1 слайд

Кросс-валидация — это метод, предназначенный для оценки качества работы модели, широко применяемый в машинном обучении. Он помогает сравнить между собой различные модели и выбрать наилучшую для конкретной задачи. В этом разделе мы рассмотрим наиболее распространённые методы кросс-валидации, а также обсудим возможные проблемы, которые могут возникнуть в процессе их применения.

2 слайд

Hold-out

Метод **hold-out** представляет из себя простое разделение на train и test:

Такое разделение очень легко реализовать с помощью [sklearn](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html):

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).

3 слайд

Если вы перебираете какие-то модели для вашей задачи, то оптимизировать их качества стоит на валидационном множестве, а окончательное сравнение моделей проводить на тестовом множестве. Оптимизация качеств модели может включать в себя подбор гиперпараметров, подбор архитектуры (в случае нейросеток), подбор оптимального трешолда для максимизации значений целевой метрики (например, вы делаете двуклассовую классификацию, а модель выдаёт непрерывные значения от 0 до 1, которые нужно бинаризовать так, чтобы получить максимальный скор по F1) и т. д. Если же оптимизировать качества моделей и проводить их сравнение на одном и том же множестве, то можно неявно заложить в модели информацию о тестовом множестве и получить результаты хуже ожидаемых на новых данных.

4 слайд

При простом случайном разделении на тренировочное и тестовое множества (как в примерах выше) может случиться так, что распределения тренировочного и тестового множеств окажутся не такими, как у всего исходного множества. Проиллюстрируем такую ситуацию на примере случайного разбиения датасета [Iris](https://archive.ics.uci.edu/ml/datasets/iris) на трейн и тест. Распределение классов в данном датасете равномерное:

* 33.3 Setosa
* 33.3 Versicolor
* 33.3 Virginica

Случайное разбиение, в котором две трети цветов (100) отправились в трейн, а оставшаяся треть (50) отправилась в тест, может выглядеть, например, так:

* трейн: 38 × Setosa, 28 × Versicolor, 34 × Virginica (распределение 38)
* тест: 12 × Setosa, 22 × Versicolor, 16 × Virginica (распределение 24)

Если распределение цветов в исходном датасете отражает то, что в природе они встречаются одинаково часто, то мы только что получили два новых датасета, не соответствующих распределению цветов в природе. Распределения обоих датасетов вышли не только несбалансированными, но ещё и разными: самый частый класс в трейне соответствует наименее частому классу в тесте.

На помощь в такой ситуации может прийти **стратификация**: разбиение на трейн и тест, сохраняющее соотношение классов, представленное в исходном датасете. В sklearn такое разбиение можно получить с помощью параметра stratify:

В целом на достаточно больших датасетах (порядка хотя бы 10 тысяч семплов) со сбалансированными классами можно не очень сильно беспокоиться об описанной выше проблеме и использовать обычный random split. Но если у вас очень несбалансированные данные, в которых один класс встречается сильно чаще другого (как, например, в задачах фильтрации спама или сегментации осадков на спутниковых снимках), стратификация может довольно сильно помочь

5 слайд.

Метод **k-Fold** чаще всего имеют в виду, когда говорят о кросс-валидации. Он является обобщением метода hold-out и представляет из себя следующий алгоритм:

1. Фиксируется некоторое целое число k (обычно от 5 до 10), меньшее числа семплов в датасете.
2. Датасет разбивается на k одинаковых частей (в последней части может быть меньше семплов, чем в остальных). Эти части называются *фолдами*.
3. Далее происходит k итераций, во время каждой из которых один фолд выступает в роли тестового множества, а объединение остальных — в роли тренировочного. Модель учится на k−1 фолде и тестируется на оставшемся.
4. Финальный скор модели получается либо усреднением k получившихся тестовых результатов, либо измеряется на отложенном тестовом множестве, не участвовавшем в кросс-валидации.

В коде выше получилось два фолда: в первый вошли объекты с индексами 2 и 3, во второй — объекты с индексами 0 и 1. На первой итерации алгоритма фолд с индексами 2 и 3 будет тренировочным, а на второй — фолд с индексами 0 и 1.

6 слайд

В sklearn есть также метод cross\_val\_score, принимающий на вход классификатор, данные и способ разбиения данных (либо число фолдов) и возвращающий результаты кросс-валидации:

Метод k-Fold даёт более надёжную оценку качества модели, чем hold-out, так как обучение и тест модели происходят на разных подмножествах исходного датасета. Однако проведение k итераций обучения и теста может быть вычислительно затратным, и поэтому метод обычно применяют либо когда данных достаточно мало, либо при наличии большого количества вычислительных ресурсов, позволяющих проводить все k итераций параллельно. В реальных задачах данных зачастую достаточно много для того, чтобы hold-out давал хорошую оценку качества модели, поэтому k-Fold в больших задачах применяется не очень часто.

7 слайд

Метод **leave-one-out (LOO)** является частным случаем метода k-Fold: в нём каждый фолд состоит ровно из одного семпла. LOO тоже есть в sklearn:

Этот метод может понадобиться в случае, если у вас очень мало данных (например, в задаче сегментации клеток на изображениях с оптического микроскопа) и вы хотите использовать максимальное их количество для обучения модели: для валидации на каждой итерации методу требуется всего один семпл. Однако и итераций будет столько, сколько семплов в данных, поэтому метод неприменим для средних и больших задач.

8 слайд

Метод **stratified k-Fold** — это метод k-Fold, использующий стратификацию при разбиении на фолды: каждый фолд содержит примерно такое же соотношение классов, как и всё исходное множество (подробнее о стратификации говорилось выше). Такой подход может потребоваться в случае, например, очень несбалансированного соотношения классов, когда при обычном random split некоторые фолды могут либо вообще не содержать семплов каких-то классов, либо содержать их слишком мало. Этот метод также представлен в sklearn:

9 слайд

### Кросс-валидация на временных рядах

Существует такая задача, как прогнозирование временных рядов. На практике она часто возникает в форме «Что будет с показателями нашего продукта в ближайший день / месяц / год?». При этом имеются какие-то исторические данные этих показателей за предыдущее время, которые можно визуализировать в виде некоторого графика по времени:

Этот график — пример графика временного ряда, и наша задача — спрогнозировать, как будет выглядеть данный график в будущие моменты времени. Кросс-валидация моделей для такой задачи осложняется тем, что данные не должны пересекаться по времени: тренировочные данные должны идти до валидационных, а валидационные — до тестовых. С учётом этих особенностей фолды в кросс-валидации для временных рядов располагаются вдоль временной оси так, как показано на следующей картинке:

10 слайд

В sklearn реализована такая схема кросс-валидации: