Системы искусственного интеллекта

Лекция 5

SVM и логистическая регрессия

Запорожцев Иван Федорович
zaporozhtsev.if.work@gmail.com

Линейные решающие модели в задаче бинарной классификации

Пусть $X = \mathbb{R}^n$, $Y = \{\pm 1\}$

(обратите внимание на метки)

Обучающая выборка: $\{(x_i, y_i)\}_{i=1}^m$

Хотим линейную модель:
(формально зависимость нелинейная)

$$a(x) = \operatorname{sgn}(w^{\mathrm{T}}x + b) = \begin{cases} +1, & w^{\mathrm{T}}x + b > 0 \\ -1, & w^{\mathrm{T}}x + b < 0 \end{cases}$$

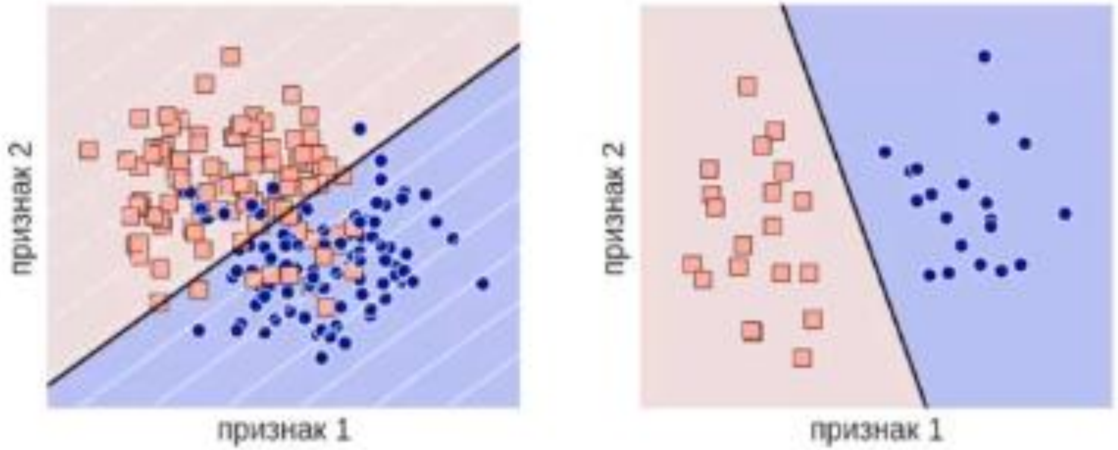
Случай $w^{\mathrm{T}}x + b = 0$ **нам тут не особо важен**

Линейный классификатор

$$a(x) = \operatorname{sgn}(w^{\mathrm{T}} x)$$

Если пополнить признаковое пространство фиктивным признаком

Геометрический смысл: делим пространство гиперплоскостью на две части



Иногда это может здорово работать!

Линейный классификатор: обучение

Общая идея – 0-1-loss – число ошибок

$$L(X_{\text{train}}, a) = \sum_{t=1}^{m} L(y_t, a(x_t)) \to \min$$

$$L(y_t, a(x_t)) = \begin{cases} 1, & \operatorname{sgn} w^{\mathrm{T}} x_t \neq y_t \\ 0, & \operatorname{sgn} w^{\mathrm{T}} x_t = y_t, \end{cases}$$

Естественно минимизировать число ошибок, но

- производная = 0 (не везде дифференцируема)
- выдаёт мало информации
  только число ошибок, а не их «фатальность»
- оптимизация здесь – NP-полная задача

Зазор (Margin)

$$L(y_t, a(x_t)) = \begin{cases} 1, & \operatorname{sgn} w^{\mathrm{T}} x_t \neq y_t \\ 0, & \operatorname{sgn} w^{\mathrm{T}} x_t = y_t, \end{cases}$$

Можно переписать:

$$L(y_t, a(x_t)) = I[y_t w^{\mathrm{T}} x_t \leq 0] = \theta(-z_t)$$

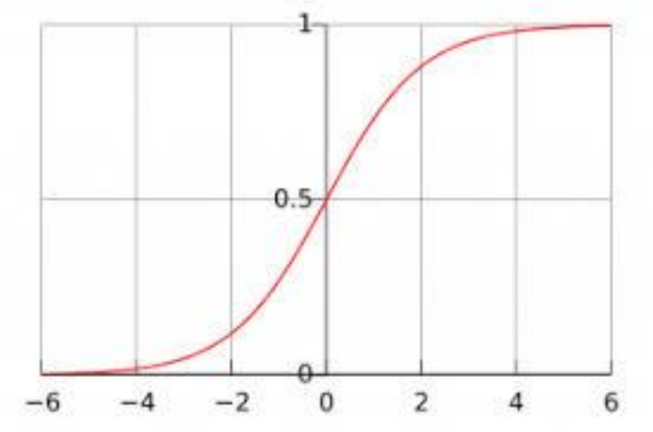
> Где $\theta(z) = I[z \geq 0]$
>
> $z_t = y_t w^{\mathrm{T}} x_t$ – зазор (чем меньше, тем хуже)

Пропорционален расстоянию до ГП
с точностью до $\|w\|$ (дальше), знак ~ ошибка

Суррогатные функции

Surrogate loss functions

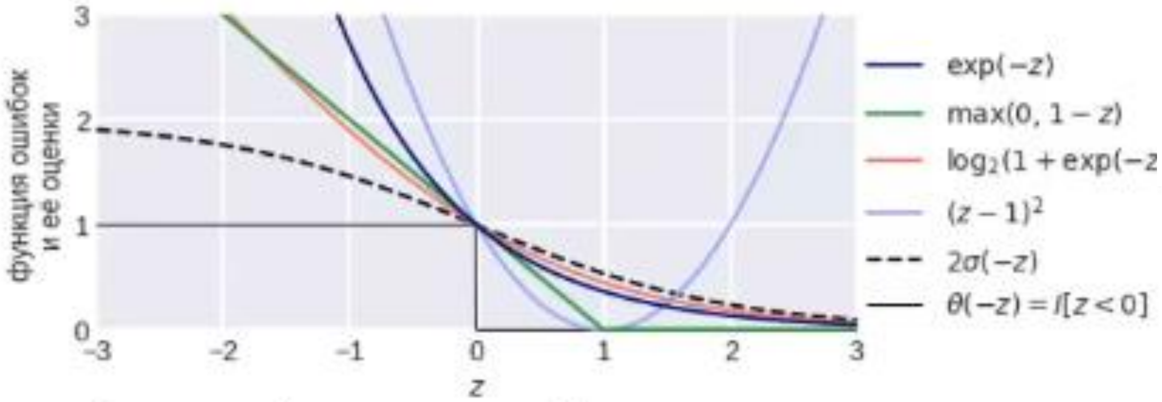
$$\sigma(z) = \frac{1}{1+e^{-z}}$$



$$\sum_{t=1}^{m} L(y_t, a(x_t)) \le \sum_{t=1}^{m} L'(y_t, a(x_t)) \to \min$$

$$L(y_t, a(x_t)) = \theta(-z_t) \le f(-z_t) = L'(y_t, a(x_t))$$

$$z_t = y_t w^{\mathrm{T}} x_t$$



ReLU

Оценка функции ошибок через гладкую функцию, которую проще оптимизировать

Реализация в scikit-learn

```
sklearn.linear_model.SGDClassifier
sklearn.linear_model.SGDRegressor
```

