



КЛАССИФИКАЦИЯ И АЛГОРИТМЫ

2022



БИНАРНАЯ КЛАССИФИКАЦИЯ

как задача

Напоминание про регрессию

- У нас есть обучающая выборка из n объектов, у каждого из которых есть признаки (x_1, x_2, \dots, x_m) , а y – ответы для наших объектов.
- Модель линейной регрессии:

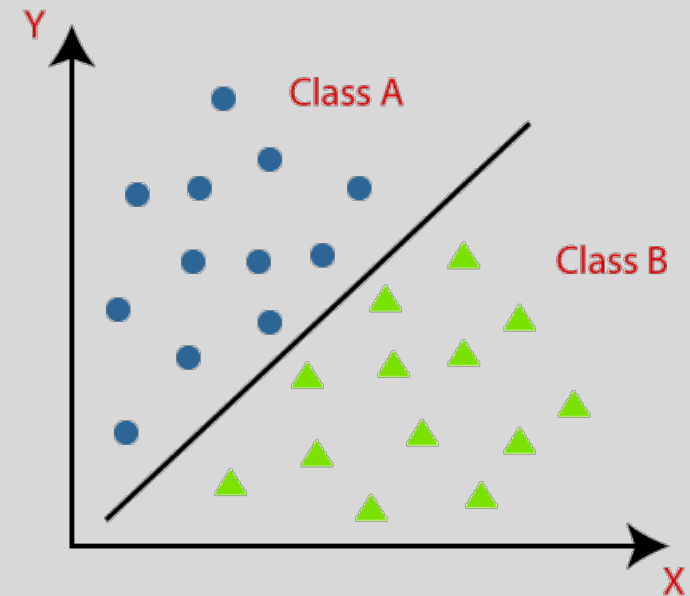
$$a(x, w) = \sum_{i=1}^n w_i x_i$$

- Метод обучения – метод наименьших квадратов (минимизируем разность между предсказанием и правильным ответом):

$$Q(w) = \sum_{i=1}^n (a(x, w) - y_i)^2 \rightarrow \min_w$$

Бинарная классификация

- Строим линию таким образом, чтобы она не ложилась на наши объекты, а разделяла их
- у нас может быть -1 или 1 (отрицательный или положительный класс)
- Как нам модифицировать нашу функцию для зависимости y от X , если мы предсказываем не случайное число, а -1 или 1?

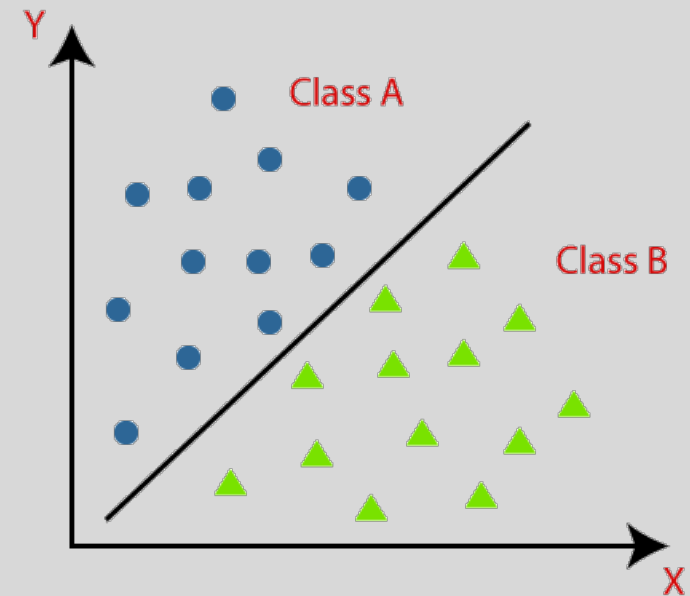


Бинарная классификация

- Строим линию таким образом, чтобы она не ложилась на наши объекты, а разделяла их
- у у нас может быть -1 или 1 (отрицательный или положительный класс)
- Модель линейного классификатора:

$$a(w, x) = \text{sign}\left(\sum_{i=1}^n w_i x_i\right)$$

- Если $\sum_{i=1}^n w_i x_i > 0$, то $\text{sign}(\sum_{i=1}^n w_i x_i) = +1$, то есть, объект отнесен к положительному классу
- Если $\sum_{i=1}^n w_i x_i < 0$, то $\text{sign}(\sum_{i=1}^n w_i x_i) = -1$, то есть, объект отнесен к отрицательному классу
- А если $\sum_{i=1}^n w_i x_i = 0$?

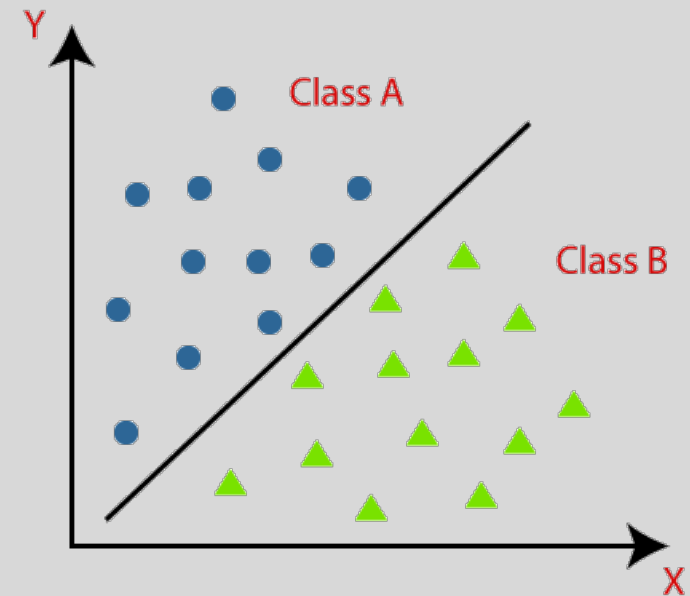


Бинарная классификация

- Строим линию таким образом, чтобы она не ложилась на наши объекты, а разделяла их
- у нас может быть -1 или 1 (отрицательный или положительный класс)
- Модель линейного классификатора:

