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

Таблица 1: Матрица ошибок

Ещё раз про метрики качества

Пять вопросов коллоквиума касаются метрик качества алгоритмов. На мой взгляд, это одна из самых сложных тем курса, поэтому часть семинара снова посвятим разговору о метриках.

Матрица ошибок

Рассмотрим задачу бинарной классификации. Пусть $\mathbb{Y} = \{+1, -1\}$, то есть объекты могут быть либо класса «+1», либо «−1».

Класс «+1» назовём положительным, а класс «−1» — отрицательным.

Пусть мы построили классификатор $a(x)$, с помощью которого классифицируем объект x_i из обучающей или контрольной выборки. Очевидно, возможны четыре случая:

- $a(x_i) = +1,\ y_i = +1$;
- $a(x_i) = +1,\ y_i = -1$;
- $a(x_i) = -1,\ y_i = +1$;
- $a(x_i) = -1,\ y_i = -1$;

Применим классификатор ко всей контрольной выборке и посчитаем, сколько объектов отвечает каждому из этих четырёх исходов. Результат занесём в таблицу, которая часто называется матрицей ошибок.

Каждая из четырёх ячеек таблицы имеет своё название (см. 1). Каждое из названий состоит из двух слов:

$$\underbrace{\text{Классификатор ответил верно?}}_{\text{True или False}}\ \underbrace{\text{К какому классу алгоритм отнёс ответ?}}_{\text{Positive или Negative}}$$

	$y=+1$	$y=-1$
$a(x)=+1$	10	15
$a(x)=-1$	5	110

Таблица 2: Фильтрация спама. Матрица ошибок алгоритма $a(x)$.

	$y=+1$	$y=-1$
$a(x)=+1$	0	0
$a(x)=-1$	15	125

Таблица 3: Фильтрация спама. Матрица ошибок глупого алгоритма.

27. Что такое доля правильных ответов? В чём заключаются её проблемы?

Доля правильных ответов (accuracy) — это... доля правильных ответов.

Давайте запишем её через матрицу ошибок:

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

Эта метрика не очень хороша, особенно когда TP $<$ FP. Эта ситуация называется accuracy paradox.

Пусть мы решаем задачу фильтрации спама, то есть объекты — письма. Пусть «+1» — спам, а «−1» — не спам.

Для решения мы построили алгоритм $a(x)$, матрица ошибок которого представлена в таблице 2.

Посчитаем долю правильных ответов:

$$\text{accuracy} = \frac{10 + 110}{10 + 15 + 5 + 110} = \frac{120}{140} \approx 0.86.$$

Кажется, что результат вполне неплох, но так ли это?

Теперь рассмотрим матрицу ошибок глупого алгоритма, который всегда говорит, что письмо — не спам (см. 3).

Посчитаем долю правильных ответов этого алгоритма:

$$\text{accuracy} = \frac{0 + 125}{0 + 0 + 15 + 125} = \frac{125}{140} \approx 0.89.$$

Доля правильных ответов у глупого алгоритма выше! Такое поведение не позволяет назвать accuracy хорошей метрикой качества алгоритма.

28. Что такое точность и полнота?

Точность (precision) — это доля положительных объектов среди объектов, выделенных классификатором как положительные.

Полнота (recall) — это доля объектов, выделенных классификатором как положительные, среди всех положительных объектов.

Если понять эти две строчки, то можно выписать соответствующие формулы через матрицу ошибок:

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

В идеале мы хотим, чтобы и точность, и полнота, были равны единице.

Точность — это про то, как редко алгоритм неправильно относит к положительному классу.

Полнота — это про то, как редко алгоритм называет положительный объект отрицательным.

Ясно, что точность без полноты и полнота без точности являются очень глупыми метриками. Например, пусть «алгоритм» — это врач, класс «+1» — пациент болен, а «−1» — здоров. Тогда ленивый врач, который будет говорить пациенту, что он болен, только если это совсем очевидно, будет иметь стопроцентную точность. А неуверенный в себе врач, который каждому пациенту скажет, что он болен, а потом отправит на дальнейшее лечение, будет иметь высокую полноту. Но профессионализм и того, и другого, сомнителен.

Ещё есть так называемая F-мера (F-score), которая позволяет учесть и точность, и полноту:

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

Чем больше F-мера, тем лучше.

29. В чём заключается разница между метриками Accuracy и Precision?

Здесь стоит опять рассказать про accuracy и precision.

Это вопрос-шутка: и accuracy, precision на русский язык можно перевести с английского языка как точность. Но это совсем разные вещи.

Напомню, accuracy мы переводим как «долю правильных ответов», а precision — как «точность».

30. Что такое ROC-кривая? Что такое AUC-ROC? Для чего он используется?

Здесь мы сначала вспоминаем про то, что очень часто алгоритм может выдавать некоторую оценку вероятности того, что объект лежит в классе «+1». Например, в scikit-learn во многих алгоритмах есть метод predict_proba.

А дальше бинаризовать ответ можно по некому порогу $t \in [0, 1]$. Для математического удобства будем считать, что $\mathbb{Y} = \{1, 0\}$. Пусть $b(x)$ — алгоритм, возвращающий оценку вероятности принадлежности классу «1», а

$$a(x) = [b(x) > t]$$

— это классификатор.

Каждому значению порога t соответствует классификатор $a(x) = [b(x) > t]$. Для этого классификатора можно посчитать две характеристики:

- Долю отрицательных объектов[1], про которые классификатор говорит, что они положительны — False Positive Rate (FPR).
- Долю положительных объектов, про которые классификатор верно говорит, что они положительные — True Positive Rate (TPR).

Выпишем соответствующие формулы через матрицу ошибок:

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

Чем больше TPR и чем меньше FPR, тем лучше. [2]

Будем считать, что главное выбрать хороший алгоритм $b(x)$, а порог t можно подобрать потом. Поэтому наша цель — оценить качество алгоритма $b(x)$.

Теперь разберёмся, сколько различных значений порога t вообще можно выбрать. Вообще — бесконечное число, но на самом деле по обучающей выборке объема ℓ различимы только $\ell + 1$ вариантов выбора порога.

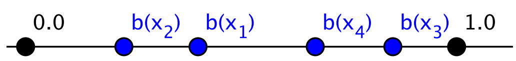

Рис. 1: Пример результата работы функции $b(x)$ на выборке из четырёх элементов.

Пусть, например, выборка из четырёх элементов, и алгоритм b оценил веротности принадлежности классу «1» так, как показано на рисунке 1.

[1]Сейчас для определённости считаем, что объекты класса «1» положительные, а «0» — отрицательные

[2]Лично мне проще вспомнить сначала формулы, а потом смысл FPR и TPR. Действительно, с числителем и одним слагаемым в знаменателе всё ясно. А второе слагаемое в знаменателе восстанавливается, если запомнить, что обе эти характеристики усредняются по y, а не по $a(x)$ (то есть по столбцам матрицы ошибок).

Тогда принципиально различимы пять вариантов выбора t: из $[0, b(x_2))$, из $[b(x_2), b(x_1))$, из $[b(x_1), b(x_4))$, из $[b(x_4), b(x_3))$ и из $[b(x_3), 1]$.

Для определённости выберем пороги равными $b(x_2) - \varepsilon$ (где ε — какое-то маленькое число), $b(x_2)$, $b(x_1)$, $b(x_4)$ и $b(x_3)$.

Теперь построим кривую таким образом: для всех значений порога в порядке возрастания посчитаем FPR и TPR алгоритма $a(x) = [b(x) > t]$. Отметим точку с координатами (FPR, TPR) на графике и соединим с предыдущей точкой, если такая есть. Такая кривая называется ROC-кривой.

Ясно, что все ROC-кривые проходят через точки (0, 0) и (1, 1), ROC-кривая идеального алгоритма проходит через точку (1, 0).

Существует эффективный алгоритм, который строит ROC-кривую за один проход по выборке, но о нём вам не рассказывали, и знать его не обязательно.

Пример ROC-кривой для маленькой выборки можно посмотреть на слайде 43 лекции №10.

Теперь введём понятие AUC-ROC — площади под ROC-кривой. Чем больше площадь, тем в среднем при большем количестве значений порога получается хороший классификатор.

Идеальная AUC-ROC равен 1, ужасная AUC-ROC примерно равна $1/2$.

Заметим, что так как FPR и TPR нормируются на размеры классов, ROC-AUC не поменяется при изменении баланса классов.