– общая реализация линейного алгоритма, обучающегося градиентным спуском работает с разреженными данными!

```
loss="hinge"
```

Функция ошибки, варианты:

- `"hinge"` – SVM $f(z) = max(0, z)$ ReLU
- `"log"` – логистическая регрессия
- `"modified_huber"`
- `"squared_hinge"`
- `"perceptron"` – персептрон

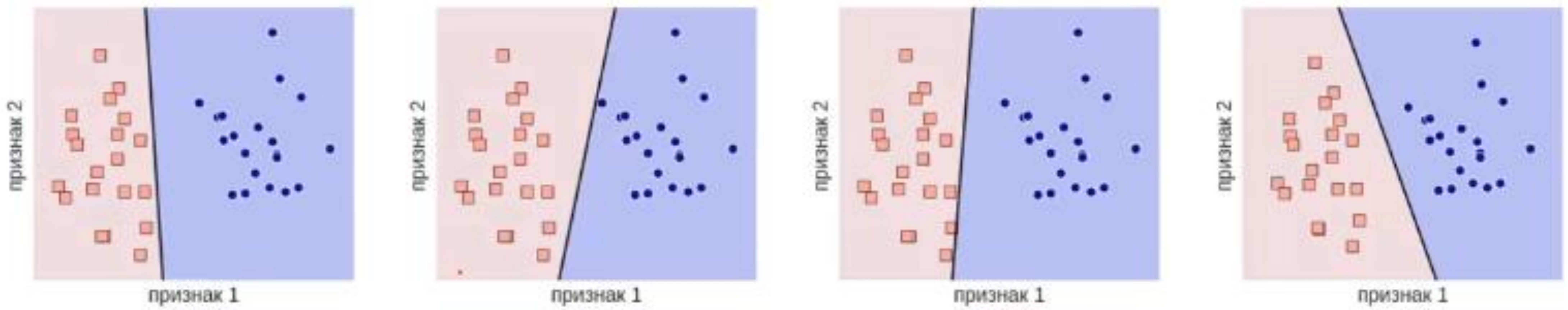
Для регрессии:

- `"squared_loss"`
- `"huber"`
- `"epsilon_insensitive"`
- `"squared_epsilon_insensitive"`

Support Vector Machine (SVM)

OneClassSVM

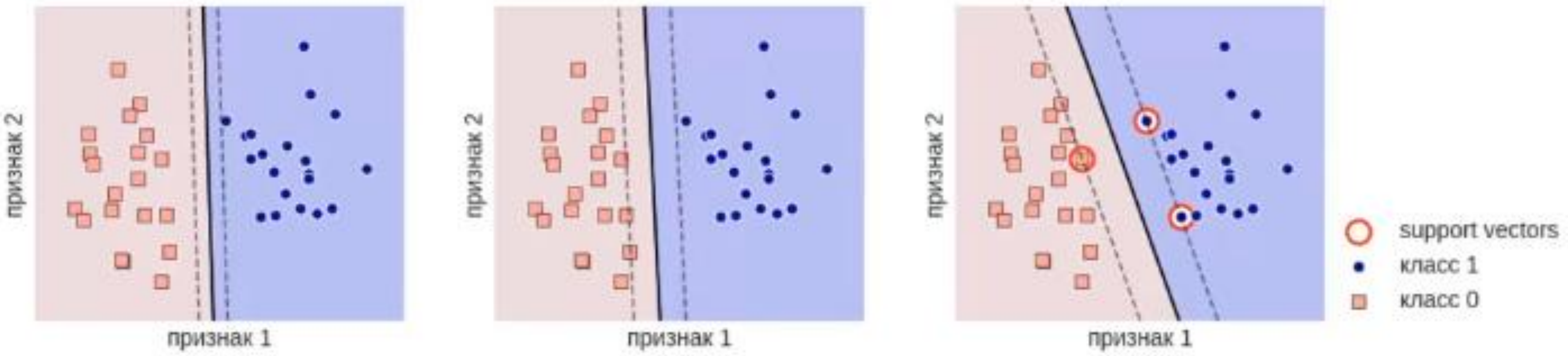
Метод опорных векторов – самый популярный метод 1990х
Здесь для начала предполагаем, что классы линейно разделимы



До сих пор пытались просто разделить точки...

А какой линейный классификатор лучше?

SVM: идея максимального зазора



Если немного ошибёмся с коэффициентами / если объект задан неточно, всё равно решение должно быть правильным

SVM: постановка задачи

Пусть обучающая выборка:

$$\{(x_i, y_i)\}_{i=1}^{m}$$

$$y_i \in Y = \{\pm 1\}$$

Хотим разделить точки двух разных классов гиперплоскостью

$$a(x) = \operatorname{sgn}(w^{\mathrm{T}} x + b)$$

(здесь не вводим фиктивный константный признак)

Коэффициенты нужны с точностью до множителя $\alpha > 0$

$$\operatorname{sgn}(w^{\mathrm{T}} x + b) = \operatorname{sgn}(\alpha w^{\mathrm{T}} x + \alpha b)$$

$$w^T x + b = 0 \Leftrightarrow w_1 x_1 + \ldots + w_n x_n + b = 0 \Leftrightarrow \alpha w_1 x_1 + \ldots + \alpha w_n x_n + \alpha b = 0$$

Должно быть

$$w^{\mathrm{T}} x_i + b \geq +1 \quad \text{если} \quad y_i = +1$$

$$w^{\mathrm{T}} x_i + b \leq -1 \quad \text{если} \quad y_i = -1$$

(из-за нормировки это возможно)

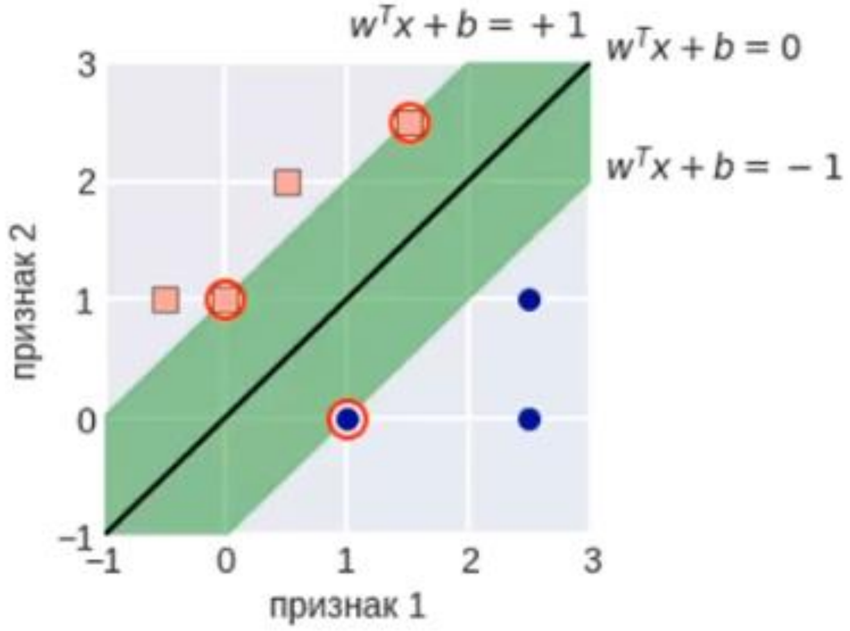
Другая форма записи:

$$y_i (w^{\mathrm{T}} x_i + b) \geq 1$$

Можно считать (из-за нормировки), что

$$\min_i | w^{\mathrm{T}} x_i + b | = 1$$

SVM: постановка задачи



$$y_i(w^{\mathrm{T}} x_i + b) \geq 1$$

$$\min_i | w^{\mathrm{T}} x_i + b | = 1$$

SVM: постановка задачи

Расстояние от точки до гиперплоскости:

$$\rho\left(x_i, \left\{x \in \alpha : w^T x + b = 0\right\}\right) = \frac{|w^{\mathrm{T}} x_i + b|}{\|w\|}$$

$$\downarrow$$

Хотим, чтобы минимум из этих расстояний был максимален

нормированный зазор (margin) –

$$\min_i \frac{|w^{\mathrm{T}} x_i + b|}{\|w\|} = \frac{1}{\|w\|} \to \max$$

$$\alpha : Ax + By + Cz + D = 0, \quad x^0 : (x_0, y_0, z_0)$$

$$\rho(x^0, \alpha) = \frac{|Ax_0 + By_0 + Cz_0 + D|}{\sqrt{A^2 + B^2 + C^2}}$$

SVM: постановка задачи

В общем случае, когда $X = \mathbb{R}^n$, $Y = \{\pm 1\}$,
обучающая выборка: $\{(x_i, y_i)\}_{i=1}^{m}$

Алгоритм со скоринговой функцией (score function):

$$a(x) = \mathrm{sgn}\big(b(x)\big), b(x) \in \mathbb{R},$$

> $|b(x_i)|$ – уверенность в ответе
>
> $y_i b(x_i)$ – зазор (насколько уверенность оправдала ожидание)
>
> $\mathrm{sgn}(\cdot)$ – деформация

SVM: постановка задачи

Задача квадратичного программирования

QP = Quadratic Program
с m ограничениями (constraints)

$$\frac{\|w\|^2}{2} \to \min$$

$$y_i(w^{\mathrm{T}}x_i + b) \geq 1,\ i \in \{1,2,\ldots,m\}$$

Заметим, что здесь также, как при регуляризации линейной регрессии, хотим квадрат нормы весов сделать меньше

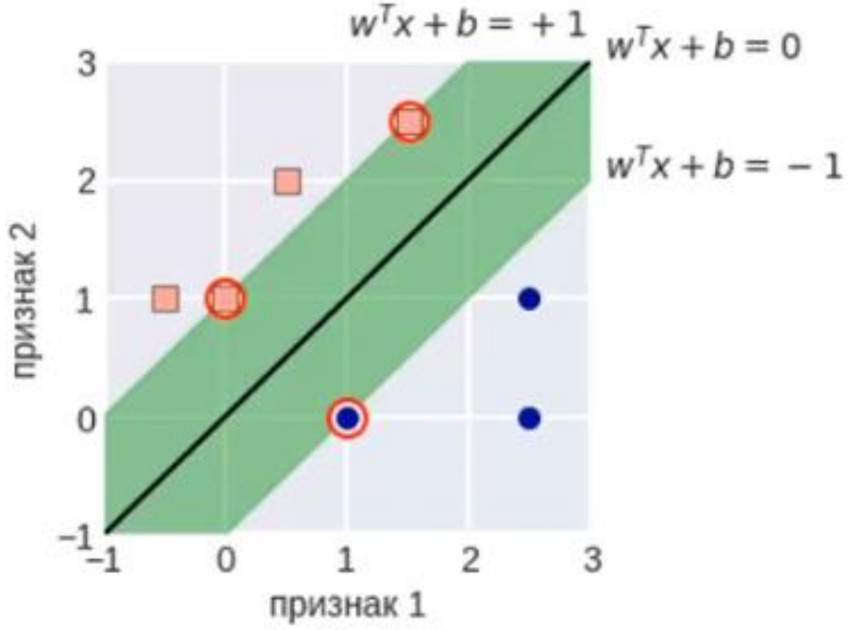
«квадрат» – для удобства оптимизации

Заметим, что решение существует и единственно

Есть много солверов...

SVM: постановка задачи

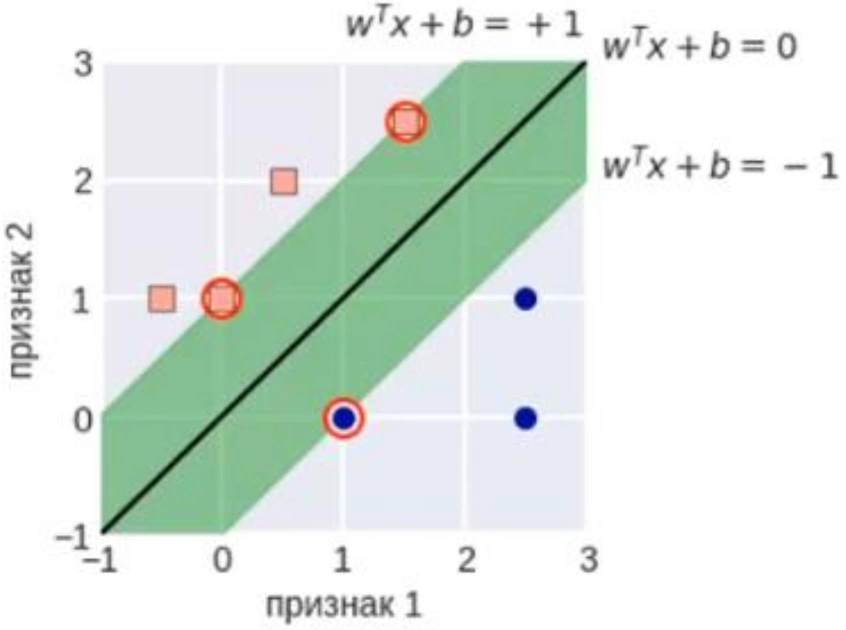
Ширина зеленой полосы?



$$y_i(w^{\mathrm{T}} x_i + b) \geq 1$$

$$\min_i |w^{\mathrm{T}} x_i + b| = 1$$

SVM: постановка задачи

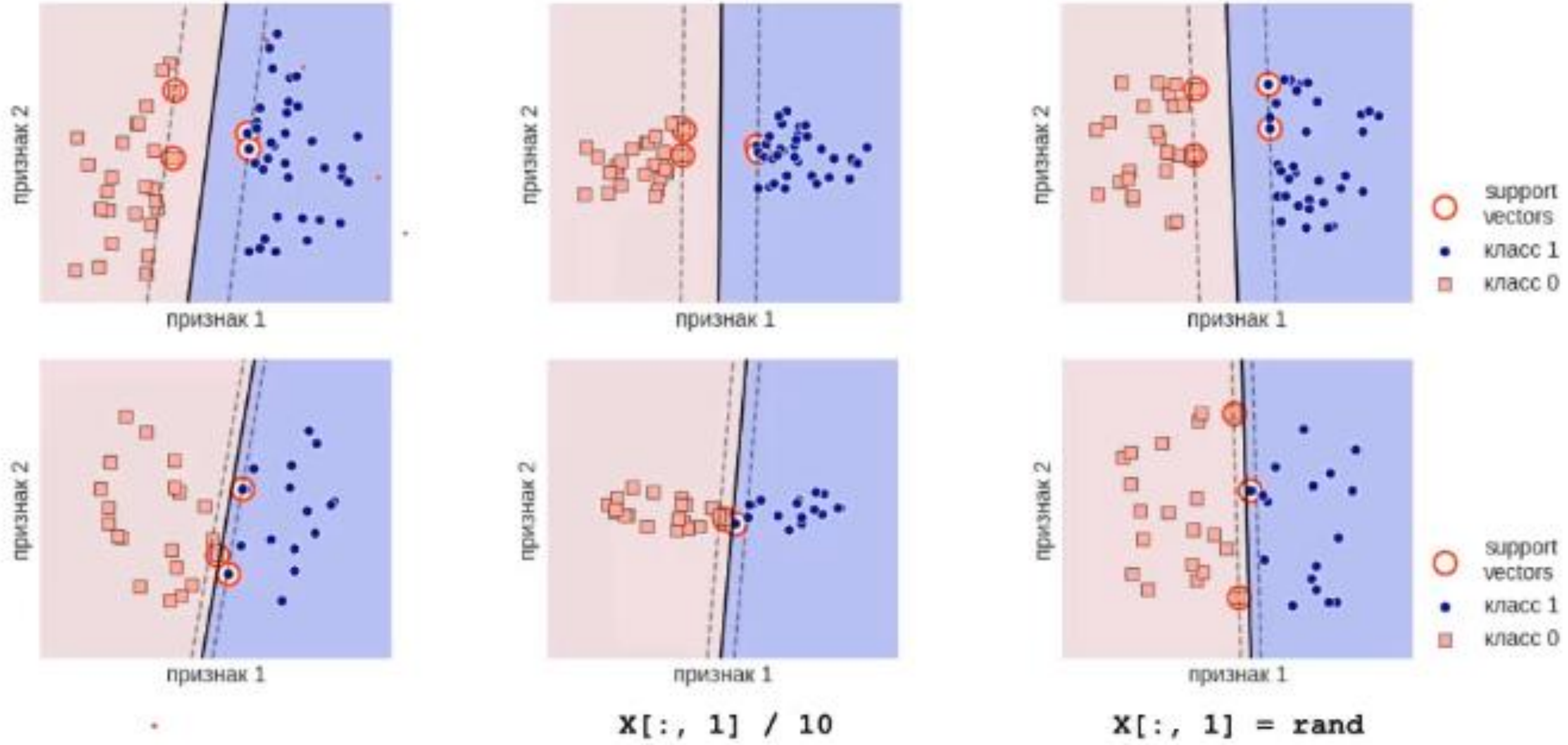


$$width\left(\{x_i\}_{i=\overline{1,n}}, \{x \in \alpha : w^T x + b = 0\}\right) = 2\rho\left(x^*, \{x \in \alpha : w^T x + b = 0\}\right)$$

$$= 2\min_i \frac{|w^T x_i + b|}{\|w\|}$$

Чувствительность к масштабу и шумам

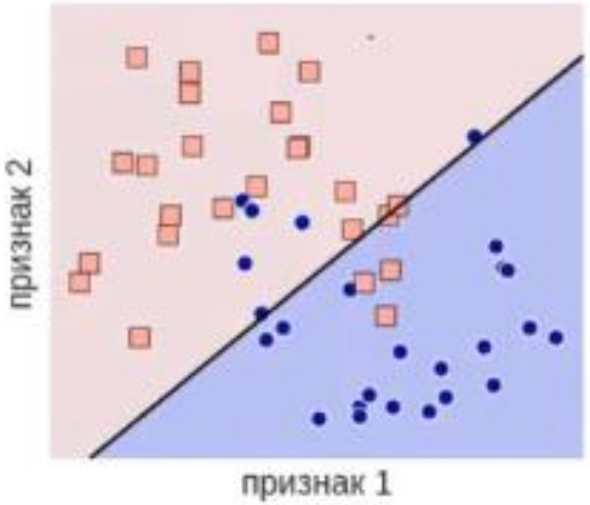
Укажите ошибки на изображениях!



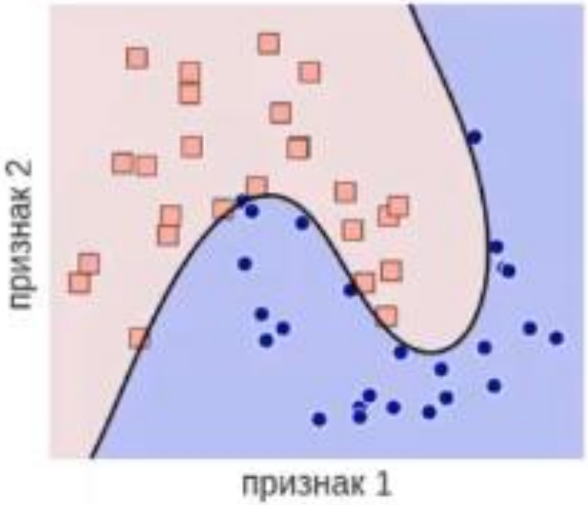
Если нет линейной разделимости

Два подхода (часто используются вместе)

1. Разделять так, чтобы ошибок было мало



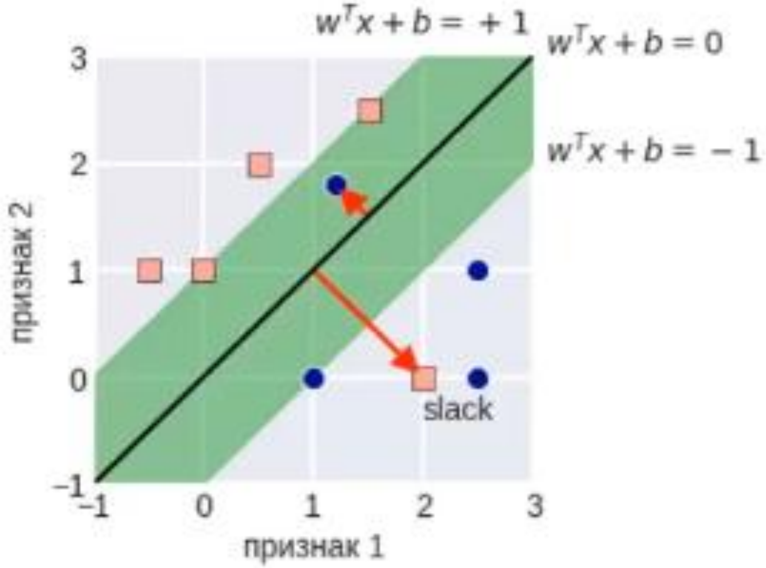
2. Использование нелинейных разделяющих поверхностей



Переход в другое признаковое пространство

Soft-Margin SVM

Разделение допуская ошибки



Позволить объектам «залезать»
в полупространство другого класса

$$y_i(w^{\mathrm{T}}x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

но не хотим, чтобы было много больших «залезаний»:
штрафующие переменные (slack variables)

Soft-Margin SVM: разделение допуская ошибки

Прямая задача:

$$\frac{\|w\|^2}{2} + C\sum_{i=1}^{m}\xi_i \to \min$$

$$y_i(w^\top x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i \in \{1, 2, \ldots, m\}$$

тоже задача QP, но в два раза больше ограничений

Можно было бы рассматривать задачу с таким ограничением $\sum_{i=1}^{m}\xi_i \leq C$

Понятно, что можно было бы и $+C\sum_{i=1}^{m}|\xi_i|^d$

C – **баланс между оптимизацией зазора и ошибки на обучении**

Если $C = 0$, то получим решение $w = \tilde{0}$

Если $C \to +\infty$, то получим решение как в «hard-margin objective»

Soft-Margin SVM: численное решение

$$\frac{\|w\|^2}{2} + C\sum_{i=1}^{m}\xi_i \to \min$$

$$y_i(w^{\mathrm{T}}x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i \in \{1,2,\ldots,m\}$$

Преобразуем:

$$\begin{cases}\xi_i \geq 1 - y_i(w^{\mathrm{T}}x_i + b)\\ \xi_i \geq 0\end{cases} \Leftrightarrow \quad \xi_i \geq \max[1 - y_i(w^{\mathrm{T}}x_i + b), 0]$$

т.к. минимизируем сумму берём минимально возможное:

$$\xi_i = \max[1 - y_i(w^{\mathrm{T}}x_i + b), 0]$$

$$\frac{\|w\|^2}{2} + C\sum_{i=1}^{m}\xi_i \to \min$$

Soft-Margin SVM: численное решение

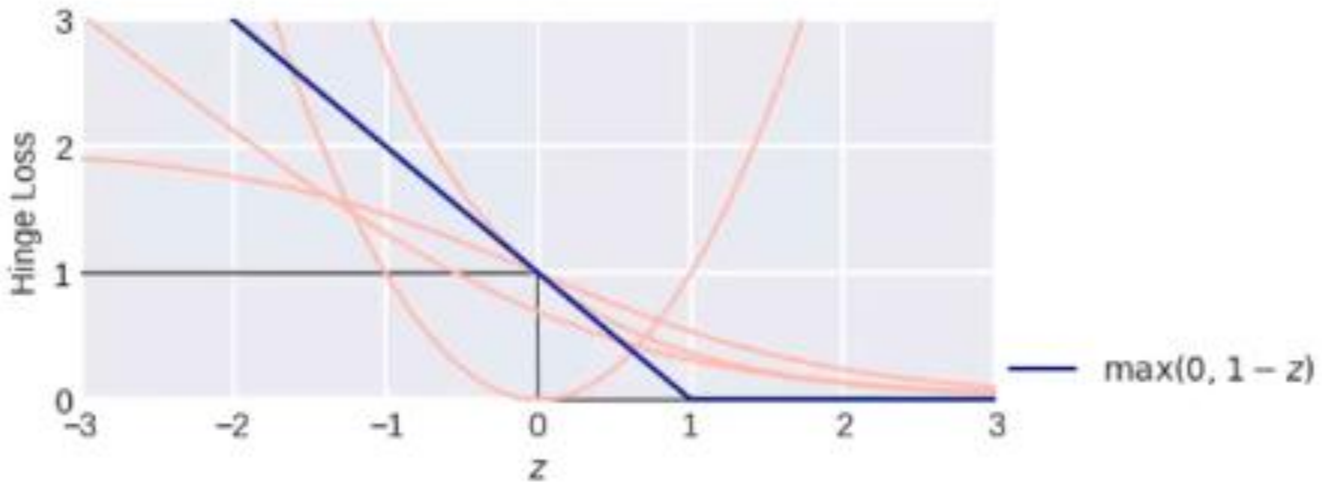
Поставленная задача свелась к ...

$$\underbrace{\frac{\|w\|^2}{2}}_{L_2-\text{регуляризация}} + C\underbrace{\sum_{i=1}^{m}\max[1-y_i(w^\mathrm{T}x_i+b),0]}_{\text{Hinge Loss}} \to \min$$

Получили уже знакомое:
Hinge Loss + L2-регуляризация,
но тут нет дифференцируемости
(в каждой точке) из-за max

Также видно, что если $1-y_i(w^\mathrm{T}x_i+b)\le 0$
т.е. Зазор ≥ 1 (достаточно большой),
то объект не влияет на решение решение
зависит только от опорных объектов

Hinge loss



А в логистической регрессии: логистическая функция ошибки + регуляризатор

Понятно, почему в реализации SVM не коэффициент регуляризации,

а (1 / коэф. регул.)

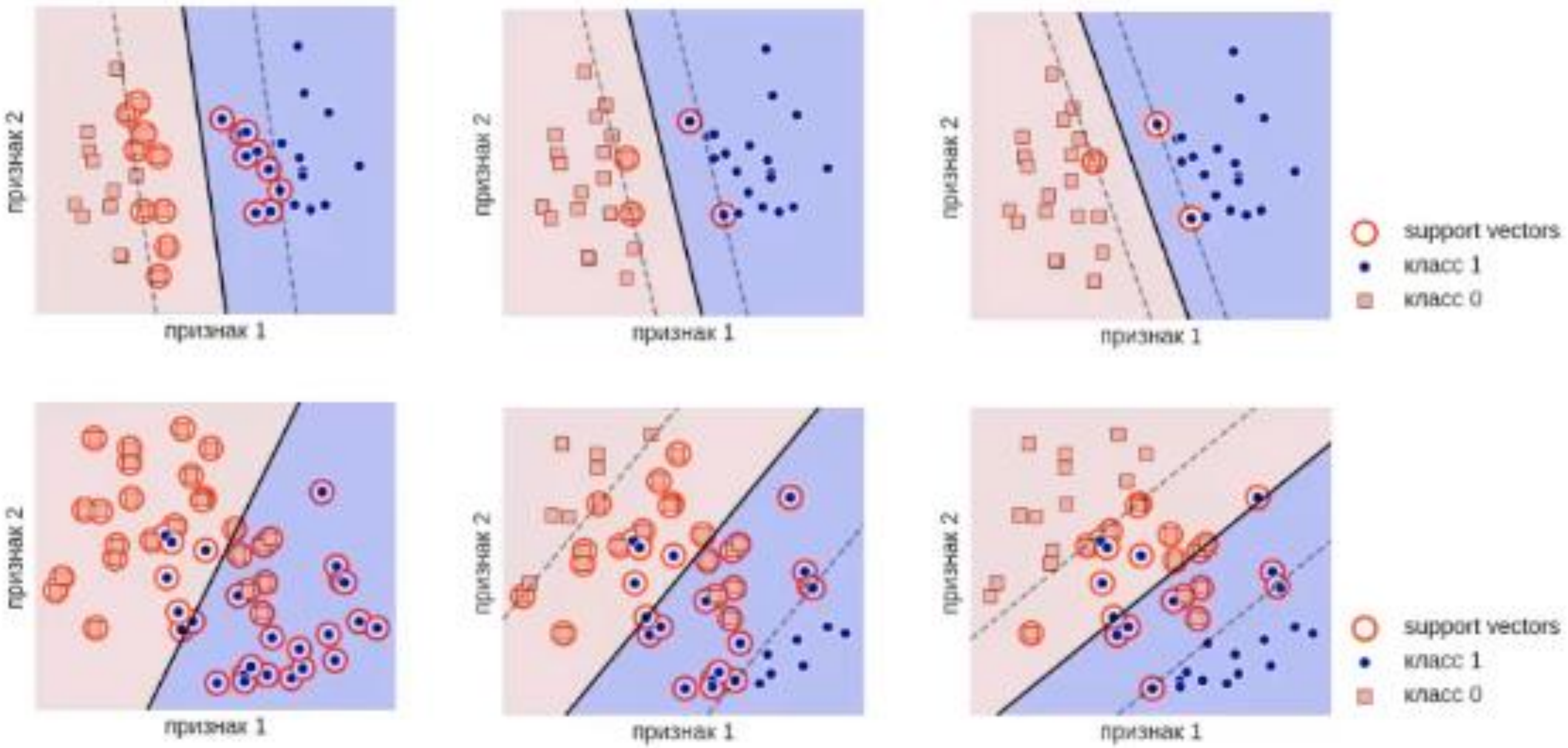
$$\sum_{i=1}^{m} \text{hinge}(y_i, w^{\mathrm{T}} x_i + b) + \frac{1}{2C} \| w \|^2 \to \min$$

Реализация в scikit-learn

```
from sklearn.svm import LinearSVC
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

clf = make_pipeline(StandardScaler(),
                    LinearSVC(penalty='l2', # Тип регуляризации
                    loss='squared_hinge',   # Ошибка регуляризации, более
                                            # традиционная "hinge"
                    dual='warn', # Какую задачу решать,
                                 # dual=False когда n_samples > n_features
                    C=1.0, # Обратная величина к коэффициенту регуляризации
                    multi_class='ovr',      #  = one-vs-rest
                    fit_intercept=True,     # Свободный член
                    intercept_scaling=1)    # Значение фиктивного признака
clf.fit(X, y)
```

Пример



$C = 0.01$ $C = 0.1$ $C = 1.0$

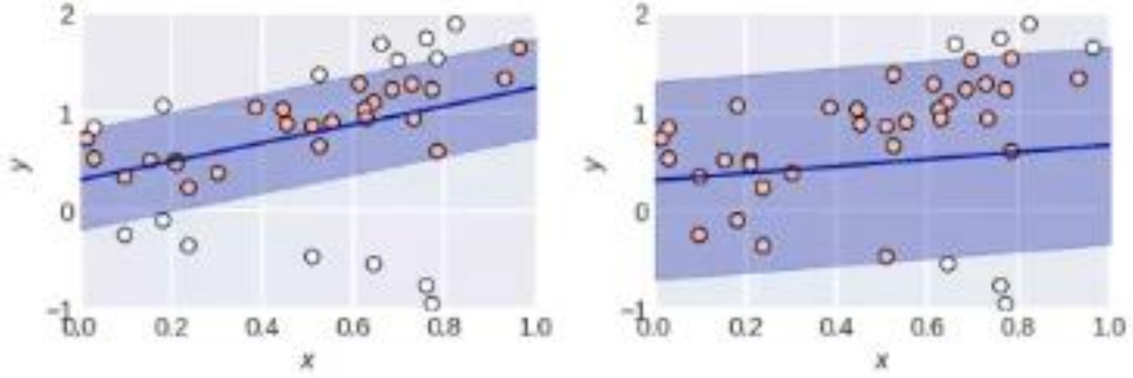
здесь опорными отмечены те объекты, которые пометила функция в sklearn

SVM Regression

Хотим использовать регуляризацию и как можно лучше подстраиваться с ε-точностью:

$$\frac{\|w\|^2}{2} \to \min$$

$$|w^{\mathrm{T}} x_i + b - y_i| \le \varepsilon, \ \ i \in \{1, 2, \ldots, m\}$$



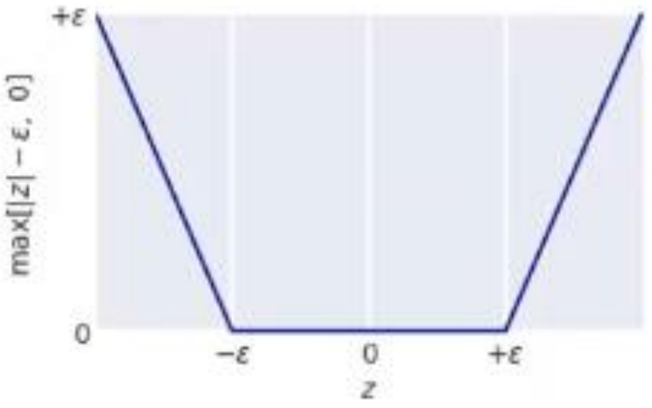
Выполнение неравенства – попадание точки в полосу