$$a(w, x) = \text{sign}\left(\sum_{i=1}^n w_i x_i\right)$$

- Если $\sum_{i=1}^n w_i x_i > 0$, то $\text{sign}(\sum_{i=1}^n w_i x_i) = +1$, то есть, объект отнесен к положительному классу
- Если $\sum_{i=1}^n w_i x_i < 0$, то $\text{sign}(\sum_{i=1}^n w_i x_i) = -1$, то есть, объект отнесен к отрицательному классу
- Если $\sum_{i=1}^n w_i x_i = 0$, то перед нами **уравнение разделяющей границы** между классами. Это **уравнение плоскости** (или прямой в двумерном случае), поэтому классификатор является линейным.





КАК ОБУЧИТЬ КЛАССИФИКАТОР?

Обучение классификатора

- Минимизируем функцию ошибок, но какая она для этой задачи?
- Функция ошибок для бинарного классификатора вроде бы простая:

$$Q(w) = \frac{1}{n} \sum_{i=1}^n [a(x_i) \neq y_i] \rightarrow \min$$

- где $[a(x_i) \neq y_i] = 1$, если предсказание на объекте неверное, иначе 0.
- Обозначим $M_i = y_i \cdot (w, x)_i$ – **отступ** на i -м объекте.
- Если у нас класс положительный, а мы предсказали отрицательный, или наоборот, то $M_i < 0$ ($-1 \cdot +1$).
- Следовательно, мы можем переписать функцию ошибок:

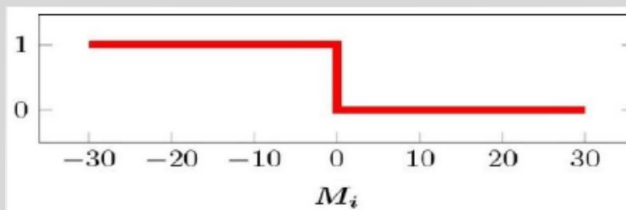
$$Q(w) = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

Обучение классификатора

- Наша функция ошибок еще называется пороговой функцией потерь:

$$Q(w) = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

- Пороговая функция потерь **разрывна**, и этот факт сильно затрудняет процесс минимизации.



- Для решения этой проблемы обычно используют другие функции потерь – непрерывные или гладкие, являющиеся аппроксимациями пороговой функции. После замены функции потерь минимизируется не сама пороговая функция, а ее верхняя оценка:

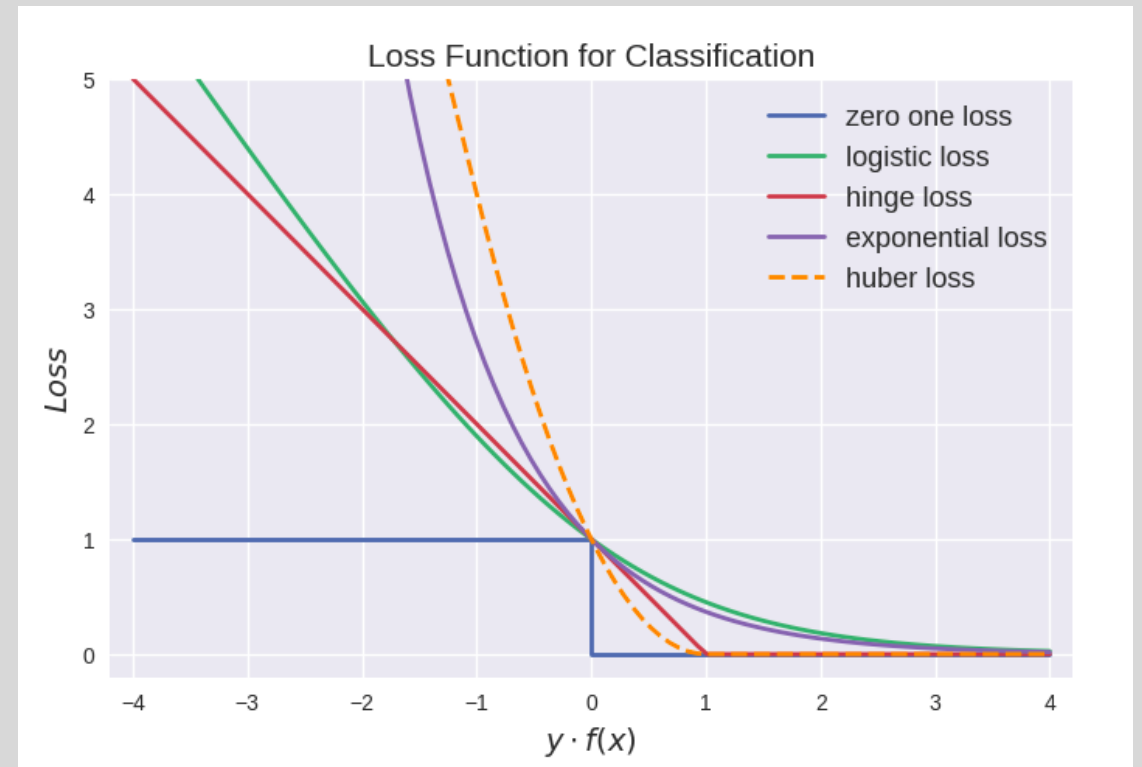
$$Q(w) \leq \tilde{Q}(w) = \sum_{i=1}^n L(M(x_i))$$

- Аппроксимации способны улучшать обобщающую способность классификатора + с ними работает градиентный спуск.

