B [1]:

```
% matplotlib  встроенный
```

Сравните влияние различных скейлеров на данные с выбросами

Характеристика 0 (средний доход в блоке) и характеристика 5 (количество домохозяйств) `California housing dataset <https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html> _` имеют очень разные масштабы и содержат несколько очень больших выбросов. Эти две характеристики затрудняют визуализацию данных и, что более важно, могут ухудшить прогнозную производительность многих алгоритмов машинного обучения. Немасштабированные данные также могут замедлить или даже предотвратить сходимость многих оценок на основе градиента.

Действительно, многие оценщики разработаны с предположением, что каждая характеристика принимает значения, близкие к нулю или, что более важно, что все функции различаются в сопоставимых масштабах. В частности, оценщики на основе метрик и градиентов часто предполагают приблизительно стандартизованные данные (центрированные объекты с единичной дисперсией). Заметным исключением являются оценщики на основе дерева решений, устойчивые к произвольному масштабированию данных.

В этом примере используются различные средства масштабирования, преобразователи и нормализаторы для приведения данных в предварительно определенный диапазон.

Масштабирующие устройства - это линейные (или, точнее, аффинные) преобразователи, которые отличаются друг от друга способом оценки параметров, используемых для сдвига и масштабирования каждой функции.

`QuantileTransformer` обеспечивает нелинейные преобразования, в которых расстояние между маргинальными выбросами и выбросами сокращается. `PowerTransformer` обеспечивает нелинейные преобразования, при которых данные отображаются в нормальное распределение для стабилизации дисперсии и минимизации асимметрии.

В отличие от предыдущих преобразований, нормализация относится к преобразованию для каждого образца, а не для преобразования элемента.

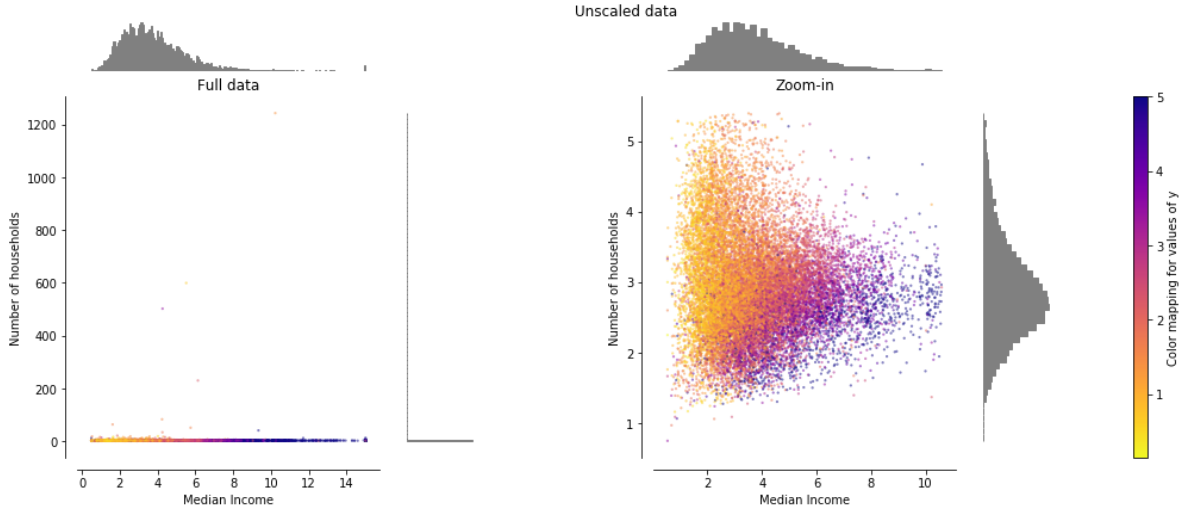
Следующий код немного подробен, не стесняйтесь переходить непосредственно к анализу результатов_.

Исходные данные

Каждое преобразование построено с отображением двух преобразованных функций, причем левый график показывает весь набор данных, а правый увеличенный, чтобы показать набор данных без маргинальных выбросов. Подавляющее большинство выборок сжаты до определенного диапазона: [0, 10] для среднего дохода и [0, 6] для количества домохозяйств. Обратите внимание, что есть некоторые незначительные выбросы (в некоторых кварталах насчитывается более 1200 домохозяйств). Следовательно, конкретная предварительная обработка может быть очень полезной в зависимости от приложения. Далее мы представляем некоторые идеи и поведение этих методов предварительной обработки в присутствии маргинальных выбросов.

B [4]:

```
make_plot ( 0 )
```



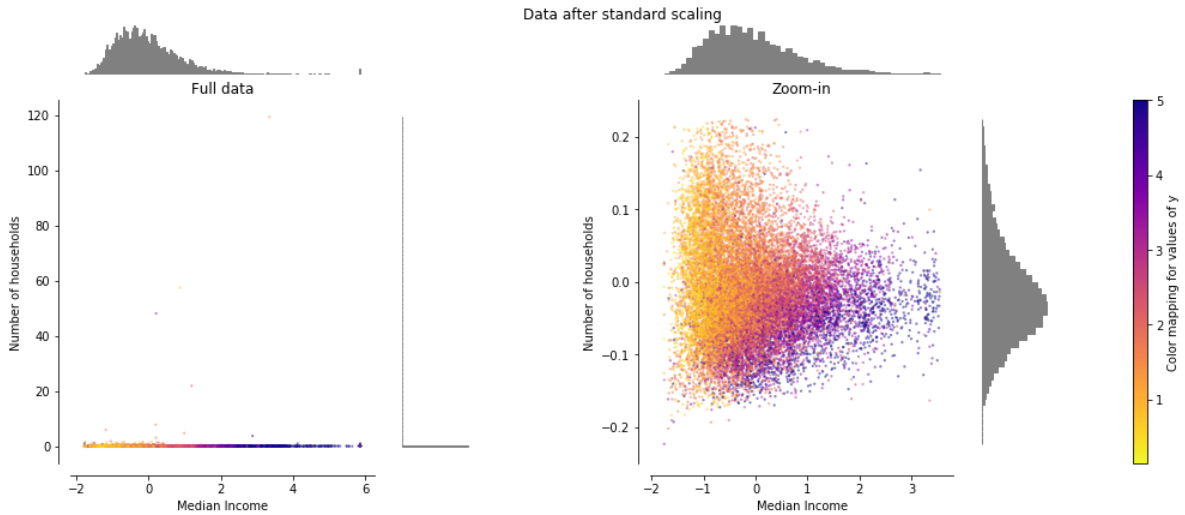
StandardScaler

`StandardScaler` удаляет среднее и масштабирует данные до единичной дисперсии. Однако выбросы влияют на вычисление среднего эмпирического значения и стандартного отклонения, которые сокращают диапазон значений признаков, как показано на левом рисунке ниже. В частности, обратите внимание на то, что, поскольку выбросы по каждой характеристике имеют разную величину, разброс преобразованных данных по каждой характеристике сильно различается: большая часть данных находится в диапазоне [-2, 4] для преобразованной характеристики медианного дохода, в то время как те же самые данные сжаты в меньшем диапазоне [-0,2, 0,2] для преобразованного числа домохозяйств.

`StandardScaler` поэтому не может гарантировать сбалансированные масштабы признаков при наличии выбросов.

B [5]:

```
make_plot ( 1 )
```



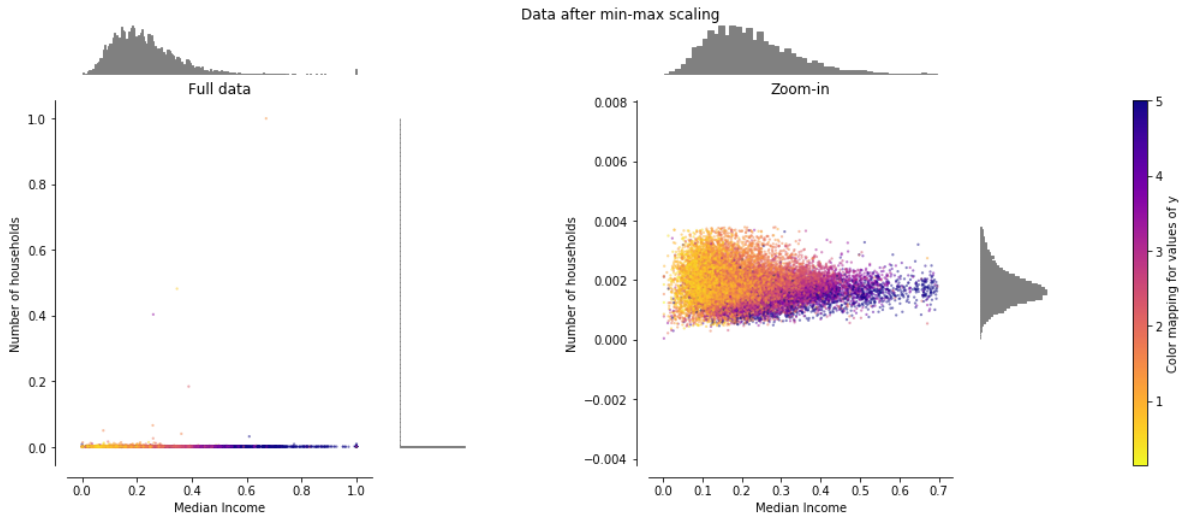
MinMaxScaler

`MinMaxScaler` изменяет масштаб набора данных так, чтобы все значения функций находились в диапазоне [0, 1], как показано на правой панели ниже. Однако это масштабирование сжимает все вставки в узком диапазоне [0, 0,005] для преобразованного количества домохозяйств.

Как `StandardScaler` , `MinMaxScaler` очень чувствителен к наличию выбросов.

В [6]:

```
make_plot ( 2 )
```

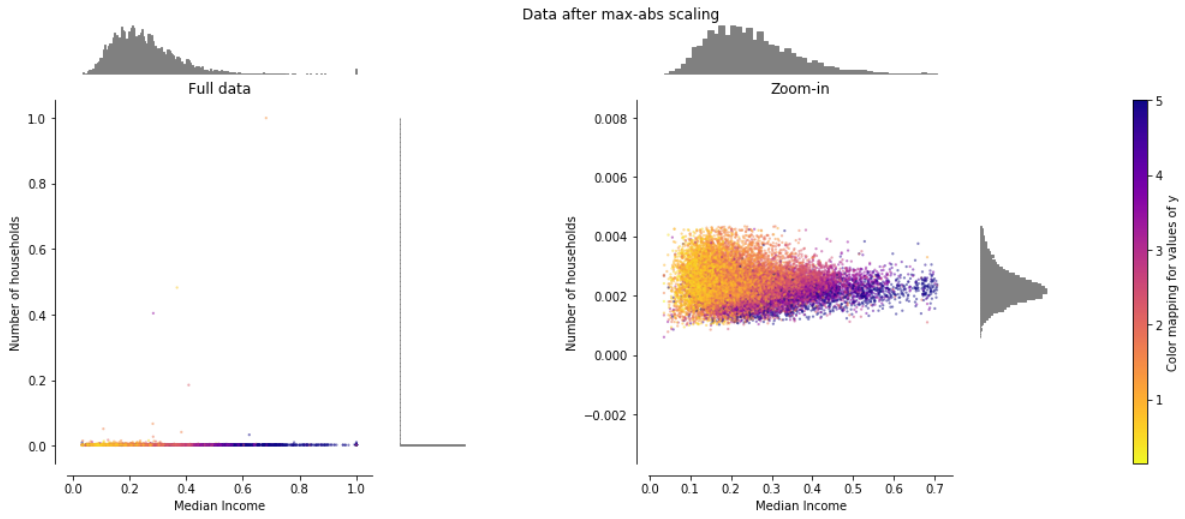


MaxAbsScaler

`MaxAbsScaler` отличается от предыдущего средства масштабирования тем, что абсолютные значения отображаются в диапазоне [0, 1]. При `MinMaxScaler` наличии только положительных данных этот скейлер ведет себя аналогично и, следовательно, страдает от наличия больших выбросов.