Сейчас формализуем – что мы хотим «от попадания»

SVM Regression

$$\frac{\| w \|^2}{2} \to \min$$

$$| w^{\mathrm{T}} x_i + b - y_i | \le \varepsilon, \quad i \in \{1, 2, \ldots, m\}$$

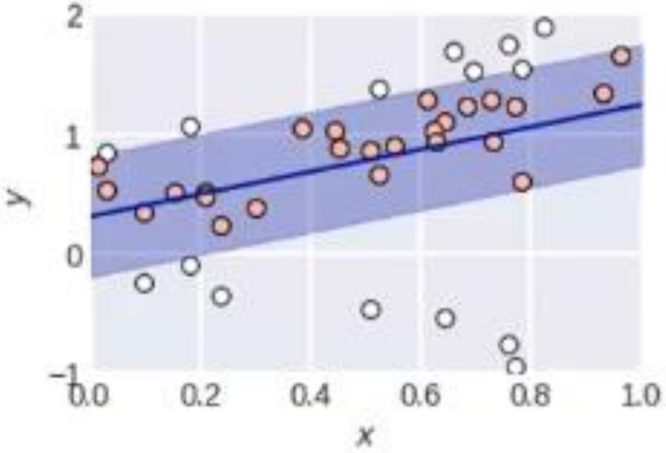


Записываем такую функцию ошибки:

$$\frac{\| w \|^2}{2} + C \sum_{i=1}^{m} \max[| w^{\mathrm{T}} x_i + b - y_i | - \varepsilon, 0] \to \min$$

Решение будет зависеть от объектов,
для которых ошибка превышает порог...

После замены переменных задача сводится к QP



SVM

Естественное определение «оптимальной разделяющей гиперплоскости»

+ геометрическая интерпретация

+ некоторая защита от проклятия размерности

Решение определяется «опорными объектами»

их сложнее всего классифицировать
– только их можно оставить в выборке

Есть теоретическое обоснование (не всё рассказали)

большой зазор → меньше переобучение

Есть нелинейные модификации «kernel tricks»

это, вообще говоря, не линейный метод!

При оптимизации нет проблем локальных минимумов

Линейные скоринговые модели в задаче бинарной классификации

Пусть $X = \mathbb{R}^n$, $Y = \{0,1\}$

Как решать задачи классификации с помощью линейной модели: будем получать вероятность принадлежности к классу 1

$$a(x) \in [0, 1]$$

Любая линейная функция на $\mathbb{R}^n$ будет получать значения в $\mathbb{R}$, поэтому нужна деформация (transfer function):

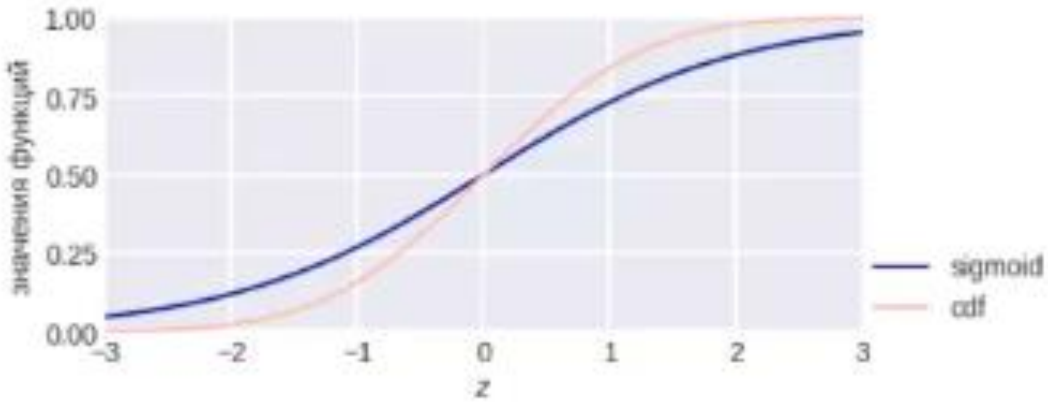
$$\sigma: \mathbb{R} \to [0, 1]$$

Решаем задачу так:

$$\sigma(w^{\mathrm{T}} x)$$

проекция на одномерное пространство и деформация

Функции деформации



В логистической регрессии

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

Логистическая функция (сигмоида)

В Probit-регрессии

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp(-t^2/2)\partial t$$

Normal Cumulative distribution function

Как получается логистическая функция (сигмоида)?

$$Y=0: X \sim N(\mu_0,\sigma^2),\ Y=1: X \sim N(\mu_1,\sigma^2),$$

$$P(Y=0)=p_0, \qquad P(Y=1)=p_1=1-p_0,$$

$$P(Y=1\,|\,X=x)=\frac{P(X=x\,|\,Y=1)P(Y=1)}{P(X=x\,|\,Y=1)P(Y=1)+P(X=x\,|\,Y=0)P(Y=0)}=$$

$$=\frac{1}{1+\dfrac{P(X=x\,|\,Y=0)P(Y=0)}{P(X=x\,|\,Y=1)P(Y=1)}}$$

Как получается логистическая функция (сигмоида)?

$$\ln\left(\frac{P(X=x\,|\,Y=1)P(Y=1)}{P(X=x\,|\,Y=0)P(Y=0)}\right)=-\ln\left(\frac{P(X=x\,|\,Y=0)P(Y=0)}{P(X=x\,|\,Y=1)P(Y=1)}\right)=\left\{P(X=x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}\right\}=$$

$$=-\ln\left(\frac{p_0}{p_1}e^{-\frac{(x-\mu_0)^2}{2\sigma^2}+\frac{(x-\mu_1)^2}{2\sigma^2}}\right)=$$

$$=\ln\left(\frac{p_1}{p_0}\right)+\left(-\frac{(x-\mu_1)^2}{2\sigma^2}+\frac{(x-\mu_0)^2}{2\sigma^2}\right)=\ln\left(\frac{p_1}{p_0}\right)+\frac{1}{2\sigma^2}\left(-x^2+2x\mu_1-\mu_1^2+x^2-2x\mu_0+\mu_0^2\right)=$$

$$=\ln\left(\frac{p_1}{p_0}\right)+\frac{1}{2\sigma^2}\left(2x(\mu_1-\mu_0)+\mu_0^2-\mu_1^2\right)=\ln\left(\frac{p_1}{p_0}\right)+\frac{x(\mu_1-\mu_0)}{\sigma^2}+\frac{\mu_0^2-\mu_1^2}{2\sigma^2}=$$

$$=\ln\left(\frac{P(X=x\,|\,Y=1)P(Y=1)}{P(X=x\,|\,Y=0)P(Y=0)}\right)=\ln\left(\frac{p_1}{p_0}\right)+\frac{\mu_0^2-\mu_1^2}{2\sigma^2}+\frac{(\mu_1-\mu_0)}{\sigma^2}x=\alpha+\beta x$$

