Выбор числа соседей

Данное задание основано на материалах лекций по метрическим методам и посвящено подбору числа соседей в методе kNN.

Вы научитесь:

- работать со методом k ближайших соседей
- выбирать в нем параметр k
- правильно готовить данные перед использованием данного метода

Введение

Метрические методы основаны на гипотезе компактности, суть которой состоит в том, что объекты с похожими признаковыми описаниями имеют похожие значения целевой переменной. Если эта гипотеза верна, то строить прогноз для нового объекта можно на основе близких к нему объектов из обучающей выборки — например, путем усреднения их ответов (для регрессии) или путем выбора наиболее популярного среди них класса (для классификации). Методы такого типа и называются метрическими. Они имеют несколько особенностей:

- Процедура обучения, по сути, отсутствует — достаточно лишь запомнить все объекты обучающей выборки
- Можно использовать метрику, учитывающую особенности конкретного набора данных — например, наличие категориальных (номинальных) признаков

- При правильном выборе метрики и достаточном размере обучающей выборки метрические алгоритмы показывают качество, близкое к оптимальному

Метрические методы чувствительны к масштабу признаков — так, если масштаб одного из признаков существенно превосходит масштабы остальных признаков, то их значения практически не будут влиять на ответы алгоритма. Поэтому важно производить масштабирование признаков. Обычно это делается путем вычитания среднего значения признака и деления на стандартное отклонение.

Реализация в Scikit-Learn

Метод k ближайших соседей реализован в классе sklearn.neighbors.KNeighborsClassifier. Основным параметром является n_neighbors, который задает число соседей для построения прогноза.

Вам понадобится производить кросс-валидацию по блокам. Кросс-валидация заключается в разделении выборки на m непересекающихся блоков примерно одинакового размера, после чего выполняется m шагов. На i-м шаге i-й блок выступает в качестве тестовой выборки, объединение всех остальных блоков — в качестве обучающей выборки. Соответственно, на каждом шаге алгоритм обучается на некоторой обучающей выборке, после чего вычисляется его качество на тестовой выборке. После выполнение m шагов мы получаем m показателей качества, усреднение которых и дает оценку кросс-валидации. Подробнее вы можете послушать про кросс-валидацию в видео "Проблема переобучения. Методология решения задач машинного обучения"из первого модуля, а также почитать на Википедии (на русском или на английском) или в документации scikit-learn.

Вам понадобится производить кросс-валидацию по блокам. Это делается в два этапа:

1. Создается генератор разбиений sklearn.cross_validation.KFold, который задает набор разбиений на обучение и валидацию. Число блоков в кросс-валидации определяется параметром n_folds. Обратите внимание, что порядок следования объектов в выборке может быть неслучайным, это может привести к смещенности кросс-валидационной оценки. Чтобы устранить такой эффект, объекты

выборки случайно перемешивают перед разбиением на блоки. Для перемешивания достаточно передать генератору KFold параметр shuffle=True.

2. Вычислить ошибку на всех разбиениях можно при помощи функции sklearn.cross_validation.cross_val_score. В качестве параметра estimators передается классификатор, в качестве параметра cv — генератор разбиений с предыдущего шага. С помощью параметра scoring можно задавать меру качества, по умолчанию в задачах классификации используется доля верных ответов (accuracy). Результатом является массив, значения которого нужно усреднить.

Приведение признаков к одному масштабу можно делать с помощью функции sklearn.preprocessing.scale, которой на вход необходимо подать матрицу признаков и получить масштабированную матрицу, в которой каждый столбец имеет нулевое среднее значение и единичное стандартное отклонение.

Инструкция по выполнению

В этом задании вам нужно подобрать оптимальное значение k для алгоритма kNN. Будем использовать набор данных Wine, где требуется предсказать сорт винограда, из которого изготовлено вино, используя результаты химических анализов.

Выполните следующие шаги:

1. Загрузите выборку Wine по адресу https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

2. Извлеките из данных признаки и классы. Класс записан в первом столбце (три варианта), признаки — в столбцах со второго по последний. Более подробно о сути признаков можно прочитать по адресу https://archive.ics.uci.edu/ml/datasets/Wine

3. Оценку качества необходимо провести методом кросс-валидации по 5 блокам (5-fold). Создайте генератор разбиений, который перемешивает выборку перед формированием блоков (shuffle=True). Для

воспроизводимости результата, создавайте генератор KFold с фиксированным параметром random_state=42. В качестве меры качества используйте долю верных ответов (accuracy).

4. Найдите точность классификации на кросс-валидации для метода k ближайших соседей (sklearn.neighbors.KNeighborsClassifier), при k от 1 до 50. При каком k получилось оптимальное качество? Чему оно равно (число в интервале от 0 до 1)? Данные результаты и будут ответами на вопросы 1 и 2.

5. Произведите масштабирование признаков с помощью функции sklearn.preprocessing.scale. Снова найдите оптимальное k на кросс-валидации.

6. Какое значение k получилось оптимальным после приведения признаков к одному масштабу? Как изменилось значение качества? Приведите ответы на вопросы 3 и 4.

Если ответом является нецелое число, то целую и дробную часть необходимо разграничивать точкой, например, 0.5. При необходимости округляйте дробную часть до двух знаков.

Ответ на каждое задание — текстовый файл, содержащий ответ в первой строчке. Обратите внимание, что отправляемые файлы не должны содержать пустую строку в конце. Данный нюанс является ограничением платформы Coursera. Мы работаем над тем, чтобы убрать это ограничение.