Лекция 4
Линейная классификация

Е. А. Соколов
ФКН ВШЭ

28 сентября 2016 г.

1 Линейные модели классификации

Мы начнём с задачи бинарной классификации, а многоклассовый случай обсудим позже. Пусть $\mathbb{X} = \mathbb{R}^d$ — пространство объектов, $Y = \{-1, +1\}$ — множество допустимых ответов, $X = \{(x_i, y_i)\}_{i=1}^{\ell}$ — обучающая выборка. Иногда мы будем класс «+1» называть положительным, а класс «−1» — отрицательным.

Линейная модель классификации определяется следующим образом:

$$a(x) = \text{sign}\left(\langle w, x\rangle + w_0\right) = \text{sign}\left(\sum_{j=1}^{d} w_j x_j + w_0\right),$$

где $w \in \mathbb{R}^d$ — вектор весов, $w_0 \in \mathbb{R}$ — сдвиг (bias).

Если не сказано иначе, мы будем считать, что среди признаков есть константа, $x_{d+1} = 1$. В этом случае нет необходимости вводить сдвиг w_0, и линейный классификатор можно задавать как

$$a(x) = \text{sign}\langle w, x\rangle.$$

Геометрически линейный классификатор соответствует гиперплоскости с вектором нормали w. Величина скалярного произведения $\langle w, x\rangle$ пропорциональна расстоянию от гиперплоскости до точки x, а его знак показывает, с какой стороны от гиперплоскости находится данная точка. Таким образом, линейный классификатор разделяет пространство на две части с помощью гиперплоскости, и при этом одно полупространство относит к положительному классу, а другое — к отрицательному.

§1.1 Обучение линейных классификаторов

В задаче регрессии имеется континуум возможных ответов, и при таких условиях достаточно странно требовать полного совпадения ответов модели и истинных ответов — гораздо логичнее говорить об их близости. Более того, как мы выяснили, попытка провести функцию через все обучающие точки легко может привести к переобучению. Способов посчитать близость двух чисел (прогноза и истинного ответа) достаточно много, и поэтому при обсуждении регрессии у нас возникло большое количество функционалов ошибки.

В случае с бинарной классификацией всё гораздо проще: у нас всего два возможных ответа алгоритма и, очевидно, мы хотим видеть как можно больше правильных ответов. Соответствующий функционал называется *долей правильных ответов* (accuracy):

$$Q(a, X) = \frac{1}{\ell}\sum_{i=1}^{\ell}[a(x_i) = y_i].$$

Нам будет удобнее решать задачу минимизации, поэтому будем вместо этого использовать долю неправильных ответов:

$$Q(a, X) = \frac{1}{\ell}\sum_{i=1}^{\ell}[a(x_i) \neq y_i] = \frac{1}{\ell}\sum_{i=1}^{\ell}[\langle w, x_i\rangle \neq y_i] \to \min_w \tag{1.1}$$

Этот функционал является дискретным относительно весов, и поэтому искать его минимум с помощью градиентных методов мы не сможем. Более того, у данного функционала может быть много глобальных минимумов — вполне может оказаться, что существует много способов добиться оптимального количества ошибок. Чтобы не пытаться решать все эти проблемы, попытаемся свести задачу к минимизации гладкого функционала.

Отступы. Заметим, что функционал (1.1) можно несколько видоизменить:

$$Q(a, X) = \frac{1}{\ell}\sum_{i=1}^{\ell}[\underbrace{y_i\langle w, x_i\rangle}_{M_i} < 0] \to \min_w$$

Здесь возникла очень важная величина $M_i = y_i\langle w, x_i\rangle$, называемая *отступом* (margin). Знак отступа говорит о корректности ответа классификатора (положительный отступ соответствует правильному ответу, отрицательный — неправильному), а его абсолютная величина характеризует степень уверенности классификатора в своём ответе. Напомним, что скалярное произведение $\langle w, x\rangle$ пропорционально расстоянию от разделяющей гиперплоскости до объекта; соответственно, чем ближе отступ к нулю, тем ближе объект к границе классов, тем ниже уверенность в его принадлежности.

Верхние оценки. Функционал (1.1) оценивает ошибку алгоритма на объекте x с помощью пороговой функции потерь $L(M) = [M < 0]$, где аргументом функции является отступ $M = y\langle w, x\rangle$. Оценим эту функцию сверху (см. рис. 1):

$$L(M) \leqslant \tilde{L}(M).$$

После этого можно получить верхнюю оценку на функционал (1.1):

$$Q(a, X) \leqslant \frac{1}{\ell}\sum_{i=1}^{\ell}\tilde{L}(y_i\langle w, x_i\rangle) \to \min_w$$

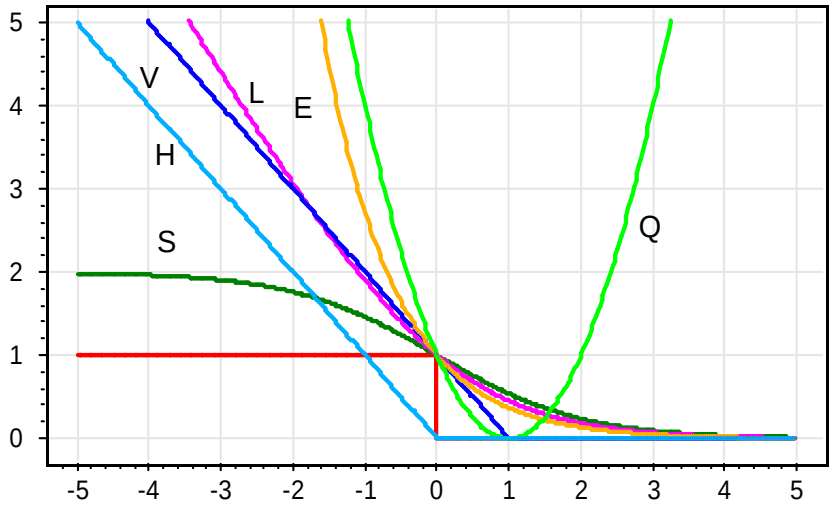

Рис. 1. Верхние оценки на пороговую функцию потерь.

Если верхняя оценка $\tilde{L}(M)$ является гладкой, то и данная верхняя оценка будет гладкой. В этом случае её можно будет минимизировать с помощью, например, градиентного спуска. Если верхнюю оценку удастся приблизить к нулю, то и доля неправильных ответов тоже будет близка к нулю.

Приведём несколько примеров верхних оценок:

1. $\tilde{L}(M) = \log\left(1 + e^{-M}\right)$ — логистическая функция потерь
2. $\tilde{L}(M) = (1 - M)_+ = \max(0, 1 - M)$ — кусочно-линейная функция потерь (используется в методе опорных векторов)
3. $\tilde{L}(M) = (-M)_+ = \max(0, -M)$ — кусочно-линейная функция потерь (соответствует персептрону Розенблатта)
4. $\tilde{L}(M) = e^{-M}$ — экспоненциальная функция потерь
5. $\tilde{L}(M) = 2/(1 + e^{M})$ — сигмоидная функция потерь

Любая из них подойдёт для обучения линейного классификатора. Позже мы подробно изучим некоторые из них и выясним, какими свойствами они обладают.

2 Метрики качества классификации

Выше мы разобрали способ сведения задачи обучения линейного классификатора к минимизации гладкого функционала. При этом часто возникает необходимость в изучении различных аспектов качества уже обученного классификатора. Обсудим подробнее распространённые подходы к измерению качества таких моделей.

Будем считать, что классификатор имеет вид $a(x) = [b(x) > t]$. Линейная модель имеет именно такую форму, если положить $b(x) = \langle w, x\rangle$ и $t = 0$.

§2.1 Доля правильных ответов

Наиболее очевидной мерой качества в задаче классификации является доля правильных ответов (accuracy), которую мы уже упоминали:

$$\text{accuracy}(a, x) = \frac{1}{\ell}\sum_{i=1}^{\ell}[a(x_i) = y_i].$$

Данная метрика, однако, имеет существенный недостаток. Если взять порог t меньше минимального значения прогноза $b(x)$ на выборке или больше максимального значения, то доля правильных ответов будет равна доле положительных и отрицательных ответов соответственно. Таким образом, если в выборке 950 отрицательных и 50 положительных объектов, то при тривиальном пороге $t = \max_i b(x_i)$ мы получим долю правильных ответов 0.95. Это означает, что доля положительных ответов сама по себе не несет никакой информации о качестве работы алгоритма $a(x)$, и вместе с ней следует анализировать соотношение классов в выборке. Также полезно вместе с долей правильных ответов вычислять *базовую долю* — долю правильных ответов алгоритма, всегда выдающего наиболее мощный класс.

Отметим, что при сравнении различных методов машинного обучения принято сообщать относительное уменьшение ошибки. Рассмотрим два алгоритма a_1 и a_2 с долями правильных ответов r_1 и r_2 соответственно, причем $r_2 > r_1$. Относительным уменьшением ошибки алгоритма a_2 называется величина

$$\frac{(1 - r_1) - (1 - r_2)}{1 - r_1}.$$

Если доля ошибок была улучшена с 20% до 10%, то относительное улучшение составляет 50%. Если доля ошибок была улучшена с 50% до 25%, то относительное улучшение также равно 50%, хотя данный прирост кажется более существенным. Если же доля ошибок была улучшена с 0.1% до 0.01%, то относительное улучшение составляет 90%, что совершенно не соответствует здравому смыслу.

§2.2 Матрица ошибок

Выше мы убедились, что в случае с несбалансированными классами одной доли правильных ответов недостаточно — необходима еще одна метрика качества. В данном разделе мы рассмотрим другую, более информативную пару критериев.

Введем сначала понятие матрицы ошибок. Это способ разбить объекты на четыре категории в зависимости от комбинации истинного ответа и ответа алгоритма (см. таблицу 1). Через элементы этой матрицы можно, например, выразить долю правильных ответов:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}.$$

Гораздо более информативными критериями являются *точность* (precision) и *полнота* (recall):

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}};$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

	$y = 1$	$y = 0$
$a(x) = 1$	True Positive (TP)	False Positive (FP)
$a(x) = 0$	False negative (FN)	True Negative (TN)

Таблица 1. Матрица ошибок

Точность показывает, какая доля объектов, выделенных классификатором как положительные, действительно является положительными. Полнота показывает, какая часть положительных объектов была выделена классификатором.

Рассмотрим, например, задачу предсказания реакции клиента банка на звонок с предложением кредита. Ответ $y = 1$ означает, что клиент возьмет кредит после рекламного звонка, ответ $y = 0$ — что не возьмет. Соответственно, планируется обзванивать только тех клиентов, для которых классификатор $a(x)$ вернет ответ 1. Если классификатор имеет высокую точность, то практически все клиенты, которым будет сделано предложение, откликнутся на него. Если классификатор имеет высокую полноту, то предложение будет сделано практически всем клиентам, которые готовы откликнуться на него. Как правило, можно регулировать точность и полноту, изменяя порог t в классификаторе $a(x) = [b(x) > t]$. Если выбрать t большим, то классификатор будет относить к положительному классу небольшое число объектов, что приведет к высокой точности и низкой полноте. По мере уменьшения t точность будет падать, а полнота увеличиваться. Конкретное значение порога выбирается согласно пожеланиям заказчика.

Отметим, что точность и полнота не зависят от соотношения размеров классов. Даже если объектов положительного класса на порядки меньше, чем объектов отрицательного класса, данные показатели будут корректно отражать качество работы алгоритма.

Существует несколько способов получить один критерий качества на основе точности и полноты. Один из них — F-мера, гармоническое среднее точности и полноты:

$$F = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}.$$

Среднее гармоническое обладает важным свойством — оно близко к нулю, если хотя бы один из аргументов близок к нулю. Именно поэтому оно является более предпочтительным, чем среднее арифметическое (если алгоритм будет относить все объекты к положительному классу, то он будет иметь recall = 1 и precision $\ll 1$, а их среднее арифметическое будет больше $1/2$, что недопустимо). Можно заметить, что F-мера является сглаженной версией минимума из точности и полноты (см. рис. 2 и 3).

Другим агрегированным критерием является *R-точность*, или точка баланса (breakeven point). Она вычисляется как точность при таком t, при котором полнота равна точности:

$$\text{R-precision} = \text{precision}([b(x) > t^*]),$$
$$t^* = \arg\min_t \left|\text{precision}([b(x) > t]) - \text{recall}([b(x) > t])\right|.$$

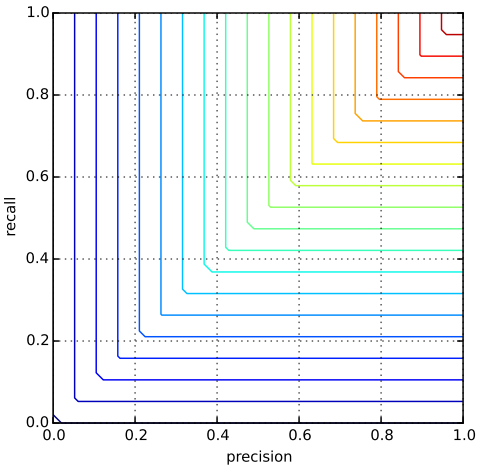

Рис. 2. Линии уровня для минимума из точности и полноты.

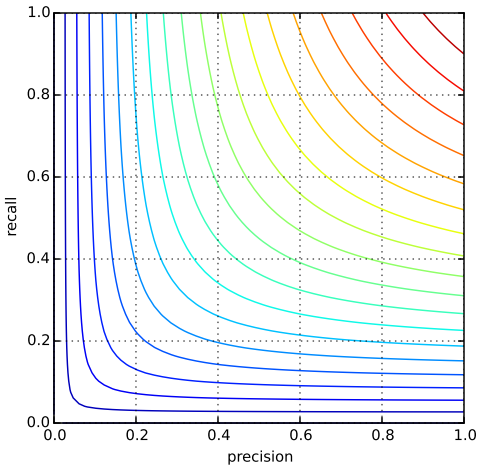

Рис. 3. Линии уровня для F-меры.

Можно показать, что R-точность равна точности при таком пороге, при котором количество отнесённых к положительному классу объектов равно количеству положительных объектов в выборке.

Часто встречаются задачи, в которых целевой признак по-прежнему бинарный, но при этом необходимо ранжировать объекты, а не просто предсказывать их класс. Например, в задаче предсказания реакции клиента можно выдавать сортированный список, чтобы оператор мог в первую очередь позвонить клиентам с наибольшей вероятностью положительного отклика. Поскольку многие алгоритмы возвращают вещественный ответ $b(x)$, который затем бинаризуется по порогу t, то можно просто сортировать объекты по значению $b(x)$. Для измерения качества ранжирования нередко используют *среднюю точность* (average precision, AP):

$$\text{AP} = \frac{1}{\ell_+}\sum_{k=1}^{\ell}[y_{(k)} = 1]\text{precision@}k,$$

где $y_{(k)}$ — ответ k-го по порядку объекта, ℓ_+ — число положительных объектов в выборке, а precision@k — точность среди первых k в списке объектов. Если алгоритм $b(x)$ так ранжирует объекты, что сначала идут все положительные, а затем все отрицательные, то средняя точность будет равна единице; соответственно, чем сильнее положительные документы концентрируются в верхней части списка, тем ближе к единице будет данный показатель.

Связь точности, полноты и доли правильных ответов Выше мы отмечали, что высокие значения доли правильных ответов вовсе не влекут за собой высокое качество работы классификатора, и ввели точность и полноту как способ решения этой проблемы. Тем не менее, при выборе точности и полноты в качестве основных метрик, следует соблюдать осторожность при выборе требований к их значениям — как мы увидим из примера, независимость данных метрик от соотношения классов может привести к неочевидным последствиям.

Рассмотрим задачу бинарной классификации с миллионом объектов ($\ell = 1.000.000$), где доля объектов первого класса составляет 1% ($\ell_+ = 10.000$). Мы знаем, что доля правильных ответов будет вести себя не вполне интуитивно на данной несбалансированной выборке, и поэтому выберем точность и полноту для измерения качества классификаторов. Поскольку мы хотим решить задачу хорошо, то введём требования, что и точность, и полнота должны быть не менее 90%. Эти требования кажутся вполне разумными, если забыть о соотношении классов. Попробуем теперь оценить, какая доля правильных ответов должна быть у классификатора, удовлетворяющего нашим требованиям.

Всего в выборке 10.000 положительных объектов, и для достижения полноты 90% мы должны отнести как минимум 9.000 к положительному классу. Получаем $\mathrm{TP} = 9000$, $\mathrm{FN} = 1000$. Так как точность тоже должна быть не меньше 90%, получаем $\mathrm{FP} = 1000$. Отсюда получаем, что доля правильных ответов должна быть равна $2.000/1.000.000 = 99.8\%$! Это крайне высокий показатель, и его редко удаётся достичь на таких выборках во многих предметных областях.

Lift. На практике часто возникают задачи, связанные с выбором подмножества: выделение лояльных клиентов банка, обнаружение уходящих пользователей мобильного оператора и т.д. Заказчика может интересовать вопрос, насколько выгоднее работать с этим подмножеством по сравнению со всем множеством. Если при рассылке предложений о кредите клиентам из подмножества и всем клиентам будет получаться одна и та же доля откликнувшихся, то подмножество не будет представлять особой ценности. Формально это измеряется с помощью *прироста концентрации* (lift), который равен отношению точности к доле положительных объектов в выборке:

$$\mathrm{lift} = \frac{\mathrm{precision}}{(\mathrm{TP} + \mathrm{FN})/\ell}.$$

Эту величину можно интерпретировать как улучшение доли положительных объектов в данном подмножестве относительно доли в случайно выбранном подмножестве такого же размера.

§2.3 Area Under Curve

Выше мы изучили точность, полноту и F-меру, которые характеризуют качество работы алгоритма $a(x) = [b(x) > t]$ при конкретном выборе порога t. Однако зачастую интерес представляет лишь вещественнозначный алгоритм $b(x)$, а порог будет выбираться позже в зависимости от требований к точности и полноте. В таком случае возникает потребность в измерении качества семейства моделей $\{a(x) = [b(x) > t] \,|\, t \in \mathbb{R}\}$.

Можно измерять качество этого множества на основе качества лучшего (в некотором смысле) алгоритма. Для этого подходит упомянутая ранее точка баланса (breakeven point). В идеальном семействе алгоритмов она будет равна единице, поскольку найдется алгоритм со стопроцентной точностью и полнотой. Данная метрика, однако, основывается лишь на качестве одного алгоритма, и не характеризует вариативность семейства.

Широко используется такая интегральная метрика качества семейства, как *площадь под ROC-кривой* (Area Under ROC Curve, AUC-ROC). Рассмотрим двумерное

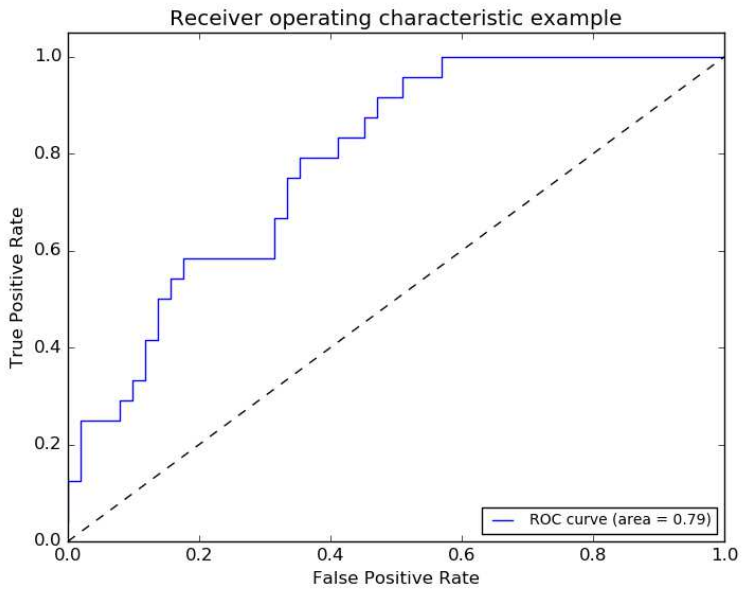

Рис. 4. Пример ROC-кривой.

пространство, одна из координат которого соответствует доле неверно принятых объектов (False Positive Rate, FPR), а другая — доле верно принятых объектов (True Positive Rate, TPR):

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}};$$
$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}.$$

Каждый возможный выбор порога t соответствует точке в этом пространстве. Всего различных порогов имеется $\ell + 1$. Максимальный порог $t_{\max} = \max_i b(x_i)$ даст классификатор с TPR $= 0$, FPR $= 0$. Минимальный порог $t_{\min} = \min_i b(x_i) - \varepsilon$ даст TPR $= 1$ и FPR $= 1$. ROC-кривая — это кривая с концами в точках $(0, 0)$ и $(1, 1)$, которая последовательно соединяет точки, соответствующие порогам $b(x_{(1)}) - \varepsilon, b(x_{(1)}), b(x_{(2)}), \ldots, b(x_{(\ell)})$ (см. рис. 4). Площадь под данной кривой называется AUC-ROC, и принимает значения от 0 до 1. Если порог t может быть подобран так, что алгоритм $a(x)$ не будет допускать ошибок, то AUC-ROC будет равен единице; если же $b(x)$ ранжирует объекты случайным образом, то AUC-ROC будет близок к 0.5.

Критерий AUC-ROC имеет большое число интерпретаций — например, он равен вероятности того, что случайно выбранный положительный объект окажется позже случайно выбранного отрицательного объекта в ранжированном списке, порожденном $b(x)$.

Индекс Джини. В задачах кредитного скоринга вместо AUC-ROC часто используется пропорциональная метрика, называемая индексом Джини (Gini index):

$$\mathrm{Gini} = 2\mathrm{AUC} - 1.$$

По сути это площадь между ROC-кривой и диагональю соединяющей точки $(0,0)$ и $(1,1)$.

Отметим, что переход от AUC к индексу Джини приводит к увеличению относительных разниц. Если мы смогли улучшить AUC с 0.8 до 0.9, то это соответствует относительному улучшению в 12.5%. В то же время соответствующие индексы Джини были улучшены с 0.6 до 0.8, то есть на 33.3% — относительное улучшение повысилось почти в три раза!

Чувствительность к соотношению классов. Рассмотрим задачу выделения математических статей из множества научных статей. Допустим, что всего имеется 1.000.100 статей, из которых лишь 100 относятся к математике. Если нам удастся построить алгоритм $a(x)$, идеально решающий задачу, то его TPR будет равен единице, а FPR — нулю. Рассмотрим теперь плохой алгоритм, дающий положительный ответ на 95 математических и 50.000 нематематических статьях. Такой алгоритм совершенно бесполезен, но при этом имеет $\text{TPR} = 0.95$ и $\text{FPR} = 0.05$, что крайне близко к показателям идеального алгоритма.

Таким образом, если положительный класс существенно меньше по размеру, то AUC-ROC может давать неадекватную оценку качества работы алгоритма, поскольку измеряет долю неверно принятых объектов относительно общего числа отрицательных. Так, алгоритм $b(x)$, помещающий 100 релевантных документов на позиции с 50.001-й по 50.101-ю, будет иметь AUC-ROC 0.95.

Precison-recall кривая. Избавиться от указанной проблемы с несбалансированными классами можно, перейдя от ROC-кривой к Precision-Recall кривой. Она определяется аналогично ROC-кривой, только по осям откладываются не FPR и TPR, а полнота (по оси абсцисс) и точность (по оси ординат). Критерием качества семейства алгоритмов выступает площадь под PR-кривой (AUC-PR). Данную величину можно аппроксимировать следующим образом. Стартуем из точки $(0,0)$. Будем идти по ранжированной выборке, начиная с первого объекта; пусть текущий объект находится на позиции k. Если он относится к классу «0», то полнота не меняется, точность падает — соответственно, кривая опускается строго вниз. Если же объект относится к классу «1», то полнота увеличивается на $1/\ell_+$, точность растет, и кривая поднимается вправо и вверх. Площадь под этим участком можно аппроксимировать площадью прямоугольника с высотой, равной $\text{precision@}k$ и шириной $1/\ell_+$. При таком способе подсчета площадь под PR-кривой будет совпадать со средней точностью:

$$\text{AUC-PR} = \frac{1}{\ell_+}\sum_{i=1}^{\ell}[y_i = 1]\text{precision@}k.$$

Отметим, что AUC-PR дает разумные результаты в рассмотренном выше примере с классификацией статей. Так, при размещении 100 релевантных документов на позициях 50.001-50.101 в ранжированном списке, AUC-PR будет равен 0.001.

Несмотря на указанные различия, между ROC- и PR-кривой имеется тесная связь. Так, можно показать, что если ROC-кривая одного алгоритма лежит полностью над ROC-кривой другого алгоритма, то и PR-кривая одного лежит над PR-кривой другого [1].

Список литературы

[1] *Davis J., Goadrich M.* (2006). The Relationship Between Precision-Recall and ROC Curves. // Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA.