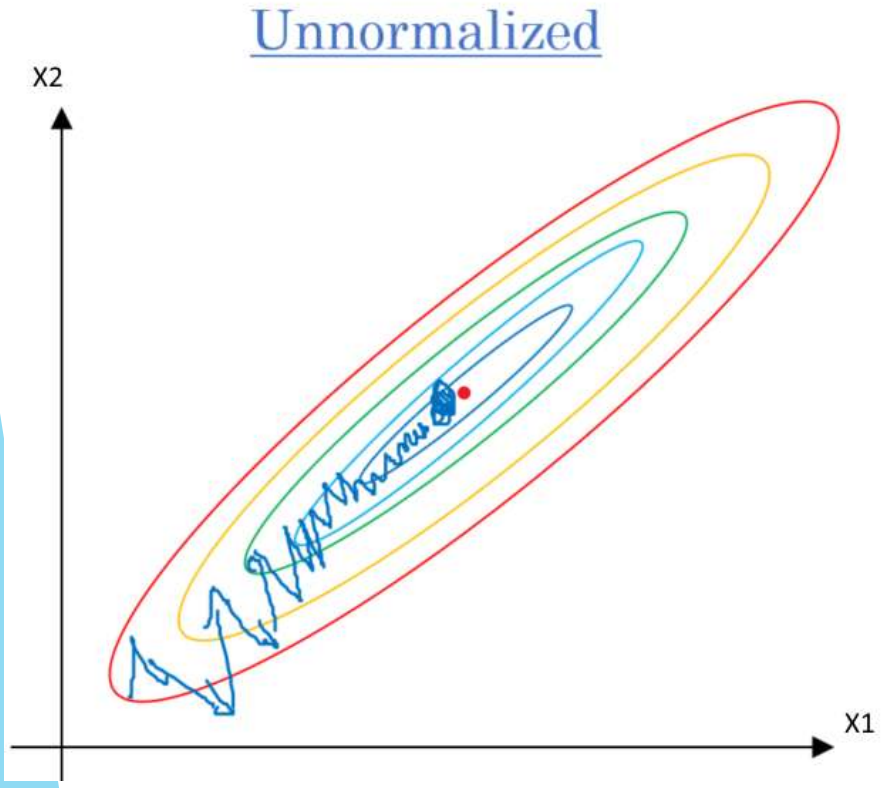
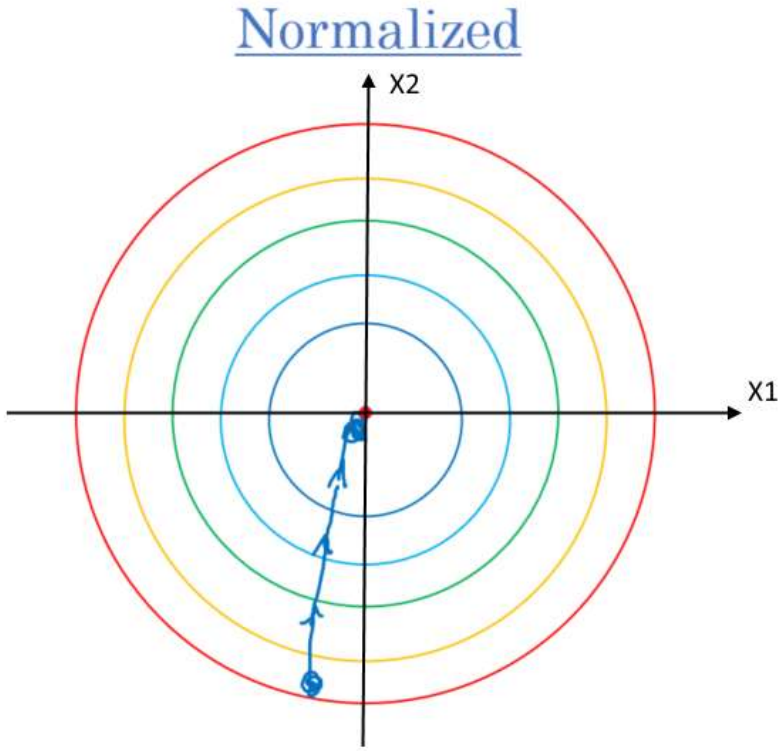
Масштабирование данных

Масштабирование данных

Если признаки сильно отличаются по масштабу, например, x1 in [100500; 201000],
а x2 in [5.1; 7.2], то обучение будет долгим и оптимальная точка может быть не достигнута





https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

Стандартизация (нормализация Z-оценки)

Выполняется преобразование

$$x' = \frac{x - \bar{x}}{\sigma}$$

$\bar{x}$ - математическое ожидание, σ - стандартное отклонение

Новое мат. ожидание 0, стандартное отклонение 1

StandardScaler

- fit(X) – расчет мат. ожидания и стандартного отклонения
- transform(X) – выполнение преобразования
- fit_transform(X) – оба действия за один шаг
- inverse_transform(X) – обратное преобразование

X – массив, размерность [n_samples, n_features]

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

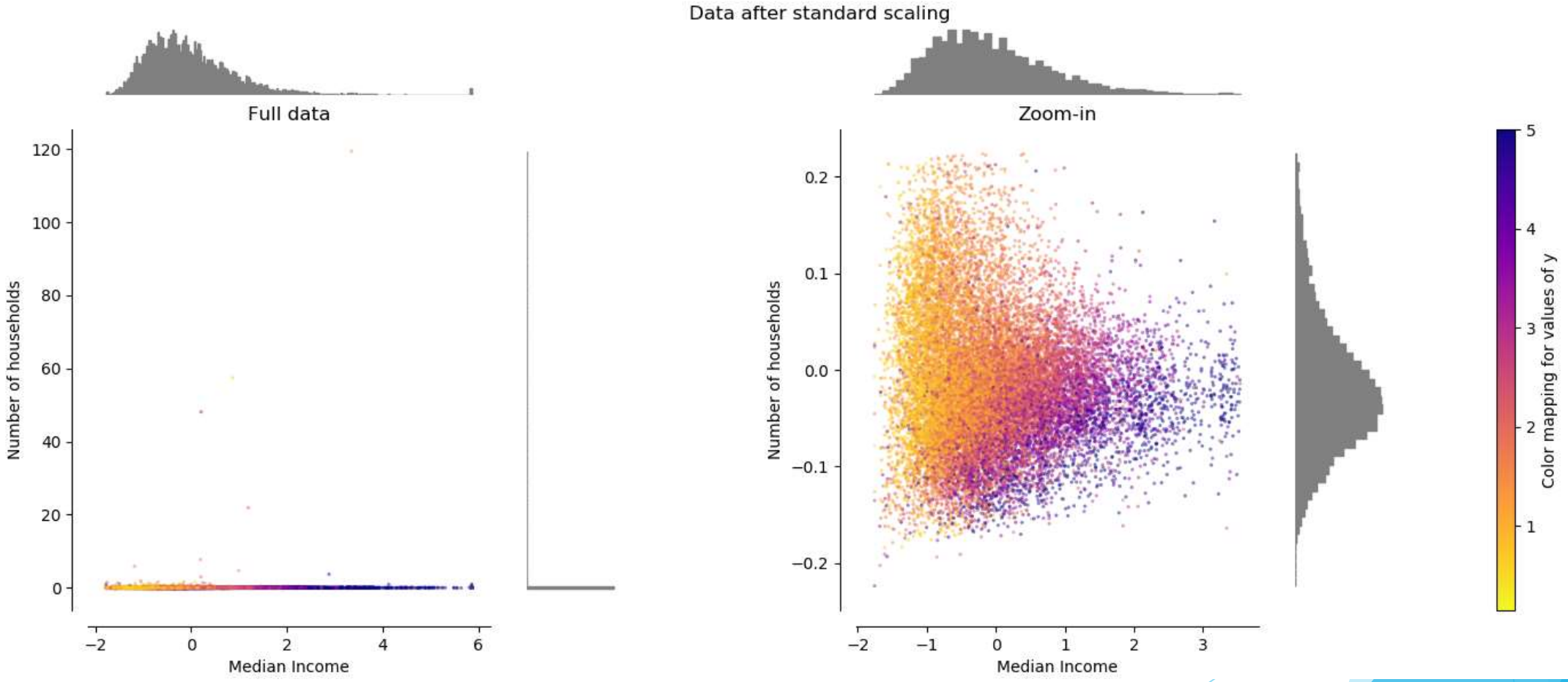
База данных вин

Химический анализ вин трех разных сортов. 1 столбец – сорт вина, столбцы 2-14 содержат данные о следующих признаках:

- Алкоголь
- Яблочная кислота
- Зола
- Щелочность золы
- Магнезия
- Общее содержание фенолов
- Флаваноиды
- Нефлаваноидные фенолы
- Проантоцианины
- Интенсивность цвета
- Оттенок
- OD280 / OD315 разбавленных (разведенных) вин
- Пролин

lection_4_Scale

Пример с большими объемами данных и непрерывной целевой функцией



https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html

Масштабирование по минимаксу

Выполняется преобразование

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

X_{min}, X_{max} – минимальное и максимальное значение, соответственно

Новая величина изменяется в пределах [0, 1]

MinMaxScaler

- fit(X) – расчет минимума и максимума
- transform(X) – выполнение преобразования
- fit_transform(X) – оба действия за один шаг
- inverse_transform(X) – обратное преобразование

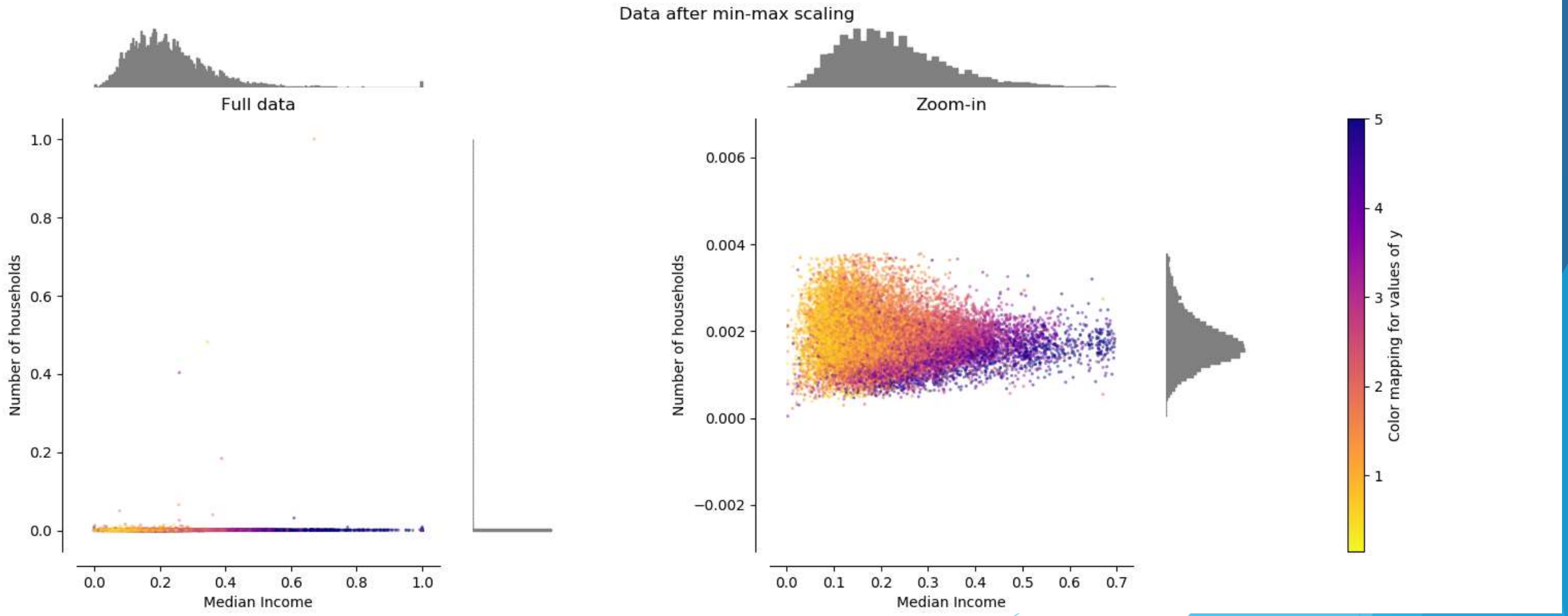
lection_4_Scale

X – массив, размерность [n_samples, n_features]

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

Пример с большими объемами данных и непрерывной целевой функцией

Масштабирование не в [0;1], а в [0;0.005]



https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html

Нормализация

Признаки каждого образца (строка) независимо от других образцов нормируются.

Новые величины имеют единичную длину. Возможна нормализация L1, L2

$$x_j^{(i)\prime} = \frac{x_j^{(i)}}{\sqrt{\left(x_1^{(i)}\right)^2 + \cdots + \left(x_n^{(i)}\right)^2}}$$

lection_4_Scale

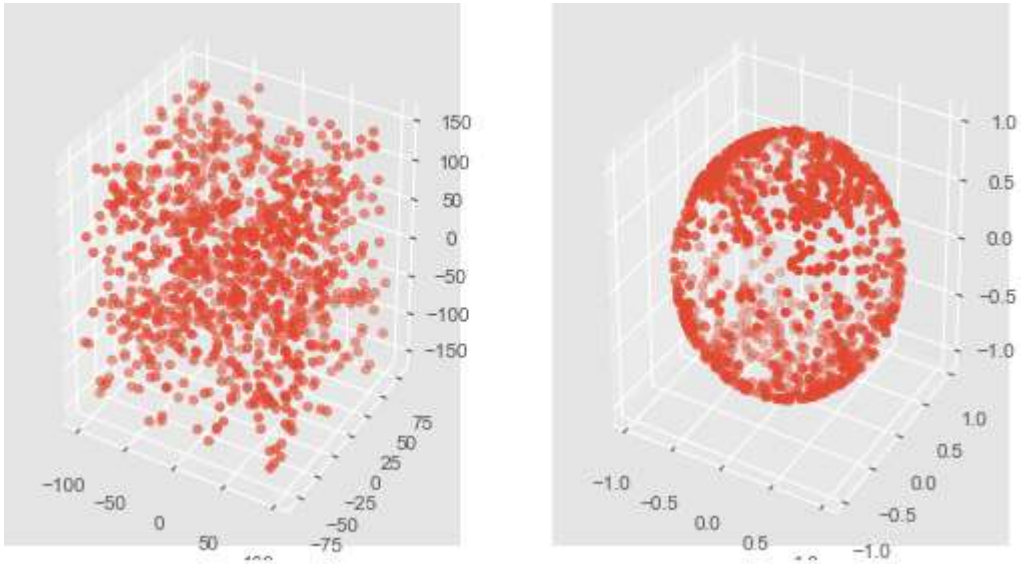
Normalizer

- fit(X) – ничего не делает (для совместимости с другими)
- transform(X) – выполнение нормализацию до единичной длины
- fit_transform(X) – выполнение нормализацию до единичной длины
- обратного преобразования не предусмотрено

X – массив, размерность [n_samples, n_features]

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html

Пример с большими объемами данных



http://benalexkeen.com/feature-scaling-with-scikit-learn/

Какое масштабирование выбрать? Для каких алгоритмов обучения применять?

- StandardScaler – практичен для алгоритмов, использующих градиентный спуск, в т.ч. алгоритмов использующих глубокое обучение. Выбросы влияют на расчет мат. ожидания и стандартного отклонения.
- MinMaxScaler – сохраняет информацию об изначальном распределении данных (просто масштабирует). Не снижает значимость выбросов.
- NormalSclaler – нормализует строку (объект), а не столбец с признаком. Обычно используется в задачах текстовой классификации, где рассчитывается **косинусное сходство** векторов

Для каких алгоритмов обучения масштабирование не обязательно?

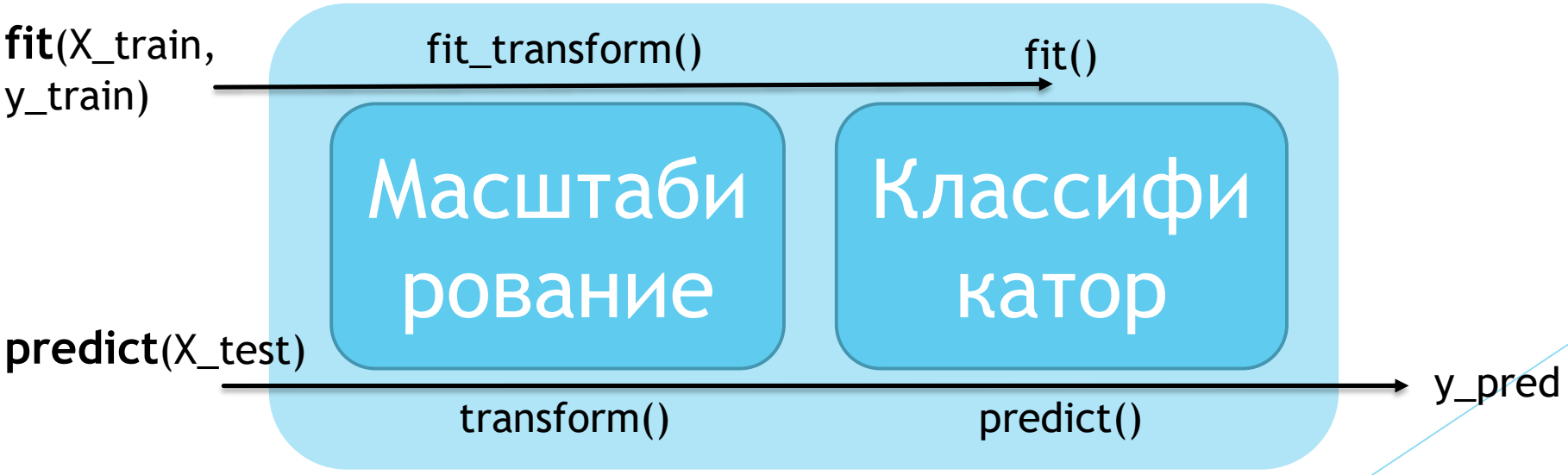
- Деревья принятия решений
- Случайные леса (на базе деревьев)

https://towardsdatascience.com/scale-standardize-or-normalize-with-scikit-learn-6ccc7d176a02

Pipeline

- Объект-конвейер принимает на вход список шагов
- Шаг – кортеж (произвольный строковый идентификатор; преобразователь или оценщик)
- Промежуточные шаги – преобразователи, последний - оценщик

```
Pipeline([ ("scalar", StandardScaler()),  ("estimator", Perceptron()) ])
```



Конвейер

Вопросы

- Признаки у образцов имеют разные масштабы. Чем это плохо? Что с этим можно сделать?

- Какие алгоритмы не подвержены влиянию разных масштабов данных?

Домашнее задание

- Загрузить данные о сортах вин.
- Разбить данные на тренировочный и тестовый набор в соотношении 7/3 с random_state=5
- Сделать массив классификаторов [Perceptron, LogisticRegression, KNeighborsClassifier(n_neighbors=7), DecisionTreeClassifier(max_depth=3)]
- Создать масштабирующие классы StandatdScaler и MinMaxScaler
- Отмасштабировать тренировочные (fit_transform()) и тестовые (transform()) признаки и тем и другим классом
- Поочередно оценить точность прогнозов на немсаштабированных данных, данных после MinMaxScaler, после StandardScaler для алгоритмов из массива
- Отобразить точность для каждого случая (название класса можно вывести как clfr.__class__.__name__)

Calling fit() more than once will overwrite what was learned by any previous fit()
https://scikit-learn.org/stable/tutorial/basic/tutorial.html

Вопросы, на которые нужно дать ответы:

- В каком случае (StandatdScaler/MinMaxScaler/без масштабирования) алгоритмы показали лучший результат
- Как повлияло масштабирование признаков на результаты работы DecisionTreeClassifier
- Для классификатора LogisticRegression сделать конвейер с лучшим алгоритмом масштабирования. Сделать предсказания с помощью конвейера.
  - Из конвейера получить натренированный классификатор (pipline.steps[0] вернет tuple с масштабирующим классом, pipline.steps[1] вернет tuple с классификатором)
  - По весам классификатора (w1,w2…) (3 массива весов, которые отделяют каждый класс от двух других) определить, какие из признаков наиболее значимы (имеют абсолютное значение больше единицы)

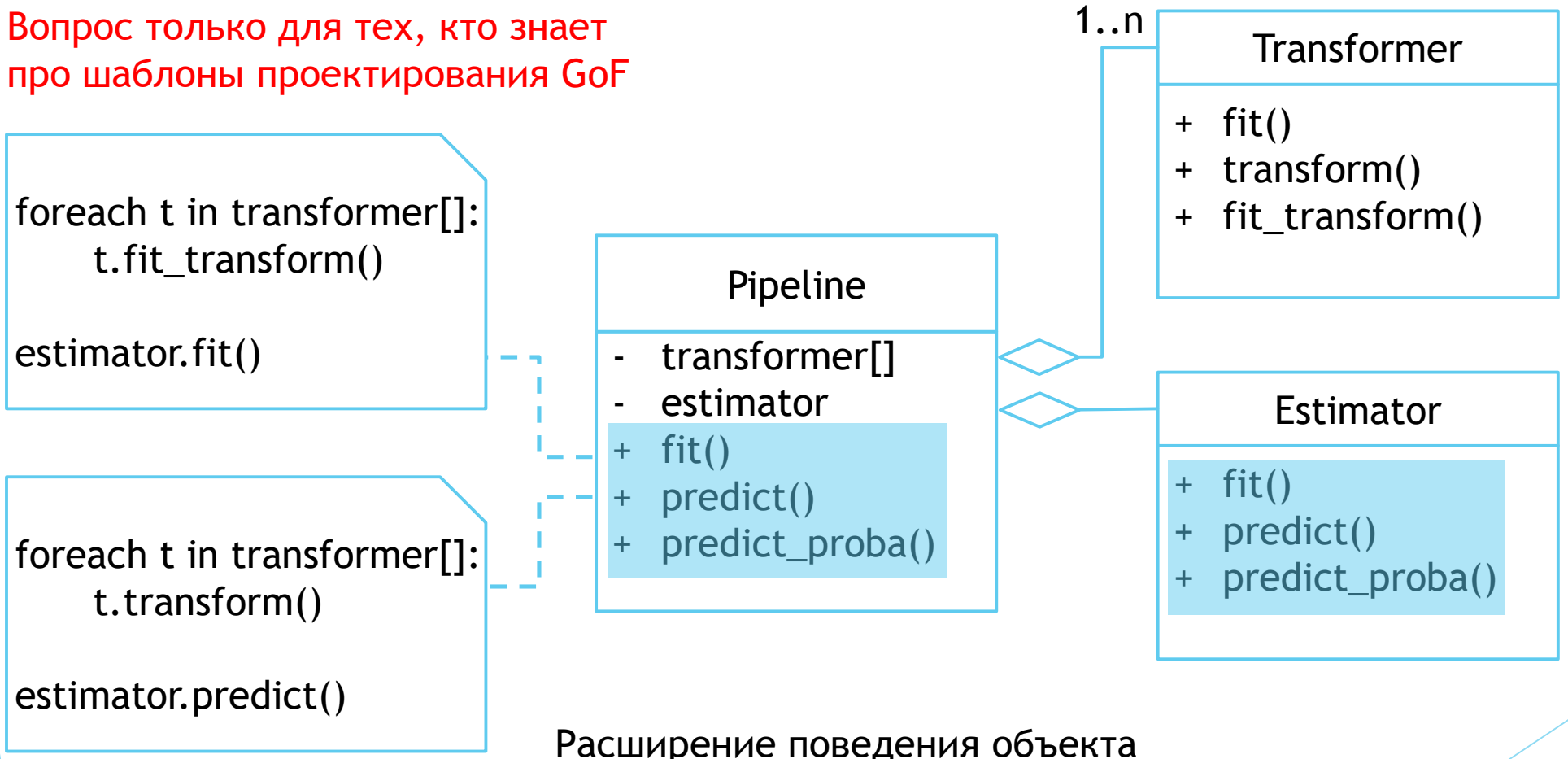
НЕ ПИСАТЬ САМОСТОЯТЕЛЬНО НИКАКИХ ФУНКЦИЙ!

Преобразование весов в индексы делать с помощью numpy см. lection_2_numpy_start
Из c=[0.134, -2.3, 4.31, -0.65, -0.54, 0,23, 1.56] получить c1=[False, True, True, False, False, False, True], Полученный вектор из True/False, передать в качестве индекса в names – узнать наименование признаков (names[c1])

names = np.array(['Алкоголь','Яблочная кислота','Зола','Щелочность золы','Магнезия','Общее содержание фенолов', 'Флаваноиды', 'Нефлаваноидные фенолы', 'Проантоцианины', 'Интенсивность цвета', 'Оттенок', 'OD280 / OD315 разбавленных (разведенных) вин', 'Пролин'])

Pipeline – аналог какого структурного шаблона проектирования?