Обучение классификатора

- Задача минимизации некоторой функции потерь называется **минимизация эмпирического риска** (сама функция потерь – эмпирический риск).
- Какую аппроксимацию выберем, такой и алгоритм!
- Наши варианты:
 - $L(M) = \log(1 + e^{-M})$ – логистическая функция потерь
 - $V(M) = (1 - M)_+ = \max(0, 1 - M)$ – кусочно-линейная функция потерь (метод опорных векторов)
 - $H(M) = (-M)_+ = \max(0, -M)$ – кусочно-линейная функция потерь (перцептрон)
 - $E(M) = e^{-M}$ – экспоненциальная функция потерь
 - $S(M) = \frac{2}{1 + e^{(-M)}}$ – сигмоидная функция потерь
- Теперь, когда определились с функцией потерь, можем использовать градиентный спуск:

$$w^{(k)} = w^{(k-1)} - \eta \cdot Q(w^{(k-1)})$$





МЕТРИКИ КАЧЕСТВА

Метрики качества



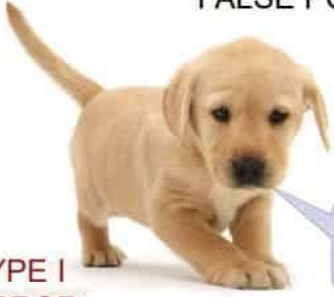

- Accuracy – доля правильных ответов:

$$accuracy(a, X) = \frac{1}{n} \sum_{i=1}^n [a(x_i) = y_i]$$

- Недостаток: при сильно несбалансированной выборке не отражает качество работы алгоритма
- Пример: одобряем кредиты. У нас есть тестовая выборка, где всего 200 человек, из них 100 действительно вернули кредит, а 100 не вернули. То есть, выборка сбалансированная.
 - Модель 1: одобрила кредит 100 людям, из них 80 вернули, 20 – нет.
 - Модель 2: одобрила кредит 50 людям, из них 48 вернули, 2 – нет.
- Пример 2: в нашей тестовой выборке 200 человек, но из них 180 кредиты не вернули и только 20 – вернули.
 - Модель взяла и всем отказала. Выгодно ли ей так делать? (А нафиг ей учиться тогда вообще?)

Матрица ошибок

Confusion matrix

		PREDICTIVE VALUES	
		POSITIVE (CAT)	NEGATIVE (DOG)
ACTUAL VALUES	POSITIVE (CAT)	<p>TRUE POSITIVE</p>  <p>3</p> <p>YOU ARE A CAT</p>	<p>FALSE NEGATIVE</p>  <p>1</p> <p>YOU ARE A DOG</p> <p>TYPE II ERROR</p>
	NEGATIVE (DOG)	<p>FALSE POSITIVE</p>  <p>2</p> <p>YOU ARE A CAT</p> <p>TYPE I ERROR</p>	<p>TRUE NEGATIVE</p>  <p>4</p> <p>YOU ARE NOT A CAT</p>

Метрики качества

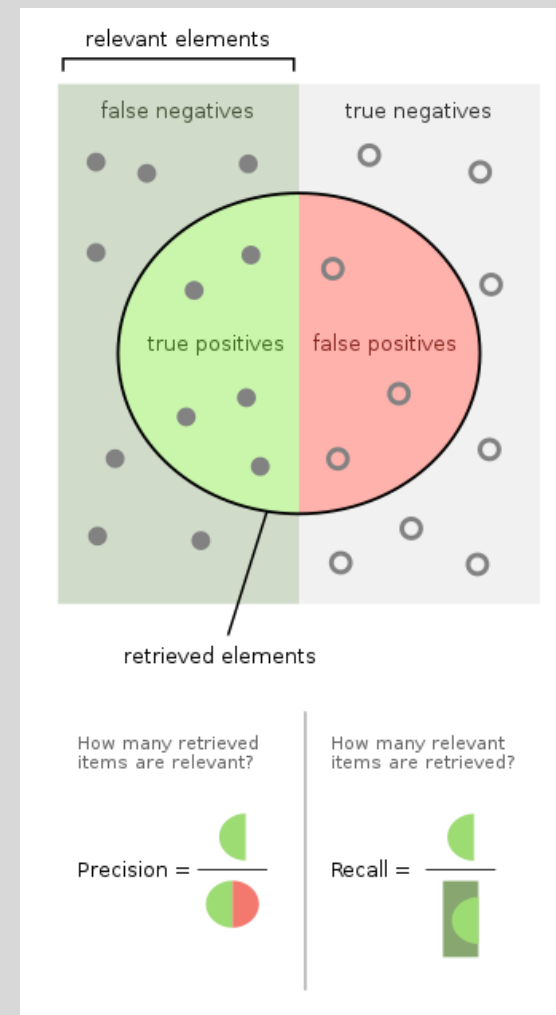
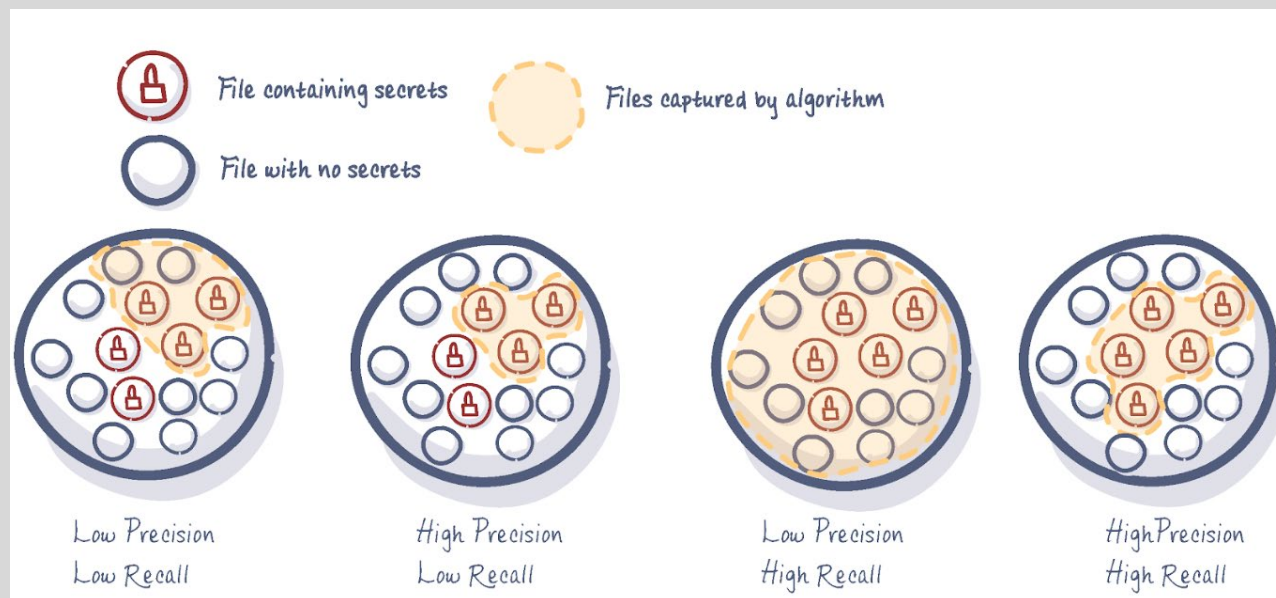
Посчитайте метрики:

