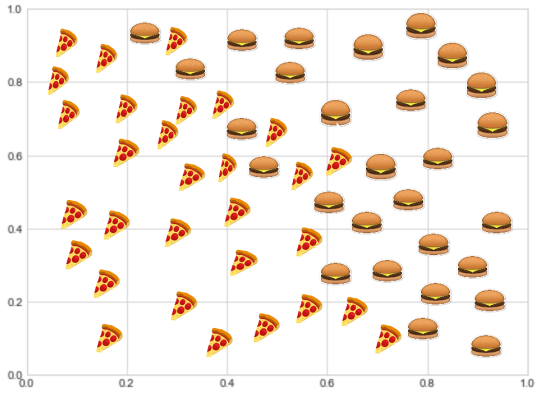
Семинар 8: Метрики классификации

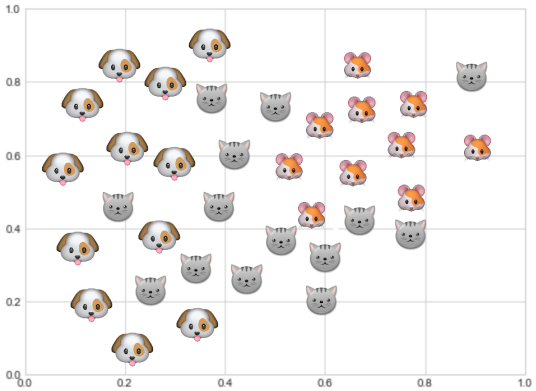
В этом семинаре мы подробнее поговорим про классификацию и метрики для неё. План будет таким:

- сформулируем задачу и поймём её специфику;
- немного поговорим про переобучение;
- поймём с помощью каких метрик можно оценить качество прогнозирования;
- попробуем разобраться какой смысл стоит за этими метриками.

Упражнение 1

Нам нужно научиться отделять пиццу от бургеров, а также котиков от пёсиков и от мышек. Проведите на картинках линии, которые отделят одни классы от других. Да, это и есть машинное обучение. Но обычно кривые рисуем не мы, а компуктер.

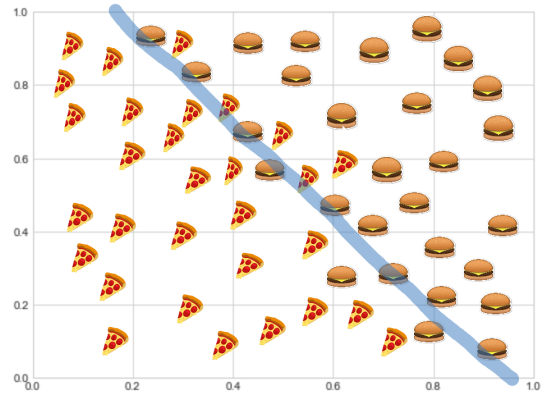


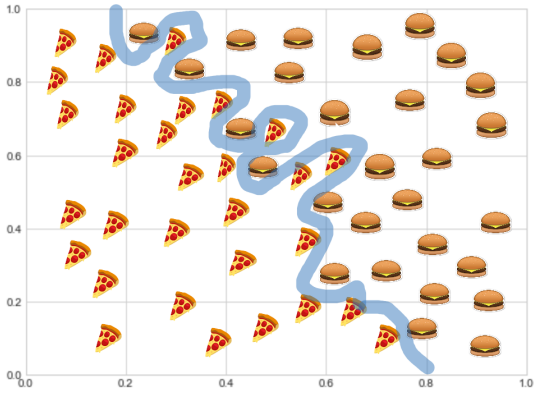


Почему нельзя провести между пиццей и бургерами слишком подробную и извилистую границу? В чём проблема самого правого верхнего котика? Что такое переобучение? Как понять переобучились ли мы?

Решение:

Сначала обсудим бургеры и пиццу. Первый вариант: провести между ними прямую. Тогда мы в части случаев ошибёмся и признаем некоторые бургеры пиццей, а некоторые пиццы бургерами. Второй вариант: провести извилистую разделительную линию, которая чётко разграничит бургеры и пиццу. Вопрос: какой из этих двух вариантов лучше?





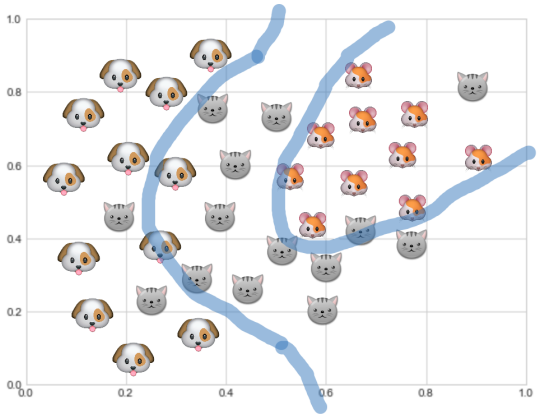
Если у нас в выборке оказались все пиццы и бургеры мира, и других быть не может, вторая граница нам подойдёт. Мы подстроимся под все особенности нашей генеральной пиццебургерной совокупности и будем всегда чётко и безошибочно отличать одно от другого.

