,2,4,3,4,5,5,4,3,3,2,3,3

Значение	Частота	Доля	Процент
2	5	0.2	20%
3	10	0.4	40%
4	6	0.24	24%
5	4	0.16	16%

Частотные распределения

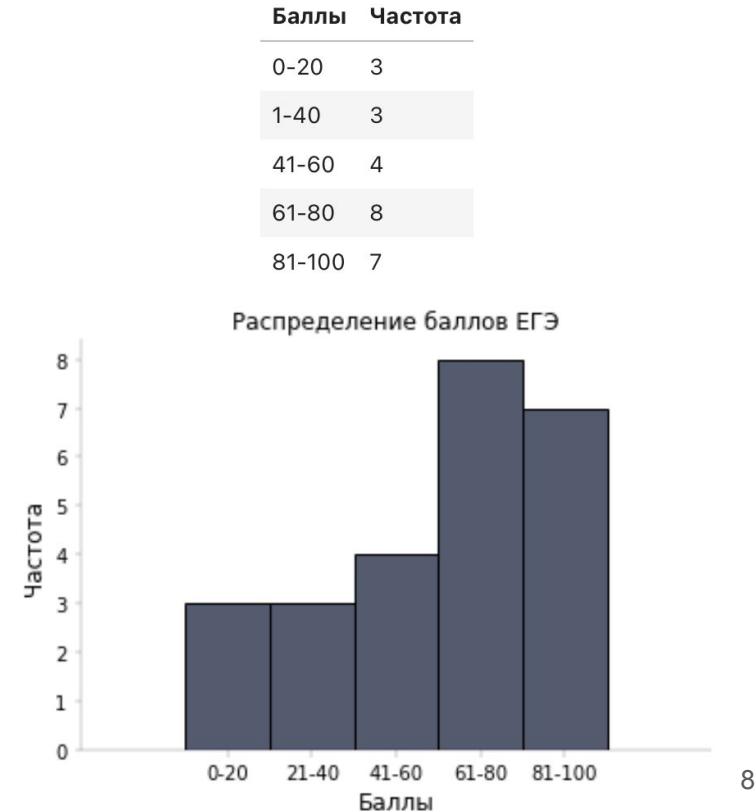
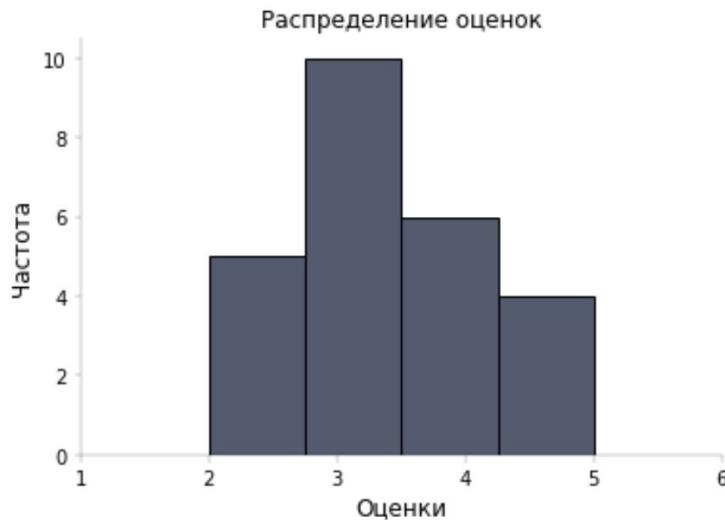
Пусть у нас есть данные, которые содержат баллы по ЕГЭ по информатике для этих же учеников:

45,55,20,63,65,34,55,88,73,78,75,20,98,63,93,60,14,88,90,75,95,73,34,27,100

Значения	Частота
0-20	3
1-40	3
41-60	4
61-80	8
81-100	7

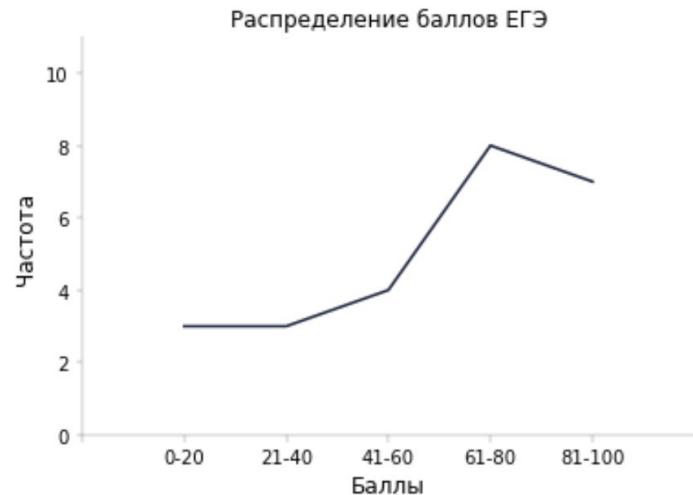
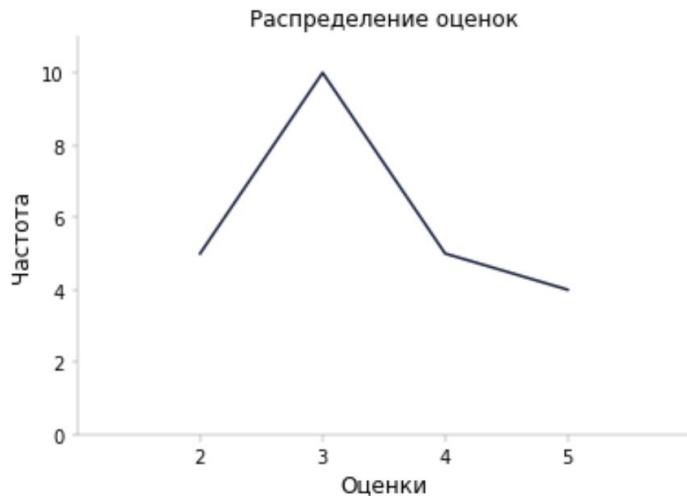
Визуализация частотного распределения

**Гистограмма
частотного
распределения
оценок/баллов**



Визуализация частотного распределения

**Полигон
частотного
распределения
оценок/баллов**

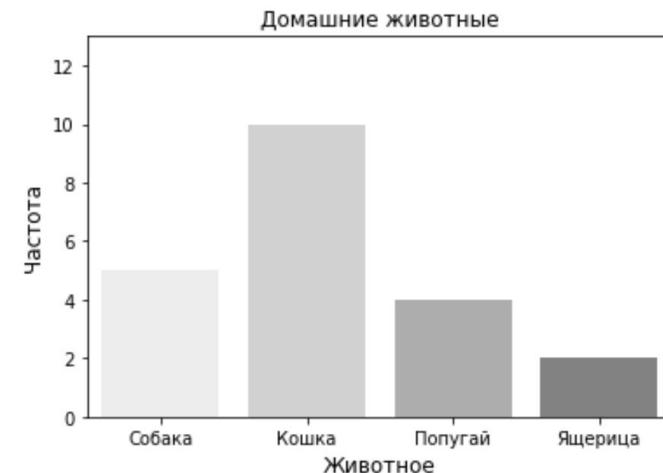


Визуализация категориальных данных

Гистограмма отлично подходит для визуализации **количественных** данных, но нам важно уметь в аналогичном виде представлять распределение частот для **порядковых** и **номинальных** данных.

Аналогом гистограммы для качественных данных является **столбчатая диаграмма**.

Животное	Частота
Собака	5
Кошка	10
Попугай	4
Ящерица	2



Группировка данных в pandas



Группировка данных

```
In [1]: import pandas as pd
```

```
bikes = pd.read_pickle('Data/BikesDataVars.pkl')
```

```
In [2]: bikes.head()
```

```
Out[2]:
```

	Date	Hour	Temperature	Humidity	Wind speed	Rainfall	Snowfall	Seasons	Holiday	Functioning Day	Rental Count	Normal Humidity	Temperature Category	Good Weather
0	2017-12-01	0	-5.2	37	2.2	0.0	0.0	Winter	0	True	257	0	Freezing	0
1	2017-12-01	1	-5.5	38	0.8	0.0	0.0	Winter	0	True	219	0	Freezing	0
2	2017-12-01	2	-6.0	39	1.0	0.0	0.0	Winter	0	True	162	0	Freezing	0
3	2017-12-01	3	-6.2	40	0.9	0.0	0.0	Winter	0	True	148	1	Freezing	0
4	2017-12-01	4	-6.0	36	2.3	0.0	0.0	Winter	0	True	97	0	Freezing	0

Метод value_counts()

Возвращает частотную таблицу значений категориальной переменной

```
In [3]: bikes['Temperature Category'].value_counts()
```

```
Out[3]: Chilly      3112  
        Nice       2778  
        Freezing   1412  
        Hot        1279  
Name: Temperature Category, dtype: int64
```

Параметр dropna в методе value_counts()

Параметр dropna (по умолчанию **True**), если ему передать переменную **False**, не выбрасывает пропущенные значения

```
In [4]: bikes['Temperature Category'].value_counts(dropna=False)
```

```
Out[4]: Chilly      3112  
        Nice       2778  
        Freezing   1412  
        Hot        1279  
        NaN        179  
Name: Temperature Category, dtype: int64
```

Метод groupby() и агрегирование

Группирует по колонке и считает агрегирующую функцию для заданной колонки (колонок)

```
In [5]: bikes.groupby('Date')['Rental Count'].sum()  
Out[5]: Date  
2017-12-01    9802  
2017-12-02    8404  
2017-12-03    8644  
2017-12-04    9556  
2017-12-05    6578  
...  
2018-11-26   13934  
2018-11-27   13964  
2018-11-28   17378  
2018-11-29   18058  
2018-11-30   17543  
  
Name: Rental Count, Length: 365, dtype: int64
```

①.

②.

③.

1. По какой колонке группируем

2. По какому признаку считаем статистику

3. Какую статистику считаем

Date Rental Count

0	2017-12-01	257
1	2017-12-01	219
2	2017-12-01	162
3	2017-12-01	148
4	2017-12-01	97
5	2017-12-01	173
...

sum()

('Date') ['Rental Count']

Объединение группировки и частотной таблицы

```
bikes.groupby('Seasons')[['Temperature Category']].value_counts()
```

Seasons		
Autumn	Chilly	1138
	Nice	894
	Hot	86
	Freezing	20
Spring	Chilly	1224
	Nice	881
	Hot	27
	Freezing	22
Summer	Hot	1166
	Nice	1003
	Chilly	0
	Freezing	0
Winter	Freezing	1370
	Chilly	750
	Hot	0
	Nice	0

Name: Temperature Category, dtype: int64

Группировка по нескольким переменным

```
bikes.groupby(['Seasons', 'Temperature Category'])['Rental Count'].sum()
```

Seasons	Temperature Category	
Autumn	Chilly	775694
	Freezing	12035
	Hot	134156
	Nice	811498
Spring	Chilly	587211
	Freezing	6331
	Hot	52993
	Nice	928572
Summer	Chilly	0
	Freezing	0
	Hot	1315525
	Nice	892664
Winter	Chilly	215221
	Freezing	258570
	Hot	0
	Nice	0

Name: Rental Count, dtype: int64

Примеры часто используемых методов для агрегирования

`mean()` – среднее

`sum()` – сумма

`count()` – подсчет количества элементов

`min()/max()` – минимальное/максимальное значение

`median()` – медиана

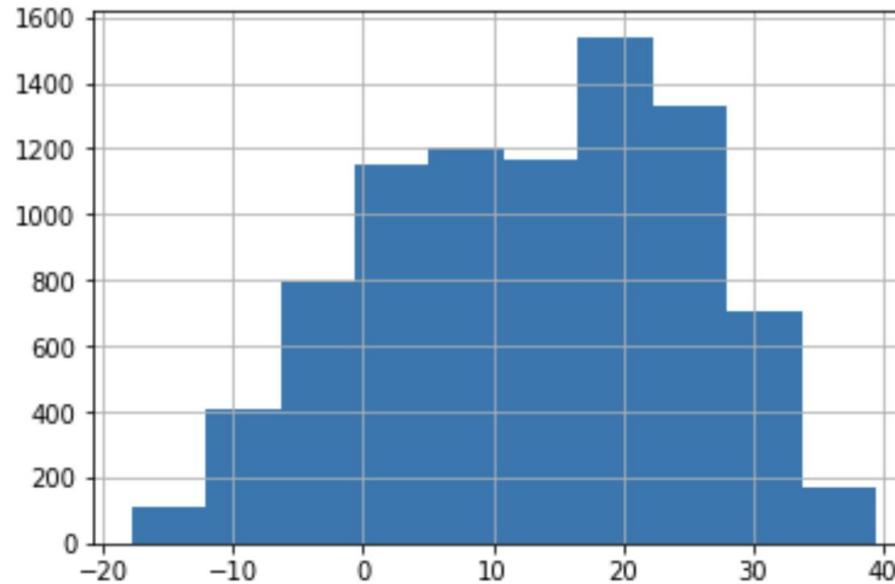
Базовая визуализация в pandas



Метод hist() для гистограммы

По умолчанию создается 10 столбиков значений **количественной** переменной

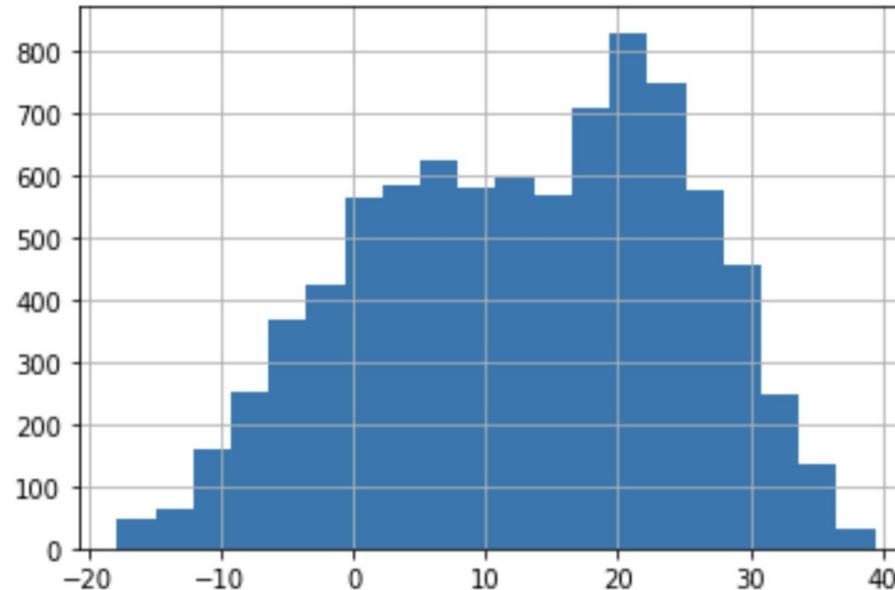
```
bikes['Temperature'].hist();
```



Параметр bins в методе hist()

Можно построить частотное распределение с большим кол-вом столбиков

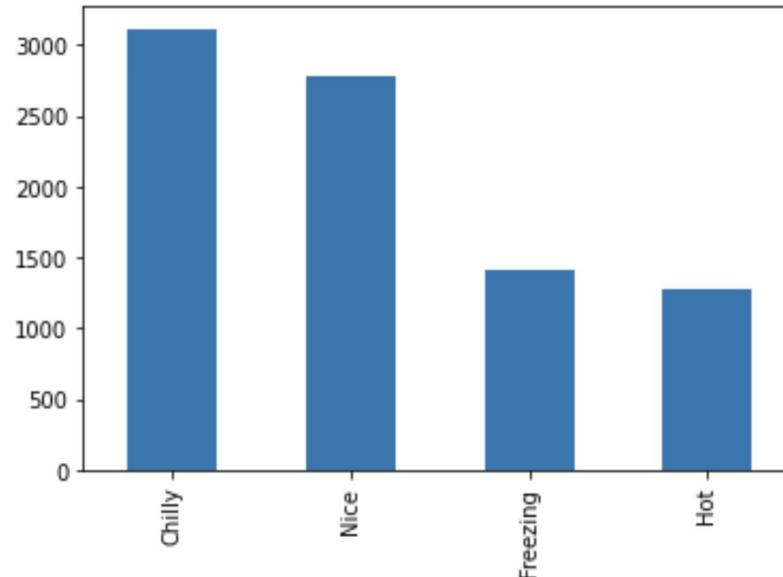
```
bikes['Temperature'].hist(bins=20);
```



Метод .plot(kind='bar') для столбчатой диаграммы

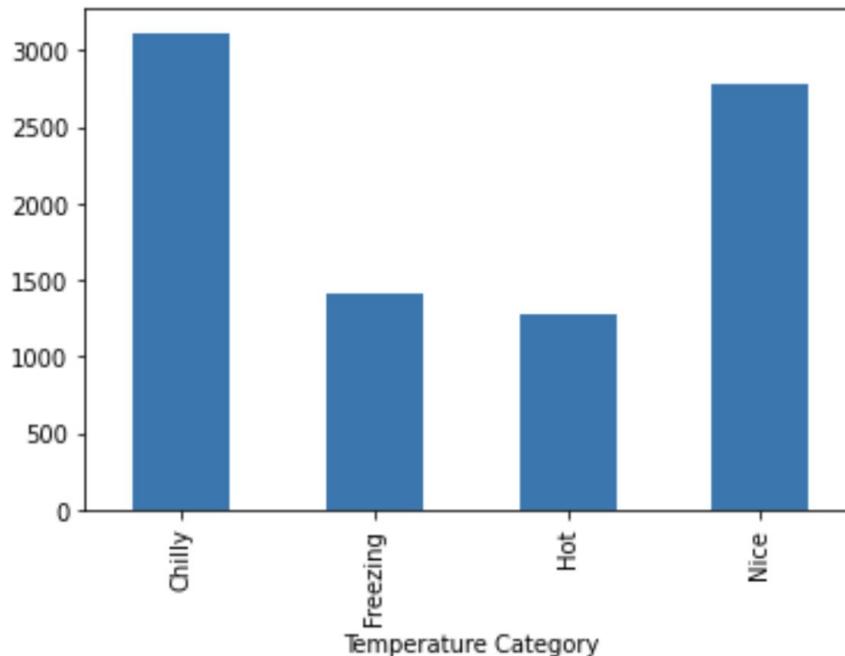
Вызывается у метода value_counts(), подсчитывающего частоты значений **категориальной** переменной

```
bikes['Temperature Category'].value_counts().plot(kind='bar');
```



Столбчатая диаграмма с категориями в алфавитном порядке

```
bikes.groupby('Temperature Category').size().plot(kind='bar');
```



Многоуровневая таблица при помощи метода .unstack()

Unstack избавляется от уровня и добавляет его к столбцам

Значения категориальной переменной из группировки становятся рядами, значения переменной, для которых считаются частоты, – колонками

```
bikes.groupby('Seasons')[['Temperature Category']].value_counts()
```

Seasons	Temperature Category	Count
Autumn	Chilly	1138
	Nice	894
	Hot	86
	Freezing	20
Spring	Chilly	1224
	Nice	881
	Hot	27
	Freezing	22
Summer	Hot	1166
	Nice	1003
	Chilly	0
	Freezing	0
Winter	Freezing	1370
	Chilly	750
	Hot	0
	Nice	0

