1. Задача классификации. Логистическая регрессия

Цель занятия

В результате обучения на этой неделе вы:

- научитесь решать задачи линейной классификации в машинном обучении
- познакомитесь с понятием правдоподобия в задачах машинного обучения
- поймете, как использовать модель логистической регрессии в задачах бинарной и мультиклассовой классификации
- узнаете различные метрики оценки качества классификации
- поймете назначение и принцип работы фреймворка PyTorch

План занятия

Конспект занятия

1. Задача линейной классификации

Формальная постановка

Задача классификации является одной из наиболее широко встречающихся задач в мире машинного обучения. Поэтому важно понять, какими свойствами обладают классификаторы (модели, которые решают задачу классификации), и на что стоит обратить внимание.

Поставим формально задачу классификации конкретно для бинарного случая. Мы работаем с двумя классами: позитивный и негативный классы; класс A, класс B; класс 1, класс 0, и т.д.

Есть некоторая выборка:

$X \in R^{n \times p}$ – объекты.

Есть значение целевой переменной:

$Y \in C^{n}$, $C = \{-1, 1\}$, $|C| < +\infty$

Наша задача – найти некоторый алгоритм классификации, который будет восстанавливать метку класса максимально точно:

$$c(X) = \hat{Y} \approx Y$$

Ограничения на метки класса:

- они не упорядочены
- их конечное число

На первый взгляд эта задача проще, чем задача регрессии. Вместо попытки предсказать континуальное число (число с числовой оси) мы предсказываем лишь один элемент из какого-то конечного набора. Поэтому задача классификации предоставляет меньшую свободу выбора, и казалось бы, решать её проще. Но, с другой стороны, в задаче регрессии у нас непрерывная целевая переменная, мы можем брать производную, оценивать какие-то градиенты. В дискретном случае посчитать производные сложно. Поэтому понадобятся другие методы решения задачи.

Линейный классификатор

В задаче классификации обычно в качестве функции потерь рассматривают ошибку классификации.

Линейный классификатор:

$$c(x) = \begin{cases} 1, & \text{if } f(x) \geq 0 \\ -1, & \text{if } f(x) < 0 \end{cases}$$

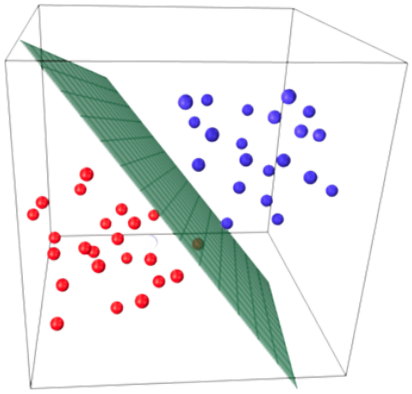
Задача бинарная, значит всего две метки класса. Для простоты предположим, что это +1 и -1. И у нас есть некоторая линейная функция $f(x)$, которая каждую точку признакового пространства отображает в число. По сути линейный функционал. Если функционал предсказывает неотрицательное число, то это метка класса 1, если отрицательное число – метка класса -1.

Можно переписать по-другому:

$$c(x) = sign(f(x)) = sign\left(x^{T}\omega\right)$$

В линейной задаче мы всегда можем обнулить числовой порог. В данном случае у нас порог 0.

Наша линейная модель делит признаковое пространство на два полупространства:



Когда мы смотрим на произведение $x^T\omega$, мы проецируем точку на плоскость, и смотрим, с какой стороны мы её спроецировали: со стороны нормального вектора или с противоположной.

Функция потерь

Чтобы ввести функцию потерь, нужно понятие **отступа** (Margin) – то, насколько глубоко "сидит" точка внутри своего класса.

$$M_i = y_i \cdot f(x_i) = y_i \cdot x_i^T\omega$$

Если отступ положительный, точка находится внутри своего класса, отрицательный – точка находится в глубине чужого класса.

$$M_i > 0 \Leftrightarrow y_i = c(x_i)$$

$$M_i \leq 0 \Leftrightarrow y_i \neq c(x_i)$$

Функция ошибки будет "смотреть" на количество ошибок классификации

$$\text{Empirical risk} = \sum_{\text{by objects}} \text{Loss on object} \rightarrow \min_{\text{model params}}$$

На каждом объекте есть истинная метка класса и предсказанная метка класса. Мы можем их сравнить: угадали или нет.

Поэтому наша функция ошибок – количество ошибок в классификации, на скольких объектах мы не угадали метку класса.

$$L_{mis}(y_i^t, y_i^p) = [y_i^t \neq y_i^p] = [M_i \leq 0]$$

Или на скольки объектах мы получили неположительный отступ.

Квадратные скобки – индикатор того, что предикат внутри скобок является истинным.

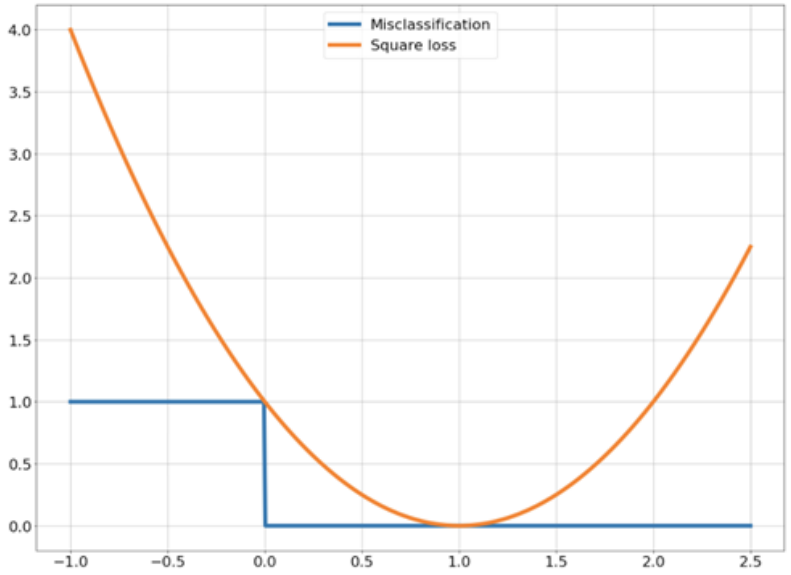
$$[P] = \begin{cases} 1, & \text{if } P \text{ is true} \\ 0, & \text{otherwise} \end{cases}$$

Функция потерь недиффиренецируема. Что с ней делать?

Точка, которая лежит глубоко внутри своего класса, и точка, которая лежит на границе своего класса, одинаково не вносят в ошибку модели. Если точка лежит на границе с чужим классом или углубилась на ε в чужой класс, это принесет ошибку 1 так же, как и ошибка, которая находится в глубине чужого класса. Из нашей функции потерь мы не можем понять, уверен наш классификатор в том или ином объекте или нет. Мы не можем оштрафовать классификатор за уверенность в том или ином объекте.

Вместо минимизации функции ошибки мы можем взять верхнюю оценку на данную функцию и минимизировать её.

$Y \in \{-1, 1\} \rightarrow Y \in R$



<u>Оптимизационная задача</u>

Введем среднюю квадратичную ошибку:

$$L_{MSE} = \left(y_i - x_i^T \omega\right)^2 = \frac{\left(y_i^2 - y_i \cdot x_i^T \omega\right)^2}{y_i^2} = \left(1 - y_i \cdot x_i^T \omega\right)^2 = \left(1 - M_i\right)^2$$

Если будем минимизировать данную функцию потерь, мы и количество ошибок классификации понизим, то есть получим решение.

Проблема этой модели: модель будет оштрафована не только за то, что она ошибается на каких-то точках, которые в глубине чужого класса, но и будет

оштрафована за то, что она будет слишком сильно “уверена”, что точка находится в глубине своего класса.

Функций потерь большое количество:

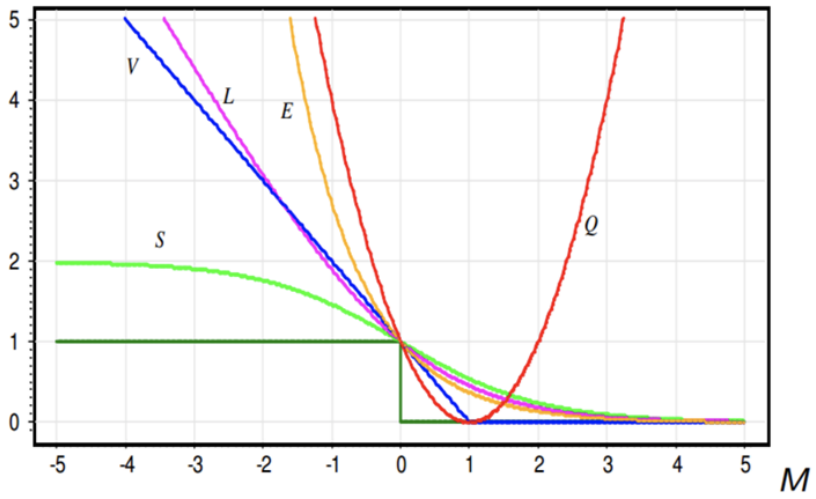
square loss $Q(M) = (1 - M)^2$

hinge loss $V(M) = (1 - M)_+$

savage loss $S(M) = 2\left(1 + e^M\right)^{-1}$

logistic loss $L(M) = log_2\left(1 + e^{-M}\right)$

exponential loss $E(M) = e^{-M}$



Основных функций потерь две: кросс-энтропия (бинарная кросс-энтропия, логистическая функция потерь) и hinge loss (соответствует методу опорных векторов).

2. Правдоподобие

Когда мы решаем оптимизационную задачу, мы используем оценку максимального правдоподобия.

Предположим, что у нас есть некоторая выборка: X – объекты, Y – таргеты. Мы бы хотели построить некоторую модель, которая очень хорошо описывает Y при условии X.

Правдоподобие – функция, которая позволяет оценить, насколько правдоподобна данная выборка при условии, что она описывается моделью с параметрами θ :

$L(\theta|X,\ Y) = P(X,\ Y|\theta)$

Если возьмем модель линейной регрессии, то θ – параметры, веса линейной регрессии.

Мы хотим понять, насколько вероятно пронаблюдать пару X и Y, если бы Y были сгенерированы с помощью весов θ.

Правдоподобие – вероятность пронаблюдать X и Y при условии θ.

Правдоподобие не является вероятностным распределением над θ. Мы бы хотели найти оптимальное значение параметров. То есть такие параметры, при которых пронаблюдать X и Y было наиболее вероятно.

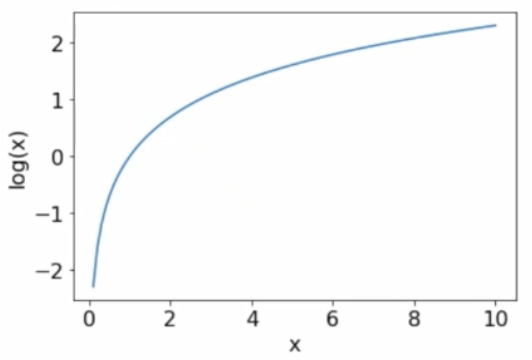
$$L(\theta|X,Y) \longrightarrow \max_{\theta}$$

Будем считать, что величины независимы, тогда их совместная вероятность будет факторизована:

$$L(\theta|X,\ Y) = P(X,\ Y|\theta) = \prod_i P\left(x_i,\ y_i|\theta\right)$$

У нас выполняется свойство i.i.d.

Чтобы оптимизировать функцию произведения, лучше всего использовать логарифм:



Логарифм – монотонно возрастающая функция. А значит решение нашей задачи оптимизации будет достигаться в одной и той же точке.

Логарифм произведения есть сумма логарифмов:

$$\log L(\theta|X,Y) = \sum_i \log P(x_i, y_i|\theta) \longrightarrow \max_{\theta}$$

Теперь мы можем использовать градиентную оптимизацию.

3. Логистическая регрессия

Пусть у нас есть некоторая бинарная классификация. Мы хотим предсказывать вероятность положительного класса при условии x:

$$p_+ = P(y = 1|x) \in [0, 1]$$

Предсказывать число от 0 до 1 мы не умеем. Мы владеем техникой линейного отображения. Линейное отображение отображает вектор в R.

$$y = x^T \omega \in R$$

Используя величину

$$\frac{p_+}{1-p_+} \in [0, +\infty)$$

мы можем перейти к числовой полуоси.