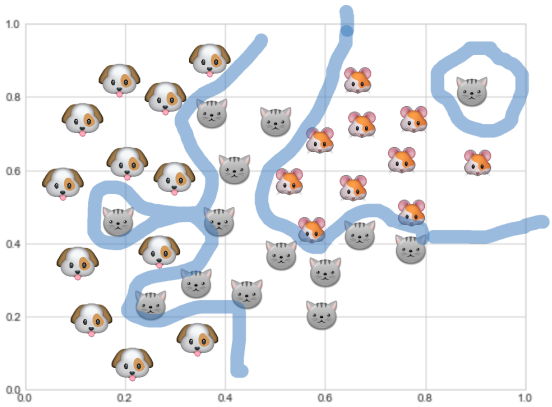
НО в нашем распряжении обычно находится не вся генеральная совокупность, а лишь какая-то её часть, выборка. В ней видим не все возможные варианты, и хотим обучить наш классификатор обобщать. Если к нам попадает новая пицца или бургер, классификатор должен адекватно сработать на них.

Скорее всего, пиццы, проникшие на территорию бургеров, обладают какими-то аномальными особенностями, на детекцию которых затачивать классификатор нет никакого смысла. Если мы попробуем сделать это, мы влезем на территорию бургеров, и на новых объектах, которые оказались обычными бургерами, будем делать ошибки, подумав, что это аномальные пиццы. Из-за этого лучше разграничить бургеры и пиццы простой линией, которая изображена на первой картинке.

Ещё раз, ещё раз. Если мы проведём подробную границу, мы заточим классификатор под особенности выборки, вместо того, чтобы научить его отличать пиццу от бургера в общем случае. **Такие ситуации называются переобучением.** И это главная головная боль людей, занимающихся машинным обучением. С переобучением у них идёт вечная борьба.

Теперь посмотрим на котиков, пёсиков и мышек. Снова мы можем провести границы между ними разными способами.





Снова мы можем провести более-менее простую границу и иногда ошибаться. Ну знаете,

есть такие собаки мелкие, похожие на кошек. Или даже на мышек[1]. И, если мы будем специфицировать границу под этих собак, мы начнём ошибаться на котах, так как подобные аномалии встречаются редко.

Основная проблема верхнего котика в том, что он аномальный. Каким-то образом он попал на территорию мышек. Выделять для него свою зону будет плохой идеей, так как в таком случае мы будем переобучать классификатор под конкретный выброс.

Осталось обсудить главный вопрос: как понять а не переобучились ли мы, когда строили модель, которая должна отделить одни объекты от других? Для этого обычно дробят выборку на две части: **тренировочную и тестовую. На тренировочной учат алгоритм** (в данном случае границу между классами), а **на тестовой проверяют насколько хорошо он работает** (сколько и каких ошибок мы допускаем на новых объектах, которые модель раньше не видела)

Если получается, что на обучающей выборке качество высокое, а на тестовой низкое — мы переобучились и вместо того, чтобы научить модель обобщать закономерности, существующие в данных, обучили его под особенности конкретной выборки. Если на тестовой выборке качество сравнимо с обучающей, значит мы научились извлекать какие-то реальные закономерности.

Бьюсь об заклад, что для простых линий, качество на тесте для бургеров и мышек будет выше, чем для сложных. Конечно же, простые границы оказываются хороши не всегда, но тем не менее всегда имеет смысл сначала построить простую модель, а после сравнивать с ней сложные.

Упражнение 2

Винни-Пух ищет неправильных пчёл. За долгие годы поиска он скопил довольно большую выборку и оценил на ней три модели: нейросеть, случайный лес и KNN. Он построил на тестовой выборке прогнозы и получил три матрицы ошибок:

	$y = 1$	$y = 0$
$\hat{y} = 1$	80	20
$\hat{y} = 0$	20	80

	$y = 1$	$y = 0$
$\hat{y} = 1$	48	2
$\hat{y} = 0$	52	98

	$y = 1$	$y = 0$
$\hat{y} = 1$	10	20
$\hat{y} = 0$	90	10000

а) Найдите для всех трёх моделей долю правильных ответов. Чем плоха эта метрика?

б) Найдите для всех трёх моделей точность (precision) и полноту (recall)

в) Предположим, что Винни-Пух колектор. Пчела, по его мнению, неправильная, если она не возвращает кредит. Переменная y принимает значение 1, если пчела вернула кредит и 0, если не вернула. ВП хочет научиться прогнозировать платёжеспособность пчелы. Какую из первых двух моделей вы бы выбрали в таком случае?

г) Предположим, что Винни-Пух врач. Пчела, по его мнению, неправильная, если она умирает от болезни. Он хочет находить таких пчёл и лечить. Переменная y принимает значение 1, если пчела больна больной болезнью с болью и 0, если она здорова. ВП хочет спрогнозировать нужно ли пчеле пройти обследование. Какую из первых двух моделей

[1] На самом деле на крыс

вы б выбрали в этом случае?

д) Найдите для всех трёх моделей f1-меру.

Решение:

а) Ежели мы совершаем ошибку, то мы делаем это, когда прогнозируем $\hat{y} = 1$ там, где реально должно быть $y = 0$, либо наоборот, прогнозируя $\hat{y} = 0$ там, где реально должно быть $y = 1$. В остальных случаях ошибки нет. Наша классная табличка выглядит следующим образом:

	$y = 1$	$y = 0$
$\hat{y} = 1$	TP	FP
$\hat{y} = 0$	FN	TN

В ней TP — True Positive, то есть мы спрогнозировали 1 и не ошиблись, FP — False Positive, то есть мы спрогнозировали 1 и ошиблись. По аналогии с FN — False Negative и TN — True Negative. Долю правильных ответов можно посчитать как

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

Давайте проделаем эту несложную процедуру для вердиктов всех трёх алгоритмов, описанных выше.

$$Accuracy_1 = \frac{80 + 80}{80 + 80 + 20 + 20} = 0.8$$
$$Accuracy_2 = \frac{48 + 98}{48 + 98 + 2 + 52} = 0.73$$
$$Accuracy_3 = \frac{10 + 10000}{10 + 10000 + 20 + 90} = 0.98$$

У этой метрики есть как минимум две существенные проблемы.

Первая проблема связана с несбалансированными выборками. Именно такая ситуация наблюдается в третьей табличке. Нулевой класс заметно перетягивает на себя выборку. Выходит, что если мы просто-напросто спрогнозируем, что все объекты в выборке нулевые, мы получим табличку

	$y = 1$	$y = 0$
$\hat{y} = 1$	0	0
$\hat{y} = 0$	100	10020

и $Accuracy = 0.99$. Наша модель показала более низкое качество, чем простое угадывание. Выходит, что наша модель с точки зрения этой метрики абсолютно бесполезна.

Чтобы «бороться» с этой проблемой, используется следующий факт. Пусть q_0 — доля объектов самого крупного класса, тогда доля правильных ответов для разумных алгоритмов

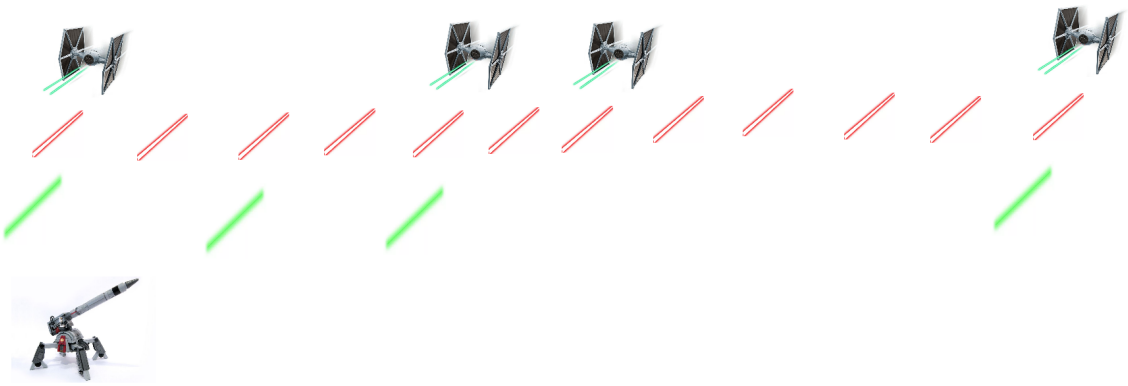
$accuracy \in [q_0; 1]$, а не $[0.5; 1]$, как это можно было бы ожидать. Поэтому, если получается высокий процент правильных ответов, это может быть связано не с тем, что построен хороший классификатор, а с тем, что какого-то класса сильно больше, чем остальных.

Вторая проблема с долей верных ответов состоит в том, что она никак не учитывает разные цены разных типов ошибок.

Например, представим себе линейное небо, которое бороздят вражеские самолёты. Мы стоим на земле, у нас есть пушка и снаряды. Сбивать вражеские самолёты можно руководствуясь двумя разными стратегиями:

Путь первый: стрелять по всему небу, куда только можно попасть. В таком случае точность наших выстрелов будет низкой, но зато мы собьём все самолёты, то есть добьёмся высокой полноты. Такая стратегия на картинке прорисована красными лазерными выстрелами из пушки.

Путь второй: стрелять поточнее, но реже. Тогда мы будем сбивать самолёты точно, потратим мало снарядов вхолостую, но собьём не все самолёты. Такая стратегия на картинке прорисована зелёными выстрелами из пушки.



Для разных задач бывают характерны разные стратегии. Если вернуться к нашей исходной таблице ошибок, то за точность классификатора будет отвечать первая строка

$$Precision = \frac{TP}{TP + FP}.$$

За полноту будет отвечать первый столбец

$$Recall = \frac{TP}{TP + FN}.$$

Посмотрим ещё один пример: если мы решаем задачу крединтого скоринга, выдаём пчёлам кредит, нам нужно получить деньги назад с процентом, иначе мы разоримся. Нам нужна модель, которая будет точно определять надёжного заёмщика. В этой ситуации для нас неважно покрыть все вражеские истребители снарядами (выдать кредиты каждой надёжной пчеле), для нас важно сделать это точно. Поэтому основное внимание мы уделяем ошибке FP, и ищем точный алгоритм.

Если мы пытаемся найти больных больной болезнью с болью и отправить их делать дополнительные анализы, для нас страшнее FN ошибка. Если мы отправим лишнего человека на анализы, ничего страшного с ним не произойдёт. Если мы забудем проверить больного, он умрёт. Тут лучше добиться высокой полноты, при небольшой точности.

В разных ситуациях ошибки имеют разные цены. $\mathtt{Accuracy}$ не видит этого, поэтому на практике обычно используют $\mathtt{Precision}$ и $\mathtt{Recall}$.

б) Найдём их для наших трёх моделей:

$$\begin{aligned} \mathtt{Precision}_1 &= 0.8 & \mathtt{Recall}_1 &= 0.8 \\ \mathtt{Precision}_2 &= 0.96 & \mathtt{Recall}_2 &= 0.48 \\ \mathtt{Precision}_3 &= 0.33 & \mathtt{Recall}_3 &= 0.1 \end{aligned}$$

Вторая модель является очень точной, но в ущерб полноте. Третья модель очень плохая. Эти две метрики, в отличие от $\mathtt{Accuracy}$, позволили заметить этот факт.

в) При использовании первой модели кредит будет выдан 100 клиентам, 80 из которых его вернут. Во второй модели, более консервативной, кредит был выдан только 50 клиентам, причем вернули его в 48 случаях.

Выше мы обсудили, что бизнес-специфика задачи диктует нам необходимость взять модель, где побольше точность.

г) Выше мы обсудили, что нам нужна модель, где побольше полнота.

д) Чем выше точность и полнота, тем лучше. Однако в реальном мире невозможно добиться того, чтобы оба показателя были очень большими. Между ними приходится искать какой-то баланс. Чтобы видеть его, хочется объединить точность и полноту в какую-то общую метрику. Такой метрикой является **f-мера**. Посчитать её можно по формуле:

$$\mathtt{f} = 2 \cdot \frac{\mathtt{precision} \cdot \mathtt{recall}}{\mathtt{precision} + \mathtt{recall}}.$$

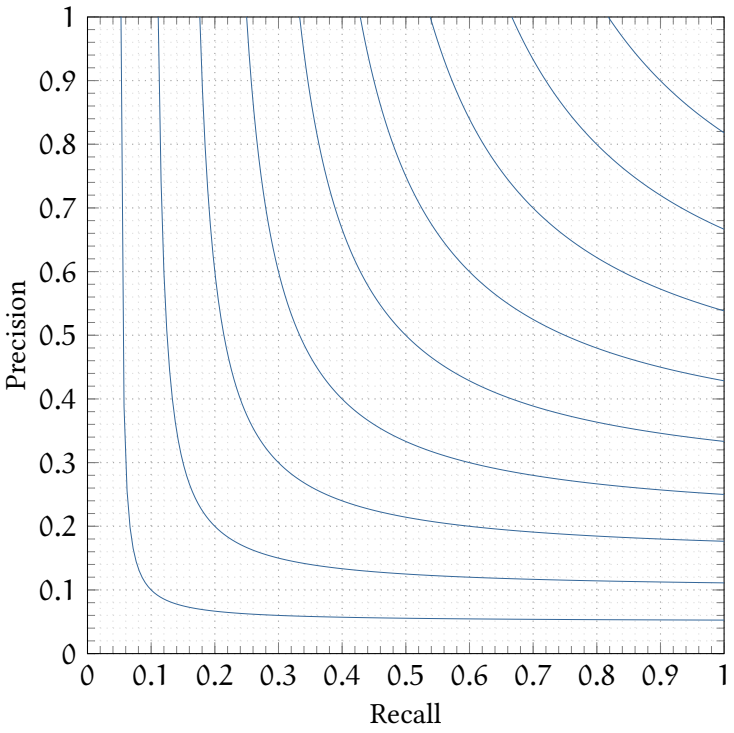
Сделаем это для наших моделей:

$$\begin{aligned} \mathtt{f}_1 &= 2 \cdot \frac{0.8 \cdot 0.8}{0.8 + 0.8} = 0.8 \\ \mathtt{f}_2 &= 2 \cdot \frac{0.96 \cdot 0.48}{0.96 + 0.48} = 0.64 \\ \mathtt{f}_3 &= 2 \cdot \frac{0.33 \cdot 0.1}{0.33 + 0.1} = 0.15 \end{aligned}$$

Видим, что первая модель оказывается в терминах f-меры самой хорошей.

F-мера — это гармоническое среднее между точностью и полнотой. 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 $\mathtt{recall} = 1$ и $\mathtt{precision} << 1$, а их среднее арифметическое будет больше 0.5, что недопустимо).

Можно попробовать нарисовать эту метрику в координах точность-полнота и получить вот такие кривые:



Вдоль каждой кривой f-мера постоянна. Например, для самой нижней кривой, a-мера равна 0.1. Чем правее по ней мы сдвигаемся тем меньше точности и тем больше полнота. Прирост полноты вдоль кривой компенсирует упадок точности и мы остаемся на том же самом уровне f-меры.

Если вы когда-нибудь изучали микроэкономику, то вам в голову должна была прийти аналогия с кривыми безразличия. Точность и полнота в данной ситуации это товары. f-мера — полезность от их потребления. Вдоль каждой кривой безразличия мы замещаем один товар другим и остаемся на том же уровне полезности.

Можно попробовать дать больший вес точности или полноте, если от нас требует этого задача и немного вытянуть кривые безразличия в сторону полноты или точности:

$$f = (\beta^2 + 1) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall}.$$

Если $\beta > 1$ приоритет отдаётся полноте, если $0 < \beta < 1$, то точности. Например, если $\beta = 2$, для нас более важна полнота. Если $\beta = 0.5$, более важна точность.

Упражнение 3

Бандерлог из Лога[2] ведёт блог, любит считать логарифмы и оценивать модели[3]. С помощью нового алгоритма Бандерлог решил задачу классификации по трём наблюдениям и получил $\hat{p}_i = \hat{P}(y_i = 1|x_i)$.

[2] деревня в Кадуйском районе Вологодской области

[3] Читай больше про приключения Бандерлога тут: https://github.com/bdemeshev/mlearn_pro

y_i	$\hat{p}_i$
1	0.7
0	0.2
0	0.3
1	0.25
0	0.1

а) Найдите ROC AUC.

б) Постройте ROC-кривую.

в) Постройте PR-кривую (кривая точность-полнота).

г) Найдите площадь под PR-кривой.

д) Как по-английски будет «бревно»?

Решение:

В предыдущей задаче мы немного обсудили метрики классификации. Когда мы на практике оцениваем модель, она, чаще всего, выплёвывает в нас не принадлежность объекта к классу в явном виде, а вероятности того, что наши объекты — единички.

Например, давайте думать в терминах оттока клиентов. Пусть наш сервис привлёк каких-то ребят в постоянные пользователи. Если они начнут унывать, им захочется свалить. Это называется оттоком. Давайте предположим, что модель выдаёт нам вероятность того, что человек решил приуныть. Если мы сможем понимать кто собрался приуныть, будем одаривать их ништяками и тогда они будут оставаться нашими клиентами.

Как понять, кто собирается приуныть, если модель выплёвывает на нас вероятности? Давайте выберем порог и будем считать, что все, у кого вероятность уныния $\geqslant 0.5$ относятся к классу 1. Есть вероятность, что они решат свалить. Их и будем одаривать ништяками. В нем случае получатся прогнозы: $1, 0, 0, 0, 0$. Если взять порог 0.3, получим прогнозы $1, 0, 1, 0, 0$.

Как выбрать порог, кого одаривать ништяками? Если у нас большой бюджет, можно попробовать поставить низкий порог, чтобы добиться большой полноты. Если маленький бюджет, то давайте поставим порог так, чтобы борьба с унынием была поточнее. Выбор порога зависит от специфики бизнеса и от того, что от нас хотят.

Видно, что точность и полнота зависят от выбора порога. А хотелось бы, чтобы та метрика, по которой мы выбираем модель, от порога не зависела. **Так рождаются идеи о метриках** `roc_auc` **и** `pr_auc`.

а) Представим себе два объекта: приунывший и нормальный. Представим себе, что модель предсказала нам вероятность уныния для первого объекта $\hat{p}_1$ и для второго $\hat{p}_2$.

$$y = 1 \qquad \hat{P}(y = 1) = \hat{p}_1$$
$$y = 0 \qquad \hat{P}(y = 1) = \hat{p}_2$$

Если у нас хорошая модель, то явно $\hat{p}_1 > \hat{p}_2$. Иначе модель всё путает и говорит, что неунывающие приуныли. Давайте посмотрим как часто на нашей выборке такая путаница происходит и **рассмотрим все возможные пары нулей и единичек.** Всего будет шесть пар.

$0.7 > 0.2$	ok
$0.7 > 0.3$	ok
$0.7 > 0.1$	ok
$0.25 > 0.2$	ok
$0.25 < 0.3$	not ok
$0.25 > 0.1$	ok

Отдельно обращаю ваше внимание, что не надо смотреть на пары из только нулей и только единичек. На самостоятельной работе часто люди смотрят на них тоже. Это ошибка.

Видим, что модель ошиблась в упорядочивании один раз. roc_auc — это доля пар, где модель оказалась права. В нашем случае это $\frac{5}{6}$. roc_auc принимает значения от 0.5 до 1, если её значения близки к 0.5, наш алгоритм ничем не лучше монетки, потому что он упорядочивает пары из унывших и нормальных случайно.

Такая метрика позволяет не привязываться к конкретному значению порога и видеть насколько классно у модели выходит упорядочивать пары объектов.

б) Величина, которую мы посчитали выше является площадью под roc-кривой (roc = receiver operating characteristic, иногда говорят «кривая ошибок»), с помощью которой часто визуализируют качество работы алгоритма. Когда говорять про roc-auc имется в виду area under the curve[4].

Чтобы нарисовать ROC-кривую, надо взять единичный квадрат на координатной плоскости, разбить его на m равных частей горизонтальными линиями и на n – вертикальными, где m – число 1 среди правильных меток теста (в нашем примере $m = 2$), n – число нулей ($n = 3$). В результате квадрат разбивается сеткой на $m \times n$ блоков.

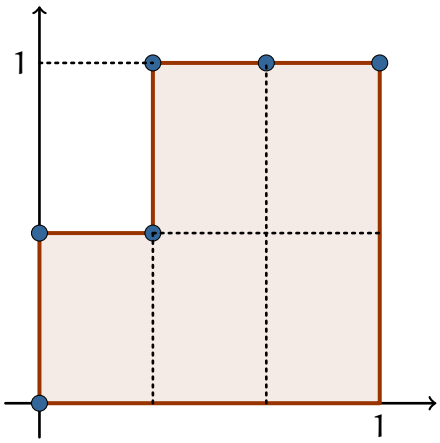
Отсортируем нашу табличку по значению вероятности, которую предсказала модель (в самостоятельной это тоже часто забывают сделать, обратите на это внимание).

y_i	$\hat{p}_i$
1	0.7
0	0.3
1	0.25
0	0.2
0	0.1

Теперь будем просматривать строки сверху вниз и прорисовывать на сетке линии, переходя их одного узла в другой. Стартуем из точки $(0, 0)$. Если значение метки класса в

[4] В блоге Дьяконова есть ещё задачки на ROC-AUC и объяснение что это за кривая, прочтите на досуге

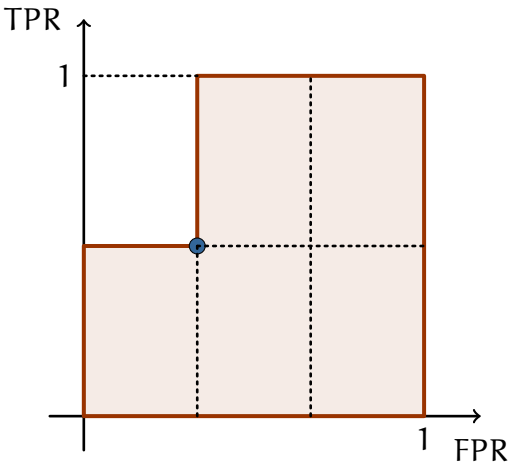
просматриваемой строке 1, то делаем шаг вверх; если 0, то делаем шаг вправо. Ясно, что в итоге мы попадём в точку $(1, 1)$, т.к. сделаем в сумме m шагов вверх и n шагов вправо.



Важный момент: если у нескольких объектов значения оценок равны, то мы делаем шаг в точку, которая на a блоков выше и b блоков правее, где a — число единиц в группе объектов с одним значением метки, b — число нулей в ней. Непонятно? Проработайте задачку из блога Дьяконова. Она там как раз такая.

Сетка разбила квадрат на $m \times n$ блоков. Ровно столько же пар вида (объект класса 1, объект класса 0), составленных из объектов тестовой выборки. Каждый закрашенный блок соответствует паре (объект класса 1, объект класса 0), для которой наш алгоритм правильно предсказал порядок (объект класса 1 получил оценку выше, чем объект класса 0), не закрашенный блок –– паре, на которой ошибся.

Можно объяснять построение кривой немного иначе, в терминах FPR — False Positive Rate и TPR — True Positive Rate. Но любой нормальный человек сразу же забывет эти стрёмные буквы. Тем не менее, по оси x на нешей картинке отложена именно FPR, а по оси y отложена TPR:



Выбору порога соответствует выбор точки на ROC-кривой. Например, на картинке выше нарисована точка, которая соответствует порогу между 0.3 и 0.25. Например,всё, что больше 0.27 мы объявляем единицей. Если двигать порог в рамках указанного выше диапазона, ничего меняться не будет.

Мы видим, что $\mathrm{FPR} = \frac{1}{3}$ — это процент точек класса 0, которые неверно классифицированы нашим алгоритмом, а $\mathrm{TPR} = 0.5$ — процент точек класса 1, которые верно классифицированы нашим алгоритмом. Кстати говоря, TPR и полнота, $\mathtt{recall}$, это одно и то же.

Метрика классная и интуитивно понятная, но страдает большой проблемой: она чувствительна к дисбалансу в классах. Также как и $\mathtt{Accuracy}$.

Например, пусть у нас есть 100000 пчёл и 100 из них неправильные. Если алгоритм идеально понимает какие 100 пчёл неправильные, мы получаем $\mathrm{TPR} = 1$ и $\mathrm{FPR} = 0$.

Посмотрим теперь на плохой алгоритм, который находит 95 неправильных пчёл из 100 и объявляет 50000 правильных пчёл плохими. Для такого алгоритма получится $\mathrm{TPR} = 0.95$, а $\mathrm{FPR} = 0.05$. Это довольно близко к идеальному алгоритму, хотя при этом точность алгоритма оставляет желать лучшего.

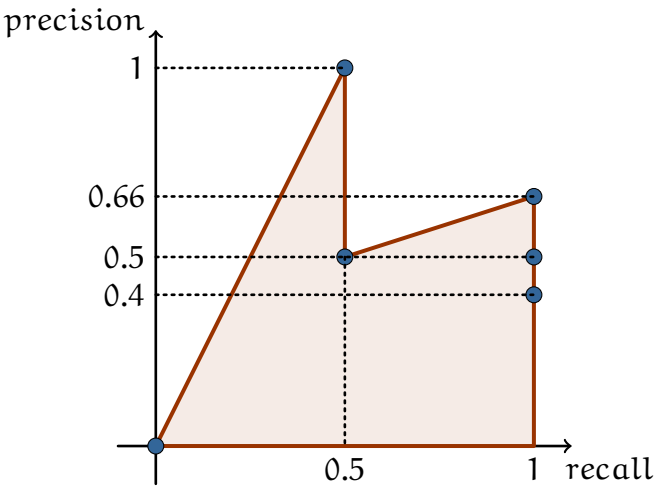
Если положительный класс существенно меньше по размеру, $\mathtt{roc\_auc}$ может давать неадекватную оценку качества алгоритма. Вылечить такой недостаток позволяет другая метрика, $\mathtt{pr\_auc}$ (precision-recall area under curve).

в) Давайте по оси x откладывать полноту, а по оси y точность. Будем по очереди перебирать разные пороги и считать для них точность и полноту. Нанесём на картинку все полученные точки и соединим их. Это и будет $\mathtt{precision - recall}$ **кривая.**

Например, если мы возьмём порог $\geqslant 0.1$, тогда все объекты будут принадлежать к классу 1 точность составит $\frac{2}{5}$, а полнота 1. Если мы возьмём порог $\geqslant 0.2$, тогда один объект будет нулевым, а остальные четыре единичными. Точность составит $\frac{2}{4}$, а полнота 1. По аналогии получим точность и полноту при порогах $\geqslant 0.25$, $\geqslant 0.3$ и $\geqslant 0.7$.

порог	точность	полнота
0.1	$\frac{2}{5}$	1
0.2	$\frac{2}{4}$	1
0.25	$\frac{2}{3}$	1
0.3	$\frac{1}{2}$	$\frac{1}{2}$
0.7	1	$\frac{1}{2}$
0.8	0	0

Строим кривую!



PR- кривая всегда начинается из точки $(0, 0)$ и заканчивается в точке $(1, r)$, где r — это доля объектов первого класса в выборке.

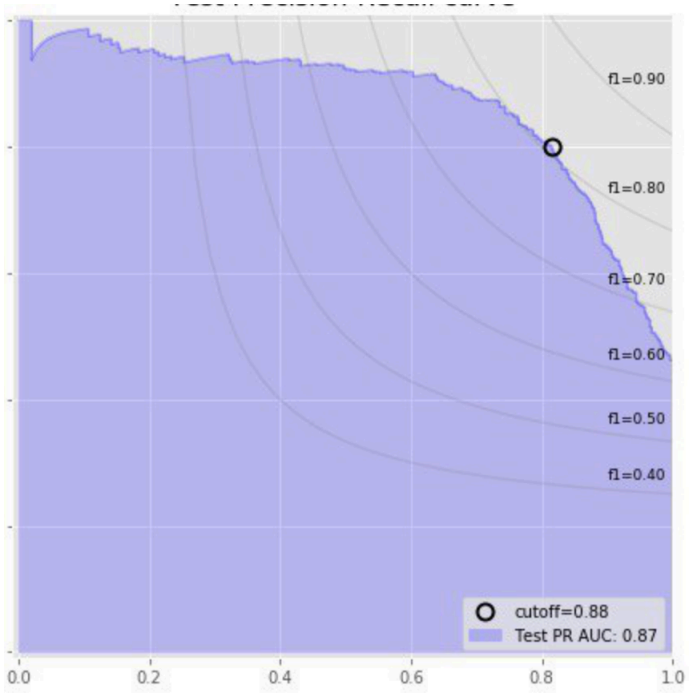
В случае идеального классификатора, то есть если существует такой порог, что и точность, и полнота равны 100%, кривая будет проходить через точку $(1, 1)$. Таким образом, чем ближе кривая пройдет к этой точке, тем лучше оценки. Площадь под этой кривой может быть хорошей мерой качества оценок принадлежности к классу 1. Такая метрика называется pr-auc, или площадь под PR-кривой.

г) Посчитав площадь под кривой получим pr-auc. Будем делать это, считая площади треугольников и прямоугольников.

$$0.5 \cdot 1 \cdot 0.5 + 0.5 \cdot 0.5 \cdot (0.66 - 0.5) + 0.5^2 = 0.25 + 0.04 + 0.25 = 0.54$$

Эта метрика подобно roc-auc не зависит от выбора порога и отражает способность модели правильно упорядочивать пары, но немножечко в другом плане. Эта метрика строится в осях точность и полнота. Из-за этого она нечувствительна к дисбалансу в выборке. В примере с пчёлами мы получаем, что $\mathtt{recall} = 0.95$, а $\mathtt{precision} = \frac{95}{50095}$, и видим, что алгоритм не очень удачный.

Кстати говоря, если снова обратиться к микроэкономике, можно провести аналогию между pr-кривой и допустимым множеством товаров. Самая классное сочетания precision и recall, доступное нам, лежит на самой высокой f-кривой (кривой безразличия), касающейся precision-recall кривой.



Заменить картинку на нормальную

д) log

Ещё задачи!

Тут лежит ещё несколько задач для самостоятельного решения. Возможно, похожие будут в самостоятельной работе...

Упражнение 4

Бандерлог начинает все определения со слов «это доля правильных ответов»:

а) accuracy — это доля правильных ответов...

б) точность (precision) — это доля правильных ответов...

в) полнота (recall) — это доля правильных ответов...

г) TPR — это доля правильных ответов...

Закончите определения Бандерлога так, чтобы они были, хм, правильными.

Задача 5

Упражнение 5

Бандерлог обучил модель для классификации и получил вектор предсказанных вероятностей принадлежностей к классу 1.

y_i	b_i
1	0.9
0	0.1
0	0.75
1	0.56
1	0.2
0	0.37
0	0.25

а) Бинаризуйте ответ по порогу t и посчитайте точность и полноту для $t = 0.3$ и для $t = 0.8$.

б) Какой порог вы бы выбрали?

в) Постройте ROC-кривую и найдите площадь под ней.

Решение:

а) Построим в нашей табличке две колонки с прогнозами. Для удобства.

y_i	b_i	$t = 0.3$	$t = 0.8$
1	0.9	1	1
0	0.1	0	0
0	0.75	1	0
1	0.56	1	0
1	0.2	0	0
0	0.37	1	0
0	0.25	0	0

Построим для обоих порогов матрицы ошибок. Слева матрица, соответствующая порогу 0.3, справа матрица, соответствующая порогу 0.8.

	$y = 1$	$y = 0$
$\hat{y} = 1$	2	2
$\hat{y} = 0$	1	2

	$y = 1$	$y = 0$
$\hat{y} = 1$	1	1
$\hat{y} = 0$	2	3

Считаем для обоих случаев точность и полноту.

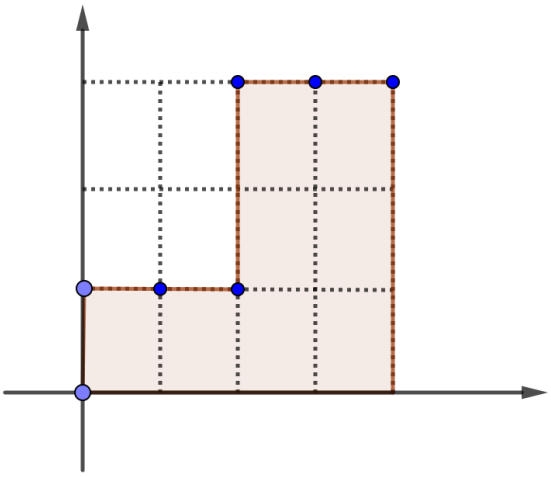
$$\text{Precision}_1 = 0.5 \qquad \text{Recall}_1 = 0.66$$
$$\text{Precision}_2 = 0.5 \qquad \text{Recall}_2 = 0.33$$

б) Обе модели дают одинаковую точность при разной полноте. У первой модели полнота повыше, имеет смысл выбрать её (но это неточно, надо бы это проверить на большем числе данных).

в) Построим roc-кривую. Для этого отсортируем все наблюдения в нашей табличке по возрастанию.

y_i	b_i
1	0.9
0	0.75
1	0.56
0	0.37
0	0.25
1	0.2
0	0.1

Заведём сетку размера 4 на 3 и начнём делать шаги по ней так, как было описано во второй задаче.



Профит, получили roc-кривую. Видим, что площадь под ней равна $\frac{8}{12}$. Иным языком говоря, среди 12 пар ноликов и единичек, вероятности отсортированы так, как нам хотелось бы 8 раз.

Для первого наблюдения $(1, 0.9)$ все четыре пары верны. Для второго $(1, 0.56)$ будет одна ошибочная сортировка, для третьей $(1, 0.2)$ будет три ошибочных сортировки.

Упражнение 6

Алгоритм бинарной классификации, придуманный Бандерлогом, выдаёт оценки вероятности $\hat{p}_i = \hat{P}(y_i = 1)$. Всего у Бандерлога 10000 наблюдений. Если ранжировать их по возрастанию $\hat{p}_i$, то окажется что наблюдения с $y_i = 1$ занимают ровно места с 5501 по 5600.

Найдите площадь по ROC-кривой и площадь под PR-кривой.

Упражнение 7

Для задачи классификации есть довольно много разных метрик, которые оценивают качество

модели для её решения. На практике иногда оказывается, что по одной метрике наша модель может иметь более высокое качество, чем другая, а по другой метрике — более низкое.

Предположим, что у нас есть две модели, предсказывающие принадлежность к одному из классов: котики, 1 или собачки, 0. Качество моделей оценивается на выборке из пяти объектов. Оказывается, что по метрике A первая модель лучше, а по метрике B, вторая. Вам нужно придумать примеры тестовых меток y и предсказаний $\hat{y}_1$ и $\hat{y}_2$ для каждой пары метрик A и B:

а) A — precision (при пороге 0.5), B — auc-roc;

б) A — precision (при пороге 0.5), B — recall (при пороге 0.5);

в) A — F1-score (при пороге доставляющем максимум), B — auc-roc.

Решение:

В решении просто примеры подобных меток и прогнозов. Понятное дело, что у задачки нет одного решения.

1. Пусть $y = (1, 1, 1, 0, 0)$. Для первого алгоритма $p_A = (0.95, 0.8, 0.8, 0.9, 0.7)$. Для второго алгоритма $p_B = (0.7, 0.8, 0.3, 0.9, 0.4)$. Порог 0.5. Всего у нас 6 пар классов. Видим, что для первого алгоритма ROC-AUC составит $\frac{4}{6} = 0.66$. Для второго ROC-AUC составит 0.33. Посчитаем точность. Для первого она составит 0.6. Для второго 0.66. Видим, что второй алгоритм лучше по точности. Первый лучше по ROC-AUC.
2. Пусть $y = (1, 1, 1, 0, 0)$. Для первого алгоритма $\hat{p}_A = (0.9, 0.8, 0.8, 0.8, 0.8)$. Для второго $\hat{p}_B = (0.8, 0.8, 0.4, 0.4, 0.8)$. Порог 0.5. Для первого алгоритма все прогнозы единичные. Точность составит 0.6. Полнота составит 1. Для второго алгоритма одна единица выпадает из прогнозирования и один ноль. Точность составит 0.66. Полнота составит тоже 0.66. Видим, что первый алгоритм оказался лучше по полноте, второй оказался лучше по точности.
3. Пусть $y = (1, 1, 1, 1, 0)$. Для первого алгоритма $\hat{p}_A = (1, 1, 1, 0, 1)$. Для второго алгоритма $\hat{p}_B = (1, 1, 1, 1, 1)$. Тогда Для первого алгоритма $ROC - AUC = 0.75$. Для второго $ROC - AUC = 0.5$. В случае f-меры получаем обратную картину. В первой ситуации $f = 2 \cdot \frac{0.75 \cdot 0.75}{0.75 + 0.75} = 0.75$ для любого порога. Во второй ситуации получаем $2 \cdot \frac{0.8 \cdot 1}{0.8 + 1} = 0.88$.

Ещё мелкие задачки на просто посчитать метрики

Мб задачка со взвешиванием ошибок на деньги