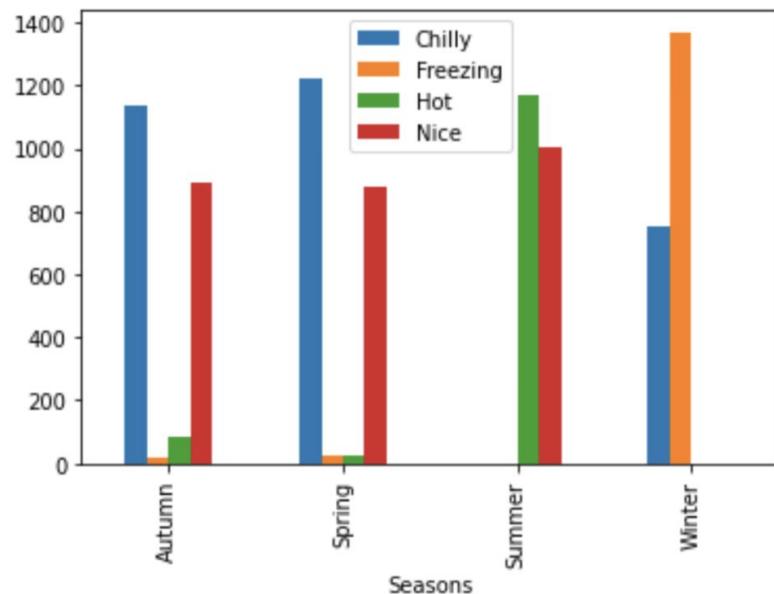
```
bikes.groupby('Seasons')[['Temperature Category']].value_counts().unstack()
```

Seasons	Chilly	Freezing	Hot	Nice
Autumn	1138	20	86	894
Spring	1224	22	27	881
Summer	0	0	1166	1003
Winter	750	1370	0	0

Name: Temperature Category, dtype: int64

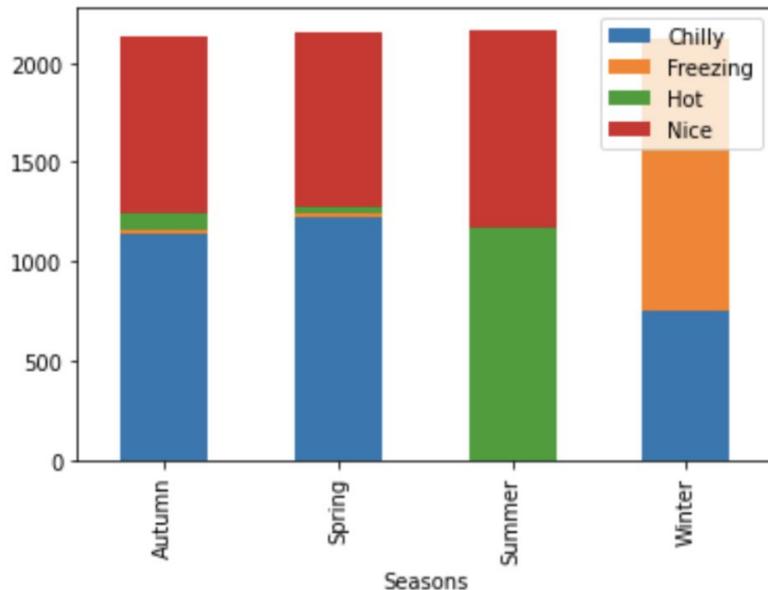
Составная столбчатая диаграмма

```
bikes.groupby('Seasons')[['Temperature Category']].value_counts().unstack().plot(kind='bar');
```



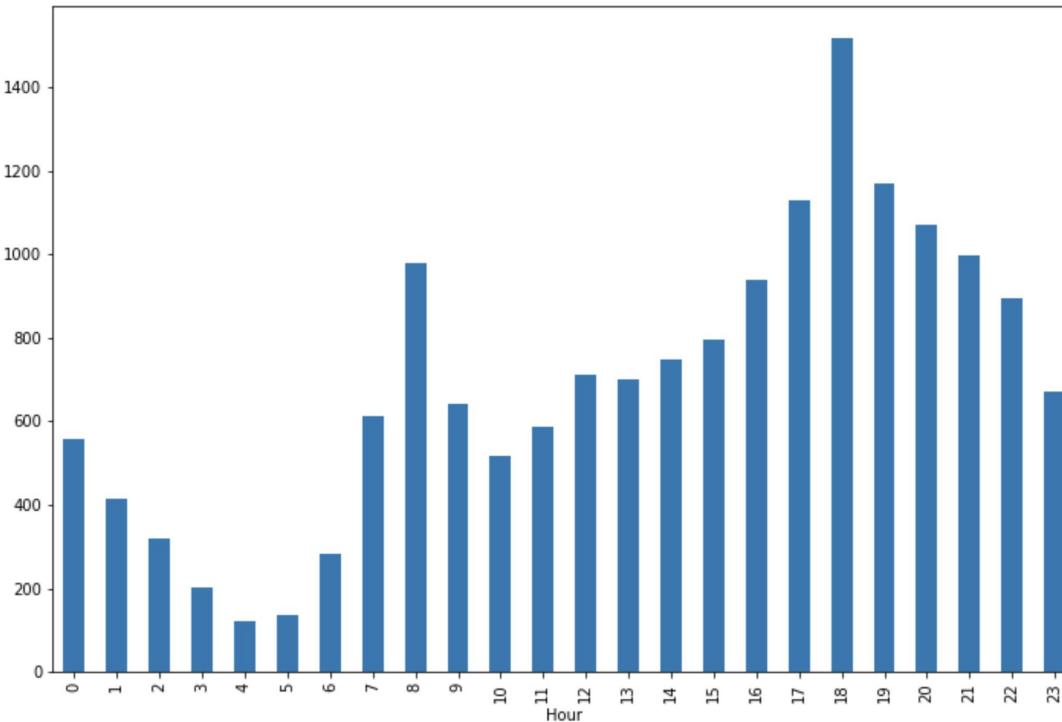
Составная столбчатая диаграмма

```
bikes.groupby('Seasons')[['Temperature Category']].value_counts().unstack().plot(kind='bar',  
                           stacked=True);
```

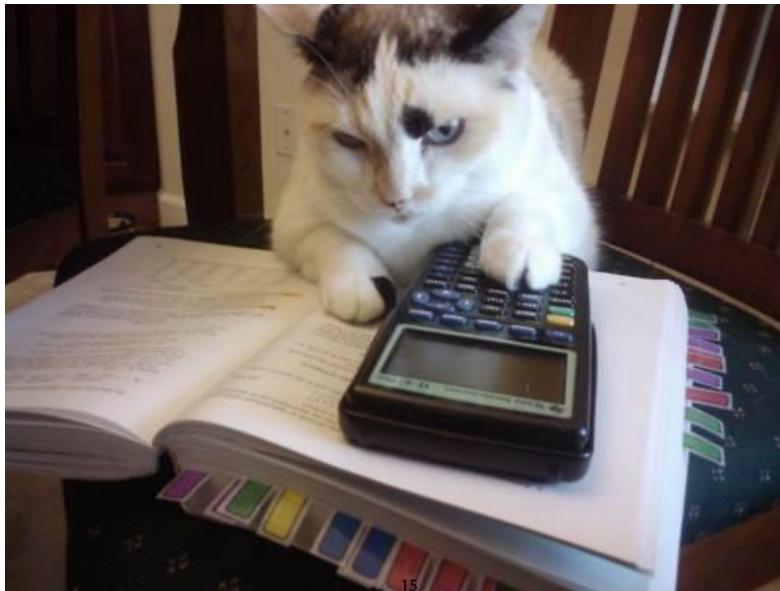


Визуализация среднего после группировки

```
bikes.groupby('Hour')['Rental Count'].mean().plot(kind='bar', figsize=(12,8));
```

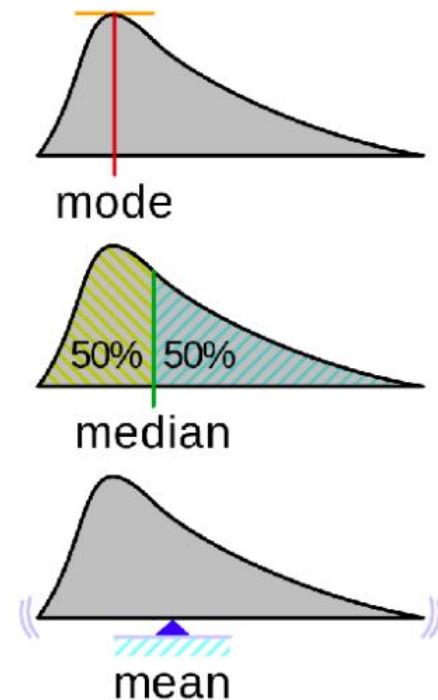


Описательные статистики: меры центральной тенденции и разброса



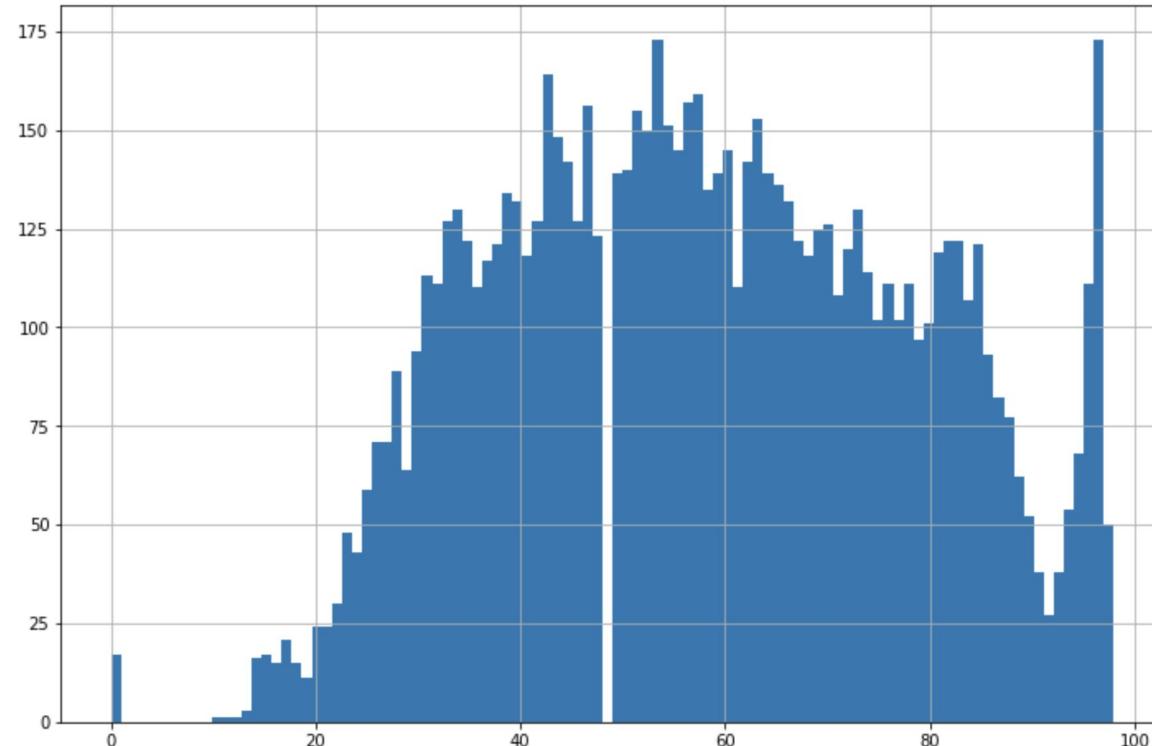
Меры центральной тенденции

- Мода (mode) – наиболее часто встречающееся значение, локальный максимум
`bikes['Humidity'].mode()`
- Медиана (median) – значение, которое делит распределение: половина значений больше медианы, половина – меньше
`bikes['Humidity'].median()`
- Среднее значение (mean) – сумма всех значений переменной, делённая на количество значений
`bikes['Humidity'].mean()`



Поиск моды визуально

```
: bikes['Humidity'].hist(bins=100, figsize=(12,8));
```



Метод describe()

```
bikes.describe()
```

	Hour	Temperature	Humidity	Wind speed	
count	8760.000000	8581.000000	8760.000000	8760.000000	8760.000000
mean	11.500000	12.878557	58.226256	1.724909	
std	6.922582	11.955551	20.362413	1.036300	
min	0.000000	-17.800000	0.000000	0.000000	
25%	5.750000	3.400000	42.000000	0.900000	
50%	11.500000	13.700000	57.000000	1.500000	
75%	17.250000	22.500000	74.000000	2.300000	
max	23.000000	39.400000	98.000000	7.400000	

