Искусственный Интеллект

Лекция 4: Классификация. Логистическая регрессия. Метрики качества классификации.

Мартынюк Полина Антоновна

telegram: @PAMartynyuk

email: pa-martynyuk@yandex.ru

В предыдущих сериях…

Что нужно для модели ML?

Задача
(Task)

Определяет функцию модели (или диапазон допустимых функций)

Функция потерь
(Loss)

Определяет информационные потери модели

Оптимизатор
(Optimizer)

Определяет способ оптимизации параметров модели

Что нужно для модели ML?

- Линейная регрессия

Задача
(Task)

Регрессия – предсказание численного значения

$$y = \sum_{i=1}^{p}(x_i w_i) + b$$

Функция потерь
(Loss)

MSE – сумма квадратов разницы истинных и предсказанных значений

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

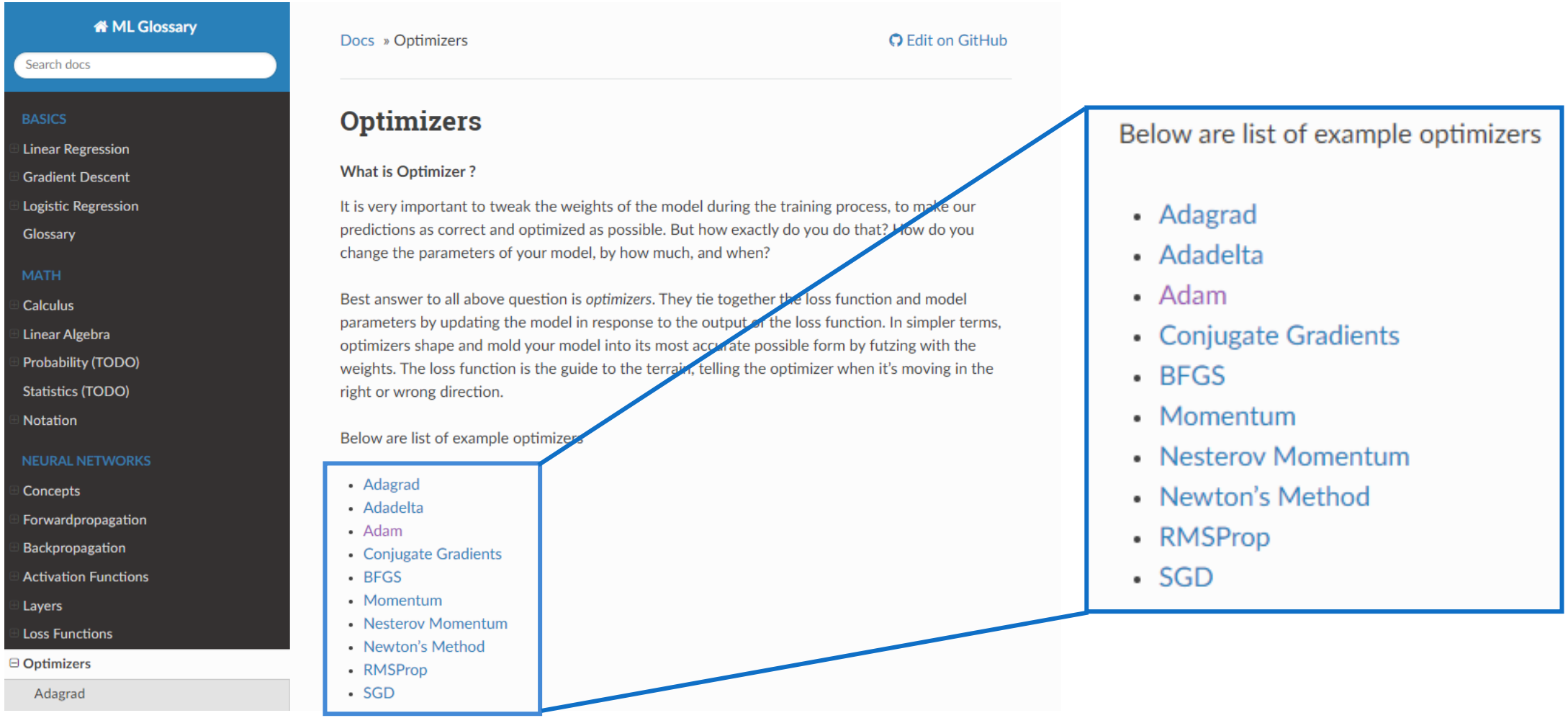
Оптимизатор
(Optimizer)

Градиентный спуск (GD)
или
Стохастический градиентный спуск (SGD)

$$w = w - \alpha\nabla L(w)$$
$$b = b - \alpha\nabla L(b)$$

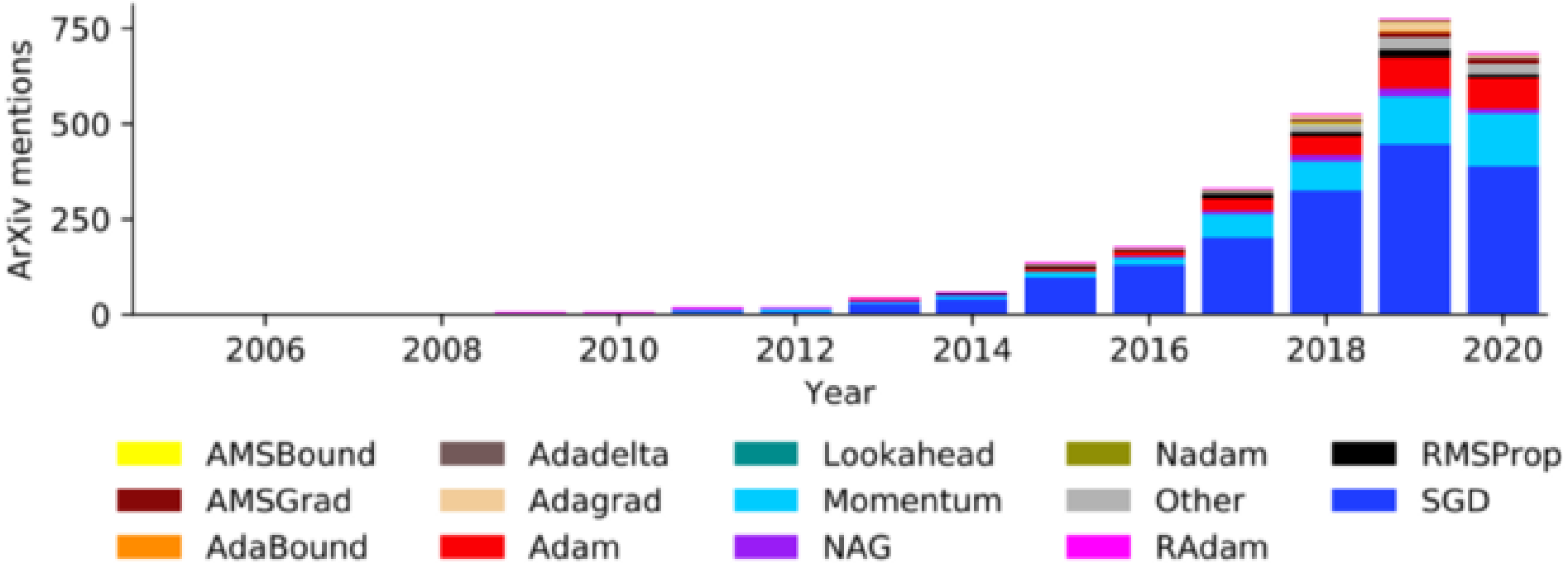
Оптимизатор

- https://ml-cheatsheet.readthedocs.io/en/latest/optimizers.html

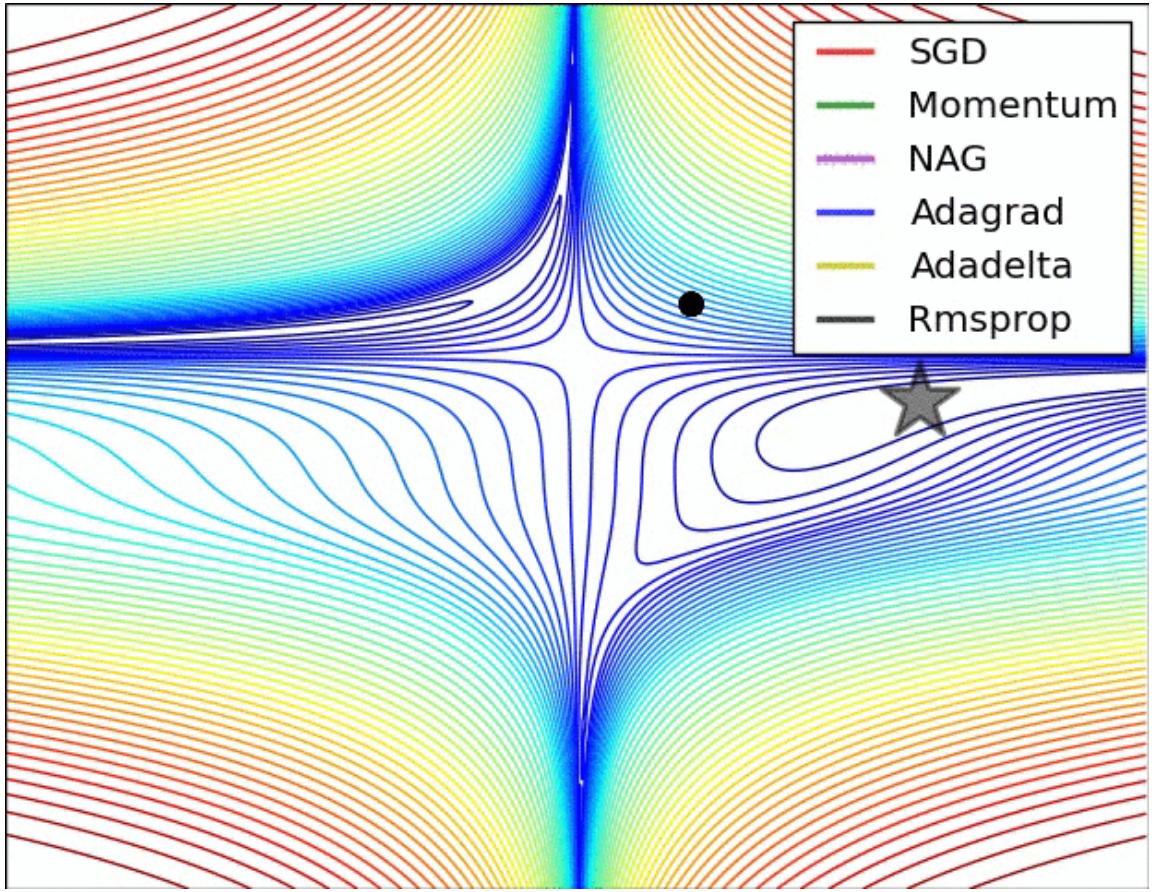


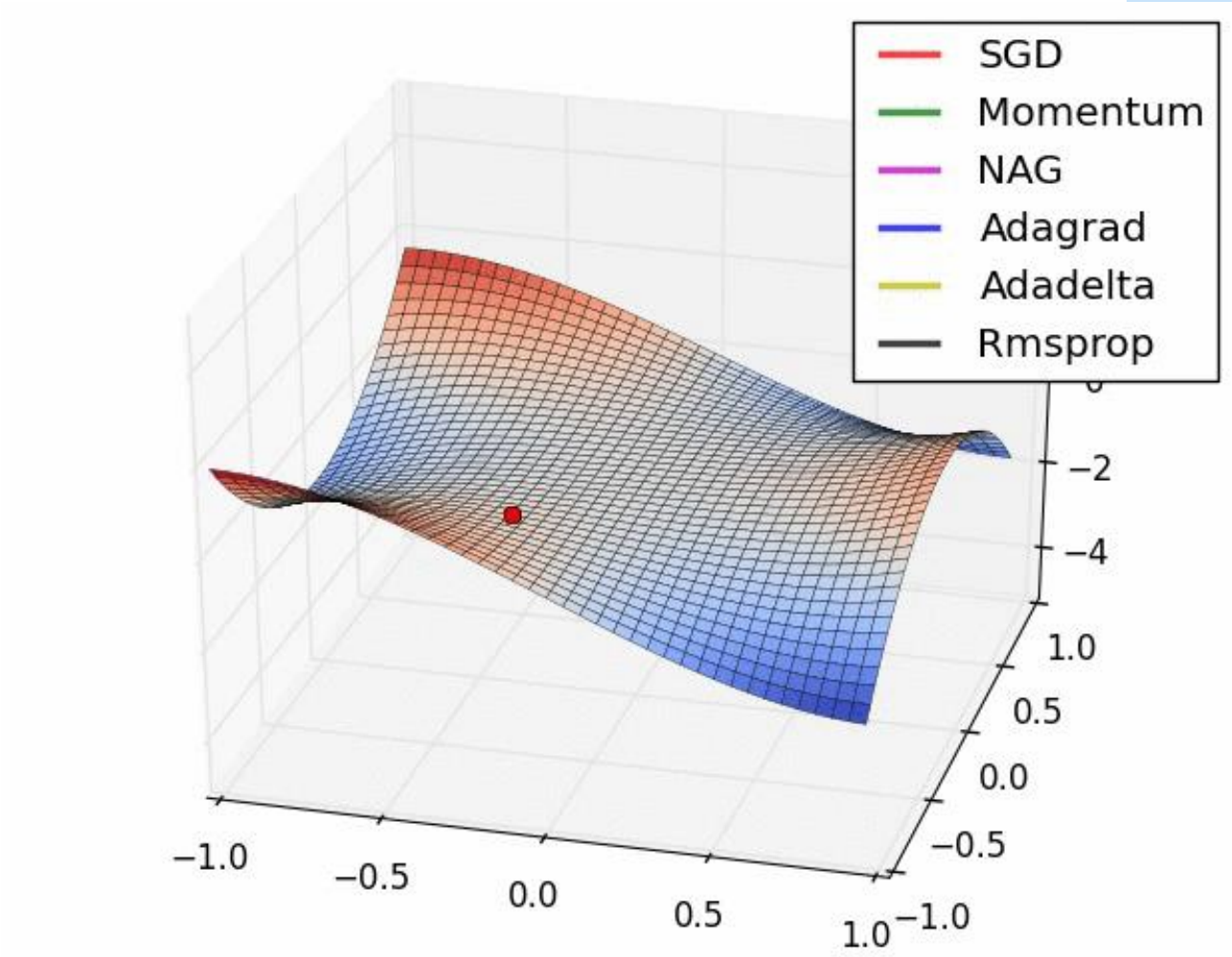
Популярные оптимизаторы

- По упоминаниям в ArXiv



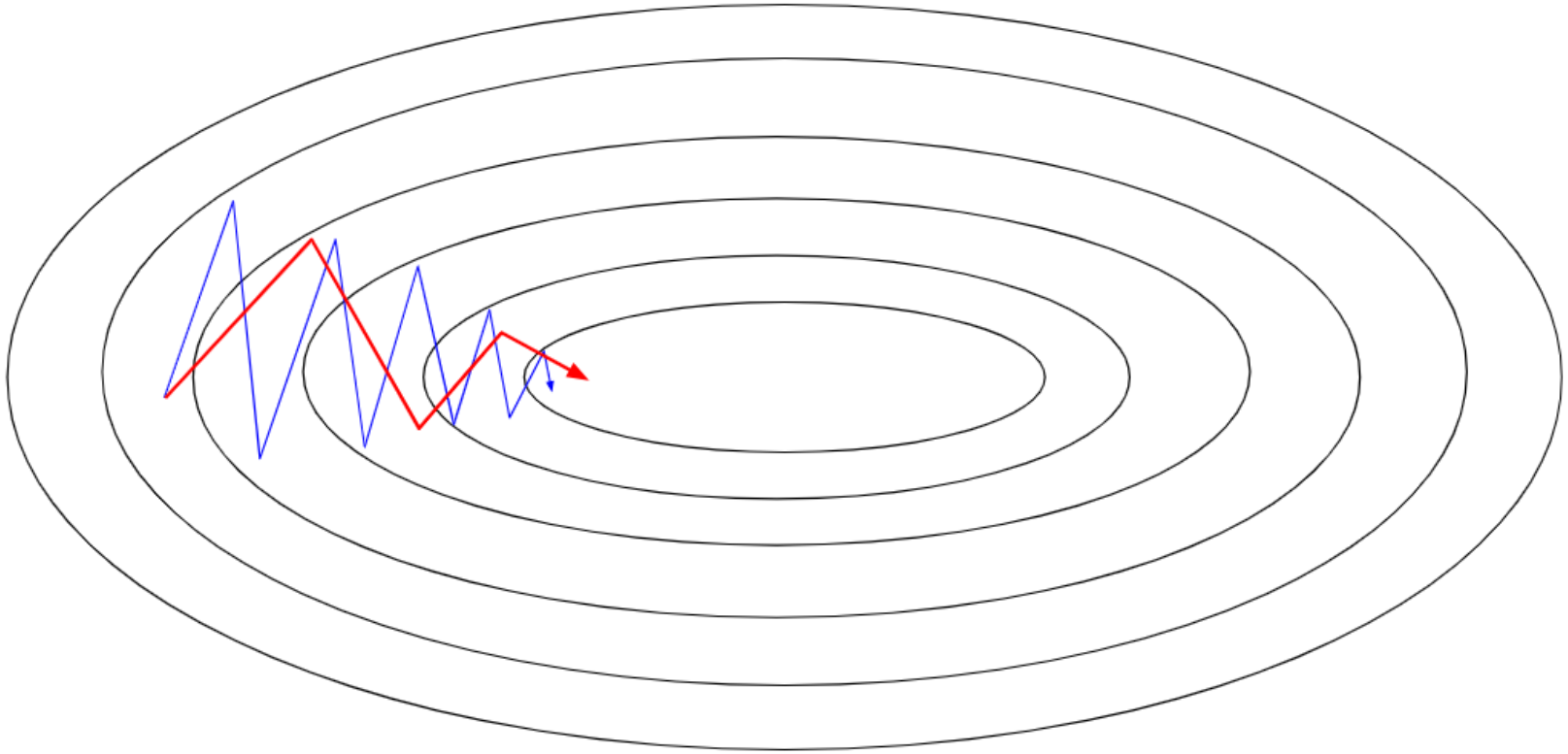
Популярные оптимизаторы





Momentum

- В SGD много колебаний. Нужно двигаться вперед, а не вверх-вниз (*сгладить осцилляцию*). Необходимо увеличить скорость обучения модели в правильном направлении, способ решения проблемы - накопление импульса.



Momentum

- **Значение β - около 0,9**
- ***v_{dW}*** больше зависит от предыдущего значения ***v_{dW}***, а не ***dW***.
- Оптимизатор импульса учитывает прошлые градиенты, чтобы сгладить обновление. Вот почему возможно минимизировать колебания.

$$v_{dW} = \beta v_{dW} + (1 - \beta) dW$$

$$v_{db} = \beta v_{db} + (1 - \beta) db$$

$$W = W - \alpha v_{dW}$$

$$b = b - \alpha v_{db}$$

А что насчет классификации?

Задача
(Task)

Классификация
–
предсказание
класса/категории

Функция потерь
(Loss)

?

Оптимизатор
(Optimizer)

?

Классификация

- предсказание категории или класса для входных данных.

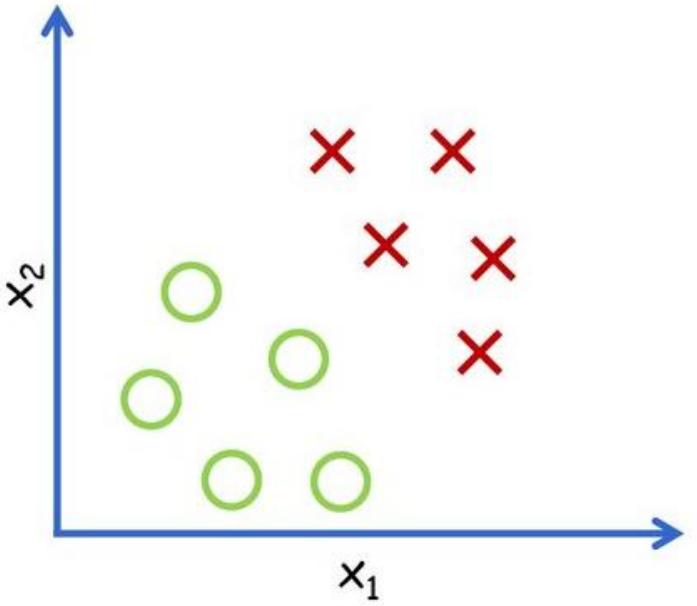
Выходная величина принимает значение одного из заранее определённых классов.

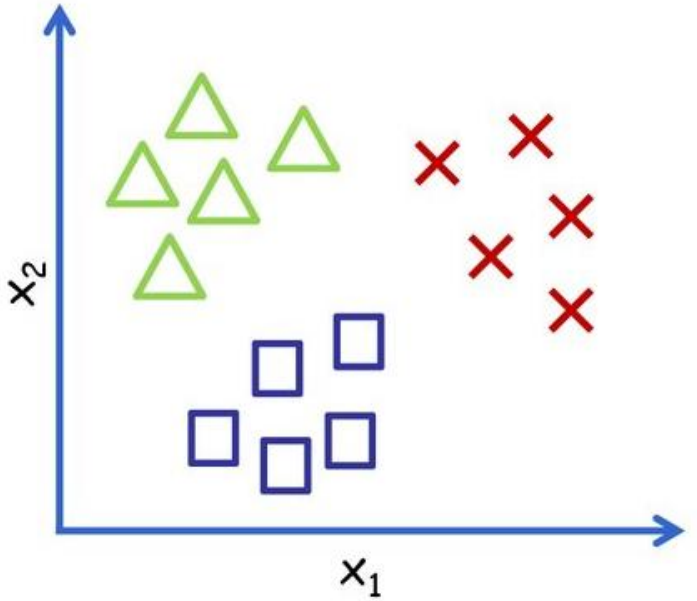
- **Бинарная классификация**: В задачах бинарной классификации метка может принимать одно из двух значений

$$Y = \{0, 1\}$$

- **Многоклассовая классификация**: В многоклассовой классификации метка может относиться к одному из нескольких классов или категорий. Например, при классификации изображений метка может указывать на тип объекта (кошка, собака, автомобиль и так далее)

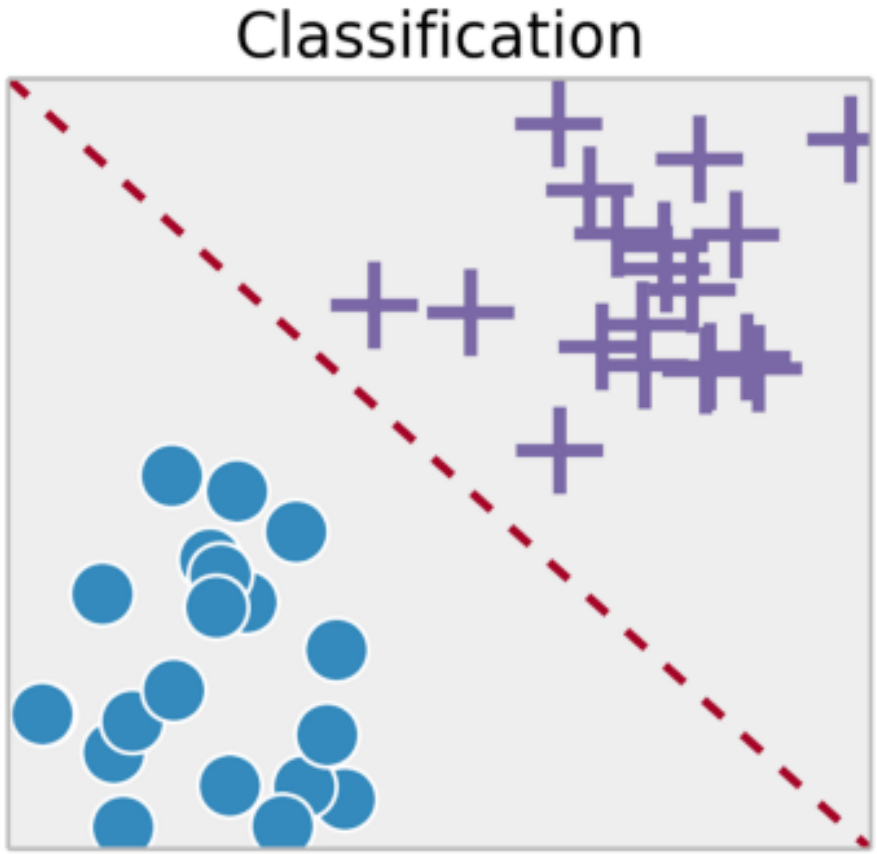
$$Y = \{0, 1, 2, \ldots, k\}$$

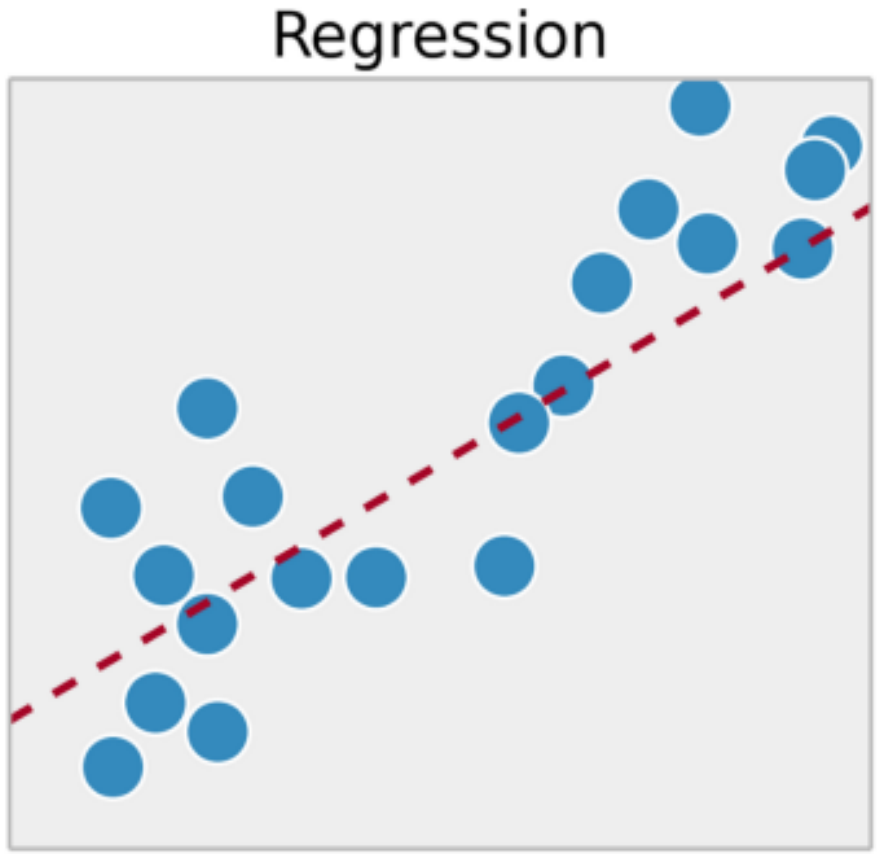




Классификация

Если в задаче регрессии мы пытаемся найти закон распределения данных, то в задаче классификации мы ищем **закон, по которому мы можем корректно разделить данные**





Классификация

Формальная постановка задачи

- Задана выборка значений признаков:

$$X_n : \{x_1, x_2, \ldots, x_n \mid x_i \in R^p\}$$

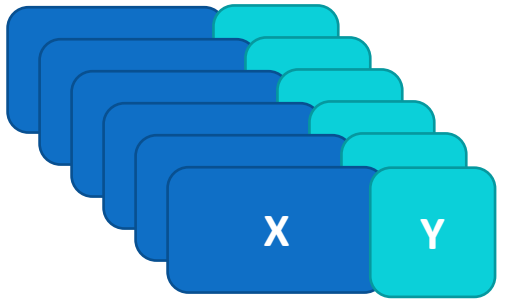
Здесь n - количество элементов в выборке входных данных, p - размерность признакового пространства.

- Задана выборка соответствующих значений целевой переменной:

$$Y_n : \{y_1, y_2, y_3, \ldots, y_n \mid y_i \in \{0, 1\}\}$$

- Получаем множество исходных данных:

$$D : \{(x, y)_i\}, \; i = 1 \ldots n$$



Классификация

- Нужно построить модель, предсказывающую по x_i значение y_i, соответствующее истинному значению y_i :

$$\hat{y}_i = f(w, x_i)$$

$$\sum_{i=1}^{n} |y_i - \hat{y}_i| \longrightarrow 0$$

- **Цель обучения – найти такие веса модели $\hat{W}$, при которых достигается максимум корректных классификаций объектов**

- **Как будет выглядеть функция потерь модели?**

Классификация

Для подбора функции потерь обратимся к статистике и введём **вероятностную модель**.

- **X – случайная величина** (вектор признаков)
- **Y – случайная величина** (целевая переменная)

Пример случайной модели (клики на рекламу):

X = (количество кликов раньше, время активности, уровень доходов)

Y = 1 если клик будет, 0 если клика не будет.

Тогда можно задать распределение вероятностей:

$$P(Y = 1 | X = (20 \text{ кликов}, 1 \text{ час}, 85\text{к рублей}))$$

Вероятность того, что пользователь с характеристиками X совершит клик по рекламному баннеру

Классификация

Пусть наша модель и будет определять вероятность принадлежности классу:

$$\hat{f}(\mathbf{x}) = P(Y = 1|\mathbf{x})$$

Назовём **правдоподобием** следующее выражение:

$$\prod_{i=1}^{n} P(Y = y_i|x_i)$$

Это **вероятность получения нашей исходной выборки** согласно предсказаниям модели.

Классификация

Хорошая модель должна предсказать **как можно больше истинных значений** для целевой переменной.

Следовательно, задача оптимизации будет сводиться к задаче **максимизации этой вероятности**:

$$\prod_{i=1}^{n} P(Y = y_i|x_i) \longrightarrow max_w$$

Логарифм – выпуклая функция. Максимум логарифма будет соответствовать искомому максимуму.

$$ln \prod_{i=1}^{n} P(Y = y_i|x_i) \longrightarrow max_w$$

Классификация

Логарифм умножения – это **сумма** логарифмов:

$$\sum_{i=1}^{n} ln(P(Y = y_i|x_i)) \longrightarrow max_w$$

Подставив минус перед суммой, преобразуем задачу максимизации в задачу **минимизации**:

$$L(w) = -\frac{1}{n}\sum_{i=1}^{n} ln(P(Y = y_i|x_i)) \longrightarrow min_w$$

Минимизация полученного выражения - то же самое, что **минимизация эмпирического риска**, где функция потерь - логарифм вероятности правильного класса.

Классификация

Оставшийся вопрос – как вычислять вероятность $P(Y = y_i | x_i)$?

Итак, мы уже умеем предсказывать численное значение при помощи линейной регрессии:

$$y = \sum_{i=0}^{p} (x_i w_i)$$

Что, если попробовать предсказываемое числовое значение преобразовывать в число, лежащее на отрезке [0, 1]?

$$\mathbb{R} \longrightarrow [0; 1]$$

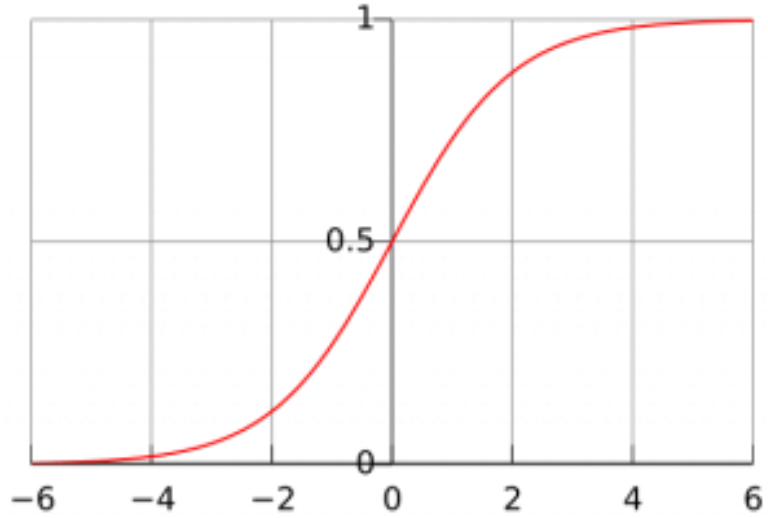
Классификация

Нам потребуется некоторая **сглаживающая функция σ**:

$$\hat{f}(\mathbf{x}) = \sigma(\sum_{i=0}^{p} w_i x_i)$$

В качестве **сглаживающей функции** используем **функцию сигмоиды:**

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



Классификация

Область значений функции сигмоиды (между 0 и 1) идеально подходит для **предсказания вероятности**.

Будем считать, что наша модель предсказывает не класс, а именно вероятность принадлежности к классу. Именно поэтому это – логистическая **регрессия**.

$$\hat{f}(\mathbf{x}) = \frac{1}{1 + e^{-(\sum_{i=0}^{p} w_i x_i)}}$$

Вероятность для двух классов можно расписать следующим образом:

$$ln(P(Y = y_i|x_i)) = y_i ln(\hat{f}(x_i)) + (1 - y_i) ln(1 - \hat{f}(x_i))$$

Классификация

Рассмотрим, как изменяются значения компонентов функции

$$y_i ln(\hat{y}_i) + (1 - y_i) ln(1 - \hat{y}_i)$$

y_i	$\hat{y}_i$	$y_i \ln(\hat{y}_i)$	$(1-y_i) \ln(1-\hat{y}_i)$	Сумма
0	0	$0 \cdot (-\infty)$	$1 \cdot 0$	0
1	1	$1 \cdot 0$	$0 \cdot (-\infty)$	0
1	0	**$1 \cdot (-\infty)$**	$0 \cdot 0$	**$-\infty$**
0	1	$0 \cdot 0$	**$1 \cdot (-\infty)$**	**$-\infty$**

Классификация

В полученную ранее формулу функции потерь можно подставить вероятность, которую предсказывает логистическая регрессия.

Функция потерь для произвольного классификатора:

$$L(w) = -\frac{1}{n}\sum_{i=1}^{n} ln(P(Y = y_i|x_i)) \longrightarrow min_w$$

Функция потерь для логистической регрессии (**LogLoss**):

$$L(w) = -\frac{1}{n}\sum_{i=1}^{n} [y_i ln(\hat{f}\,(x_i)) + (1 - y_i) ln(1 - \hat{f}\,(x_i))] \longrightarrow min_w$$

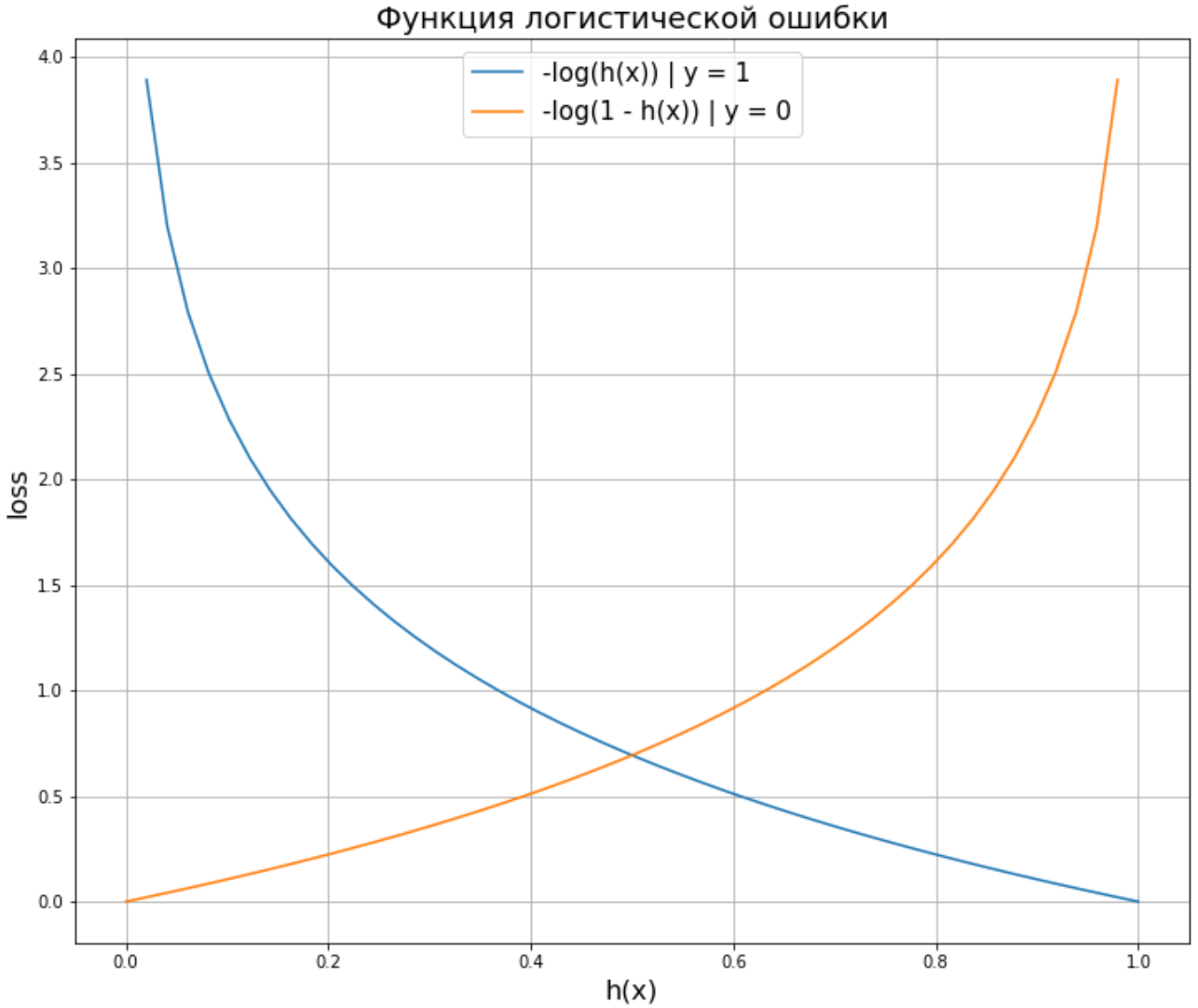
Классификация

- **Сценарий 1: y = 1**

  Предположим, что для конкретного значения в обучающем датасете истинное значение/ целевой класс записан как 1. Тогда «срабатывает» синяя ветвь графика и ошибка измеряется по ней. Соответственно, чем ближе выдаваемая моделью вероятность к единице, тем меньше ошибка.

- **Сценарий 2: y = 0**

  Предположим, что целевая переменная записана как 0. Тогда срабатывает оранжевая ветвь. Ошибка модели будет минимальна при значениях, близких к нулю.



Кросс-энтропия

Многоклассовая классификация

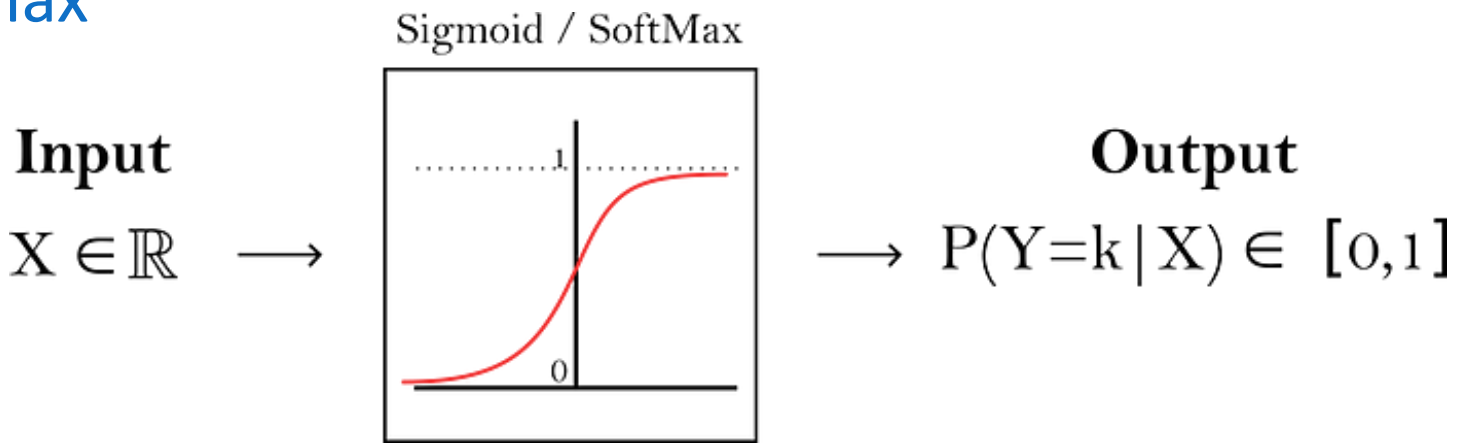
- Пусть у нас есть m классов. Введем две новые функции:

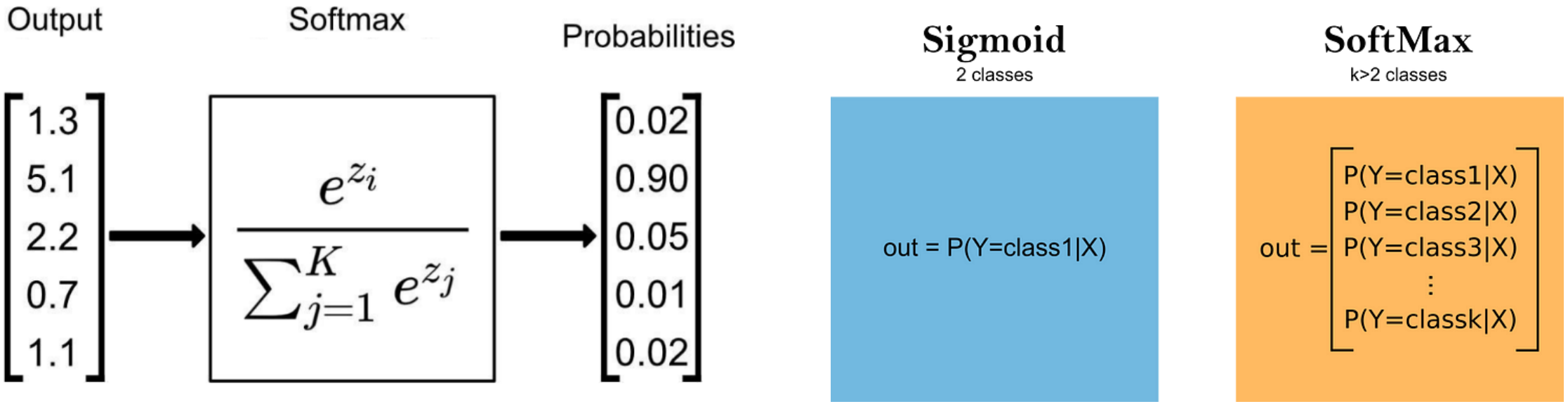
$$logits(\mathbf{x}) = \begin{pmatrix} w^{(1)} \cdot \mathbf{x} \\ w^{(2)} \cdot \mathbf{x} \\ \cdots \\ w^{(m)} \cdot \mathbf{x} \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^{p} w_i^{(1)} \cdot x_i \\ \sum_{i=0}^{p} w_i^{(2)} \cdot x_i \\ \cdots \\ \sum_{i=0}^{p} w_i^{(m)} \cdot x_i \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdots \\ \alpha_m \end{pmatrix}$$

$$Softmax(\alpha) = \left(\frac{e^{\alpha_1}}{\sum_{i=1}^{m} e^{\alpha_i}}, \frac{e^{\alpha_2}}{\sum_{i=1}^{m} e^{\alpha_i}}, \ldots, \frac{e^{\alpha_m}}{\sum_{i=1}^{m} e^{\alpha_i}} \right)$$

Многоклассовая классификация

- Пример работы Softmax





Многоклассовая классификация

- В случае многоклассовой классификации:

$$\hat{f}(\mathbf{x}) = Softmax(logits(\mathbf{x}))$$

- Выпишем предсказанную вероятность для *k*-го класса. Ее можно подставить в функцию потерь для произвольного классификатора:

$$P(Y = k|\ \mathbf{x}) = \hat{f}_k(\mathbf{x}) = \frac{e^{\sum_{i=0}^{p} w_i^{(k)} \cdot x_i}}{\sum_{k=1}^{m} e^{\sum_{i=0}^{p} w_i^{(k)} \cdot x_i}}$$

Градиентный спуск для логистической регрессии

- Функция потерь:

$$L(w) = -\frac{1}{n}\sum_{i=1}^{n}[y_i ln(\hat{f}\,(x_i)) + (1-y_i)ln(1-\hat{f}\,(x_i))]$$

- Возьмем производную:

$$\frac{\partial L}{\partial w_1} = -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i}{\hat{f}\,(x_i)}\frac{\partial \hat{f}\,(x_i)}{\partial w_1} - \frac{1-y_i}{1-\hat{f}\,(x_i)}\frac{\partial \hat{f}\,(x_i)}{\partial w_1}$$

$$\frac{\partial \hat{f}\,(x_i)}{\partial w_1} = \hat{f}\,(x_i)(1-\hat{f}\,(x_i))x_{i1}$$

- Соединим:

$$\frac{\partial L}{\partial w_1} = -\frac{1}{n}\sum_{i=1}^{n} x_{i1}[y_i(1-\hat{f}\,(x_i)) - (1-y_i)\hat{f}\,(x_i)]$$

Классификация: метрики качества

- Поговорим о метриках

Верное предсказание

Котик

Ошибочное предсказание

Котик

Классификация: метрики качества

Наиболее интуитивно понятная метрика – **доля верных ответов**

- **Точность/ Доля верных ответов**
- **Accuracy**

$$\text{доля правильных ответов} = \frac{\text{количество правильных ответов}}{\text{количество всех ответов}}$$

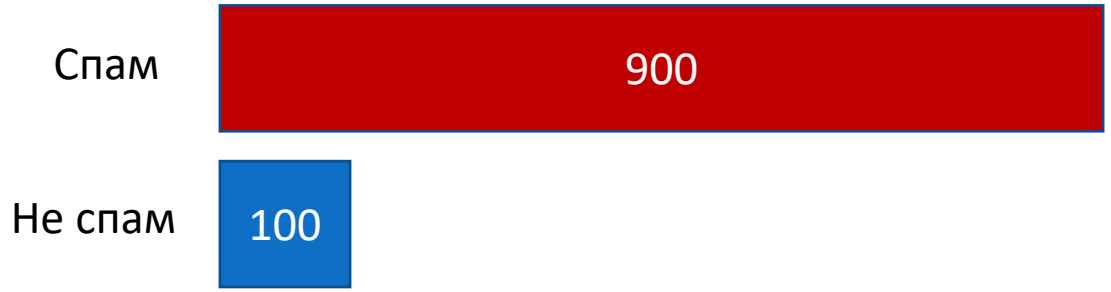
- **Какие могут быть проблемы?**

Классификация: метрики качества

Пусть наша задача – определять спам-письма

Классификация: спам / не спам

Пусть в почтовом ящике 1000 писем:



**Тривиальная модель –
всегда выдаёт класс “Спам”**

$$Accuracy = \frac{900}{1000} = 0,9$$

Плохо работает на несбалансированных классах

Классификация: метрики качества

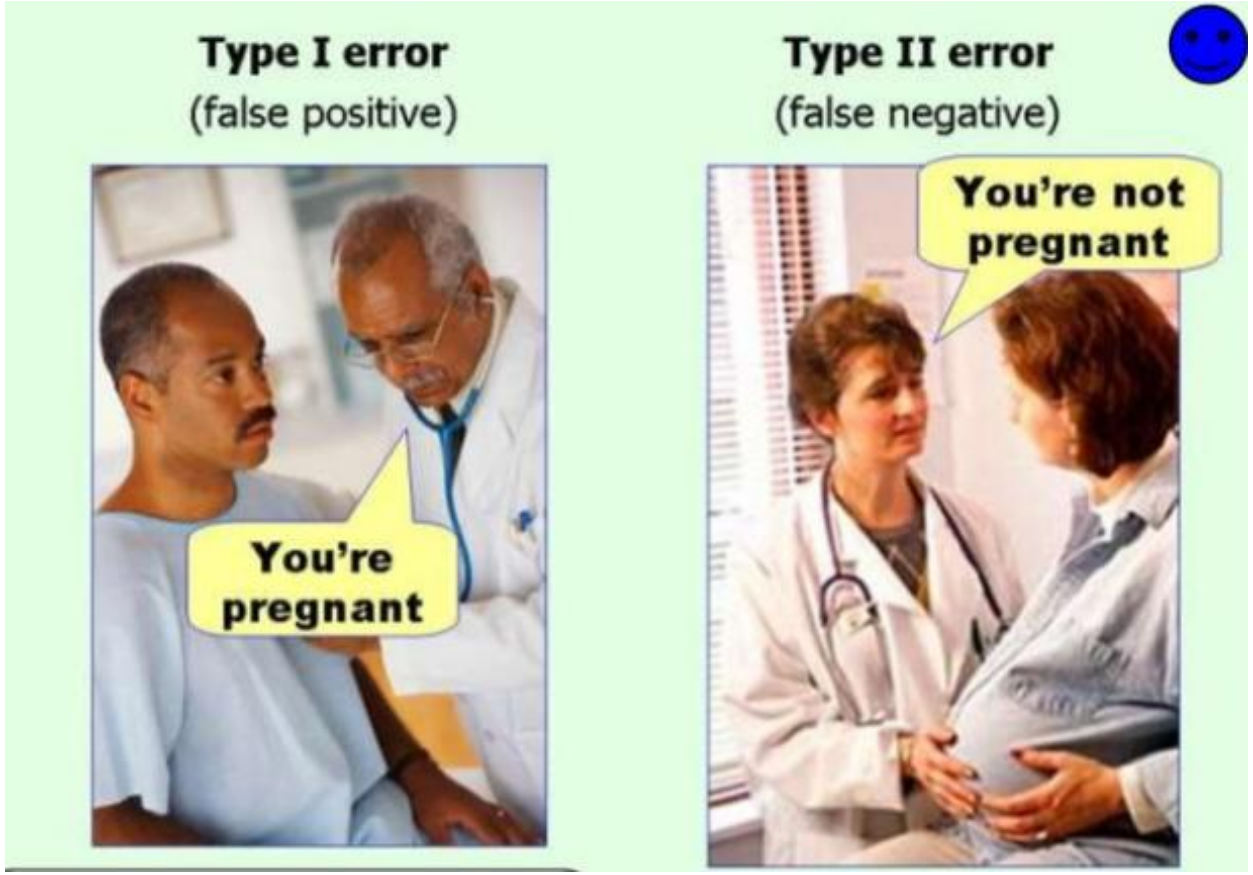
- Матрица ошибок

		алгоритм	
		0	1
истина	0	True Negative	False Positive
	1	False Negative	True Positive

Классификация: метрики качества

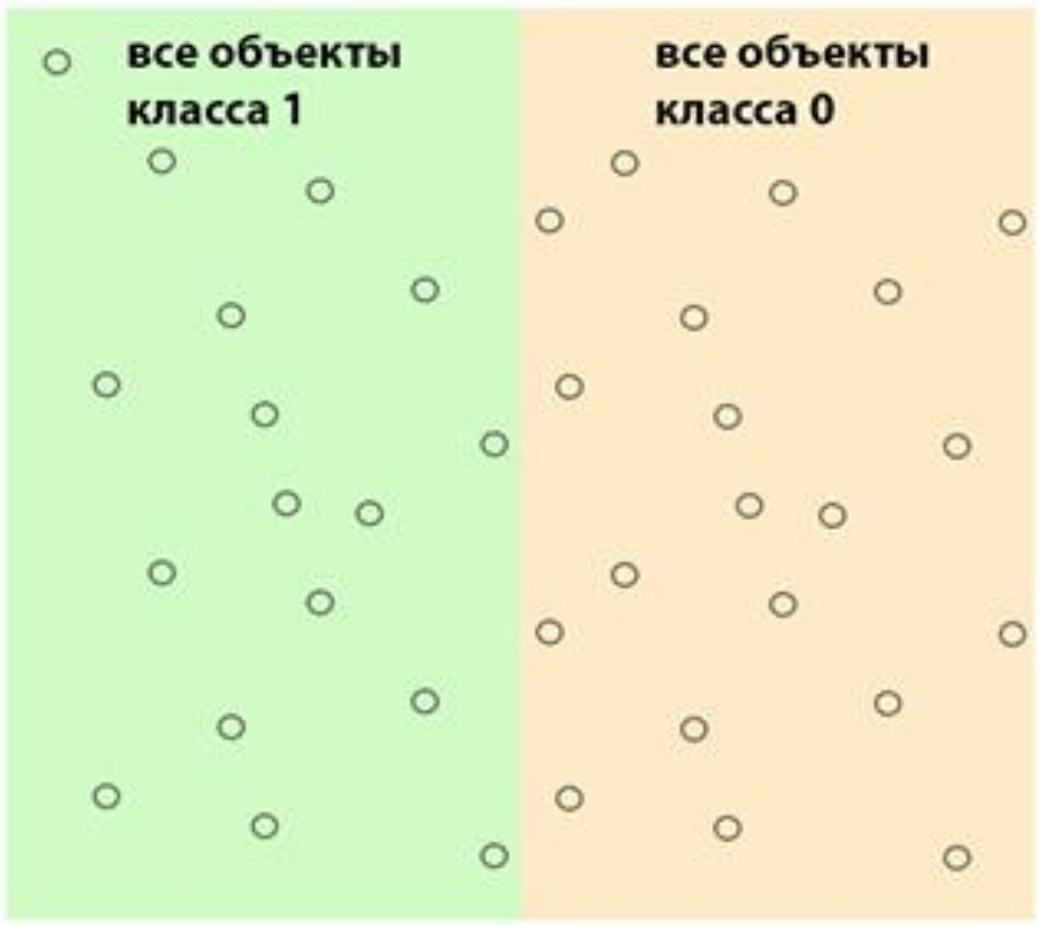
- Матрица ошибок

		алгоритм	
		0	1
истина	0	True Negative	False Positive
	1	False Negative	True Positive



