Машинное обучение, ФКН ВШЭ
Семинар №6

1 Калибровка вероятностей

Часто при обучении моделей для бинарной классификации хочется получать не только предсказанную метку класса, но и вероятность положительного класса. Предсказанная вероятность может служить как мера уверенности нашего алгоритма. Однако некоторые алгоритмы не выдают корректные вероятности классов. В таком случае калибруют вероятности модели.

Для начала определимся с тем, что хотим получить от предсказанных вероятностей. В задаче бинарной классификации откалиброванным алгоритмом называют такой алгоритм, для которого доля положительных примеров (на основе реальных меток классов) для предсказаний в окрестности произвольной вероятности p совпадает с этим значением p. Например, если взять объекты, для которых предсказанные вероятности близки к 0.7, то окажется, что среди них 70% принадлежат положительному классу. Нет критерия, которое бы установило откалиброванность алгоритма, однако можно построить калибровочную кривую. На этой кривой абсцисса точки соответствуют значению p (предсказаний алгоритма), а ордината соответствует доле положительных примеров, для которых алгоритм предсказал вероятность, близкую к p. В идеальном случае эта кривая совпадает с прямой $y = x$. Примеры такой кривой на рис. (1).

Изучим два стандартных метода для калибровки вероятностей алгоритма: калибровка Платта и изотоническая регрессия.

§1.1 Калибровка Платта

Пусть наш алгоритм выдаёт значения $f(x)$ (могут не быть вероятностями). Тогда итоговая вероятность:

$$P(y = 1|x) = \frac{1}{1 + \exp(af(x) + b)},$$

где a, b – скалярные параметры. Эти параметры настраиваются методом максимума правдоподобия (минимизируя логистическую функцию потерь) на отложенной выборке или с помощью кросс валидации. Также Платт предложил настраивать параметры на обучающей выборке базовой модели, а для избежания переобучения изменить метки объектов на следующие значения:

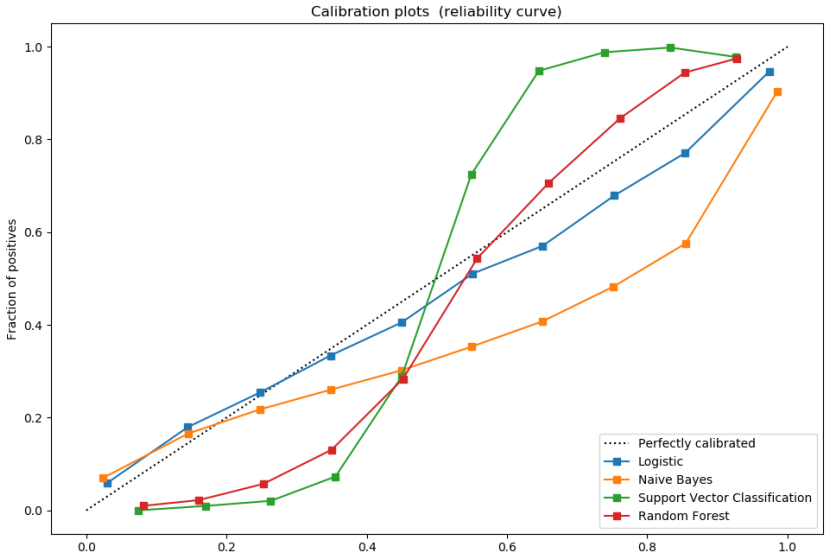

Рис. 1. Калибровочные кривые нескольких алгоритмов

$$t_+ = \frac{N_+ + 1}{N_- + 2}$$

для положительных примеров и

$$t_- = \frac{1}{N_- + 2}$$

для отрицательных.

Калибровку Платта можно представить как применения логистической регрессии поверх предсказаний другого алгоритма с отключенной регуляризацией.

§1.2 Изотоническая регрессия

В этом методе также строится отображение из предсказаний модели в откалиброванные вероятности. Для этого используем изотоническую функцию (неубывающая кусочно-постоянная функция), в которой x – выходы нашего алгоритма, а y – целевая переменная. Иллюстрация изотонической регрессии на рис. (2).

Мы хотим найти такую функцию $m(t)$: $P(y = 1|x) = m(f(x))$. Она настраивается под квадратичную ошибку:

$$m = \arg\min_z \sum (y_i - z(f(x_i))^2,$$

с помощью специального алгоритма (Pool-Adjacent-Violators Algorithm), изучать который в этом курса не будем.

В результате калибровки получаем надстройку над нашей моделью, которая применяется поверх предсказаний базовой модели. В случае мультиклассовой классификации каждый класс калибруется отдельно против остальных (one-versus-all), вероятности при предсказании нормируются.