$$log\frac{p_+}{1-p_+} \in R$$

Тогда

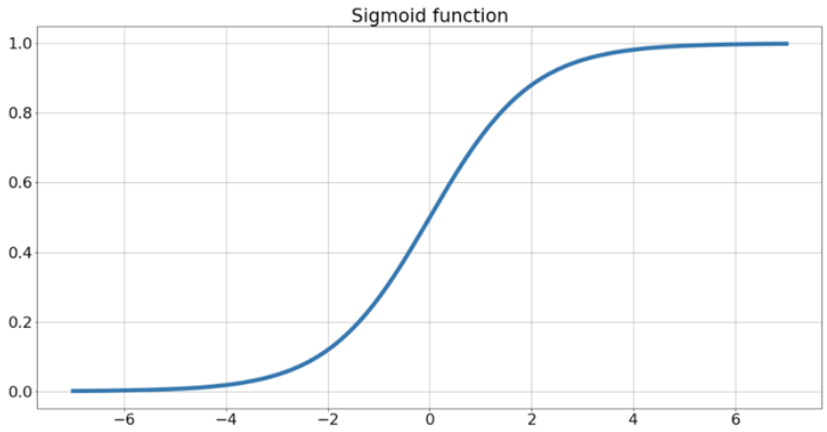
$$\frac{p_+}{1-p_+} = exp\left(x^T \omega\right)$$

Получаем логистическую регрессию

$$p_+ = \frac{1}{1+exp\left(-x^T \omega\right)} = \sigma\left(x^T \omega\right)$$ – сигмоида.

Сигмоида – функция, которая отображает числовую ось в [0, 1].

$$\sigma(x) = \frac{1}{1+exp(-x)}$$



Теперь мы можем построить бинарный классификатор. Сигмоида позволяет предсказывать вероятность для каждой точки в признаковом пространстве. Берем

линейную модель, применяем сигмоиду, получаем вероятность положительного класса.

Мы задаем в каждой точке вероятностного пространства некоторое распределение – распределение Бернулли.

У сигмоиды есть свойства:

$1 - \sigma(x) = \sigma(-x)$

$\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$

Теперь мы можем настраивать параметры модели, используя метод максимального правдоподобия

$$logL(\omega|X,Y) = logP(X,Y|\omega) = log \prod_{i=1}^{n} P(x_i, y_i|\omega)$$

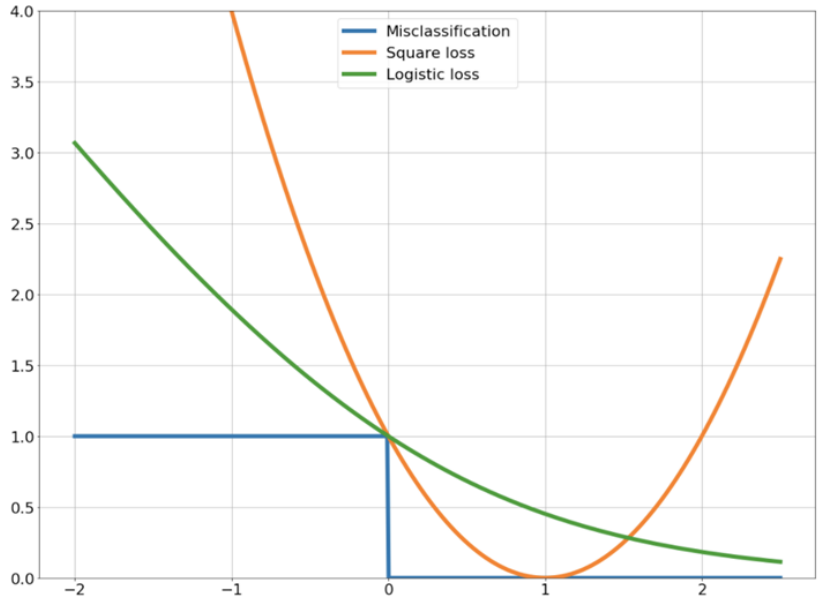
Если $y_i = 1$, то $P(x_i, 1|\omega) = \sigma_\omega(x_i) = \sigma_\omega(M_i)$

Если $y_i = -1$, то $P(x_i, -1|\omega) = 1 - \sigma_\omega(x_i) = \sigma_\omega(-x_i) = \sigma_\omega(M_i)$

$$\log L(w|X,Y) = \sum_{i=1}^{n} \log \sigma_w(M_i) = -\sum_{i=1}^{n} \log(1 + \exp(-M_i)) \to \max_w$$

Получаем логистическую функцию потерь.

$$L_{Logistic} = log(1 + exp(-M_i))$$



Логистическая функция потерь является верхней оценкой ошибки классификации, поскольку мы исходили из простых предположений. Мы бы хотели построить такую

