Занятие 6. Алгоритм SVM. Многоклассовая классификация

Колмагоров Евгений
ml.hse.dpo@yandex.ru

18 ноября 2024

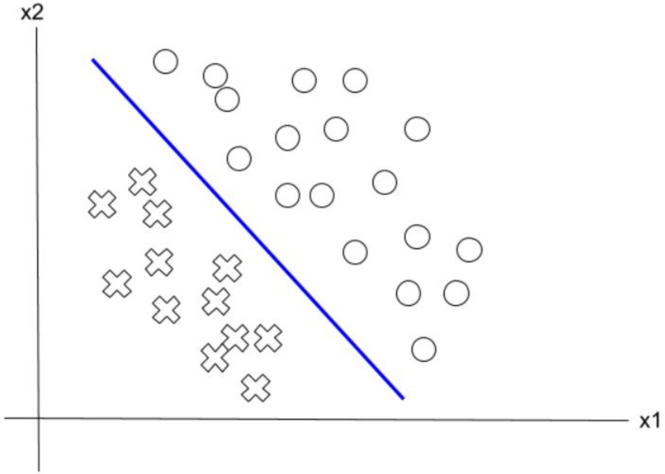
План лекции

1. Метод опорных вектор
2. “Мягкий” вариант алгоритма
3. Модели для многоклассовой классификации
4. Метрики качества в многоклассовой классификации

Напоминание. Бинарная классификация

Решаем задачу бинарной классификации методом линейной регрессии:

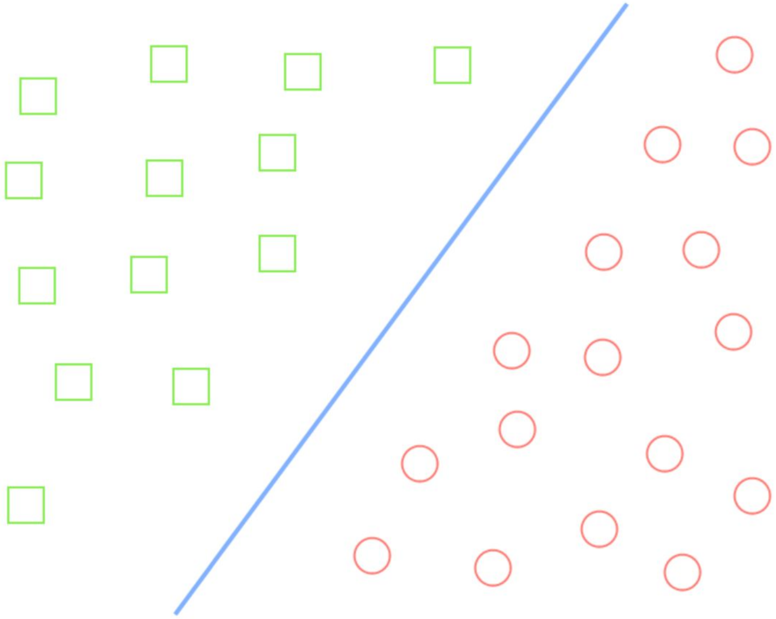
$$a(x, w) = sign(\sum_{i=0}^{d} w_i x_i)$$



$$M_i = y_i \cdot a(x_i, w) = y_i \cdot (w, x_i)$$

Линейно разделимая выборка

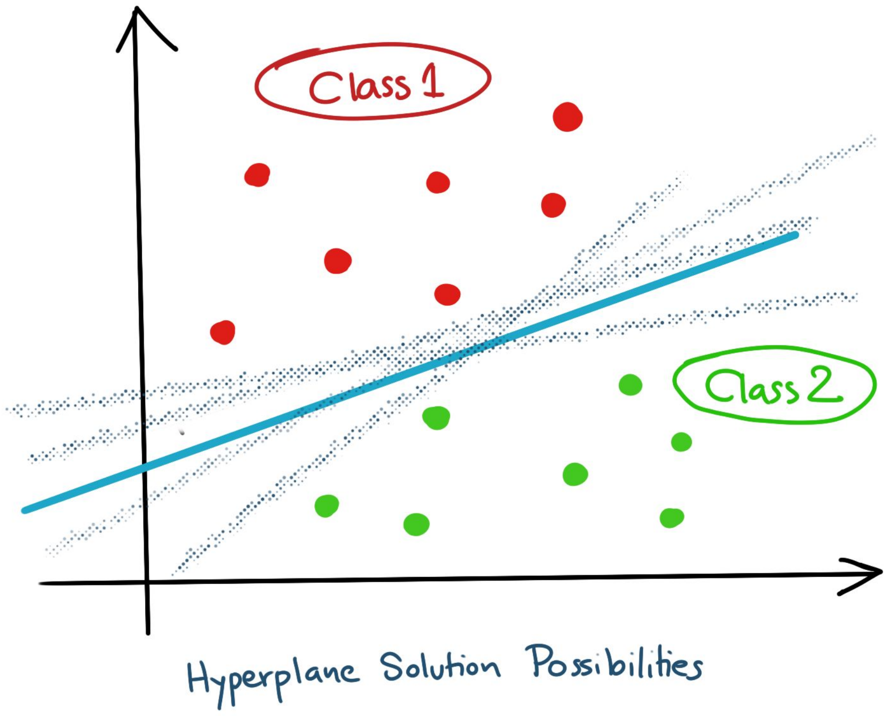
Определение: Выборка линейно разделима, если существует такой вектор параметров $\boldsymbol{w}^*$, что соответствующий классификатор $a(x)$ не допускает ошибок на этой выборке



Неоднозначность выбора классификатора

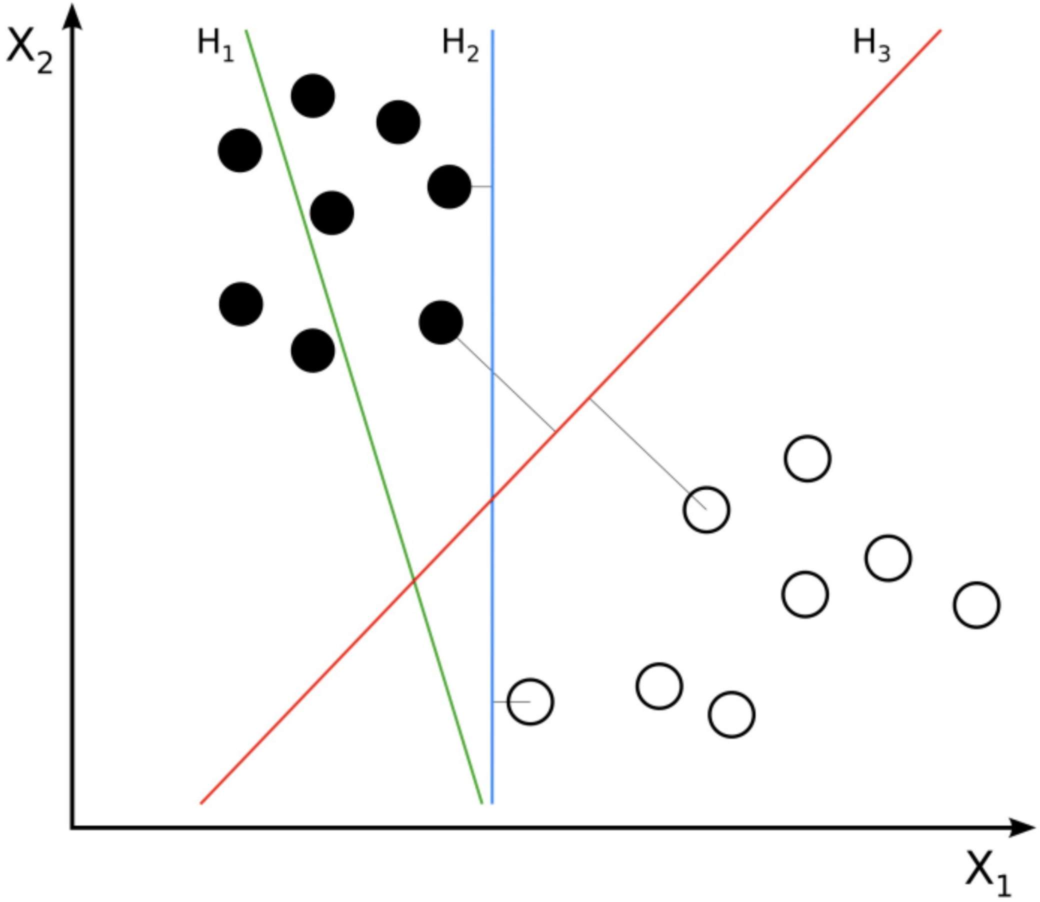
Но при построении разделяющей гиперповерхности для бинарного классификатора, который линейно разделяет выборку существует свобода в том, как именно может выглядеть разделяющая гиперповерхность

Вопрос: как выбрать среди возможных вариантов наилучшую?



Критерий “хорошей” гиперплоскости

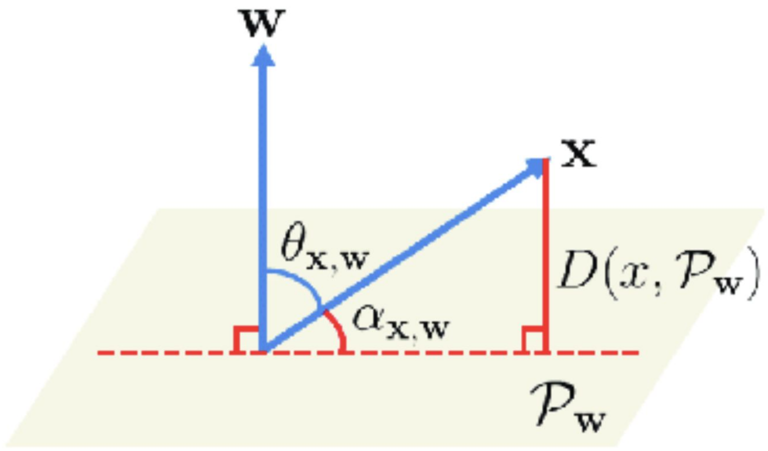
Будем считать, что гиперплоскость H_1 оптимальней гиперплоскости H_2, если расстояние от H_1 до некоторого ближайшего объекта выборки x_1^*, больше чем расстояние от H_2 до своего ближайшего объекта x_2^*



Напоминание. Расстояние от точки до гиперплоскости

Из курса линейной алгебры расстояние от точки x_0 до гиперплоскости H заданной своим вектором нормали w и смещением b определяется определяется по формуле:

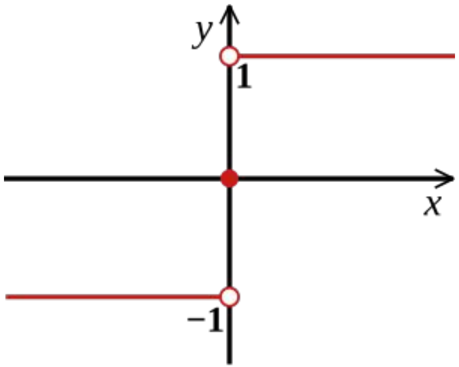
$$\rho(x_0, H) = \frac{|(w, x_0) + b|}{|w|}$$



Об одном свойстве sign функции

У функции сигнум есть следующее свойство:

$$sign(x \cdot y) = sign\, x \cdot sign\, y$$



Откуда для ответов a(x) классификатора следует, что

$$a(x) = sign((w, x)) = sign((\hat{w}, x) + b) = sign((k\hat{w}, x) + kb) =$$
$$= sign(k((\hat{w}, x) + b)) = sign(k) \cdot sign((\hat{w}, x) + b) =$$
$$= \{\text{если } k > 0\} = sign((\hat{w}, x) + b)$$

Выбор коэффициента k

Воспользуемся свободой выбора коэффициента k и отнормируем веса модели таким образом, чтобы для ближайшего объекта x^* обучающей выборки $\boldsymbol{X}$ было выполнено следующее равенство:

$$min_{x \in X} \left| (\hat{w}, x) + b \right| = 1$$

Расстояние до ближайшего объекта

Тогда по формуле расстояния выходит, что расстояние до ближайшего объекта выборки:

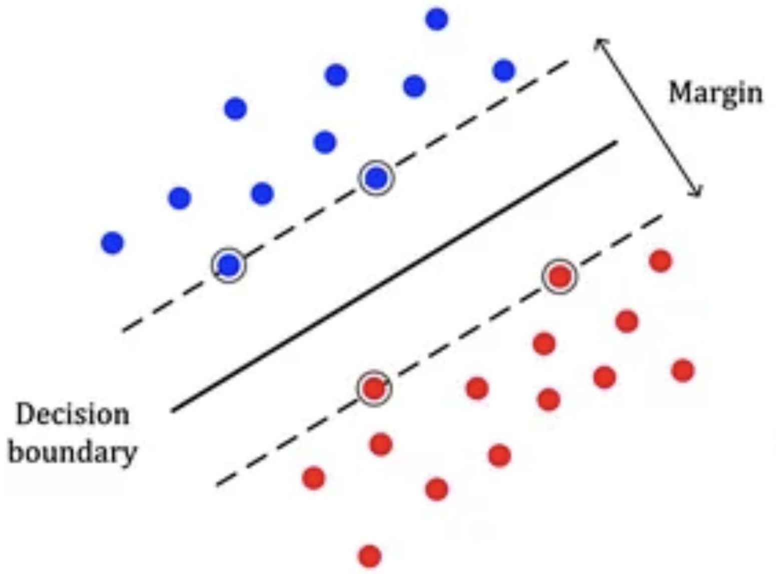
$$\rho(x^*, H) = min_{x\in X}\frac{|(\hat{w},x)+b|}{|\hat{w}|} = \frac{1}{\hat{w}}min_{x\in X}|(\hat{w},x)+b| = \frac{1}{|\hat{w}|}$$

Замечание: данная величина также носит название отступа (Margin)

Случай линейно разделимой выборки

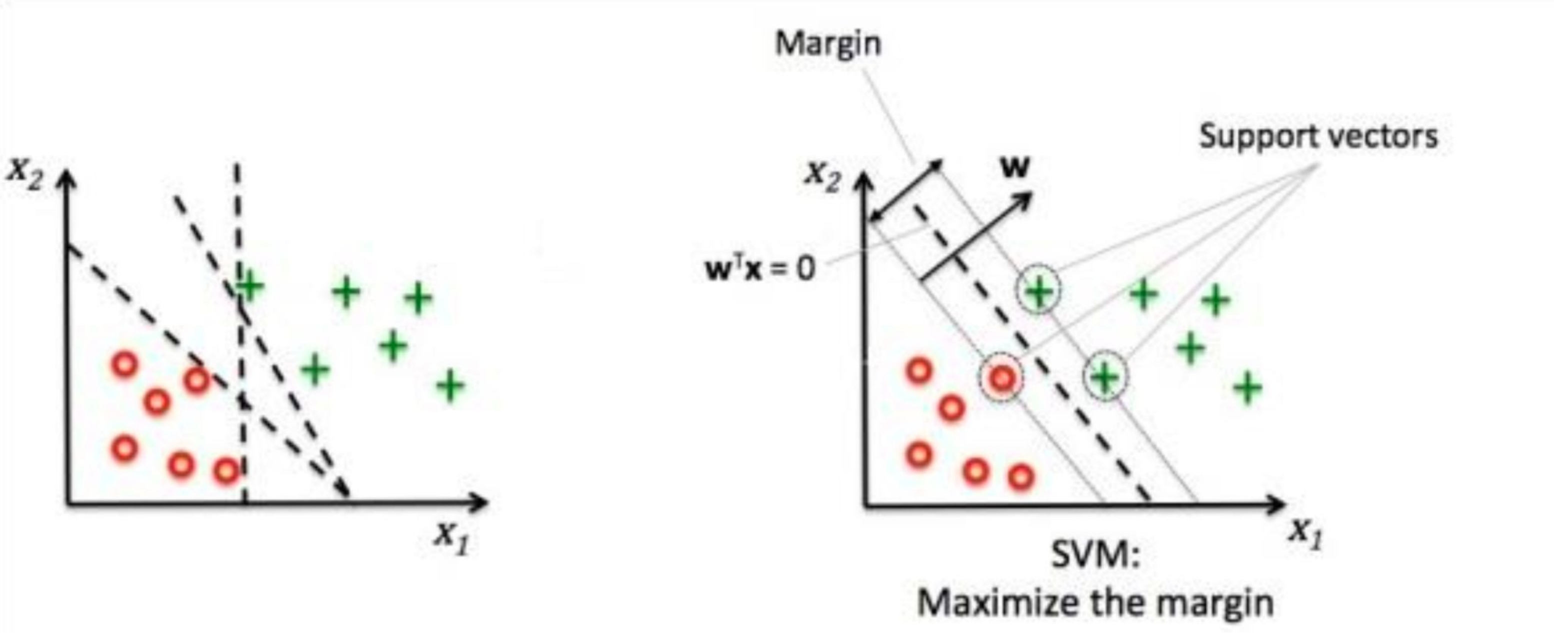
Если выборка линейно разделима, то величина ширина зазора между положительным и отрицательным классами будет удвоенной величиной:

$$M = \frac{2}{|\hat{w}|}$$



Алгоритм SVM

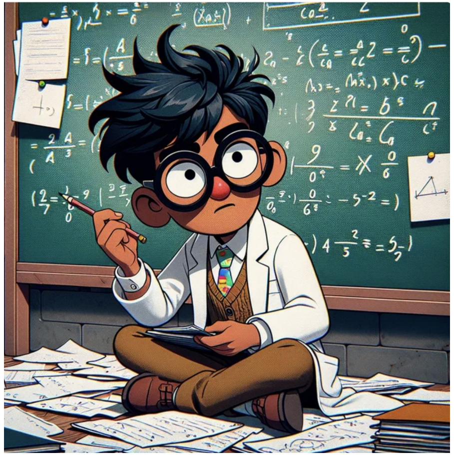
Метод классификации SVM (support vector machine) заключается в том, чтобы построить такую разделяющую гиперплоскость H так, чтобы ширина зазора M была максимальна



Формальная постановка задачи

Требуется построить классификатор, идеально разделяющий обучающую выборку и при этом имеющий максимальный отступ:

$$\begin{cases} \frac{1}{2}|\hat{w}| \to min \\ y_i((\hat{w}, x) + b) \geq 1,\ i = 1, \ldots, l \end{cases}$$



Решение данной оптимизационной задачи и будет соответствовать оптимальной гиперплоскости для SVM

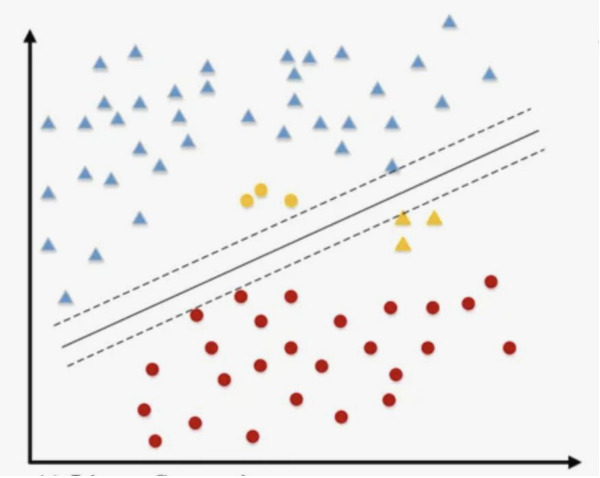
Может ли у SVM несколько гиперплоскостей

Утверждение: Данная оптимизационная задача имеет только одно единственное решение

$$\begin{cases} \frac{1}{2}|\hat{w}| \to min \\ y_i((\hat{w}, x) + b) \geq 1,\ i = 1, \ldots, l \end{cases}$$

Всегда ли имеем линейно разделимые выборки

В большинстве случаев выборки данных представляют собой нелинейно разделимое распределение.



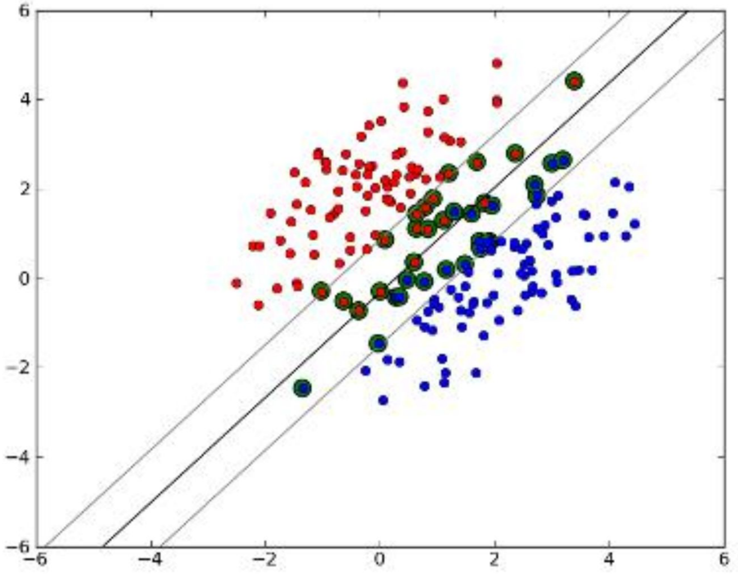
Возможны следующие ситуации

- Попадание объектов по другую сторону гиперплоскости
- Попадание объектов внутрь зазора

Линейно неразделимая выборка

- Существует хотя бы один объект x $\in$ X, что для любых параметров модели будет нарушено условие

$$\exists x_i : y_i((\hat{w}, x_i) + b) < 1, \ \forall \hat{w}, b$$



“Мягкий” вариант SVM

Смягчим ограничения и введём штрафы $\xi_i \geq 0$ за попадание объектов внутрь зазора M и с их учётом перепишем ограничения:

$$M_i = y_i((\hat{w}, x_i) + b) \geq 1 - \xi_i,\ i = 1, \ldots, l$$

Замечание: Если отступ объекта M_i: $0 \leq M_i \leq 1$, то объект верно классифицируется, но имеет ненулевой штраф

“Мягкий” вариант SVM

Таким образом в “мягком” варианте метода опорных векторов хотим, чтобы

- алгоритм классификации имел как можно меньшие штрафы ξ_i
- при этом имел как можно более широкий зазор $1/|\boldsymbol{w}|$

“Мягкий” вариант SVM

Таким образом в “мягком” варианте метода опорных векторов хотим, чтобы

- алгоритм классификации имел как можно меньшие штрафы ξ_i
- при этом имел как можно более широкий зазор *1/|**w**|*

В формальной постановке:

$$\begin{cases} \frac{1}{2}|\hat{w}| + C\sum_{i=1}^{l} \xi_i \to min_{\hat{w},\xi} \\ y_i((\hat{w}, x_i) + b) \geq 1 - \xi_i,\ i = 1, \dots, l \\ \xi_i \geq 0,\ i = 1, \dots, l \end{cases}$$

Единственность решения

Утверждение: Из теории оптимизации задача

$$\begin{cases} \frac{1}{2}|\hat{w}| + C\sum_{i=1}^{l} \xi_i \to min_{\hat{w},\xi} \\ y_i((\hat{w}, x_i) + b) \geq 1 - \xi_i,\ i = 1, \ldots, l \\ \xi_i \geq 0,\ i = 1, \ldots, l \end{cases}$$

является выпуклой и имеет единственное решение

Сведение к безусловной задаче

$$\begin{cases} \frac{1}{2}|\hat{w}| + C\sum_{i=1}^{l}\xi_i \to min_{\hat{w},\xi} \ \ (1) \\ y_i((\hat{w}, x_i) + b) \geq 1 - \xi_i, \ i = 1, \ldots, l \ (2) \\ \xi_i \geq 0, \ i = 1, \ldots, l \ \ (3) \end{cases}$$

Перепишем условия (2) и (3) в следующем виде:

$$\begin{cases} \xi_i \geq 1 - y_i((\hat{w}, x_i) + b) = 1 - M_i \\ \xi_i \geq 0 \end{cases}$$

Сведение к безусловной задаче

$$\begin{cases} \frac{1}{2}|\hat{w}| + C\sum_{i=1}^{l} \xi_i \to min_{\hat{w},\xi} \ \ (1) \\ y_i((\hat{w}, x_i) + b) \geq 1 - \xi_i, \ i = 1, \ldots, l \ (2) \\ \xi_i \geq 0, \ i = 1, \ldots, l \ \ (3) \end{cases}$$

Перепишем условия (2) и (3) в следующем виде:

$$\begin{cases} \xi_i \geq 1 - y_i((\hat{w}, x_i) + b) = 1 - M_i \\ \xi_i \geq 0 \end{cases}$$

$$\Rightarrow \xi_i = max(0, 1 - y_i((\hat{w}, x_i) + b))$$

Безусловная задача оптимизации

Подставим выражение для ξ_i в задачу минимизации:

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

Теперь задача обучения модели SVM свелась к поиску оптимального вектора весов w^*

Задача оптимизации

На задачу оптимизации SVM можно смотреть, как на оптимизацию функционала ошибки Q:

$$Q(a, X) = \frac{1}{N}\sum_{i=0}^{N} I[M_i < 0] \le \frac{1}{N}\sum_{i=0}^{N} L(x_i, y_i)$$

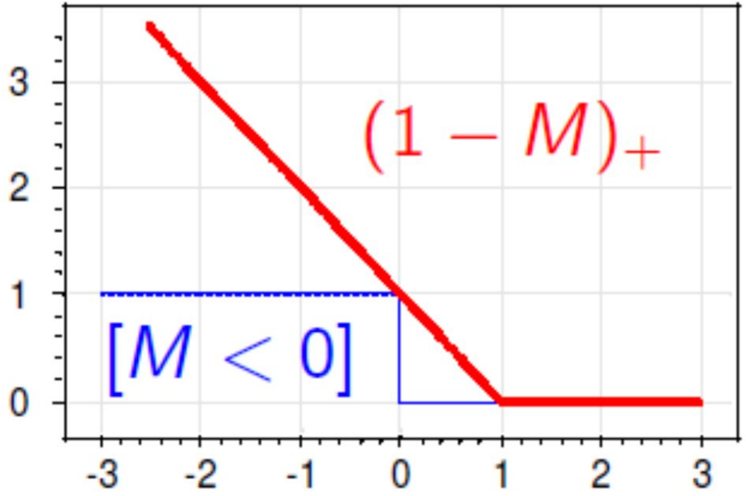
С функцией потерь $L(M_i) = max(0, 1 - M_i) = (1 - M_i)_+$ с L_2- регуляризацией:

$$Q(a, X) = \sum_{i=1}^{l}[max(0, 1 - y_i(w, x_i))] + \frac{1}{2C}|w|^2 \to min_w$$

Задача оптимизации

С функцией потерь $L(M_i) = max(0, 1 - M_i) = (1 - M_i)_+$ с L_2- регуляризацией:

$$Q(a, X) = \sum_{i=1}^{l} [max(0, 1 - y_i(w, x_i))] + \frac{1}{2C} |w|^2 \rightarrow min_w$$