- Precision: $precision(a, x) = \frac{TP}{TP+FP}$
- Recall: $recall(a, x) = \frac{TP}{TP+FN}$
- F-score: $\frac{2 \times precision \times recall}{precision + recall}$

	$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть		$y = 1$ Могут вернуть	$y = -1$ Не могут вернуть
$a(x) = 1$ Получили кредит	80	20	$a(x) = 1$ Получили кредит	48	2
$a(x) = -1$ Не получили кредит	20	80	$a(x) = -1$ Не получили кредит	52	98

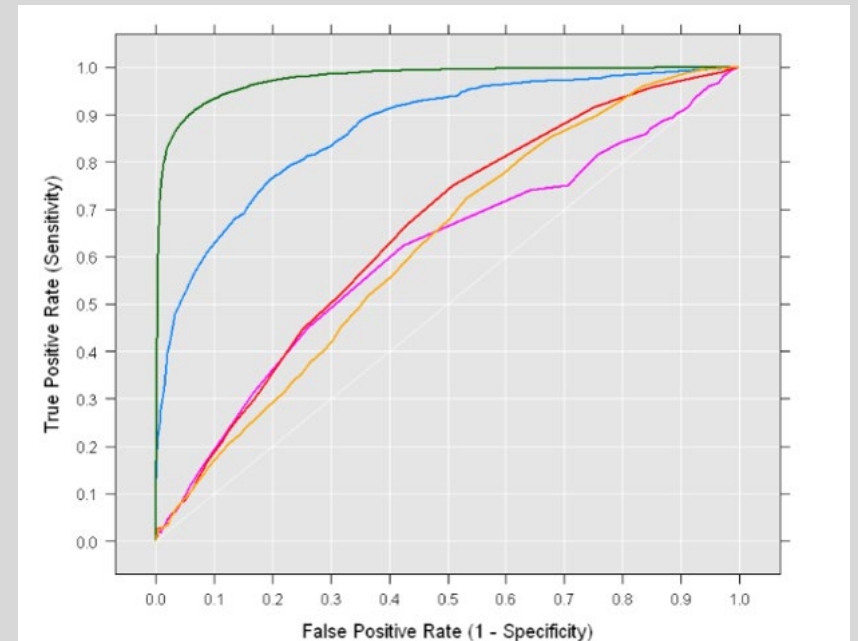
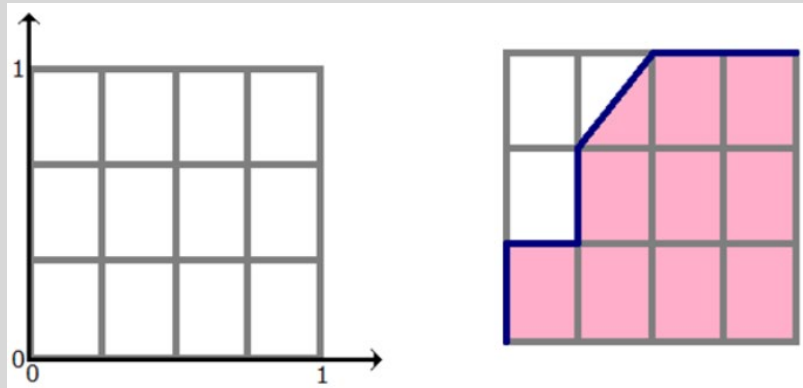
Метрики качества

- Что показывают точность и полнота?



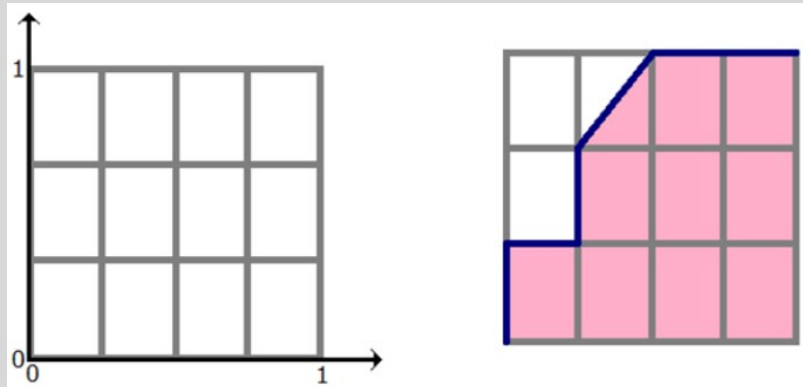
Интегральная метрика: ROC-AUC

- Берем квадрат и делим его на m рядов и n колонок: m – это число наших правильных y_true , n – число неправильных.
- Чертим линию: если наша модель правильно предсказала, идем вверх, если нет – направо. Таким образом придем к правому верхнему углу.
- Видимо, чем чаще мы идем вверх, тем лучше работает модель.
- А значит, чем выгнутее кривая в левый верхний угол, тем лучше!