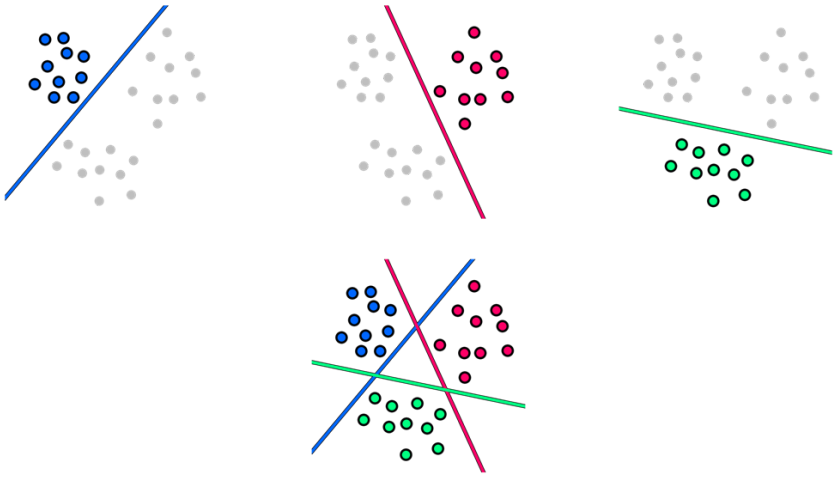
модель, которая для каждой точки пространства будет предсказывать вероятность принадлежности положительному классу так, чтобы для всех объектов положительного класса вероятность была максимально близко к единице, а для отрицательного класса – к нулю. И модель штрафуется за то, что она предсказывает низкую вероятность для объекта положительного класса или высокую вероятность для объектов отрицательного класса.

4. Мультиклассовая классификация

Логистическая регрессия позволяет решать задачи бинарной классификации гораздо эффективнее, чем, например, метод kNN. Но что если классов 5? Или 3.

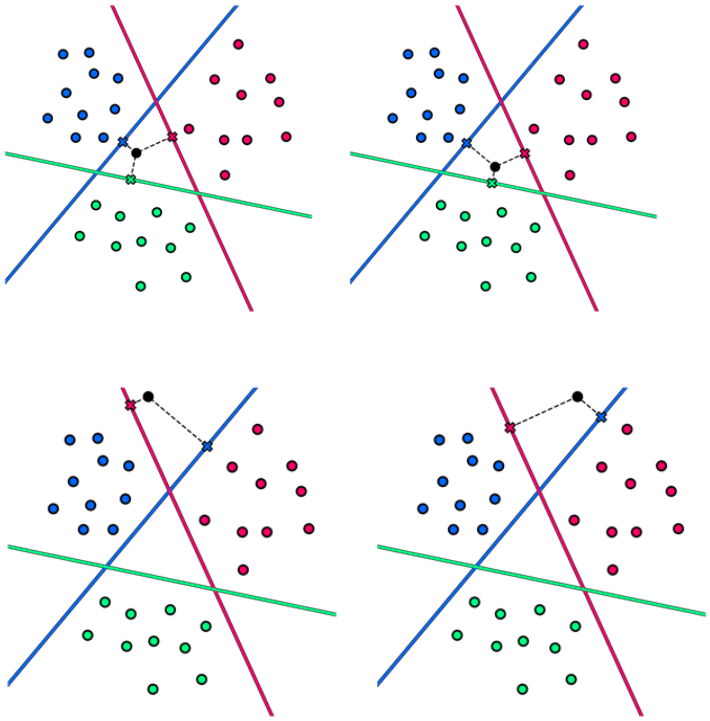
Тут нужна стратегия, которая позволит от бинарной задачи классификации перейти к многоклассовой.

Один против всех



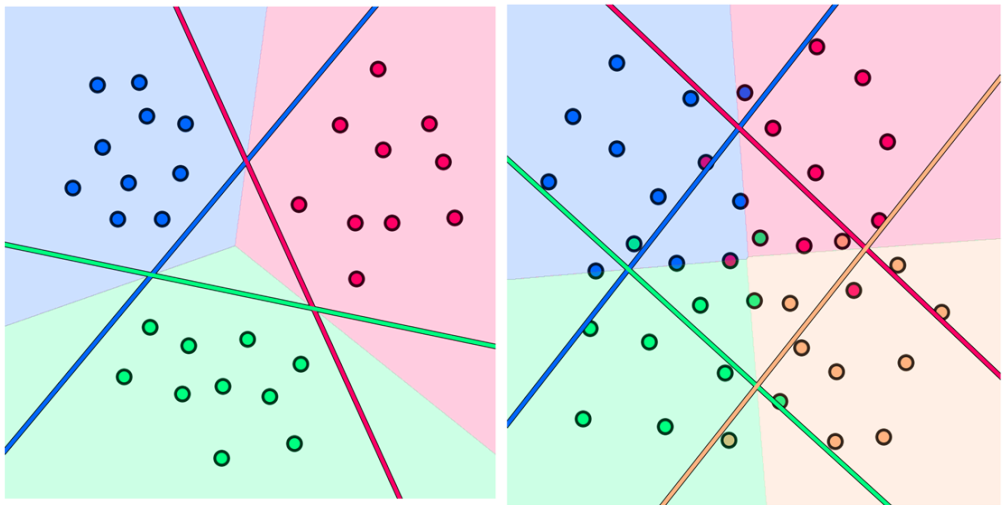
Для каждого класса строим бинарный классификатор, который отличает данный класс от всего остального.

Но что если найдутся регионы, не принадлежащие ни одному классу?



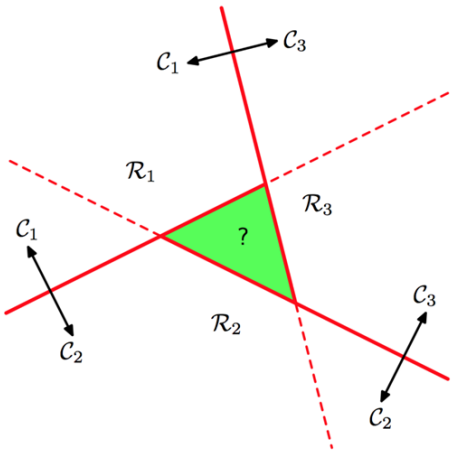
Мы предсказываем не только метку класса, но и **отступ**. Отступ можно использовать, как уверенность в том, что объект принадлежит данному классу.

На основании отступа получим следующие картины:



Один против одного

Строим бинарные классификаторы для каждой пары классов.



Сравнение подходов

	Один против всех	Один против одного
количество классификаторов	k	$\frac{k(k-1)}{2}$
dataset for each	full	subsampled

5. Метрики классификации

Поговорим о том, как можно измерить качество полученной модели, используя доступные данные, валидационную выборку, и как понять, подходит ли модель под ту или иную бизнес задачу.

В задаче регрессии использовались те же самые функции ошибки, что использовались при обучении: MSE, MAE, …

В классификации свои методы оценки качества модели.

Accuracy

$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\left[y_i^t = y_i^p\right]$ – точность (accuracy) классификации. Какая доля объектов выборки была классифицирована правильно.

Пример.

target: 1 0 1 0 0 0 0 1 0 0

predicted: 0 0 1 0 0 0 0 1 1 0

$Accuracy = \frac{8}{10} = 0.8$

Accuracy имеет недостатки. Представим датасет, который очень сильно не сбалансирован. 99% объектов относятся к одному классу, а 1% – к другому. У нас есть алгоритм, который всегда предсказывает первый класс. При этом Accuracy будет 99%. В этом случае используют Balanced accuracy:

$$Balanced\ accuracy = \frac{1}{C}\sum_{k=1}^{C}\frac{\sum_i\left[y_i^t=k\ and\ y_i^t=y_i^p\right]}{\sum_i\left[y_i^t=k\right]}$$

Balanced accuracy учитывает размер каждого класса. Мы смотрим на долю классификации каждого из классов и потом усредняем.

Точность и полнота

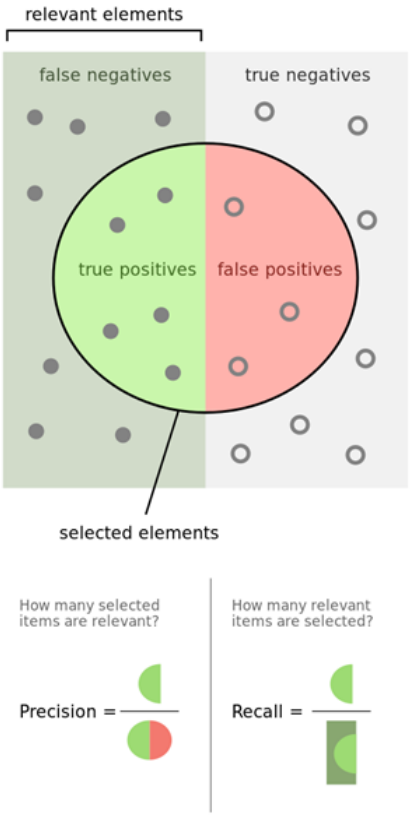
Точность и полнота (Precision and Recall) показывают нам оценку точности с точки зрения ошибки первого рода и второго рода.

У нас есть наблюдение, и мы можем правильно отклассифицировать объекты, принадлежащие к позитивному классу и к негативному классу. И неправильно отнести другие объекты к позитивному и негативному классам.

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

$Precision = \frac{TP}{TP+FP}$ – насколько точно можно оценить метку положительного класса среди всех предсказанных объектов положительного класса.

$Recall = \frac{TP}{TP+FN}$ – показатель, насколько много объектов из истинно положительного класса мы смогли охватить.



Точность позволяют нам оценить, насколько мы уверены, что предсказанная метка положительного класса релевантна. Полнота позволяет оценить, насколько мы уверены, что мы не потеряли никакие объекты из положительного класса.

В оценке точности и полноты у нас есть целевой класс TP, на который мы в первую очередь обращаем внимание. При несбалансированной выборке точность будет 90%, полнота – 100%, а классификатор будет абсолютно бесполезен.

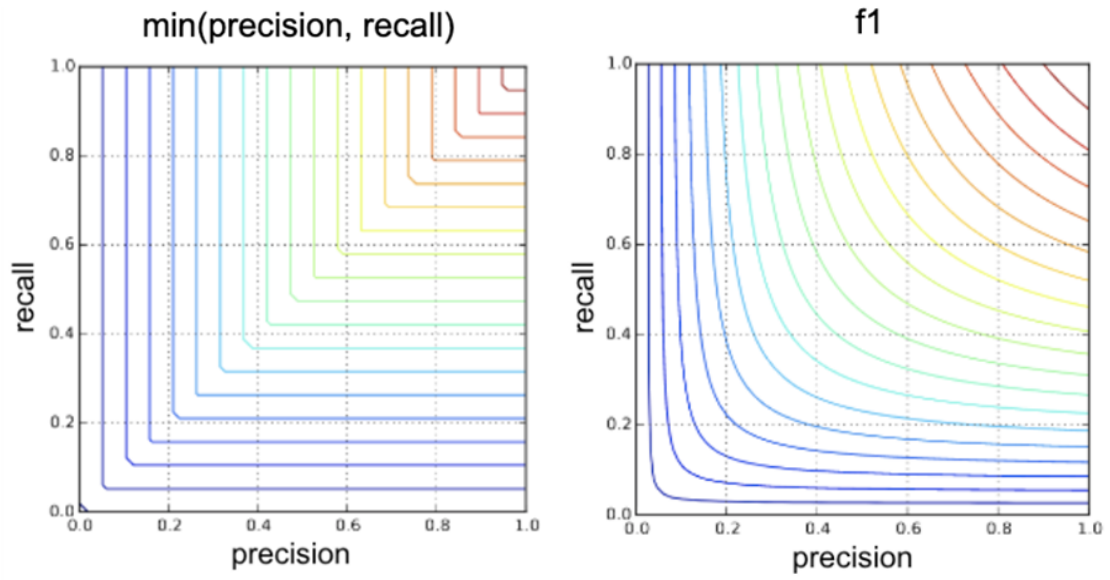
Точность и полнота – это два числа. Одновременно оптимизировать два числа трудно.

F-score

F-score – усреднение точности и полноты.

$$F_1 = \frac{2}{precision^{-1}+recall^{-1}} = 2\frac{Precision \cdot Recall}{Precision+Recall}$$

По сути это попытка аппроксимировать минимум из точности и полноты.



F-score часто используют, чтоб измерять качество классификации.

В общем случае, когда точность и полнота могут быть неравнозначны в задаче:

$$F_\beta = \left(1 + \beta^2\right)\frac{precision \cdot recall}{\beta^2 precision + recall}$$

Коэффициент β^2 позволяет сфокусироваться на точности или полноте.