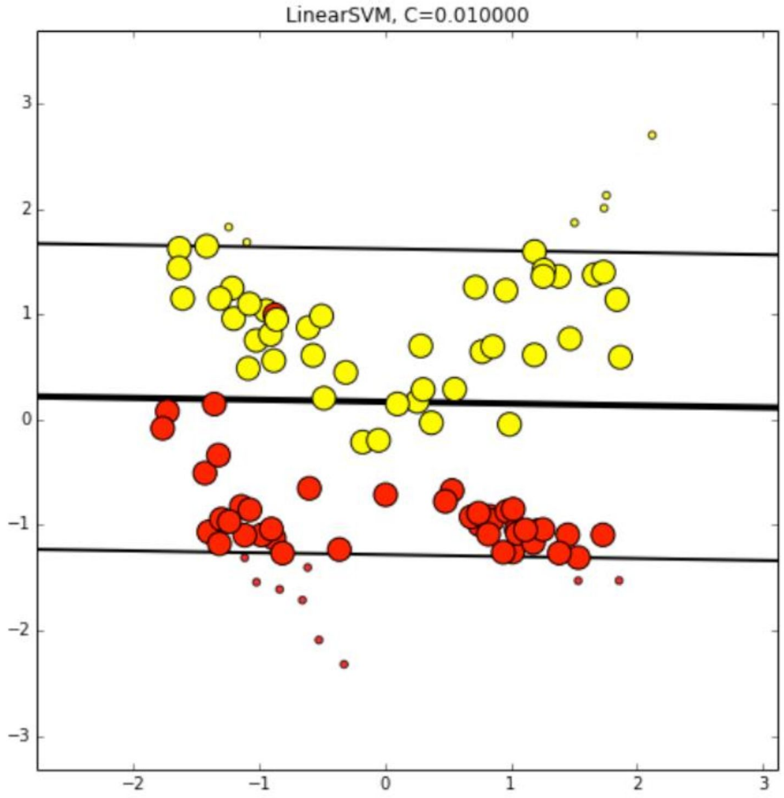


Значение константы C

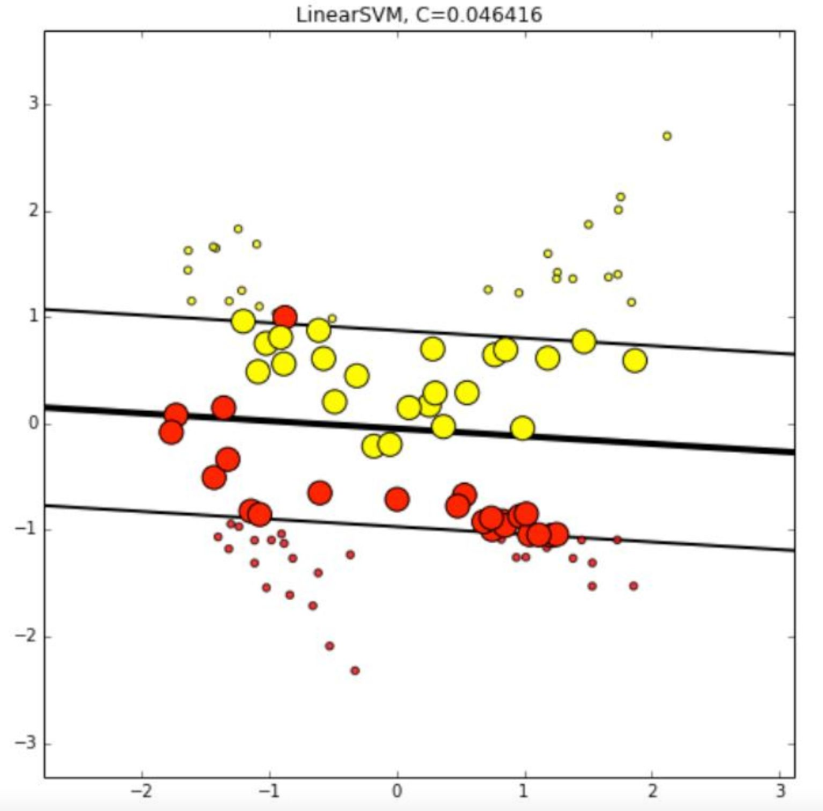
$$Q(a, X) = \sum_{i=1}^{l}[max(0, 1 - y_i(w, x_i))] + \frac{1}{2C}|w|^2 \rightarrow min_w$$

- Положительная константа *C* является управляющим параметром метода и позволяет находить компромисс между максимизацией разделяющей полосы и минимизацией суммарной ошибки.