В [7]:

```
make_plot ( 3 )
```

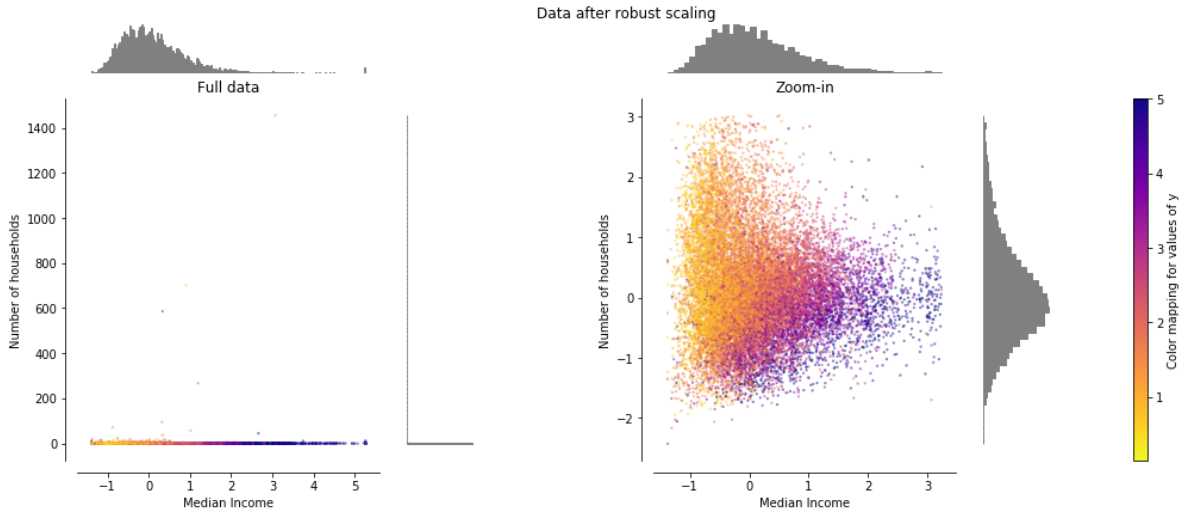


RobustScaler

В отличие от предыдущих скейлеров, статистика центрирования и масштабирования этого скейлера основана на процентилях и поэтому не зависит от небольшого числа очень больших предельных выбросов. Следовательно, результирующий диапазон преобразованных значений признаков больше, чем для предыдущих масштабаторов и, что более важно, примерно одинаков: для обеих функций большинство преобразованных значений лежат в диапазоне [-2, 3], как видно на увеличенном- на рисунке. Обратите внимание, что сами выбросы все еще присутствуют в преобразованных данных. Если желательно отдельное отсечение выбросов, требуется нелинейное преобразование (см. Ниже).