Подсчет среднего по колонке с предварительной группировкой по другой

Сколько в среднем арендовали
велосипедов в каждый час за год?

```
bikes.groupby('Hour')['Rental Count'].mean()
```

Hour

0	558.178082
1	415.720548
2	319.767123
3	201.010959
4	122.838356
5	135.863014
6	283.654795
7	612.646575
8	979.838356
9	642.136986
10	519.123288
11	585.336986
12	710.534247
13	700.706849
14	747.147945
15	796.084932
16	938.032877
17	1128.873973
18	1518.983562
19	1168.328767
20	1069.698630
21	997.942466
22	893.621918
23	671.898630

Name: Rental Count, dtype: float64

Подсчет медианы с группировкой по двум признакам

```
bikes.groupby(['Hour', 'Good Weather'])['Rental Count'].median()
```

Hour	Good Weather	
0	0	386.0
	1	1053.0
1	0	301.0
	1	621.5
2	0	230.0
	1	550.0
3	0	155.0
	1	347.0
4	0	91.0
	1	193.5
5	0	105.0
	1	353.5
6	0	188.0
	1	349.0
7	0	386.5
	1	671.0
8	0	755.0
	1	914.5

Метод .agg()

Если передать список статистик внутри .agg(), возвращается датафрейм

```
bikes.groupby('Seasons')['Temperature'].agg(['mean', 'median'])
```

	mean	median
Seasons		
Autumn	14.138821	13.8
Spring	13.038533	13.3
Summer	26.574827	26.6
Winter	-2.567783	-2.2

Метод .pivot_table()

Возвращает таблицу с агрегированными данными (сводная таблица)

```
bikes.pivot_table(index='Hour', сагрегировать по часам (ряды таблицы)
                    values=['Temperature', 'Rental Count'], какие колонки интересуют
                    aggfunc=['mean', 'median']) какие метрики интересуют
```

Hour	mean		median	
	Rental Count	Temperature	Rental Count	Temperature
	0	1	2	3
0	558.178082	11.253652	433	11.80
1	415.720548	10.972145	329	11.50
2	319.767123	10.542535	239	10.90
3	201.010959	10.355462	165	10.70
4	122.838356	10.069859	94	10.40
5	135.863014	9.849580	107	10.00
6	283.654795	9.763944	193	10.00
7	612.646575	9.454062	410	9.60
8	979.838356	9.988515	762	10.40
9	642.136986	11.459669	519	12.10
10	519.123288	12.766947	414	14.00

Метод .pivot_table() с двухуровневой агрегацией

```
bikes.pivot_table(index=['Hour', 'Seasons'],
                   values=['Temperature', 'Rental Count'],
                   aggfunc=['mean', 'median'])
```

Hour	Seasons	mean		median	
		Rental Count	Temperature	Rental Count	Temperature
0	Autumn	699.912088	12.467816	722.0	11.90
	Spring	491.760870	10.977273	430.0	11.70
	Summer	875.097826	24.752174	880.5	24.55
	Winter	158.800000	-3.613483	148.5	-2.80
1	Autumn	472.747253	12.261111	514.0	11.60
...
22	Winter	213.722222	-2.920455	184.0	-2.30
23	Autumn	761.560440	12.814444	745.0	11.90
	Spring	613.076087	11.647191	441.5	12.40
	Summer	1142.347826	25.282022	1113.5	25.00
	Winter	160.466667	-3.052273	148.0	-2.50

96 rows × 4 columns

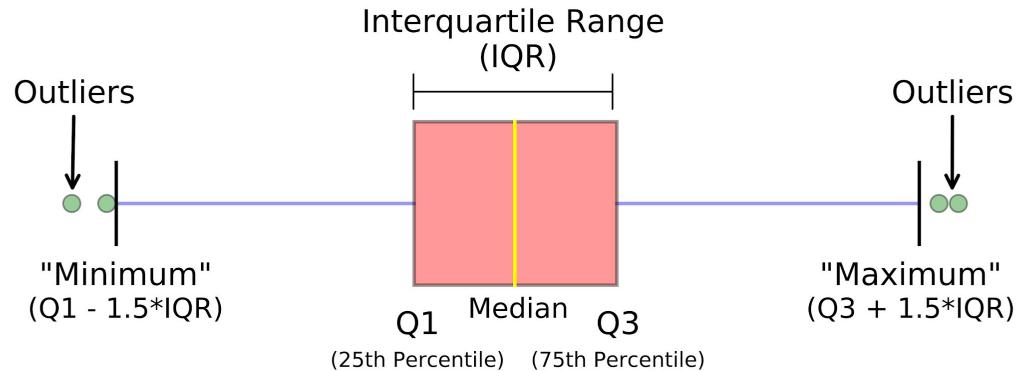
Меры разброса

- Размах
- Квартильный размах
- Дисперсия
- Стандартное отклонение

Размах

- Размах — разность между **наибольшим и наименьшим** значениями результатов наблюдений.
- Пример:
 - Дан числовой ряд: 1, 5, 3, 9, 11, 2, 14, 6
 - Расположим числа в порядке возрастания: 1, 2, 3, 5, 6, 9, 11, 14
 - Найдем размах: $14 - 1 = 13$

Интерквартильный размах (IQR)



- Q_1 – такое значение, при котором **25% наблюдений меньше** него
- Q_2 – то же самое, что и медиана (50% наблюдений меньше этого значения и 50% наблюдений больше)
- Q_3 – такое значение, при котором **25% наблюдений больше** него
- Интерквартильный размах - это разница между значениями третьего (верхнего, 75%) и первого (нижнего, 25%) квартилей; она показывает, в какое число категорий «укладываются» центральные 50% наблюдений

Стандартное отклонение и дисперсия

- Пример:
 - Предположим, что интересующая нас группа (генеральная совокупность) – это класс из восьми учеников, которым выставляются оценки по 10-балльной системе.
 - Пусть оценки учеников класса следующие: 2, 4, 4, 4, 5, 5, 7, 9.
 - 1. Тогда средняя оценка равна: $(2+4+4+4+5+5+7+9) / 8 = 5$
 - 2. Вычислим **квадраты отклонений** оценок учеников от их средней оценки:

$$(2-5)^2 = (-3)^2 = 9$$

$$(5-5)^2 = 0^2 = 0$$

$$(4-5)^2 = (-1)^2 = 1$$

$$(5-5)^2 = 0^2 = 0$$

$$(4-5)^2 = (-1)^2 = 1$$

$$(7-5)^2 = 2^2 = 4$$

$$(4-5)^2 = (-1)^2 = 1$$

$$(9-5)^2 = 4^2 = 16$$

Стандартное отклонение и дисперсия

- Продолжение примера
 - 3. Посчитаем среднее арифметическое квадратов отклонений оценок учеников от их средней оценки (значений, полученных на шаге 2) . Данную величину будем называть дисперсией.
 - .

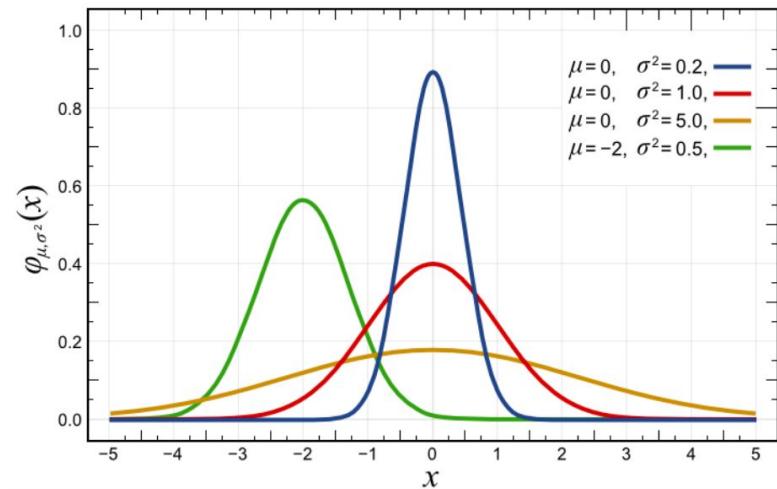
$$\sigma^2 = \frac{9+1+1+1+0+0+16}{8} = 4$$

- Ремарка: здесь используется смещённая оценка, в случае несмещённой оценки используется $n - 1$ в знаменателе, где n – число наблюдений.
- 4. Возьмем квадратный корень из дисперсии
 - Данную величину будем называть стандартным отклонением
 - .

$$\sigma = \sqrt{4} = 2$$

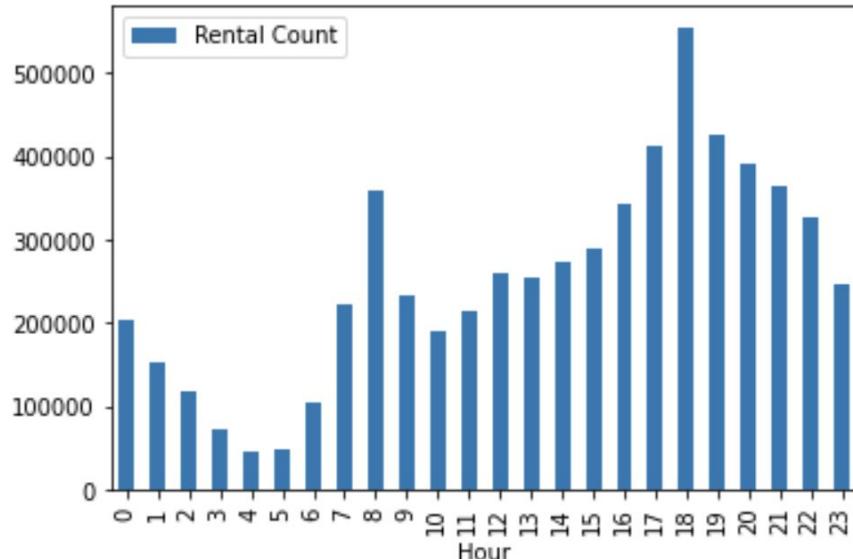
Интерпретация стандартного отклонения

- Также *стандартное отклонение* называют ещё *среднеквадратичным отклонением*
 - **Большее** значение этой характеристики означает **большой разброс** данных относительно среднего;
 - **меньшее** значение, соответственно, показывает, что значения сгруппированы вокруг среднего значения



Создадим датафрейм с агрегированной статистикой

```
: bikes_hour = pd.DataFrame(bikes.groupby('Hour')['Rental Count'].sum())  
  
: bikes_hour.plot(kind='bar');
```



Размах и интерквартильный размах

```
# размах  
bikes_hour['Rental Count'].max() - bikes_hour['Rental Count'].min()
```

509593

```
# интерквартильный размах  
bikes_hour['Rental Count'].quantile(0.75) - bikes_hour['Rental Count'].quantile(0.25)
```

166152.25

Методы .var() и .std()

Возвращают дисперсию и стандартное отклонение соответственно

```
: print(bikes_hour['Rental Count'].var())
print(bikes_hour['Rental Count'].std())
```

```
16566399888.432972
128710.5274965221
```

Сводная таблица с использованием собств. ф-ции

```
def range_values(x):
    return x.max() - x.min()

bikes.pivot_table(index='Hour',
                  values=['Temperature', 'Rental Count'],
                  aggfunc=['mean', 'std', 'var', range_values])
```

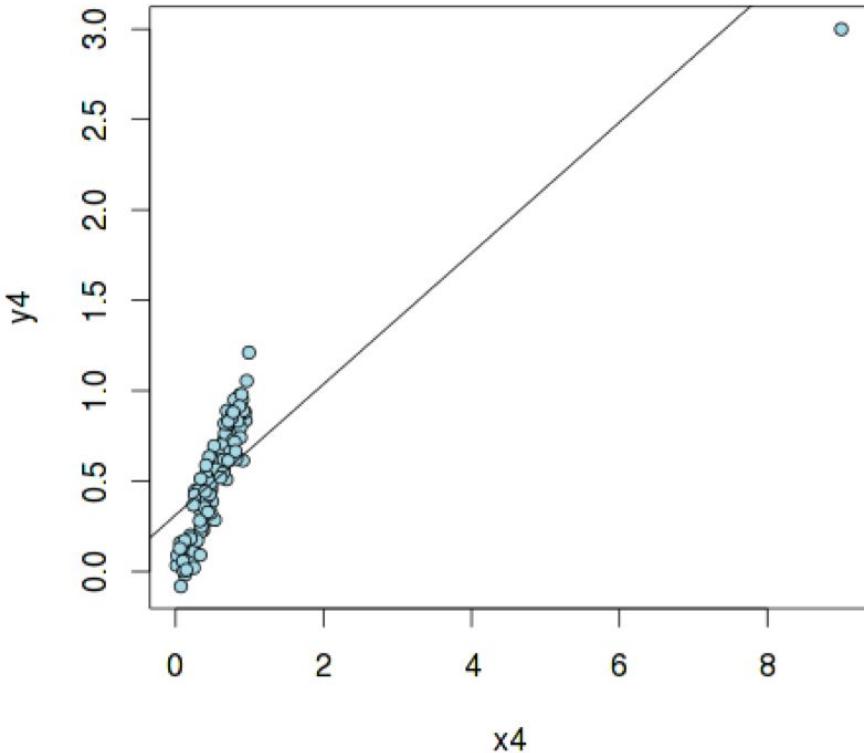
Hour	mean		std		var		range_values	
	Rental Count	Temperature						
0	558.178082	11.253652	455.016192	11.437803	2.070397e+05	130.823339	2059	48.0
1	415.720548	10.972145	338.058896	11.391632	1.142838e+05	129.769278	1485	47.8
2	319.767123	10.542535	268.797389	11.411300	7.225204e+04	130.217762	1950	47.5
3	201.010959	10.355462	162.601701	11.306762	2.643931e+04	127.842870	781	47.2
4	122.838356	10.069859	106.375778	11.284178	1.131581e+04	127.332677	608	47.5
5	135.863014	9.849580	113.191916	11.310231	1.281241e+04	127.921327	561	47.6

Выбросы



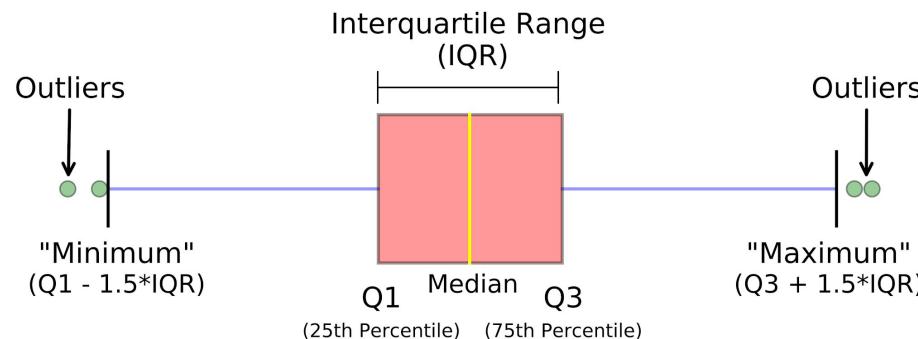
Выбросы

- Выброс — это результат измерения, выделяющийся из общей выборки
 - Поскольку множество статистических методов плохо работают на выборках с выбросами, их приходится обнаруживать и исключать



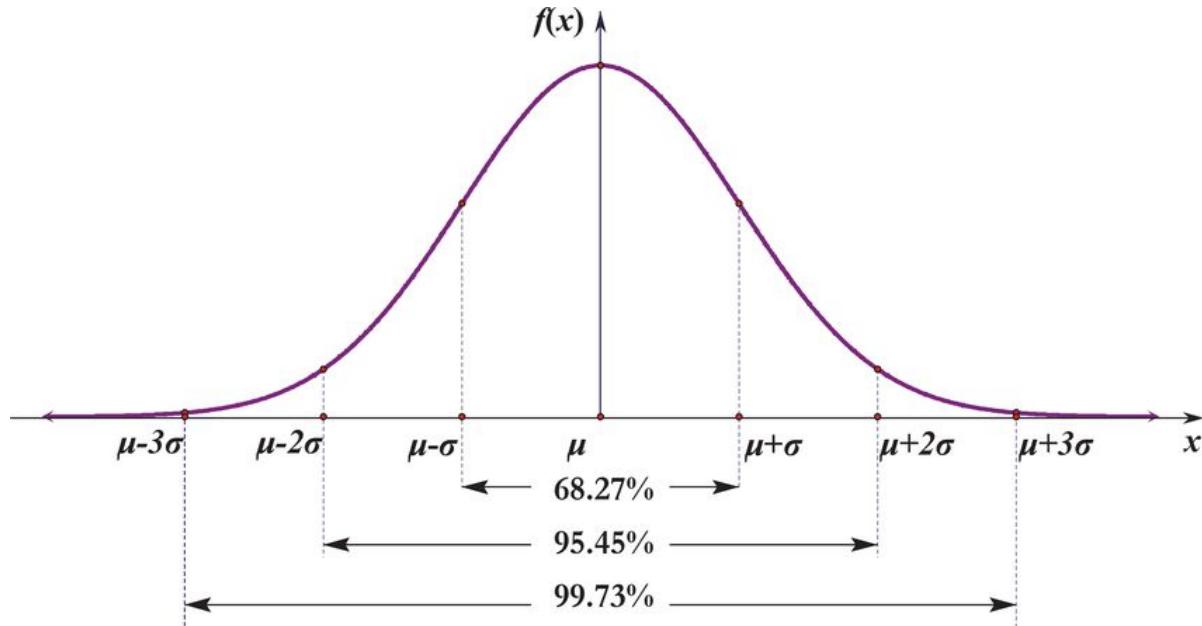
Метод межквартильного расстояния

- Подходит для данных, которые имеют не нормальное распределение
- Алгоритм
 - Вычислить Q1, Q3 (1-й и 3-й квартили)
 - Вычислить $IQR = Q3 - Q1$ (межквартильный диапазон)
 - Находим **внутренние** границы для обнаружения **незначительных** выбросов
 - $[Q1 - IQR \cdot 1.5, IQR \cdot 1.5 + Q3]$
 - Находим **внешние** границы для обнаружения **значительных** выбросов
 - $[Q1 - IQR \cdot 3, IQR \cdot 3 + Q3]$



Метод среднеквадратичного отклонения

- «Правило трех сигм»
 - В **нормальном** распределении (или схожем с ним) среднеквадратичное отклонение (σ) может определить суммарный процент значений, которое оно покрывает в выборке



Коэффициент корреляции

- $-1 < \text{corr}(x,y) < 1$
 - Чем $\text{corr}(x,y)$ ближе к 1,
 - тем больше положительная связь между двумя наборами переменных
 - Чем $\text{corr}(x,y)$ ближе к -1,
 - тем больше отрицательная связь между двумя наборами переменных
 - Чем $\text{corr}(x,y)$ ближе к 0,
 - тем меньше связи между двумя наборами переменных
- Коэффициент корреляции отражает **линейную взаимосвязь** между **переменными**
 - переменные могут быть связаны нелинейно!
- Коэффициент корреляции не говорит о **причинности**
 - причинность следует из теории