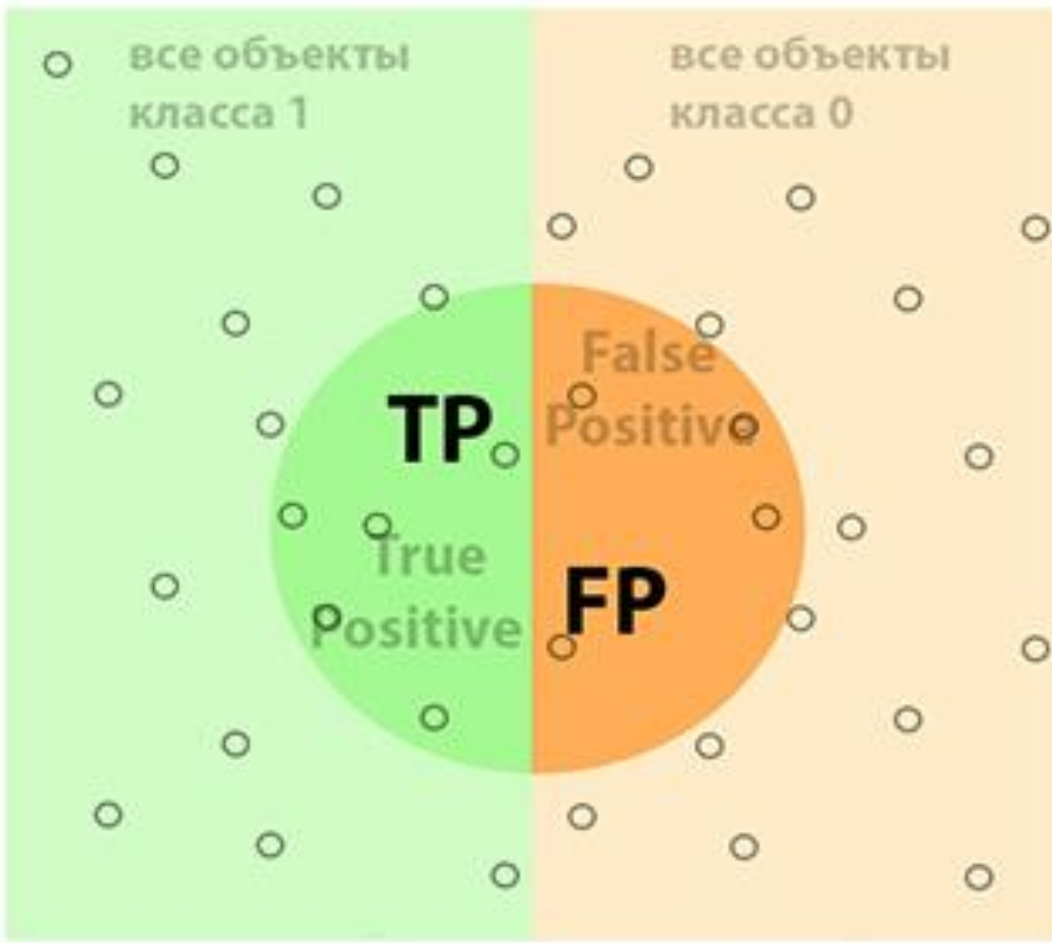
Классификация: метрики качества

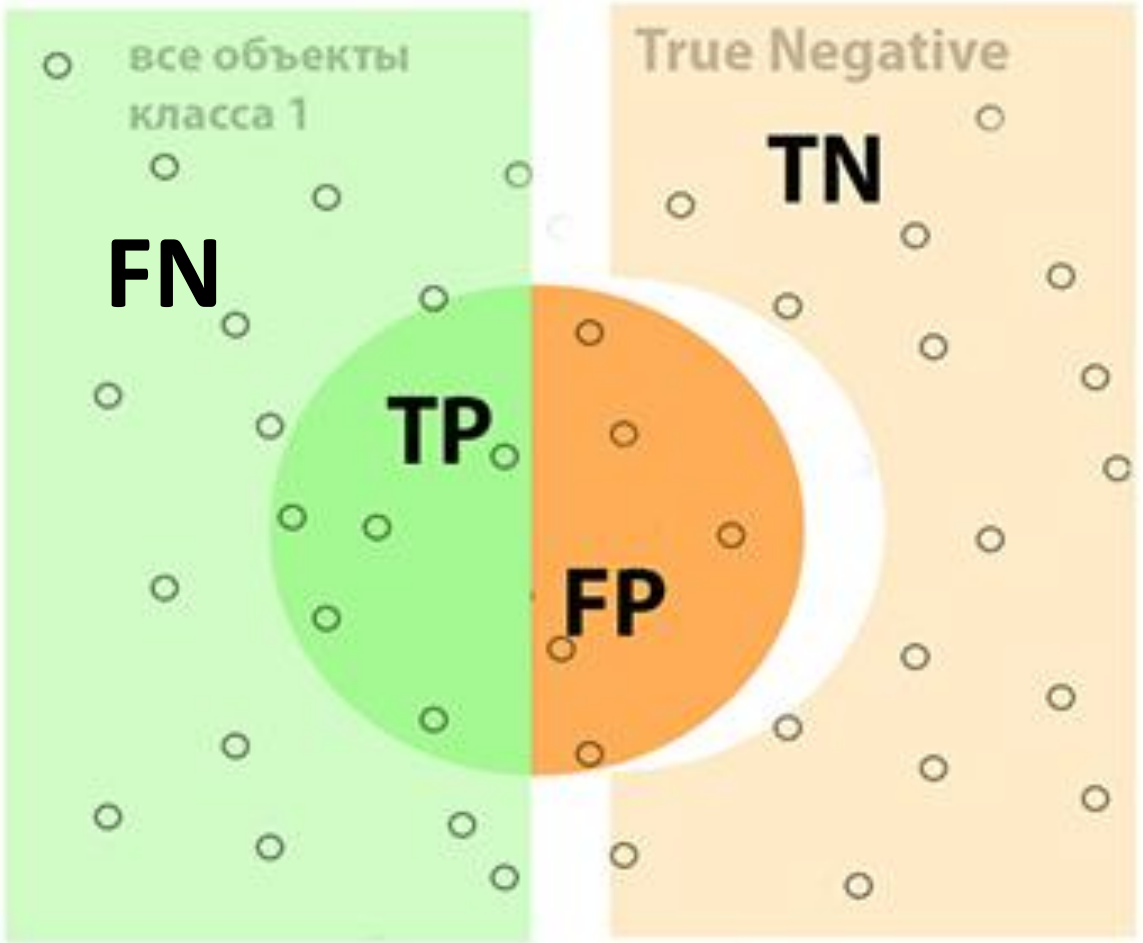
- Матрица ошибок



Классификация: метрики качества

- Матрица ошибок





Классификация: метрики качества

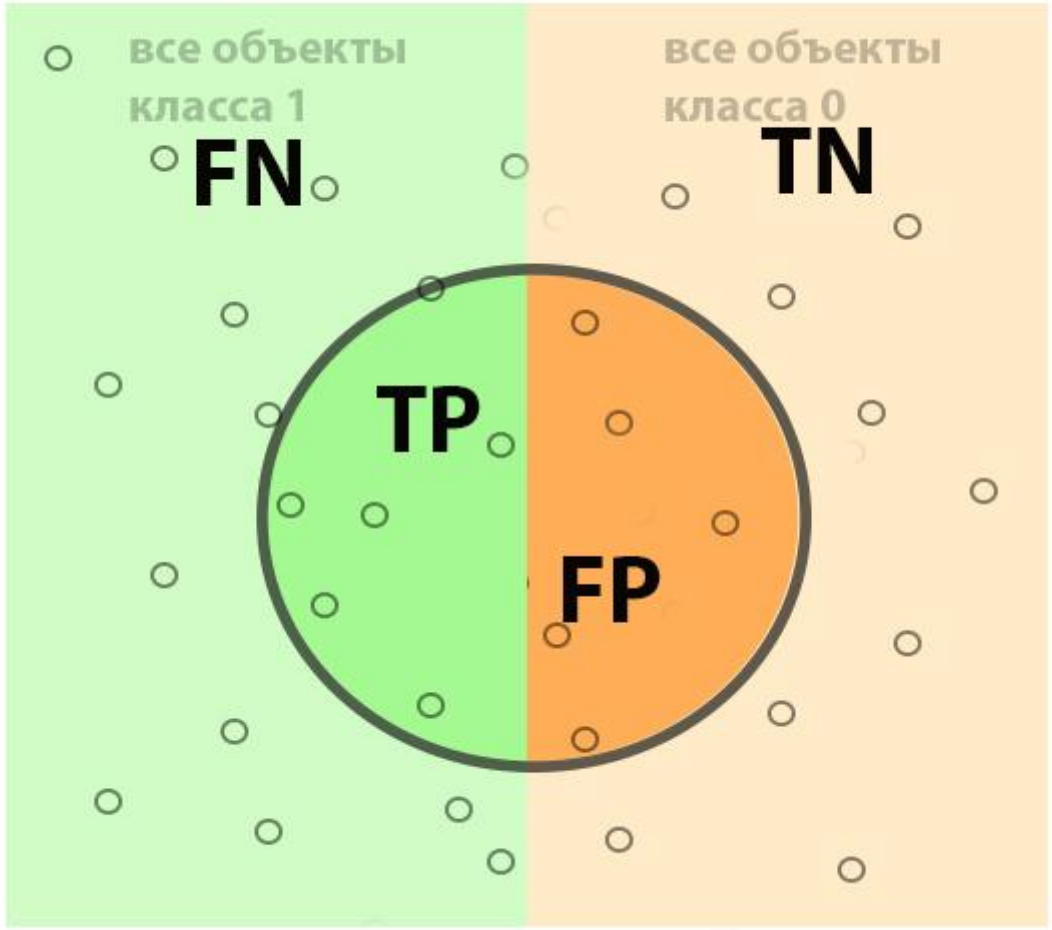
- **Точность**
- **Precision**

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

- Улучшая эту метрику, мы уменьшаем число “ложных срабатываний” FP

- **Пример:** Спецагент и сканер отпечатка пальца

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



Классификация: метрики качества

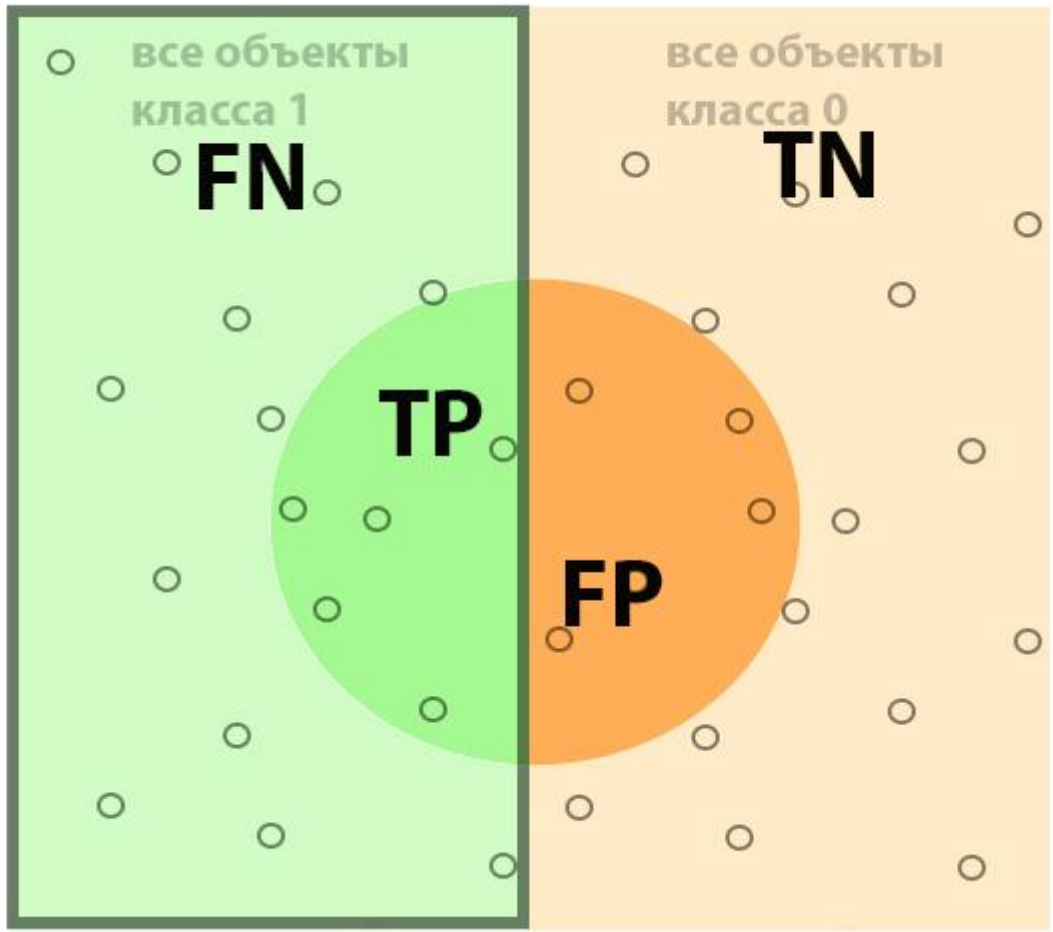
- **Полнота/Охват**
- **Recall**

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

- Улучшая эту метрику, мы уменьшаем число “недо-срабатываний” FN

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

- **Пример:** Больные опасной болезнью



Классификация: метрики качества

- **F1-мера (F1 score)**

  Гармоническое среднее между точностью и полнотой. Оценивает баланс между точностью и полнотой.

  Полезна в случаях, когда важен баланс между точностью и полнотой. Высокое значение F-меры указывает на хорошее сбалансированное предсказание.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Классификация: метрики качества

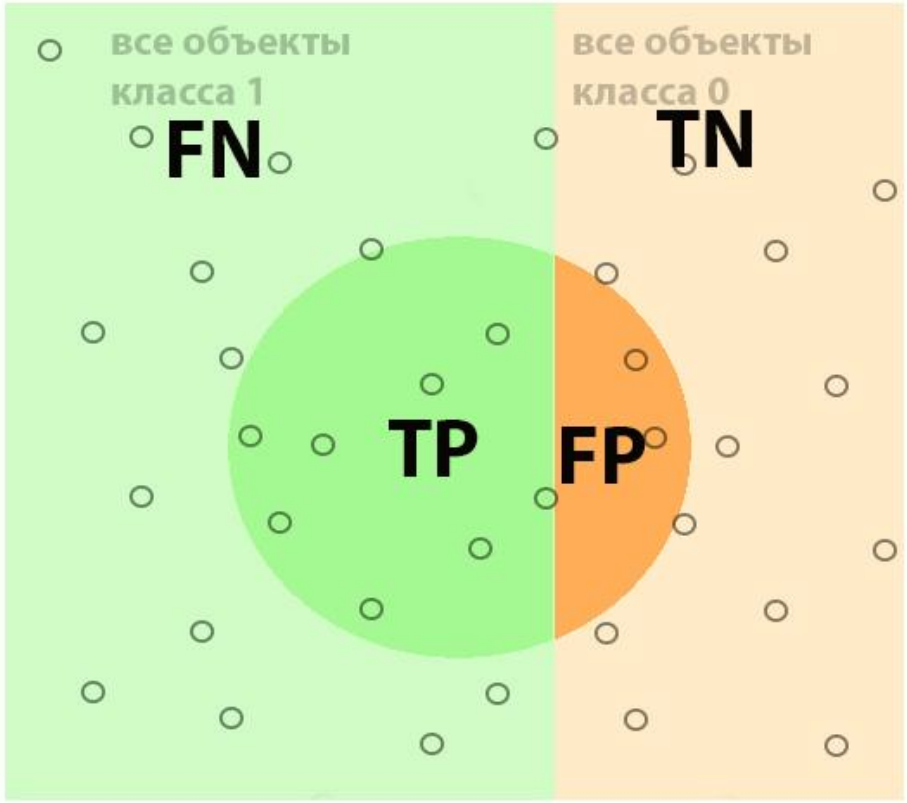
- *F1-мера* является довольно стабильной метрикой при равном балансе классов. На практике большинство задач имеют перекос в балансе классов. Посмотрим, что будет происходит с описанными понятиями в этом случае.

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

Изменится пропорционально

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

Возрастёт

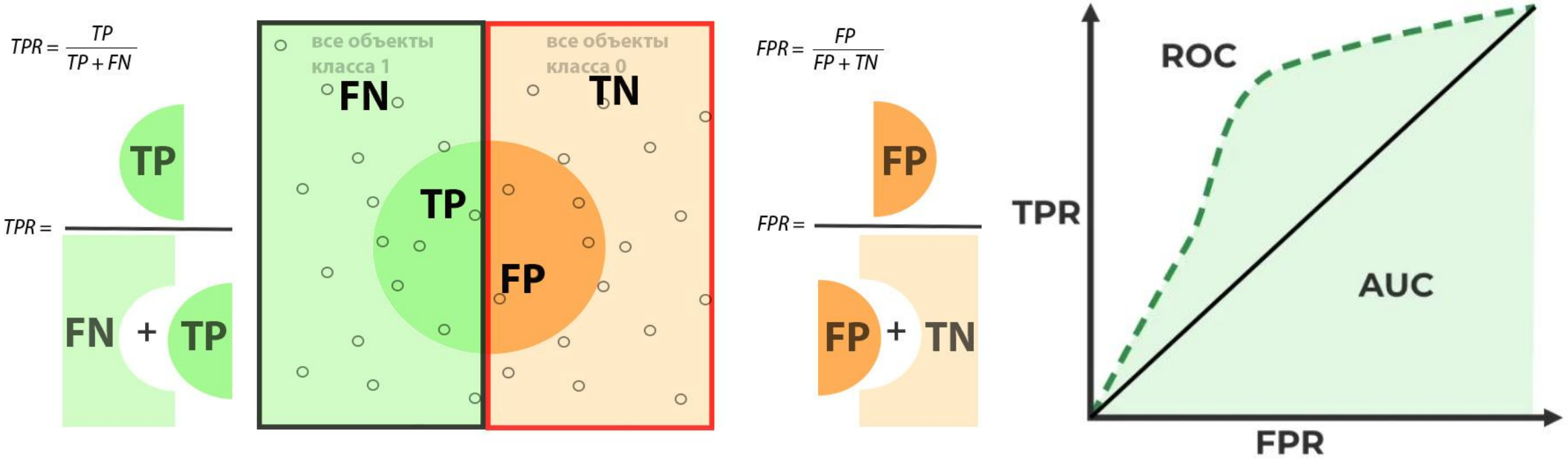


Для нивелирования перекоса предполагается использование некоторого β коэффициента — это и будет среднегармоническая *F-мера*.

$$F_\beta = (1 + \beta^2) \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\beta^2 \cdot \mathrm{Precision} + \mathrm{Recall}}$$

Классификация: метрики качества

- ROC-AUC - receiver operating characteristic, иногда говорят «кривая ошибок», и area under the curve.



https://alexanderdyakonov.wordpress.com/2017/07/28/auc-roc-%D0%BF%D0%BB%D0%BE%D1%89%D0%B0%D0%B4%D1%8C-%D0%BF%D0%BE%D0%B4-%D0%BA%D1%80%D0%B8%D0%B2%D0%BE%D0%B9-%D0%BE%D1%88%D0%B8%D0%B1%D0%BE%D0%BA/

Классификация: метрики качества

- Пусть алгоритм выдал оценки, как показано в табл. 1.
- Упорядочим строки табл. 1 по убыванию ответов алгоритма – получим табл. 2.
- В идеале её столбец «класс» тоже станет упорядочен (сначала идут 1, потом 0); в самом худшем случае – порядок будет обратный (сначала 0, потом 1); в случае «слепого угадывания» будет случайное распределение 0 и 1.

id	оценка	класс
1	0.5	0
2	0.1	0
3	0.2	0
4	0.6	1
5	0.2	1
6	0.3	1
7	0.0	0

Табл. 1

id	оценка	класс
4	0.6	1
1	0.5	0
6	0.3	1
3	0.2	0
5	0.2	1
2	0.1	0
7	0.0	0

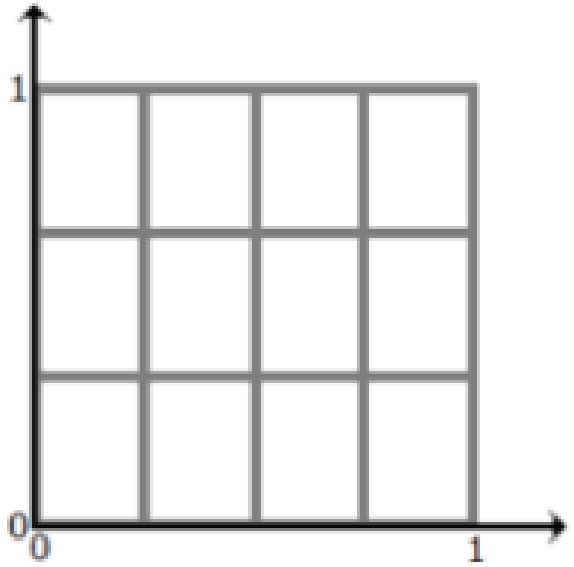
Табл. 2

Классификация: метрики качества

- 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***=3), ***n*** – число нулей (***n***=4). В результате квадрат разбивается сеткой на ***m***×***n*** блоков.

id	оценка	класс
4	0.6	1
1	0.5	0
6	0.3	1
3	0.2	0
5	0.2	1
2	0.1	0
7	0.0	0

Табл. 2

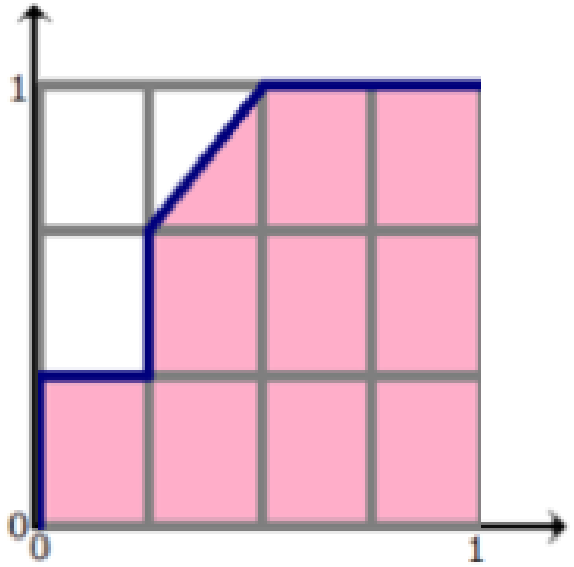


Классификация: метрики качества

- Теперь будем просматривать строки табл. 2 сверху вниз и прорисовывать на сетке линии, переходя их одного узла в другой. Стартуем из точки (0, 0). Если значение метки класса в 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.

id	оценка	класс
4	0.6	1
1	0.5	0
6	0.3	1
3	0.2	0
5	0.2	1
2	0.1	0
7	0.0	0

Табл. 2

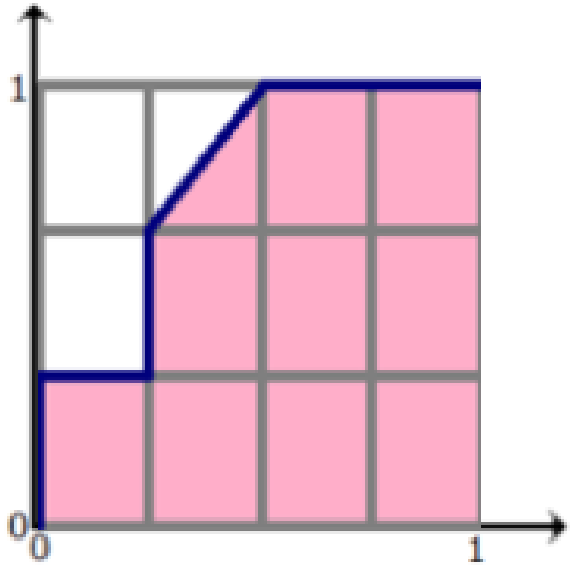


Классификация: метрики качества

- **AUC ROC** – площадь под ROC-кривой – часто используют для оценивания качества упорядочивания алгоритмом объектов двух классов. Ясно, что это значение лежит на отрезке [0, 1].
- В нашем примере AUC ROC = 9.5 / 12 ~ 0.79.

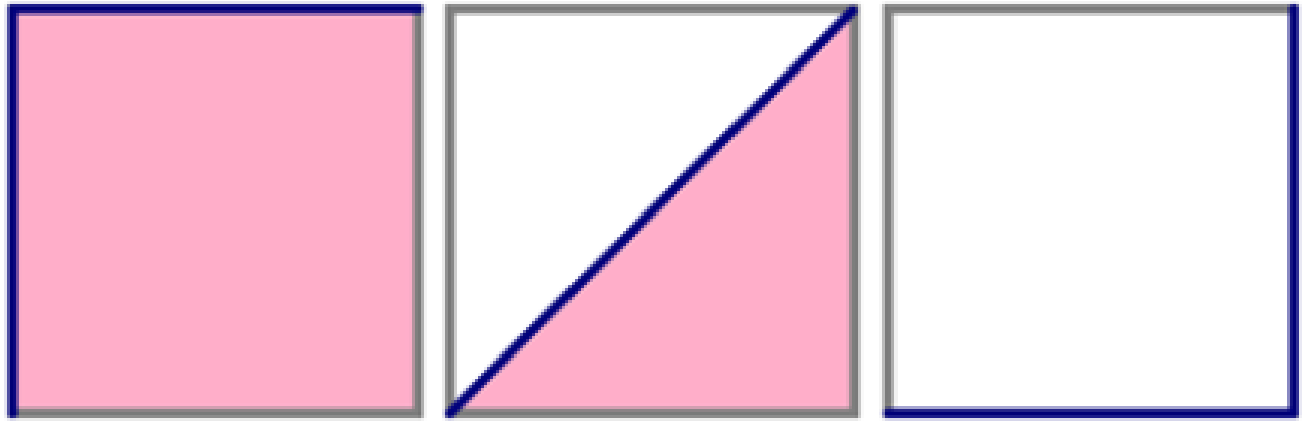
id	оценка	класс
4	0.6	1
1	0.5	0
6	0.3	1
3	0.2	0
5	0.2	1
2	0.1	0
7	0.0	0

Табл. 2



Классификация: метрики качества

- Идеальному соответствует ROC-кривая, проходящая через точку (0, 1), площадь под ней равна 1. Наихудшему – ROC-кривая, проходящая через точку (1, 0), площадь под ней – 0. Случайному – что-то похожее на диагональ квадрата, площадь примерно равна 0.5.



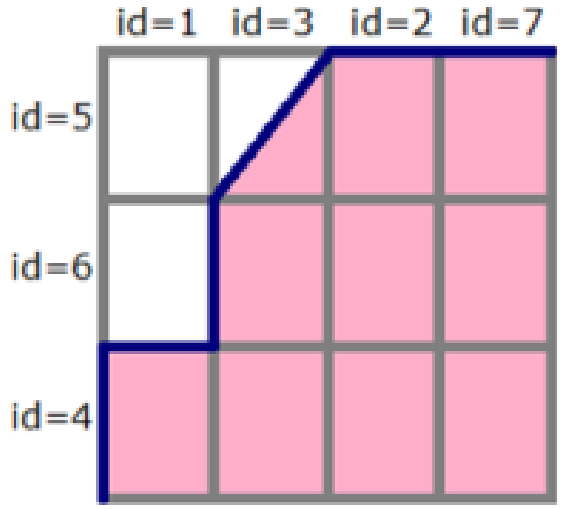
ROC-кривые для наилучшего (AUC=1), случайного (AUC=0.5) и наихудшего (AUC=0) алгоритма.

Классификация: метрики качества

- Сетка разбила квадрат на ***m*×*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), незакрашенный блок – паре, на которой ошибся.

id	оценка	класс
4	0.6	1
1	0.5	0
6	0.3	1
3	0.2	0
5	0.2	1
2	0.1	0
7	0.0	0

Табл. 2



Таким образом, **AUC ROC равен доле пар объектов вида (объект класса 1, объект класса 0), которые алгоритм верно упорядочил**, т.е. первый объект идёт в упорядоченном списке раньше.

Классификация: метрики качества

Многоклассовая классификация

- Если классов становится больше двух, расчёт метрик усложняется. Если задача классификации на *K* классов ставится как *K* задач об отделении класса *i* от остальных (*i*=1,…,*K*), то для каждой из них можно посчитать свою матрицу ошибок. Затем есть два варианта получения итогового значения метрики из *K* матриц ошибок:
- Усредняем элементы матрицы ошибок (TP, FP, TN, FN) между бинарными классификаторами. Затем по одной усреднённой матрице ошибок считаем Precision, Recall, F-меру. Это называют **микроусреднением**.

$$TP = \frac{1}{K}\sum_{i=1}^{K} TP_i$$

- Считаем Precision, Recall для каждого классификатора отдельно, а потом усредняем. Это называют **макроусреднением**.

Спасибо за внимание!