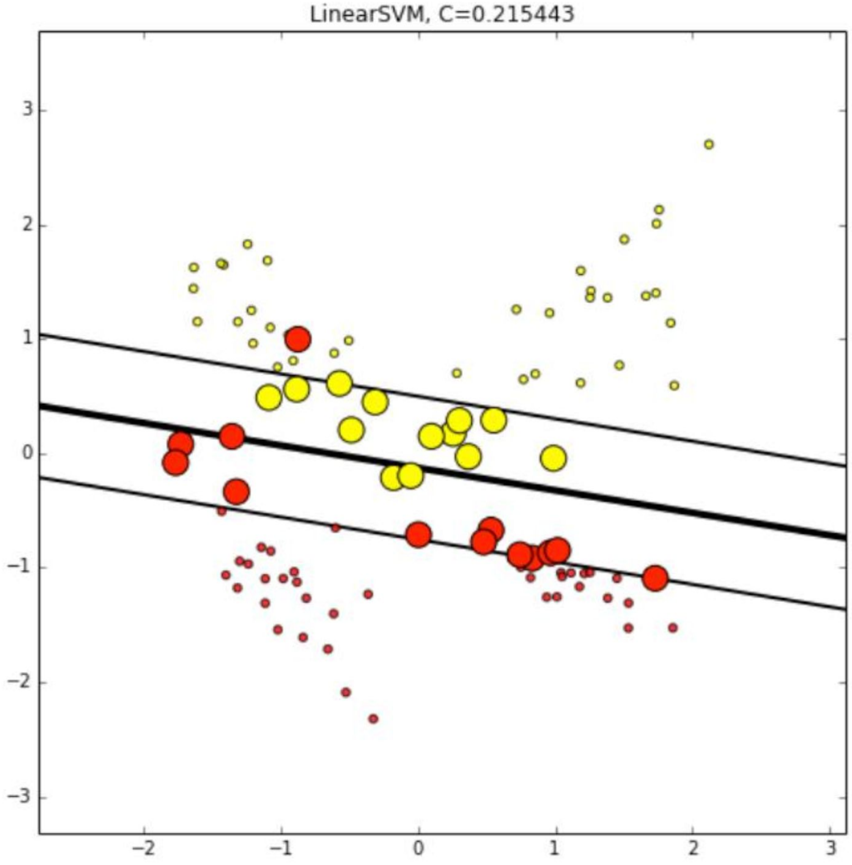
Значение константы C



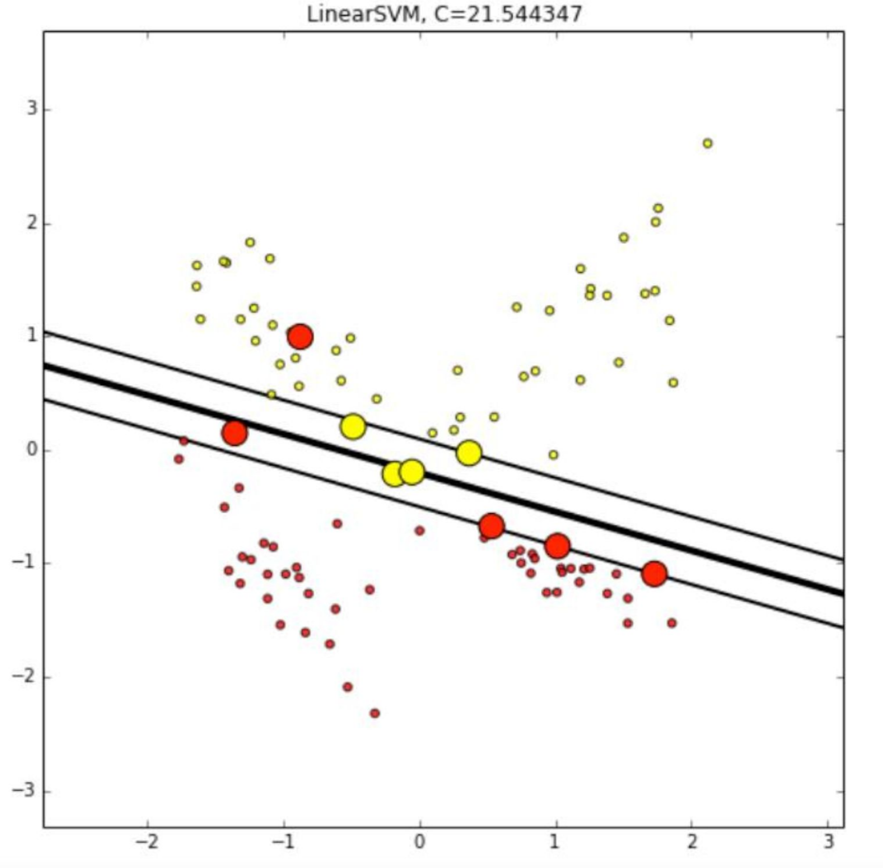
Значение константы C



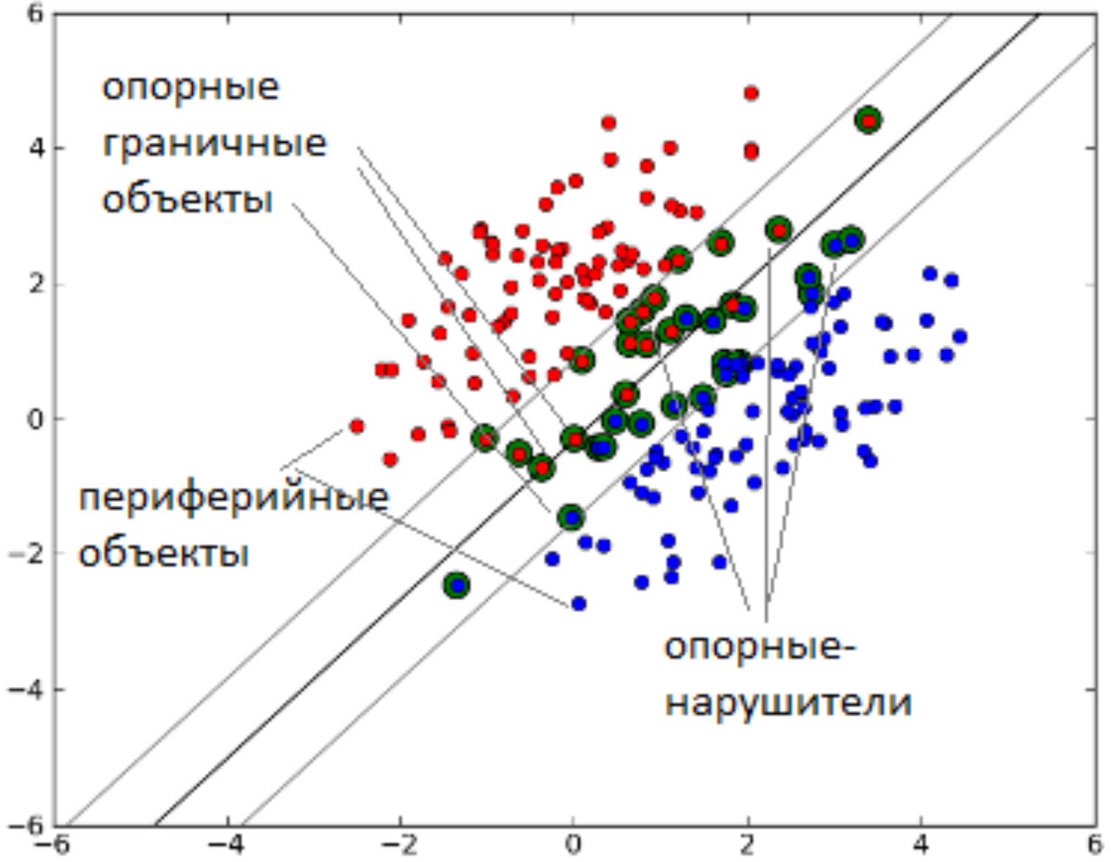
Значение константы C



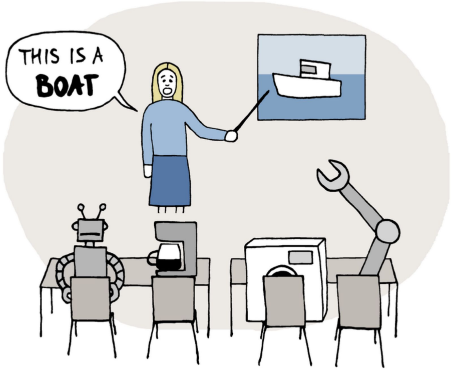
Значение константы C



Типы объектов в SVM



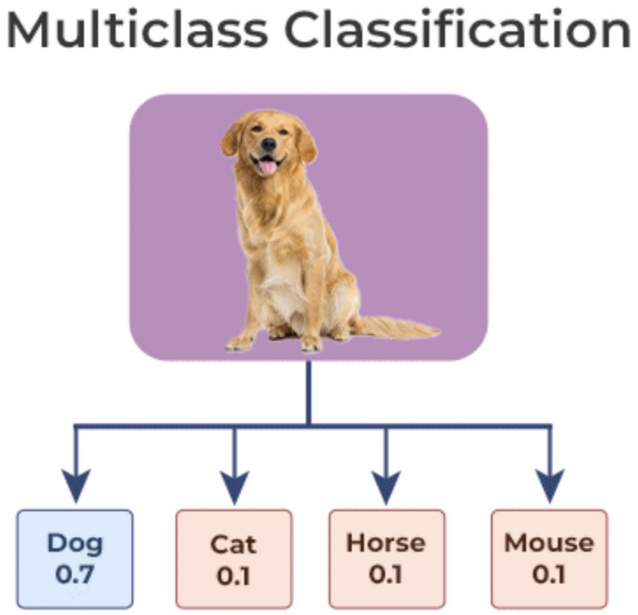
Многоклассовая классификация



Постановка задачи

Перейдём к более общей задаче классификации, в которой объект x может относиться к одному из нескольких классов:

- $y_i \in \{1, \ldots, K\}$, где $K > 2$

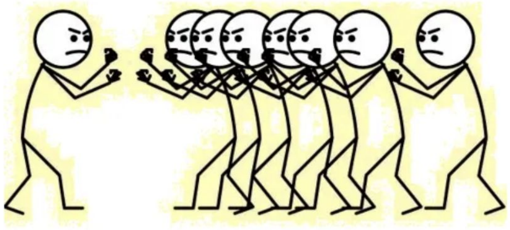


Подход one-vs-all

- Обучим K бинарных классификаторов $b_1(x)$, … , $b_K(x)$, каждый из которых решает задачу: принадлежит объект x к классу k_i или не принадлежит?

  Например, линейные классификаторы будут иметь вид

$$b_k(x) = sign(w_k \cdot x)$$



Подход one-vs-all

- Обучим K бинарных классификаторов $b_1(x)$, … , $b_K(x)$, каждый из которых решает задачу: принадлежит объект x к классу k_i или не принадлежит?

  Например, линейные классификаторы будут иметь вид

$$b_k(x) = sign(w_k \cdot x)$$

- Тогда в качестве итогового предсказания будем выдавать предсказания самого уверенного классификатора

$$a(x) = argmax_{k \in \{1,\dots,K\}}(w_k, x)$$

Подход one-vs-all

- Обучим K бинарных классификаторов $b_1(x)$, … , $b_K(x)$, каждый из которых решает задачу: принадлежит объект x к классу k_i или не принадлежит?

  Например, линейные классификаторы будут иметь вид

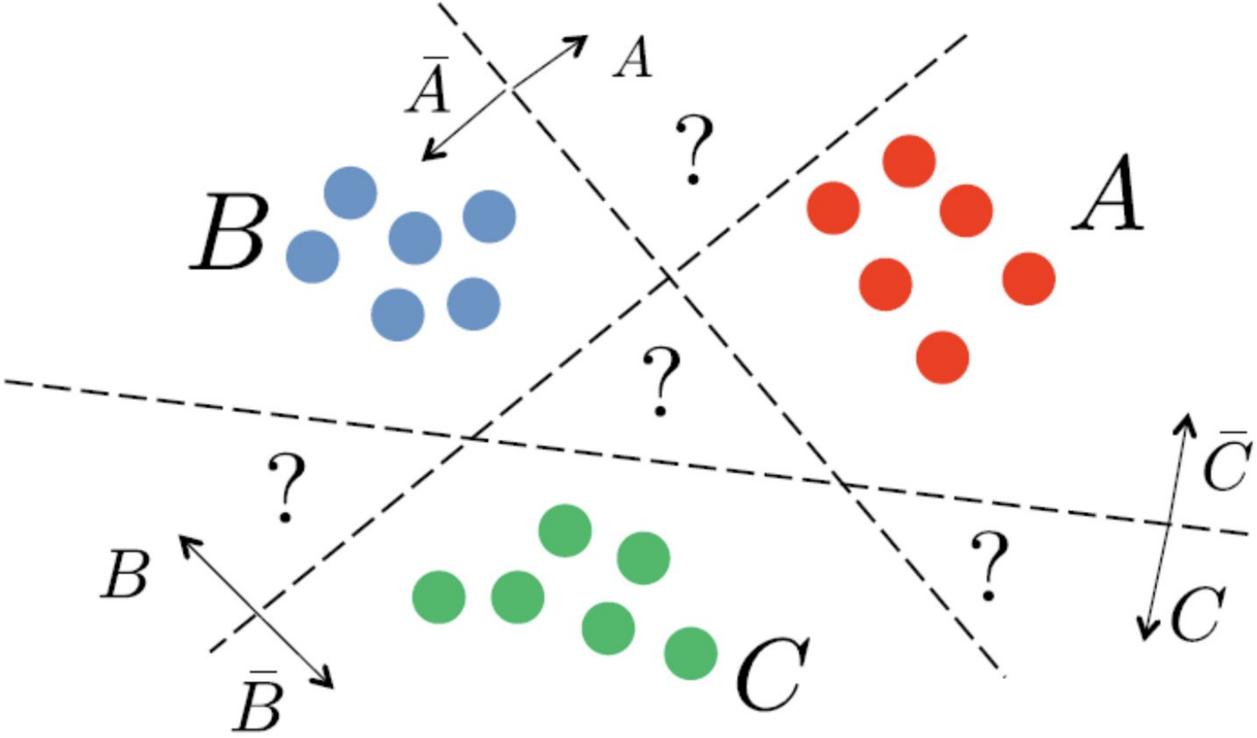
$$b_k(x) = sign(w_k \cdot x)$$

- Тогда в качестве итогового предсказания будем выдавать предсказания самого уверенного классификатора

$$a(x) = argmax_{k \in \{1,\dots,K\}}(w_k, x)$$

Проблема: Классификаторы могут иметь различные масштабы, поэтому сравнивать их некорректно

Подход all-vs-all



Подход all-vs-all

- Для каждой пары классов *i* и *j* обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс *i* или j

Подход all-vs-all

- Для каждой пары классов *i* и *j* обучим бинарный классификатор $a_{\mathrm{ij}}(x)$, который будет предсказывать класс *i* или j
- Если всего *K* классов, то получим C_K^2 классификаторов. Каждый такой классификатор будем обучать только на объектах классов *i* и *j*.

Подход all-vs-all

- Для каждой пары классов *i* и *j* обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс *i* или j
- Если всего *K* классов, то получим C_K^2 классификаторов. Каждый такой классификатор будем обучать только на объектах классов *i* и *j*.
- В качестве итогового предсказания выдадим класс, который предсказало наибольшее число классификаторов:

$$a(x) = argmax_{k \in \{1,\dots,K\}} \sum_{i=1}^{K} \sum_{i \neq j} I[a_{ij}(x) = k]$$

Проблема: Нужно обучить C_K^2 классификаторов, что при больших K может быть очень большим числом

Многоклассовые классификаторы

Некоторые методы бинарной классификации можно напрямую обощить на случай многих классов, таким образом, например, могут быть дополнены методы:

- Логистической регрессии
- Метод опорных векторов

Многоклассовая логистическая регрессия

- Предположим, что есть K линейных моделей, каждая из которых даёт оценку принадлежности выбранному классу: $b_k = (w_k, x)$
- Преобразуем оценку принадлежности выбранному классу через Softmax преобразование – некоторое обобщение сигмоиды.

$$softmax(b_1, \ldots, b_k) = \left(\frac{exp(b_1)}{\sum_{i=1}^{K} exp(b_i)}, \ldots, \frac{exp(b_K)}{\sum_{i=1}^{K} exp(b_i)}\right)$$

Многоклассовая логистическая регрессия

- Предположим, что есть K линейных моделей, каждая из которых даёт оценку принадлежности выбранному классу: $b_k = (w_k, x)$
- Преобразуем оценку принадлежности выбранному классу через Softmax преобразование – некоторое обобщение сигмоиды

$$softmax(b_1, \ldots, b_k) = \left(\frac{exp(b_1)}{\sum_{i=1}^{K} exp(b_i)}, \ldots, \frac{exp(b_K)}{\sum_{i=1}^{K} exp(b_i)}\right)$$

- Тогда вероятность класса k:

$$P(y_i = k | x_i, w_k) = \frac{exp(w_k \cdot x_i)}{\sum_{i=1}^{K} exp(w_i \cdot x_i)}$$

Обучение модели

Аналогично бинарному случаю можно расписать правдоподобие для случая, когда целевая переменная y может принадлежать нескольким классам – мультиномиальное распределение

$$\Pi = \prod_{i=1}^{n} a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} =$$

$$= \prod_{i=1}^{n} \prod_{j=1}^{K} a_j(x_i)^{[y_i=j]} \to \max_{w_1,\dots,w_K}$$

Обучение модели

И после логарифмирования получим многоклассовый аналог logloss:

$$-\sum_{i=1}^{n}\sum_{j=1}^{K}[y_i = j]\, logP(y = j|x_i, w) \to \min_{w_1,\dots,w_K}$$

Метрики качества

Подобно бинарному случаю можно составить матрицу ошибок размера K x K, где K – количество классов

		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	4	6	3
	Fish	1	2	0
	Hen	1	2	6

Пример матрицы ошибок для трёхклассовой классификации

Виды усреднений

Для подсчёта качества работы алгоритма на всех классах применяют различные способы усреднения качеств работы на каждом из классов.

Существует:

- Макро-усреднение (macro-average)
- Микро-усреднение (micro-average)
- Взвешенное усреднение (weighted-average)

Макро-усрденение

В данном подходе вычисляется значение выбранной метрики для каждого бинарного классификатора. Например, кошка/не кошка, рыба/ не рыба, курица/ не курица, а затем полученные метрики усредняются.

Например:

$$Macro - accuracy = \frac{Accuracy_1 + ... + Accuracy_K}{K}$$, $Accuracy_k$ – Accuracy на k-ом классе

$$Macro - precision = \frac{Precision_1 + ... + Precision_K}{K}$$, $Precision_k$ - Точность на k-ом классе

Макро-усреднение

Например, посчитаем для данного примера, макро-precision

		True/Actual		
		Cat ( )	Fish ( )	Hen ( )
Predicted	Cat ( )	4	6	3
	Fish ( )	1	2	0
	Hen ( )	1	2	6

$$Precision(cat) = \frac{4}{4+6+3} = 4/13$$

$$Precision(fish) = \frac{2}{2+1+0} = 2/3$$

$$Precision(Hen) = \frac{6}{6+2+1} = 2/3$$

$$Macro - precision = (Precision(cat) + Precision(fish) + Precision(Hen))/3 =$$
$$= (4/14 + 2/3 + 2/3)/3 \approx 0.55$$

Вопрос: В каких случаях метрика может завышать или занижать предсказания?

Микро-усреднение

В этом подходе мы вычисляем значения TP, TN, FP, FN по всей матрице ошибок сразу, исходя из их определения. Затем по полученным числам вычисляем выбранные метрики.

Например, для Precision и Recall микроусреднение:

$$\mathrm{Precision}_{(\mathrm{micro})} = \frac{\sum_{k=1}^{K} \mathrm{TP}_k}{\sum_{k=1}^{K} (\mathrm{TP}_k + \mathrm{FP}_k)}$$

$$\mathrm{Recall}_{(\mathrm{micro})} = \frac{\sum_{k=1}^{K} \mathrm{TP}_k}{\sum_{k=1}^{K} (\mathrm{TP}_k + \mathrm{FN}_k)}$$

Микро-усреднение

$$precision = \frac{12}{12 + 13} = \frac{12}{25}$$

		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	4	6	3
	Fish	1	2	0
	Hen	1	2	6

TP - это количество верно угаданных объектов положительного класса. В нашем случае TP = 4 + 2 + 6 = 12

FP - это суммарное количество false positive-предсказаний. Например, если cat предсказана как fish, то это false positive для fish. Таким образом, FP - это сумма всех неверных предсказаний FP = 6 + 3 + 1 + 0 + 1 + 2 = 13

Микро-усреднение

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

		True/Actual		
		Cat ( )	Fish ( )	Hen ( )
Predicted	Cat ( )	4	6	3
	Fish ( )	1	2	0
	Hen ( )	1	2	6

TP - это количество верно угаданных объектов положительного класса. В нашем случае TP = 4 + 2 + 6 = 12

FN - это сумма false negative-предсказаний. Например, если cat предсказана как fish, то это false negative для cat. Таким образом, FN - это опять же сумма всех неверных предсказаний, то есть FN = 4 + 1 + 6 + 2+ 3 + 0 = 13

- В случае микро-усреднения *precision = recall*
- Так как f1-мера это среднее гармоническое точности и полноты, то *f1 = precision = recall*

Взвешенное усреднение (weighted-average)

В этом подходе мы усредняем посчитанные для каждого класса метрики с весами, пропорциональными количеству объектов класса

Например, в данном случае

$$weighted-precision = \frac{n_1}{N} \cdot precision_1 + \ldots + \frac{n_K}{N} \cdot precision_K$$

$$weighted-recall = \frac{n_1}{N} \cdot recall_1 + \ldots + \frac{n_K}{N} \cdot recall_K$$

Взвешенное усреднение (weighted-average)

		True/Actual		
		Cat ( )	Fish ( )	Hen ( )
Predicted	Cat ( )	4	6	3
	Fish ( )	1	2	0
	Hen ( )	1	2	6

$$weighted - precision = \frac{6}{25} \cdot precision(cat) + \frac{10}{25} \cdot precision(fish) + \frac{9}{25} \cdot precision(hen) =$$
$$= \frac{6}{25} \cdot \frac{4}{13} + \frac{10}{25} \cdot \frac{2}{3} + \frac{9}{25} \cdot \frac{2}{3} \approx 0.43$$