Вопрос только для тех, кто знает
про шаблоны проектирования GoF

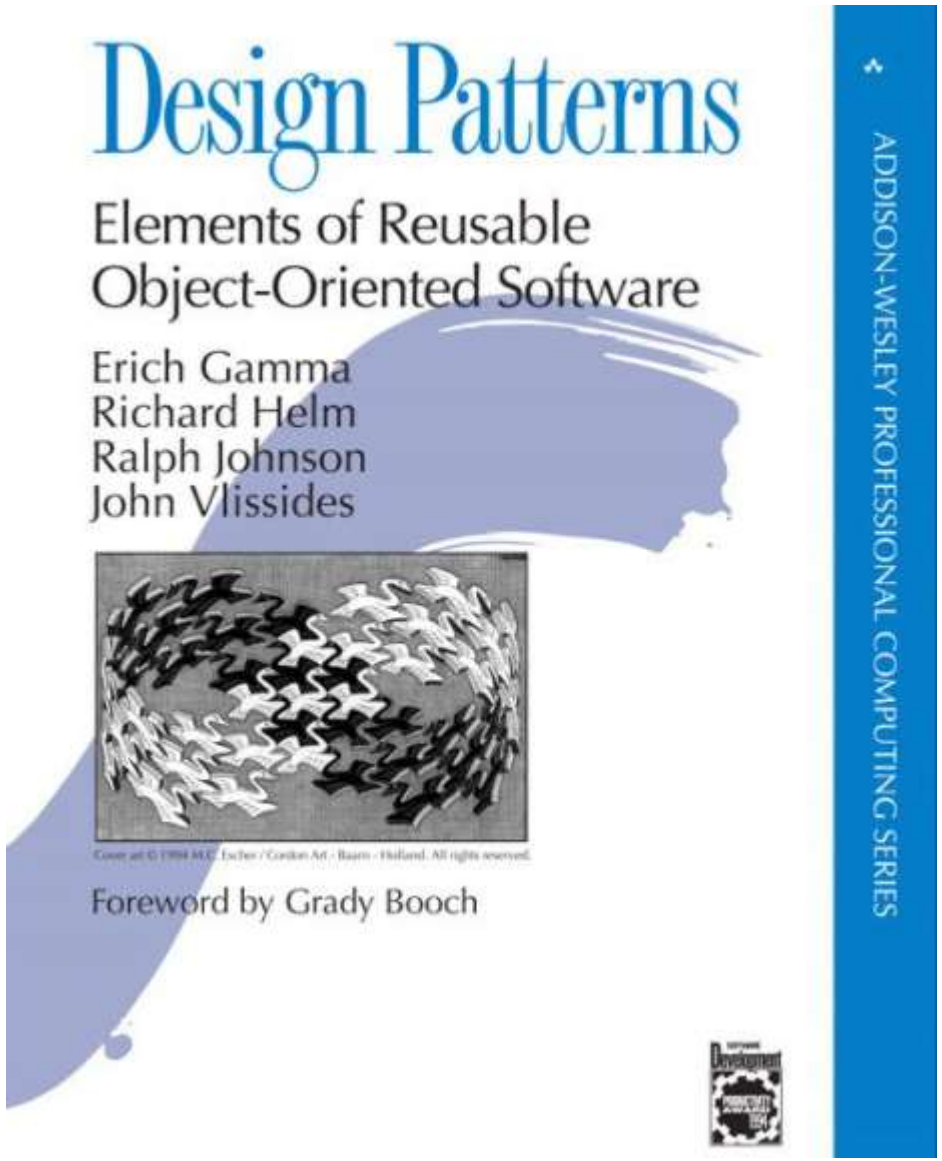


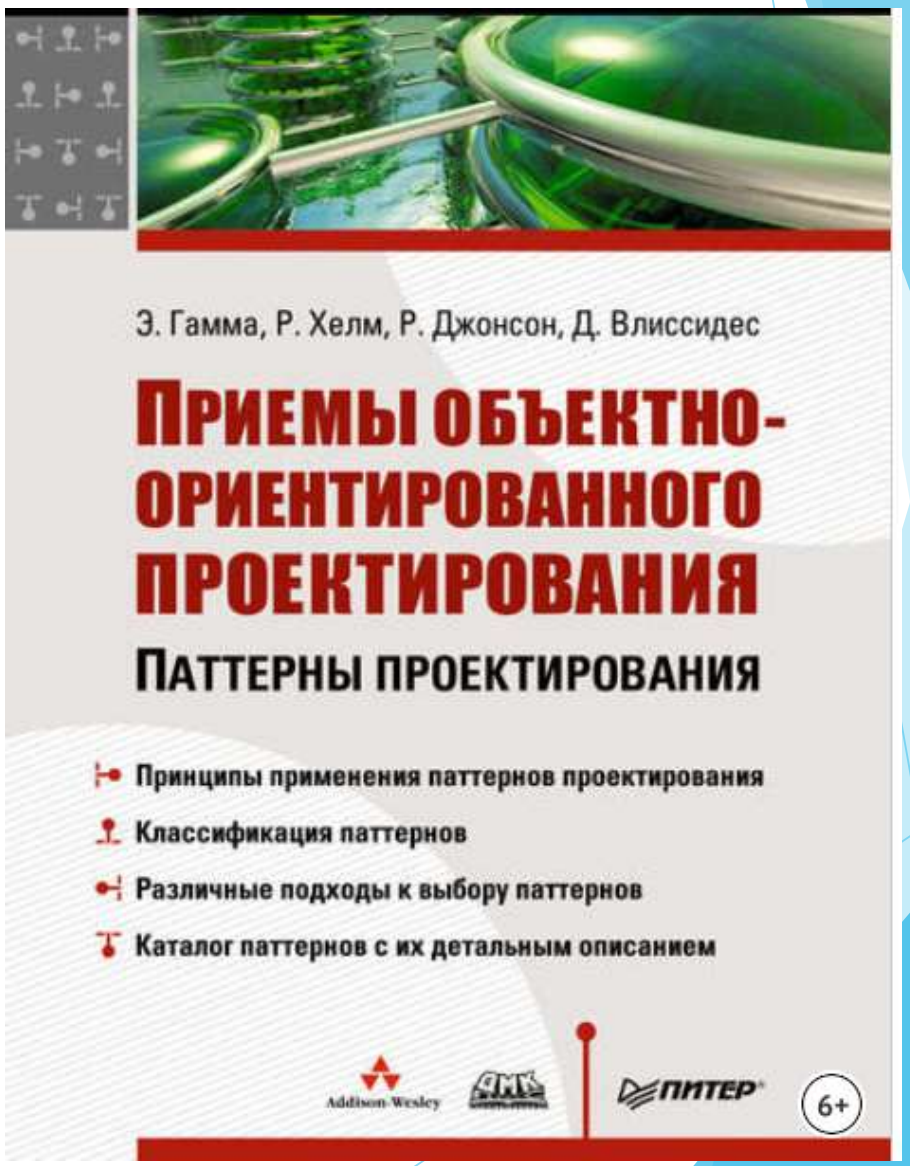
Расширение поведения объекта

Задача – модификация аргументов

Декоратор

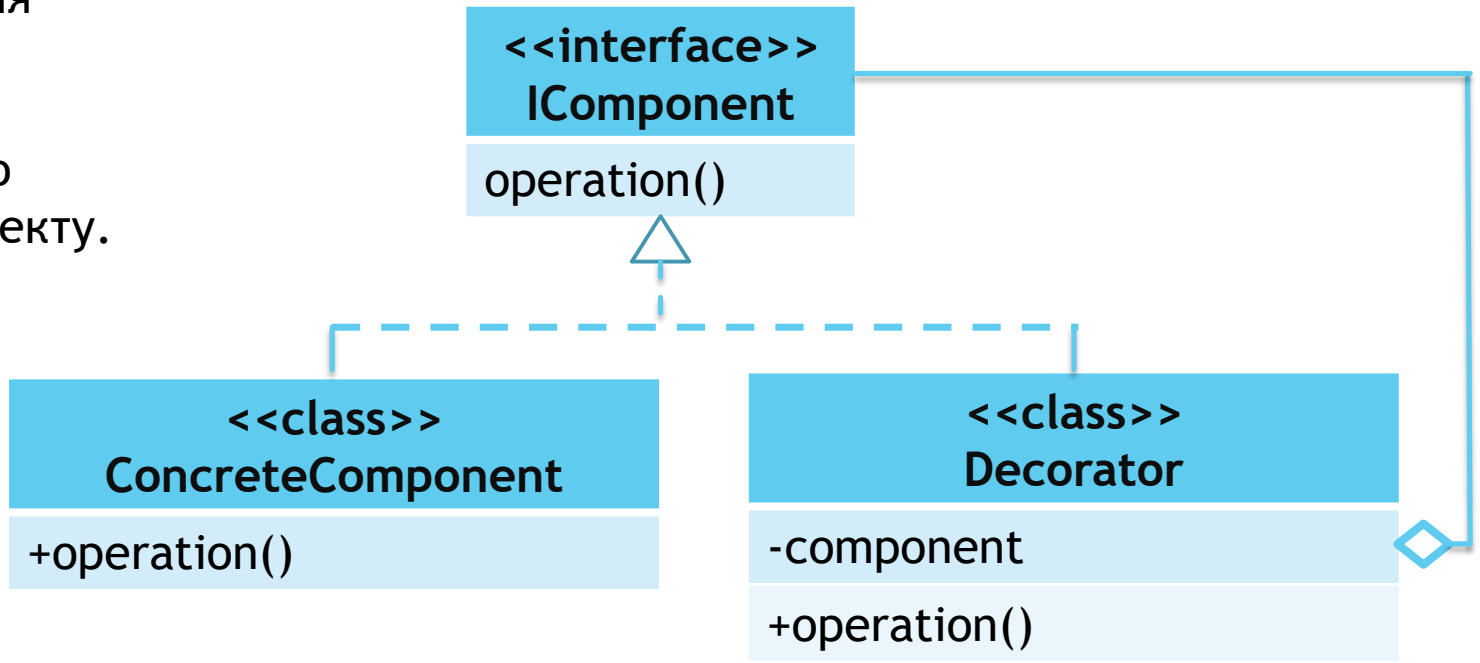
GoF

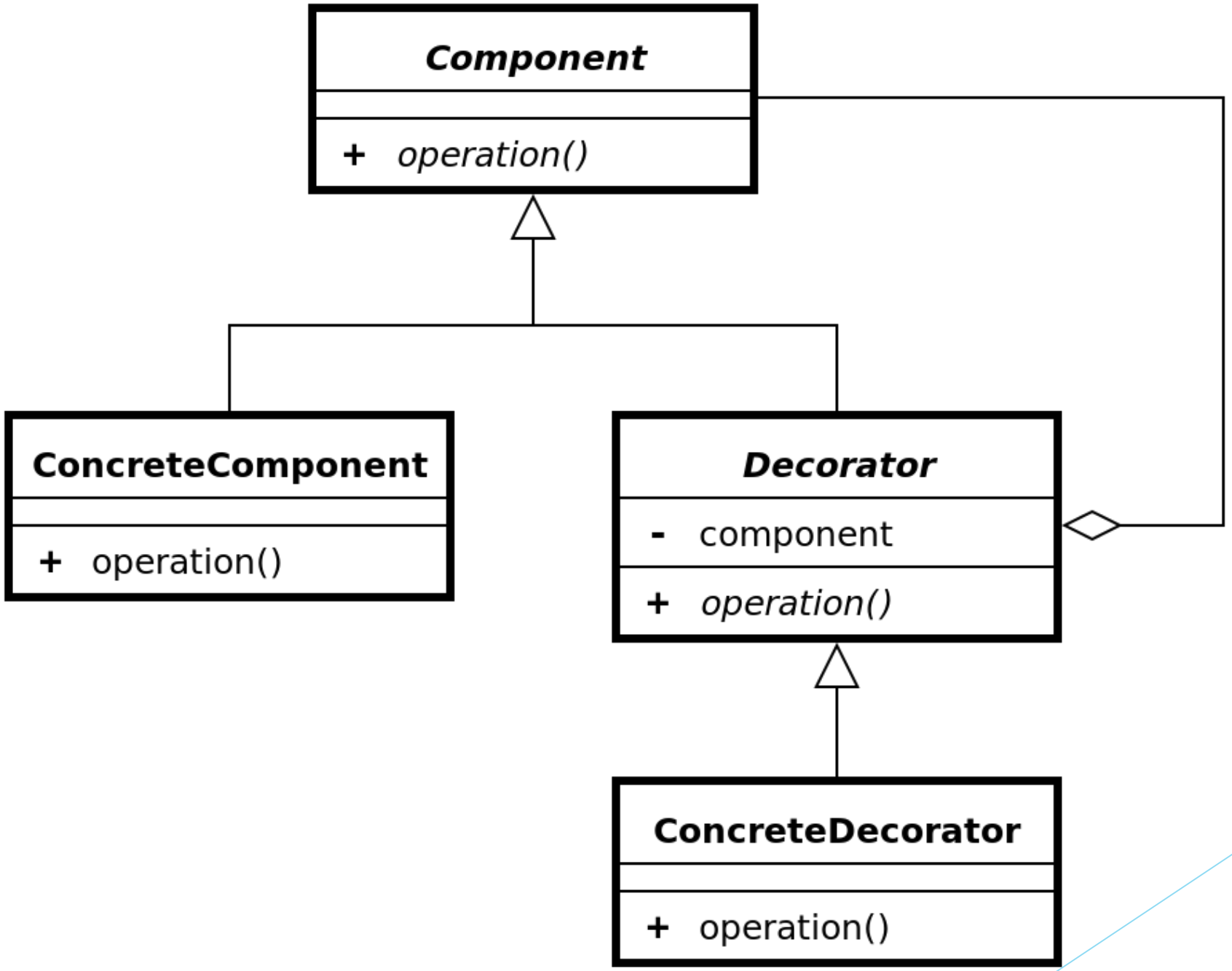

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES


Э. Гамма, Р. Хелм, Р. Джонсон, Д. Влиссидес
ПРИЕМЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОЕКТИРОВАНИЯ
ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ
Принципы применения паттернов проектирования
Классификация паттернов
Различные подходы к выбору паттернов
Каталог паттернов с их детальным описанием
Addison-Wesley
ПИТЕР
6+

Шаблон декоратор, классический вариант

предназначен для динамического подключения дополнительного поведения к объекту.




Component
+ operation()
ConcreteComponent
+ operation()
Decorator
- component
+ operation()
ConcreteDecorator
+ operation()

Бытовой пример

Розетка

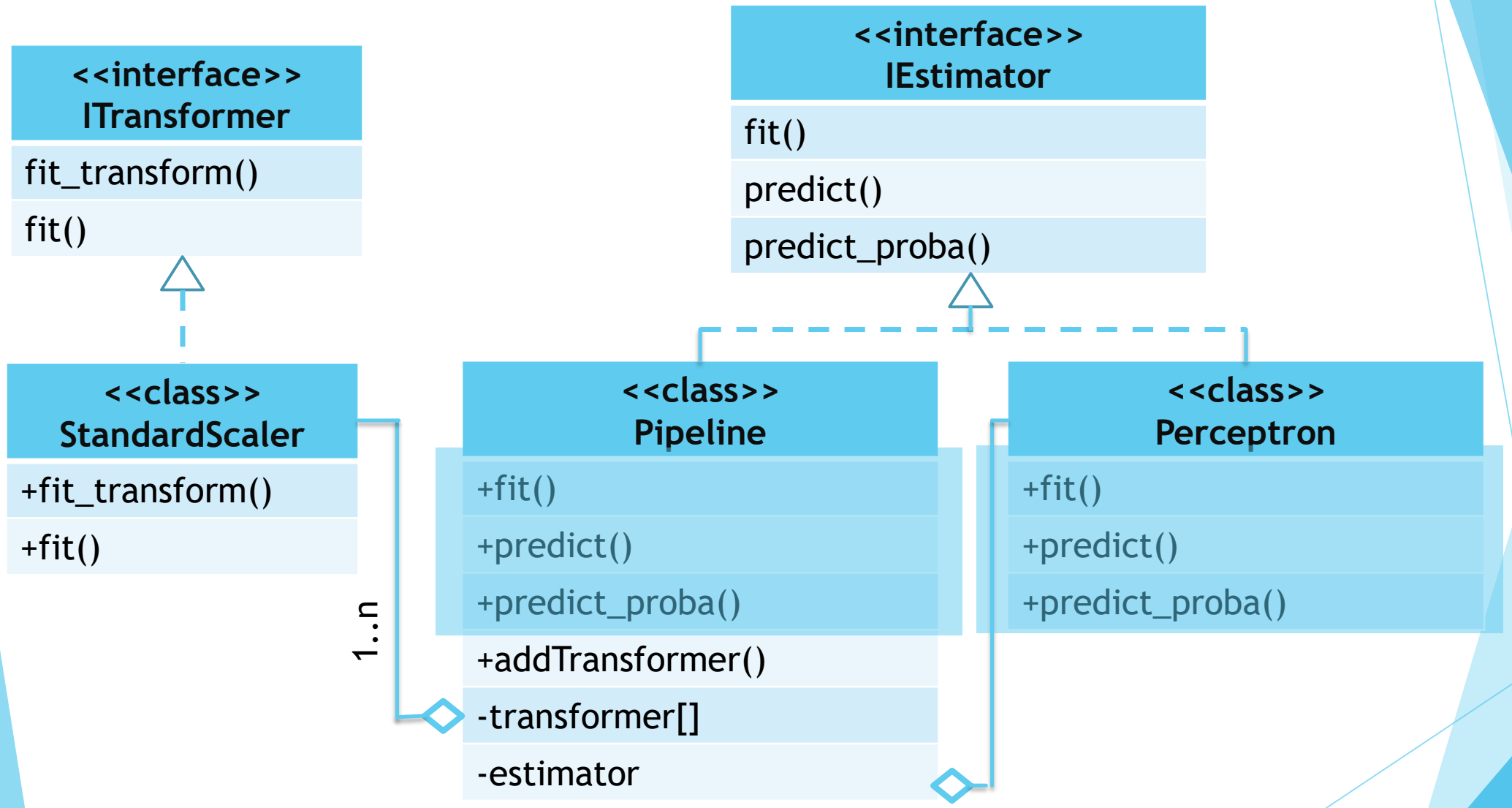
Сетевой фильтр

Умная розетка

Розетка ваттметр

Водонагреватель на даче

Шаблон декоратор



*В самом Python интерфейса IEstimator нет, шаги также нельзя добавлять динамически

Другие шаблоны и техники…