**Стандартизация**

Алгоритм K-ближайших соседей делает прогнозы относительно точки данных (семпла), используя наиболее близкие к ней наблюдения.

**Стандартизация** набора данных – это корректировка каждого значения x так, чтобы они находились примерно в одном диапазоне.

StandardScaler преобразует значения таким образом, что из каждого Наблюдения (Observation) каждого признака вычитается среднее арифметическое и результат делится на стандартное отклонение этого признака.

Создадим экземпляр класса StandardScaler с именем scaler, который можем обучить на нашем датасете, используя метод fit.

модель смотрит на x\_train и y\_train и пытается найти какие-то взаимосвязи или настроить коэффициенты

scaler.fit(iris\_csv.drop('variety', axis = 1)) # обучение датасета (без последнего столбца)

axis=1 : применить вычисление «построчно»

fit\_transform – обучаем\_преобразуем

Теперь мы можем применить метод transform для стандартизации всех признаков, чтобы они имели примерно одинаковый масштаб. Мы сохраним преобразованные семплы в переменной scaled\_features

В качестве результата мы получили массив NumPy со всеми точками данных из датасета, но нам желательно преобразовать его в формат DataFrame библиотеки pandas.

Чтобы оценить модель, вы обучаете её на тренировочном множестве, а результаты измеряете на тестовом. У sklearn по дефолту выставлен параметр shuffle=True, то есть перед разделением на тренировочное и тестовое множества происходит перемешивание семплов (и для воспроизводимости такого разбиения нужно фиксировать random\_state).

random\_state присваивается целочисленное значение, чтобы гарантировать воспроизводимость результатов псевдослучайной генерации. При повторном использовании начального значения одна и та же последовательность должна воспроизводиться от запуска к запуску, пока не запущено несколько потоков. любой, кто повторно запустит код, получит точно такие же результаты.

**Разделение датасета на обучающие и тестовые данные**

train - обучающая выборка, test - тестовая выборка

train\_test\_split - Эта функция используется для разделения набора данных на наборы для обучения и тестирования.

Значения x представляют собой DataFrame scaled\_data, который мы создали ранее. Значения y хранятся в столбце " variety " исходной таблицы

будем использовать test\_size 30%:

## X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, stratify=y,random\_state=42)

X и y – это исходные данные и метки классов, а random\_state – это параметр, который гарантирует одинаковое разделение при каждом запуске.

**Стратификация** — это процесс разделения исходной совокупности на несколько однородных групп (страт) перед процедурой сэмплинга.

Для стратифицированного разделения мы также можем использовать функцию train\_test\_split, указав параметр stratify. Теперь в каждой из выборок будет одинаковое соотношение классов, что может быть полезным, если у нас есть несбалансированные данные.

**Обучение модели K-ближайших соседей**

Обучение модели K-ближайших соседей

kNN - k Nearest Neighbor - k Ближайших Соседей — это один из алгоритмов классификации. Задача классификации— это отнесение объекта к одному из заранее определенных классов на основании его формализованных признаков

алгоритм будет использоваться для построения модели из обучающих данных, а также для прогнозирования новых точек данных. Он также будет содержать информацию, которую алгоритм извлек из обучающих данных. В случае KNeighborsClassifier он будет просто хранить учебный набор.

Для построения модели на обучающем множестве вызывается метод fit объекта knn, который принимает в качестве аргументов массив NumPy x\_train, содержащий обучающие данные, и массив NumPy y\_train соответствующих обучающих меток.

accuracy\_score - Классификационная оценка точности.

При классификации по нескольким меткам эта функция вычисляет точность подмножества: набор меток, предсказанный для выборки, должен точно соответствовать соответствующему набору меток в y\_test. Возвращает долю правильно классифицированных выборок (с плавающей запятой)

confusion\_matrix – матрица ошибок

Матрица ошибок — это способ разбить объекты на четыре категории в зависимости от комбинации истинного ответа и ответа алгоритма.

1 ошибка – 3 класс стал 2