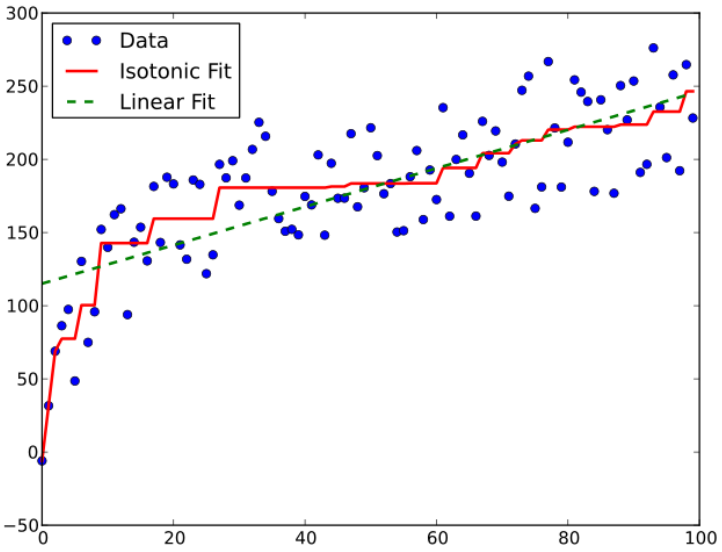

Рис. 2. Изотоническая регрессия

2 Обработка категориальных признаков

Часто в данных встречаются категориальные по смыслу признаки. Если они представлены в виде чисел, то, подавая напрямую в модель, задаём порядок над этими категориями, что обычно неправильно. Например, если красный цвет кодировался как «1», зелёный – «2», а синий – «3», то модель будет считать зелёный цвет находящимся ровно между красным и синим. Если же категориальный признак представлен в выборке не в виде чисел, то не можем использовать его для обучения модели. Изучим способы кодирования категориальных признаков.

§2.1 Label encoding

В простом случае, если категориальный признак представлен в виде нечисловых данных, можно построить обратимое отображение для каждого уникального значения в некоторое число. Это позволит использовать признак для обучения модели. Например, зелёному цвету будет соответствовать «1», красному – «2» и так далее.

У этого подхода две основные проблемы: задание порядка над категориями и работа с неизвестные в процессе обучения значения категориального признака (нужно не забывать обрабатывать такой случай отдельно).

§2.2 One-hot encoding

Другой способ кодирования заключается в добавлении признаков-индикаторов категориальных значений. Например, появятся новые бинарные признаки: «красный цвет», «зелёный цвет» и так далее. В этом случае решается проблема с заданием порядка над категориями и новыми значениями категориального признака (у такого объекта просто будут «нули» по всем признакам индикаторам).

Проблема этого подхода в увеличении количества признаков (а это затрачиваемая память) пропорционально количеству категорий. Эти разреженные признаки допускают хранение в виде разреженных матриц, но только некоторые алгоритмы умеют работать с ними. Также можно для экономии памяти не создавать признаки для редко встречающихся категорий.

§2.3 Mean target encoding

Более сложный метод кодирования категориальных признаков — кодирование средним значением целевой переменной. Идея в том, что алгоритму для предсказания цены необходимо знать не конкретный цвет автомобиля, а то, как этот цвет сказывается на цене. Поэтому заменим каждую категорию на среднее значение целевой переменной по всем объектам этой категории. Для бинарной классификации новый признак будет выглядеть следующим образом:

$$g_j(x, X) = \frac{\sum_{i=1}^{\ell}[f_j(x) = f_j(x_i)][y_i = +1]}{\sum_{i=1}^{\ell}[f_j(x) = f_j(x_i)]},$$

где $f_j(x_i)$ – j-й признак i-го объекта, y_i – класс i-го объекта.

Но в таком случае для редких категорий получим некорректные средние значения целевой переменной. Например, в выборке было только три золотистых автомобиля, которые оказались старыми и дешёвыми. Из-за этого наш алгоритм начнёт считать золотистый цвет дешёвым. Для исправления этой проблемы будем регуляризировать средним значением целевой переменной по всем категориям так, чтобы у редких категорий значение было близко к среднему по всей выборке, а для популярных к среднему значению по категории. Формально для задачи бинарной классификации это выражается так:

$$g_j(x, X) = \frac{\sum_{i=1}^{\ell}[f_j(x) = f_j(x_i)][y_i = +1] + C \times \text{global mean}}{\sum_{i=1}^{\ell}[f_j(x) = f_j(x_i)] + C},$$

где C – коэффициент, отвечающий за баланс между средим значением по категории и глобальным средним значением.

Однако если сделать это «в лоб», то столкнёмся с переобучением, так как внесли информацию о целевой переменной в признаки (новый признак слабая, но модель, предсказывающая целевое значение). Поэтому вычисление таких признаков следует производить по фолдам, то есть вычислять средние значения на основе одних фолдов для заполнения на другом фолде (аналогично процессу кросс валидации). Если же ещё планируется оценка качества модели с помощью кросс валидации по фолдам, то придётся применить «двойную кросс валидацию» для подсчёта признаков. Этот подход заключается в кодировании категориальных признаков по фолдам внутри глобальных фолдов, по которым оценивается качество модели.

Разберём этот процесс. Иллюстрация на рис. (3). Представим, что хотим посчитать качество модели на 3-м фолде. Для этого:

1. Разбиваем все фолды кроме 3-го на другие фолды. Количество внутренних фолдов может не совпадать с количеством внешних (на иллюстрации их также 3).