B [8]:

```
make_plot ( 4 )
```

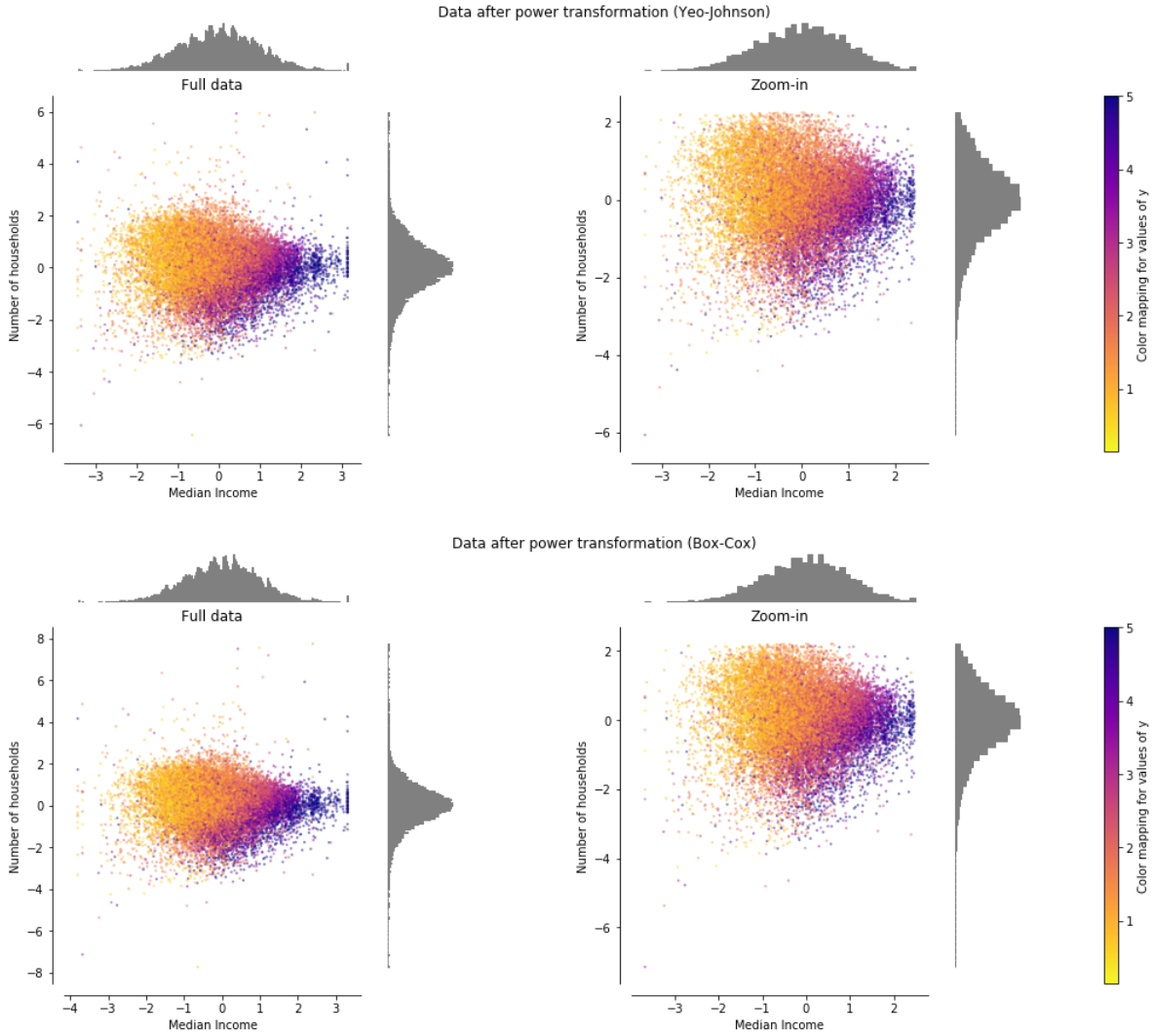


PowerTransformer

`PowerTransformer` применяет степенное преобразование к каждому объекту, чтобы сделать данные более похожими на гауссовские. В настоящее время `PowerTransformer` реализует преобразования Йео-Джонсона и Бокса-Кокса. Преобразование мощности находит оптимальный коэффициент масштабирования для стабилизации дисперсии и уменьшения асимметрии посредством оценки максимального правдоподобия. По умолчанию `PowerTransformer` также применяется нормализация единичной дисперсии с нулевым средним к преобразованному результату. Обратите внимание, что Box-Cox может применяться только к строго положительным данным. Доход и количество домашних хозяйств оказываются строго положительными, но если присутствуют отрицательные значения, предпочтительнее преобразование Йео-Джонсона.

B [9]:

```
make_plot ( 5 )
make_plot ( 6 )
```

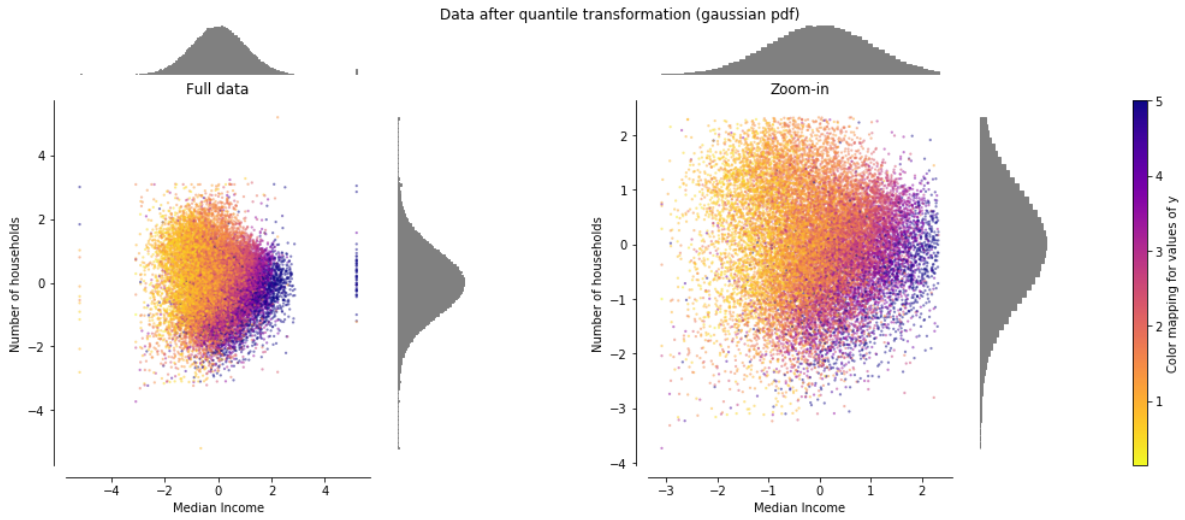


QuantileTransformer (выход по Гауссу)

`QuantileTransformer` имеет дополнительный `output_distribution` параметр, позволяющий сопоставить распределение Гаусса вместо равномерного распределения. Обратите внимание, что этот непараметрический преобразователь вводит артефакты насыщения для экстремальных значений.

B [10]:

```
make_plot ( 7 )
```



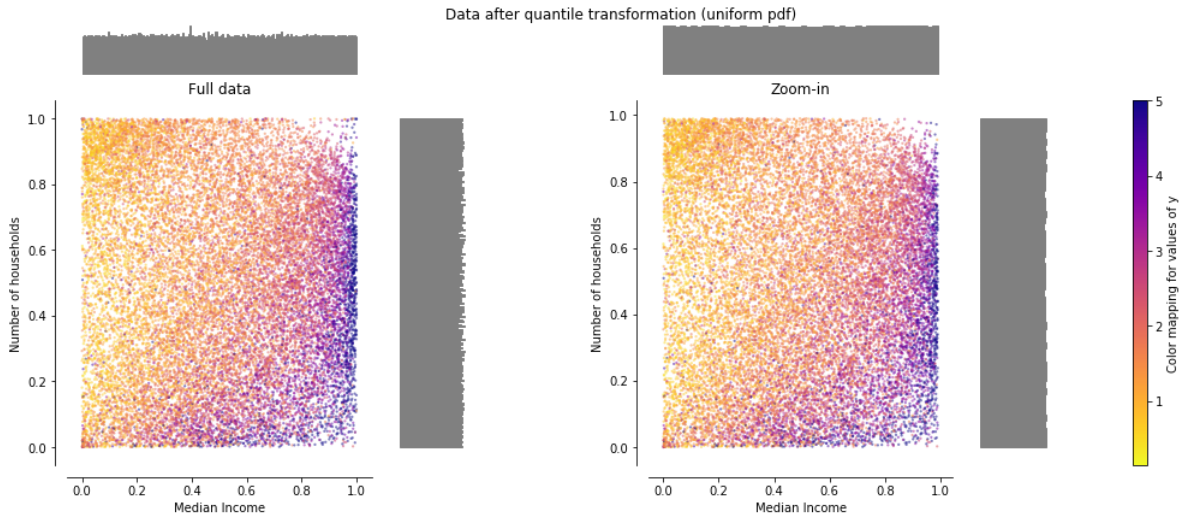
QuantileTransformer (равномерный вывод)

`QuantileTransformer` применяет нелинейное преобразование, так что функция плотности вероятности каждого признака будет отображаться в равномерное распределение. В этом случае все данные будут отображены в диапазоне [0, 1], даже выбросы, которые больше нельзя отличить от выбросов.

Как `RobustScaler` , `QuantileTransformer` является устойчивым к выбросам в том смысле , что добавление или удаление выбросов в обучающем наборе будет давать примерно такое же преобразование на проводимых вне данных. Но, в отличие от `RobustScaler` , `QuantileTransformer` также автоматически сбрасывает любые выбросы, устанавливая их на заранее определенные границы диапазона (0 и 1).

B [11]:

```
make_plot ( 8 )
```



Normalizer

В `Normalizer` перемасштабирует вектор для каждого образца , чтобы иметь единичную норму, независимо от распределения образцов. Это можно увидеть на обоих рисунках ниже, где все образцы нанесены на единичный круг. В нашем примере две выбранные функции имеют только положительные значения; поэтому преобразованные данные лежат только в положительном квадранте. Этого не было бы, если бы некоторые исходные характеристики имели сочетание положительных и отрицательных значений.

В [12]:

```
make_plot ( 9 )

plt.show()
```