Классификатор ближайших соседей

Классификация по соседству – это тип обучения, основанный на основе экземпляров: не строит общую внутреннюю модель, а просто хранит экземпляры обучающих данных. Классификация объектов выполняется следующим образом: каждому объекту присваивается такой класс, который имеет наибольшее количество представителей среди ближайших соседей объекта.

Scikit-learn представляет два разных классификатора ближайших соседей: kNN реализует обучение, опираясь на k ближайших соседей каждого объекта. k указывается пользователем и может быть только целым числом. Классификатор радиуса ближайших соседей определяет количество соседей в пределах фиксированного радиуса r для каждого объекта обучения. Значение r указывается пользователем и является не целым значением.

Классификация k-соседей в kNN чаще всего используется для двух методов. Оптимальный выбор значения k зависит от данных. Большое значение k подавляет эффекты шума, но граница между разными классами становится менее отчетлива. В классификаторе радиуса ближайших соседей значение r задается пользователем так, чтобы точки (объекты) в более редких окрестностях использовали меньшее количество ближайших соседей для классификации. Но этот метод не подходит для больших выборок.

В задаче классификации ближайших соседей используются однородные веса: значение веса для каждой точки равно количеству «голосов» ближайших соседей этой точки. Бывают случаи, когда лучше соседям, расположенным ближе всего к объекту, присваивать большее значение весов, чем более отдаленным точкам. Все это выполнимо с помощью использования переменной weight = «равномерность»: каждому соседу присваиваются одинаковые веса (все точки в каждой окрестности взвешены одинаково).

Weight = «расстояние»: значение веса равно обратному расстоянию от классифицируемой точки до всех остальных точек окрестности. Обычно, для вычисления весов используется функция расстояния.

kNN – классификатор ближайших соседей

**KNeighborsClassifier**(*n\_neighbors=5*, *weights=’uniform’*, *algorithm=’auto’*, *leaf\_size=30*, *p=2*, *metric=’minkowski’*, *metric\_params=None*, *n\_jobs=1*, *\*\*kwargs*)

Параметры:

**n\_neighbors**: целое, необязательное (по умолчанию = 5)

Количество соседей, используемых по умолчанию для запросов [**kneighbors**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.kneighbors).

**Weights**: строка или пользовательская функция

Весовая функция, используемая в прогнозировании. Возможные значения:

* “uniform”: равномерные веса. Все точки в каждой окрестности оцениваются одинаково.
* “distance”: вес точек указывает на обратное расстояние. В этом случае, более близкие соседи точки будут иметь большее влияние, чем соседи, находящиеся дальше.
* [callable] (пользовательская функция): функция, определяемая пользователем, которая в качестве входных данных использует массив расстояний и возвращает массив такого же размера, содержащий веса.

**algorithm** : {‘auto’, ‘ball\_tree’, ‘kd\_tree’, ‘brute’}

Алгоритм, используемый для вычисления ближайших соседей:

* “ball\_tree” используется в функции BallTree.
* “kd\_tree” используется в функции KDTree.
* “brute” использует поиск перебором.
* “auto” пытается подобрать наиболее подходящий алгоритм, опираясь на значения, передаваемые методу fit.

**leaf\_size**: целое

Может повлиять на скорость построения запроса, а также на память, необходимую для хранения дерева.

**p**: целое

Параметр мощности для показателя Минковского. Для р=1 применяется «манхэттонское расстояние», и эвклидово расстояние для р=2. В остальных случаях, для произвольного р, используется «минковское расстояние».

**metric**: строка или пользовательская функция

расстояние, используемое в дереве. По умолчанию используется метрика Минковского, в для р=2 – Эвклидова метрика.

**metric\_params**:

Дополнительные ключевые аргументы для функции метрики.

**n\_jobs**: целое

Число параллельных задач для поиска соседей. Если равно -1, то число задач зависит от количества ядер процессора.

Методы:

[**fit**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.fit)(X, y): подгонка модели, используя Х в качестве обучающей выборки, и у в качестве целевых значений.

[**get\_params**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.get_params)([deep]): получает данные для этого метода оценивания.

[**kneighbors**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.kneighbors)([X, n\_neighbors, return\_distance]): находит k-соседей точки.

[**kneighbors\_graph**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.kneighbors_graph)([X, n\_neighbors, mode]): вычисляет (взвешенный) граф k-соседей для точек из Х.

[**predict**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.predict)(X): определяет метки классов для предоставленных данных.

[**predict\_proba**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.predict_proba)(X): возвращаются оценки вероятности для проверки данных Х.

[**score**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.score)(X, y[, sample\_weight]): возвращает среднюю точность тестовых данных и меток.

[**set\_params**](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier.set_params)(\*\*params): необходимо задать параметры для этой оценки.