2. Для каждого из внутренних фолдов считаем значение mean target признаков на основе средних значений целевой переменной по фолдам, исключая текущий. Для 3-го внешнего фолда вычисляем как среднее вычисленных признаков по каждому из внутренних фолдов.

3. Обучаем модель на всех фолдах кроме 3-го, делаем предсказание на 3-м и считаем на нём качество.

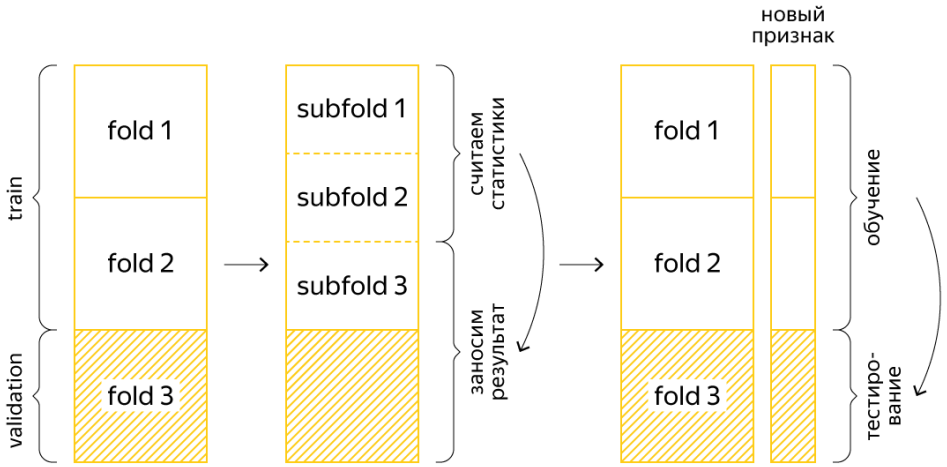

Рис. 3. Кросс валидация при кодировании средним значением

Существуют альтернативы кодированию категориальных признаков по фолдам. Во-первых, кодирование по порядку (среднее значение считается по всем объектам, расположенным выше текущего), порядок при этом может задаваться случайно или соответствовать временной шкале. Такой способ применяется в библиотеке CatBoost. Во-вторых, добавление шума к закодированным значениям.

3 Извлечение признаков из текстов

При решении некоторых задач мы сталкиваемся с тем, что объекты выборки целиком или частично описываются в виде текстов (под текстами имеем в виду строки, содержащие как минимум пару слов, иначе такой признак можно рассматривать как категориальный признак). Поэтому стоит задача представления текста в виде векторов чисел фиксированной длины.

§3.1 Bag-of-words

Простой способ заключается в подсчёте, сколько раз встретилось каждое слово в тексте. Получаем вектор длиной в количество уникальных слов, встречающихся во всех объектах выборки. В таком векторе много нулей, поэтому его удобнее хранить в разреженном виде. Такой способ представления текстов называют мешком слов.

§3.2 TF-IDF

Очевидно, что не все слова полезны в задаче прогнозирования. Например, мало информации несут слова, встречающиеся во всех текстах. Это могут быть как стоп-слова, так и слова, свойственные всем текстам выборки (в текстах про автомобили употребляется слово «автомобиль»). Эту проблему решает TF-IDF преобразование текста. Вычисляются две величины:

TD (Term Frequency) – количество вхождений слова в отношении к общему числу слов в тексте:

$$\mathrm{tf}(t, d) = \frac{n_{td}}{\sum_{t \in d} n_{td}},$$

где n_{td} — количество вхождений слова t в текст d.

IDF (Inverse Document Frequency):

$$\mathrm{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|},$$

где $|\{d \in D : t \in d\}|$ – количество текстов в коллекции, содержащих слово t.

Тогда для каждой пары (слово, текст) (t, d) вычислим величину:

$$\text{tf-idf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D).$$

Это и будем значением нового признака (вместо количество каждого слова в тексте в случае мешка слов). Отметим, что значение $\mathrm{tf}(t, d)$ корректируется для часто встречающихся общеупотребимых слов при помощи значения $\mathrm{idf}(t, D)$.

§3.3 Лемматизация и стемминг

Заметим, что одно и то же слово может встречаться в различных формах (особенно для русского языка), но описанные выше методы интерпретируют их как различные слова, что делает признаковое описание избыточным. Устранить эту проблему можно при помощи лемматизации и стемминга.

Стемминг – это процесс нахождения основы слова. В результате применения данной процедуры однокоренные слова, как правило, преобразуются к одинаковому виду. Например, вагон – вагон, вагонов – вагон и важная – важн, важно – важн.

Лемматизация – процесс приведения слова к его нормальной форме (лемме):

- для существительных – именительный падеж, единственное число;
- для прилагательных – именительный падеж, единственное число, мужской род;
- для глаголов, причастий, деепричастий — глагол в инфинитиве.

Лемматизация – процесс более сложный по сравнению со стеммингом. Стеммер просто «режет» слово до основы. Реализация лемматизаторов и стеммеров можно найти в различных библиотеках (nltk, pymorphy).