Интегральная метрика: ROC-AUC

- Берем квадрат и делим его на m рядов и n колонок: m – это число наших правильных y_{true} , n – число неправильных.
- Чертим линию: если наша модель правильно предсказала, идем вверх, если нет – направо. Таким образом придем к правому верхнему углу.



Посчитаем на примерах.

Здесь класс – это истинный ответ, а оценка – это вероятность, с которой модель относит наш объект к положительному классу (в правой табличке уже сразу ответ).

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

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

id	> 0.25	класс
4	1	1
1	1	0
6	1	1
3	0	0
5	0	1
2	0	0
7	0	0

Интегральная метрика: ROC-AUC

- Посчитаем в формулах. Нам понадобятся:
- **False Positive Rate** (доля неверно принятых объектов отрицательного класса):

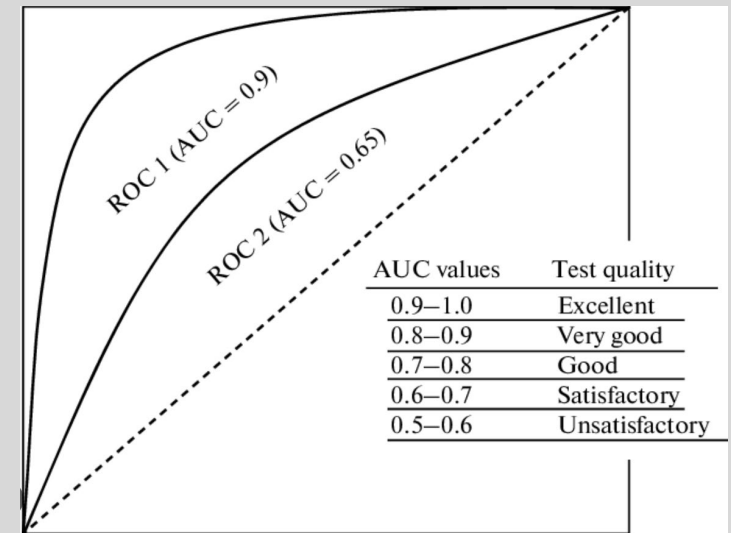
$$FPR = \frac{FP}{FP + TN} = \frac{\sum_i [y_i = -1][a(x_i) = +1]}{\sum_i [y_i = -1]}$$

- **True Positive Rate** (доля верно принятых объектов положительного класса):

$$FPR = \frac{TP}{TP + FN} = \frac{\sum_i [y_i = +1][a(x_i) = +1]}{\sum_i [y_i = +1]}$$

- На нашем графике FPR и TPR – координаты, и мы вычисляем их для каждого объекта выборки.
- **AUC (area under curve)** – площадь под ROC-кривой, ROC (receiver operating characteristic) - кривая ошибок
 - AUC = 0.5 – случайная классификация;
 - AUC = 1 – идеальная классификация.

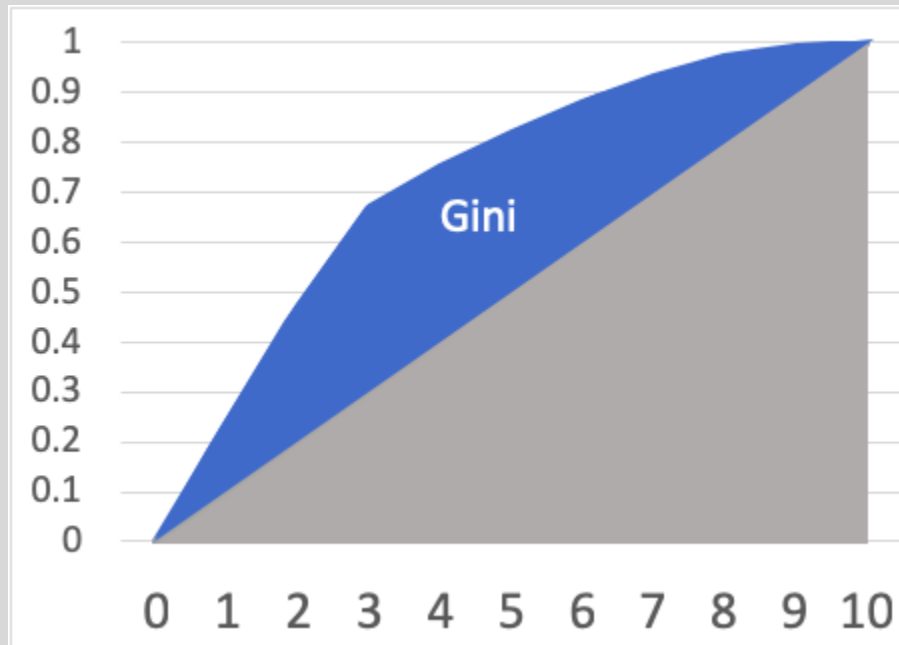
[хорошая статья на тему](#)



Индекс Джини

- Индекс Джини – это удвоенная площадь между кривой ROC и диагональной прямой.

$$Gini = 2 \cdot AUC - 1$$



AUC	Gini	Description
0.9 - 1.0	0.8 - 1.0	Excellent
0.8 - 0.9	0.6 - 0.8	Good
0.7 - 0.8	0.4 - 0.6	Fair
0.6 - 0.5	0.2 - 0.4	Poor



ВИДЫ КЛАССИФИКАТОРОВ

Линейные

Линейные vs нелинейные

Линейные

- Логистическая регрессия
- Метод опорных векторов (SVM – support vector machine)

Нелинейные

- Наивный Байес
- Метод К ближайших соседей (KNN)
- Деревья решений

Логистическая регрессия

- Хотим предсказывать не классы, а вероятности классов.

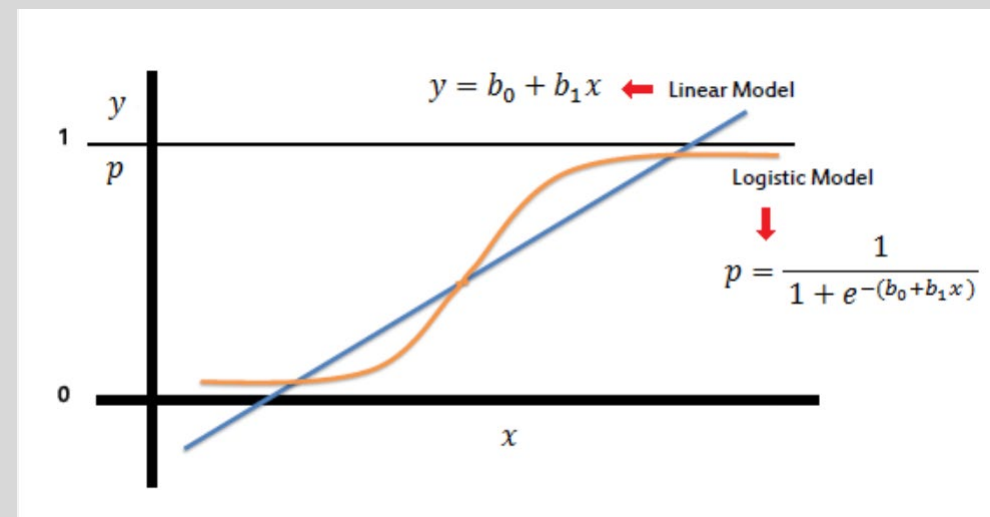
- Линейная регрессия:

$$a(w, x) = (w, x) \in R$$

- Логистическая регрессия:

$$a(w, x) = g((w, x))$$

- где $g(w, x) = \frac{1}{1+e^{-x}}$ - сигмоида (логистическая функция)
- Значения y будут от 0 до 1; $g(w, x)$ = вероятность того, что класс нашего объекта - положительный.
- Можно установить предел, например, 0.5: все объекты, у которых вероятность выше предела, относим к классу +1.
- Это и будет наша разделяющая граница.



Логистическая регрессия: функция потерь

- Если взять MSE, то возникнут проблемы:

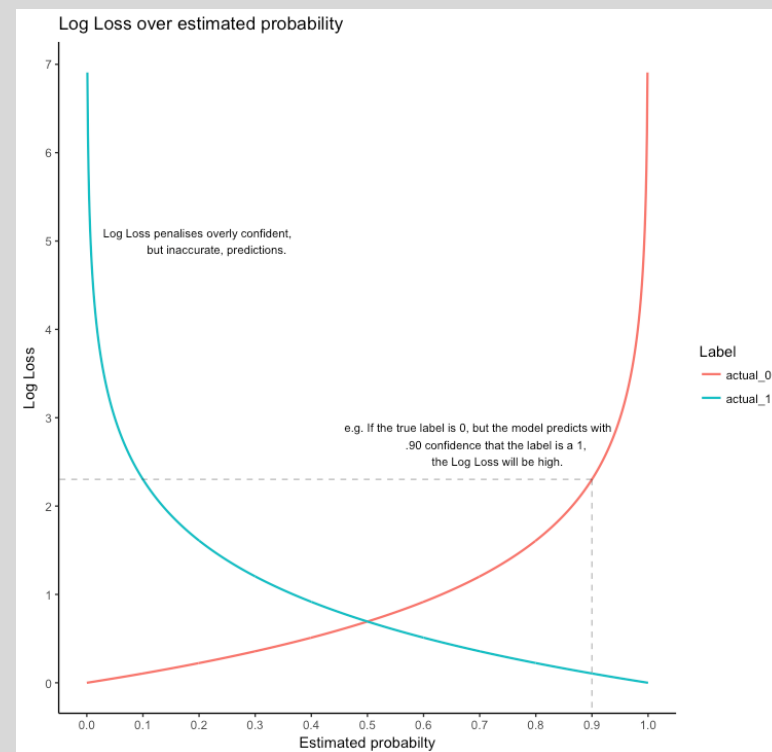
$Q(a, X) = \frac{1}{n} \sum_{i=1}^n \left(\frac{1}{1+e^{(-x,w)}} - y \right)^2$ - не выпуклая функция (можем не попасть в глобальный минимум при оптимизации)

- На совсем неправильном предсказании маленький штраф: допустим, предсказали вероятность 0% на объекте класса +1, тогда штраф всего $(1 - 0)^2 = 1$

- Возьмем логистическую функцию потерь:

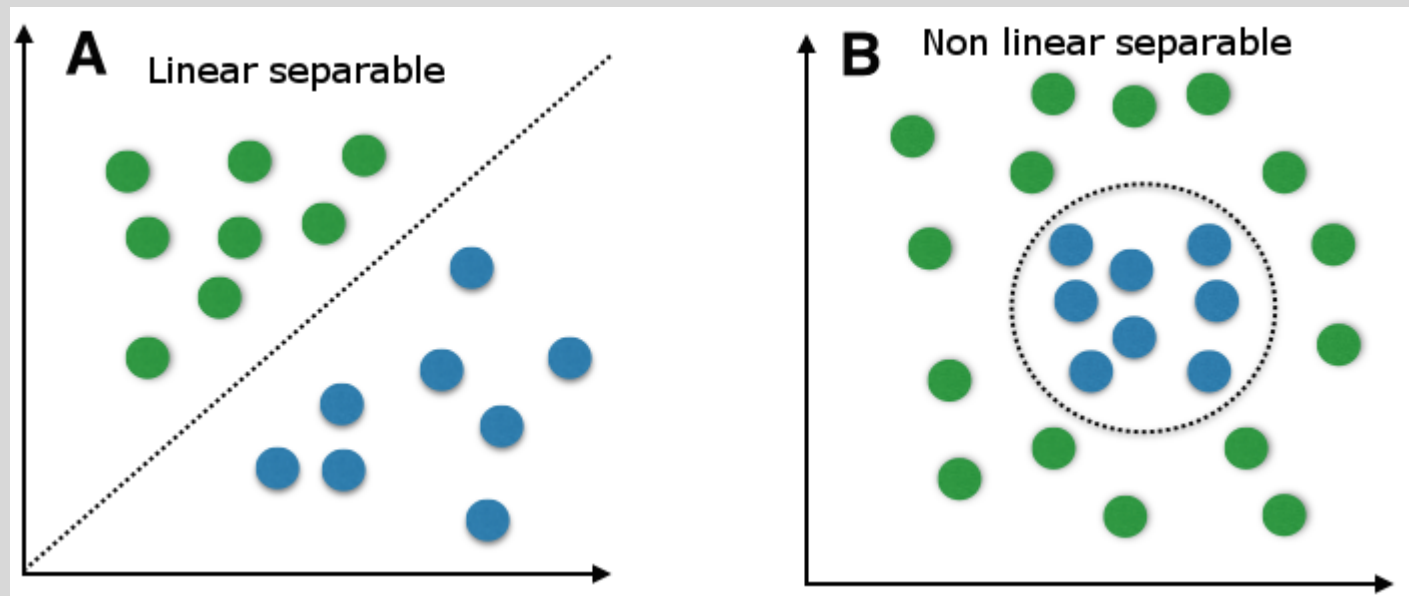
$$Q(w) = - \sum_{i=1}^n ([y_i = +1] \cdot \log(a(x_i, w))) + [y_i = -1] \cdot \log(1 - a(x_i, w)))$$

- если $a(w, x) = 1$ и $y = +1$, то штраф $L(a, y) = 0$
- если $a(w, x) \rightarrow 0$ и $y = +1$, то штраф $L(a, y) \rightarrow +\infty$

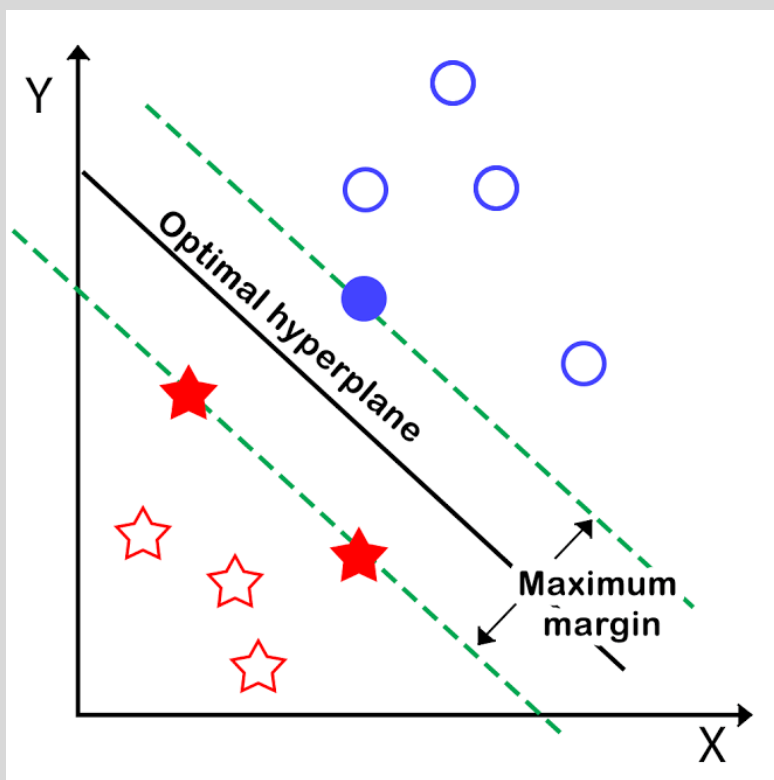


Метод опорных векторов

- Выборка может быть линейно разделимой и нет.



SVM и линейно разделимая выборка



- Цель SVM – максимизировать ширину разделяющей полосы.
- Расстояние от точки x_0 до разделяющей гиперплоскости, задаваемой классификатором:

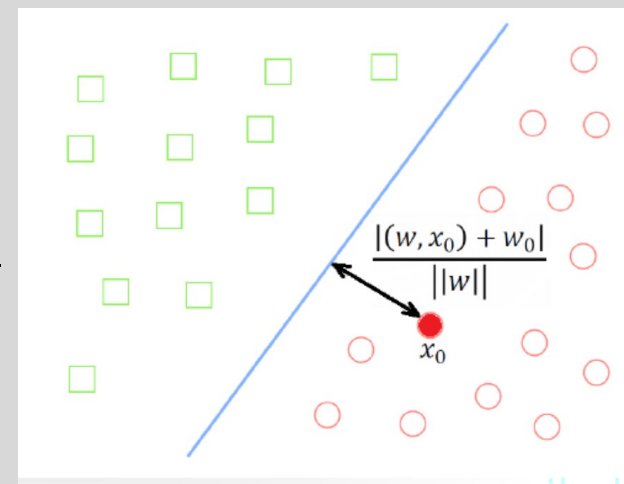
$$\rho(x_0, a) = \frac{|(w, x_0) + w_0|}{\|w\|}$$

- ($\|w\|$ - сумма квадратов коэффициентов плоскости)
- Будем оптимизировать так, что

$$\min_{x \in X} |(w, x) + w_0| = 1.$$

- Тогда расстояние до ближайшего объекта $x \in X$:

$$\min_{x \in X} |(w, x) + w_0| = \frac{1}{\|w\|} \min_{x \in X} |(w, x) + w_0| = \frac{1}{\|w\|}$$

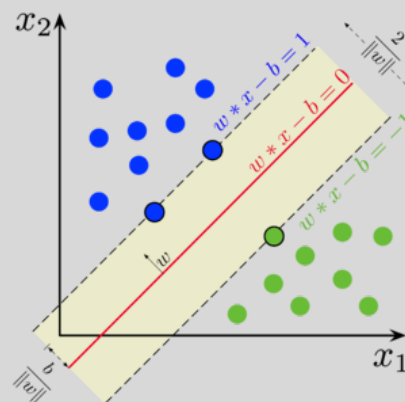


Оптимизационная задача SVM для линейно разделимой выборки

$$\begin{cases} \frac{1}{2} \|w\|^2 \rightarrow \min_w \\ y_i((w, x_i) + w_0) \geq 1, i = 1, \dots, l \end{cases}$$

Данная оптимизационная задача имеет единственное решение (доказывать не будем...)

$\frac{1}{2} \|w\|^2 \rightarrow \min$ – максимизируем ширину разделяющей полосы (квадрат – чтобы удобнее было считать производную);
 $y_i((w, x_i) + w_0)$ – отступ (margin), который > 0 , если правильная классификация; минимальное расстояние – 1, значит, все расстояния от точек до плоскости д.б. 1 и больше.



SVM и линейно неразделимая выборка

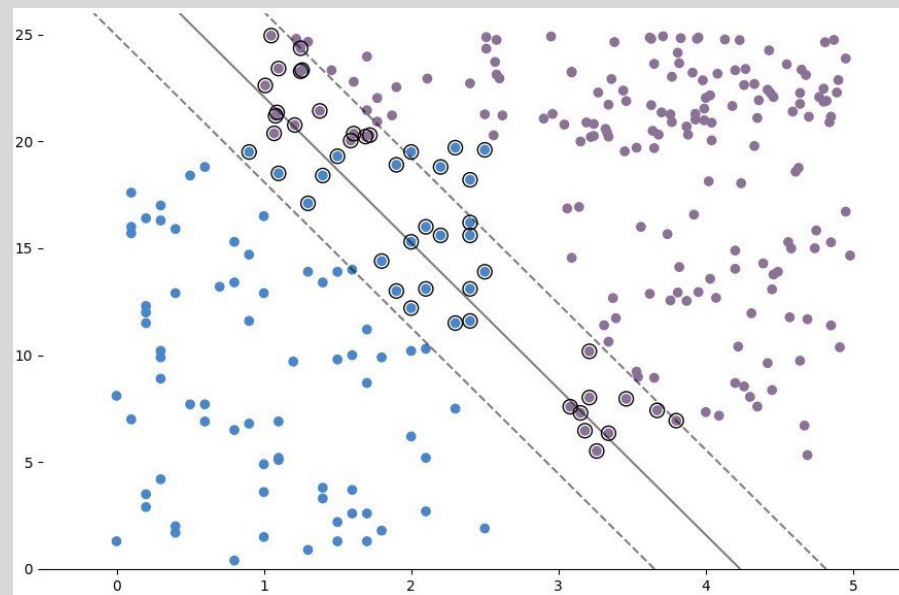
- Существует хотя бы один объект $x \in X$ такой, что
$$y_i((w, x_i) + w_0) < 1.$$

- Смягчим ограничения, введя штрафы $\xi_i \geq 0$:

$$y_i((w, x_i) + w_0) \geq 1 - \xi_i, i = 1, \dots, l$$

- Хотим:

1. Минимизировать штрафы $\sum_{i=1}^l \xi_i$
2. Максимизировать отступ $\frac{1}{\|w\|}$



Оптимизационная задача SVM для линейно неразделимой выборки

$$\begin{cases} \frac{1}{2} \|w\|^2 + c \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, i = 1, \dots, l \\ \xi_i \geq 0, i = 1, \dots, l \end{cases}$$

Является выпуклой и тоже имеет единственное решение. Доказывать опять не будем. Более простой (ну кому как, конечно) вид этой задачи:

$$\frac{1}{2} \|w\|^2 + c \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

SVM: задача оптимизации

- Итак, наша задача:

$$\frac{1}{2} \|w\|^2 + c \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

- Ничего не напоминает?

SVM: задача оптимизации

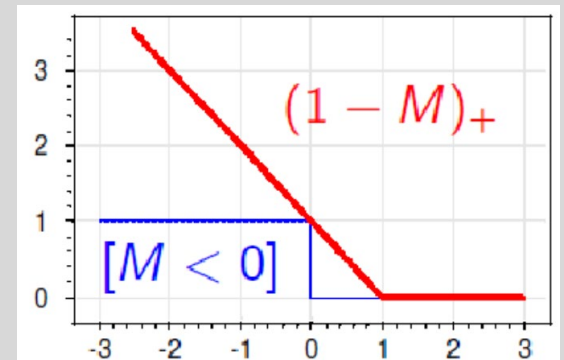
- Итак, наша задача:

$$\frac{1}{2} \|w\|^2 + c \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

- Можно рассматривать как оптимизацию функции потерь с регуляризацией:

$$Q(a, X) = \sum_{i=1}^n (1 - M_i(w, w_0))_+ + \frac{1}{2C} \|w\|^2 \rightarrow \min_{w, w_0}$$

$$L(w) = \sum_{i=1} \underbrace{\max(0, 1 - y_i[w^T x_i + b])}_{\text{Loss function}} + \underbrace{\lambda \|w\|_2^2}_{\text{regularization}}$$



Значение константы C

$$\frac{1}{2}\|w\|^2 + C \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

Что за C?

Это положительная константа, которая является управляющим параметром метода и позволяет находить компромисс между максимизацией разделяющей полосы и минимизацией суммарной ошибки.