Как получается логистическая функция (сигмоида)?

$$P(Y=1\mid X=x)=\frac{1}{1+\dfrac{P(X=x\mid Y=0)P(Y=0)}{P(X=x\mid Y=1)P(Y=1)}}=\frac{1}{1+e^{-\ln\left(\frac{P(X=x|Y=1)P(Y=1)}{P(X=x|Y=0)P(Y=0)}\right)}}=\frac{1}{1+e^{-(\alpha+\beta x)}}$$

$$P(Y=1\mid X=x)=\frac{1}{1+e^{-\underbrace{\ln\left(\frac{P(X=x|Y=1)P(Y=1)}{P(X=x|Y=0)P(Y=0)}\right)}_{=z}}}=\{z=\alpha+\beta x\}=\frac{1}{1+e^{-z}}=\sigma(z)-\textit{вероятность...}$$

$z-$"*логит* $-$ *функция*"

Свойства сигмоиды:

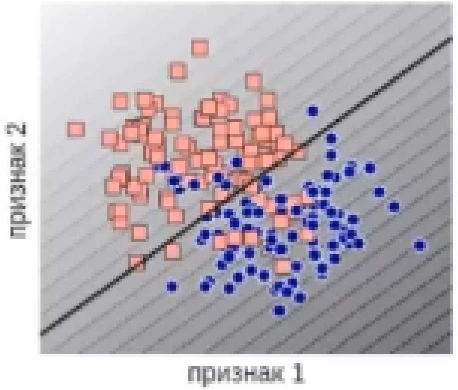
1) $\sigma(-z)=1-\sigma(z)$,

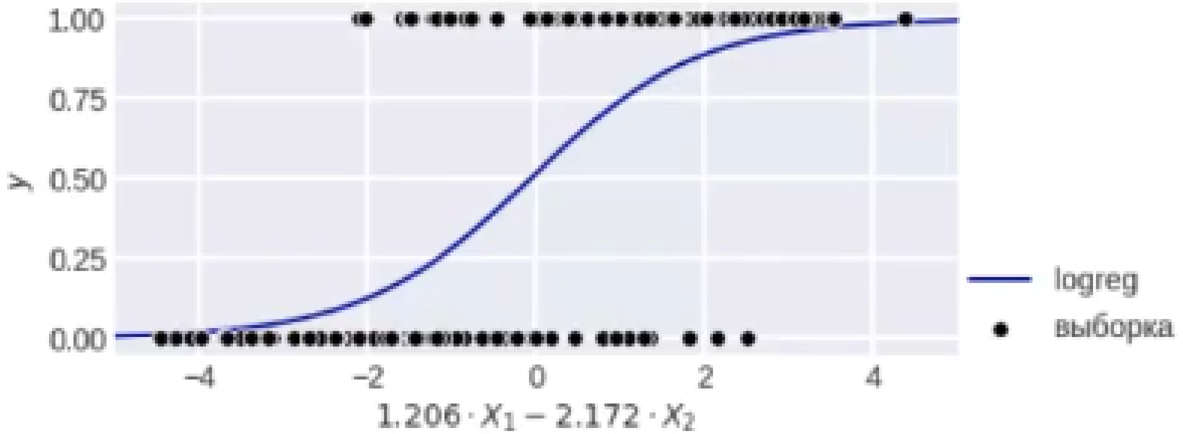
2) $\sigma'(z)=\sigma(z)(1-\sigma(z))$,

3) $z=\ln\left(\dfrac{\sigma}{1-\sigma}\right)$ – монотонное преобразование, которое называют logit-transformation

Геометрический смысл логистической регрессии

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X, y)
a = model.predict_proba(X_test)[:,1]
```





$z = w_0 + w_1 X_1 + \ldots + w_n X_n$ – проекция на прямую (один признак)

в однопризнаковом случае надо решить задачу классификации

Обучение логистической регрессии

Метод максимального правдоподобия

$$L(w_0,\dots,w_n) = \prod_{i:y_i=1} \sigma(w^{\mathrm{T}} x_i) \prod_{i:y_i=0} (1-\sigma(w^{\mathrm{T}} x_i)) \to \max$$

$a_i = a(x_i \mid w)$ – ответ алгоритма, зависящего от параметров w, на $\boldsymbol{i}$-м объекте

$$p(y \mid X, w) = \prod_i p(y_i \mid x_i, w) = \prod_i a_i^{y_i} (1-a_i)^{1-y_i} \to \max$$

здесь тоже выпуклая задача оптимизации

логирифмируем....

$$\log L(w_0,\dots,w_n) = \sum_{i:y_i=1} \log(\sigma(w^{\mathrm{T}} x_i)) + \sum_{i:y_i=0} \log(\sigma(-w^{\mathrm{T}} x_i)) \to \max$$

для удобства записи $y_i' = 2y_i - 1$, тогда

$$\log L = \sum_i \log(\sigma(y_i' w^{\mathrm{T}} x_i)) = -\sum_i \log(1+\exp(-y_i' w^{\mathrm{T}} x_i)) \to \max$$

Обучение логистической регрессии

$$\log L = -\sum_i \log(1+\exp(-y_i' w^{\mathrm{T}} x_i)) \to \max$$

Полученное выражение называют «логистической функцией ошибки» (logistic loss)

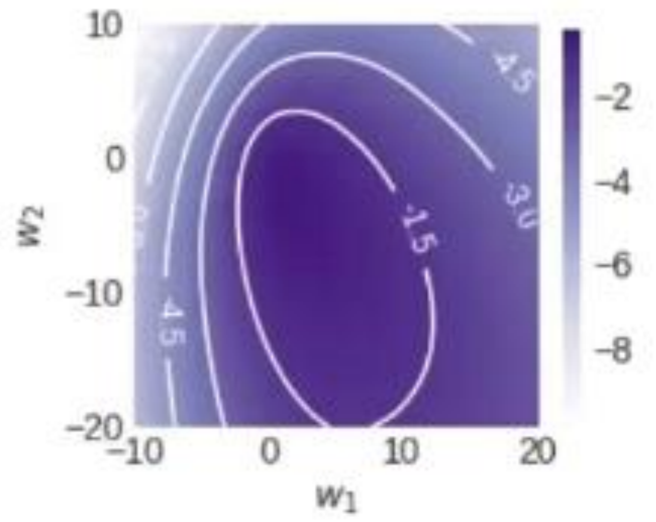
Вычислим градиент

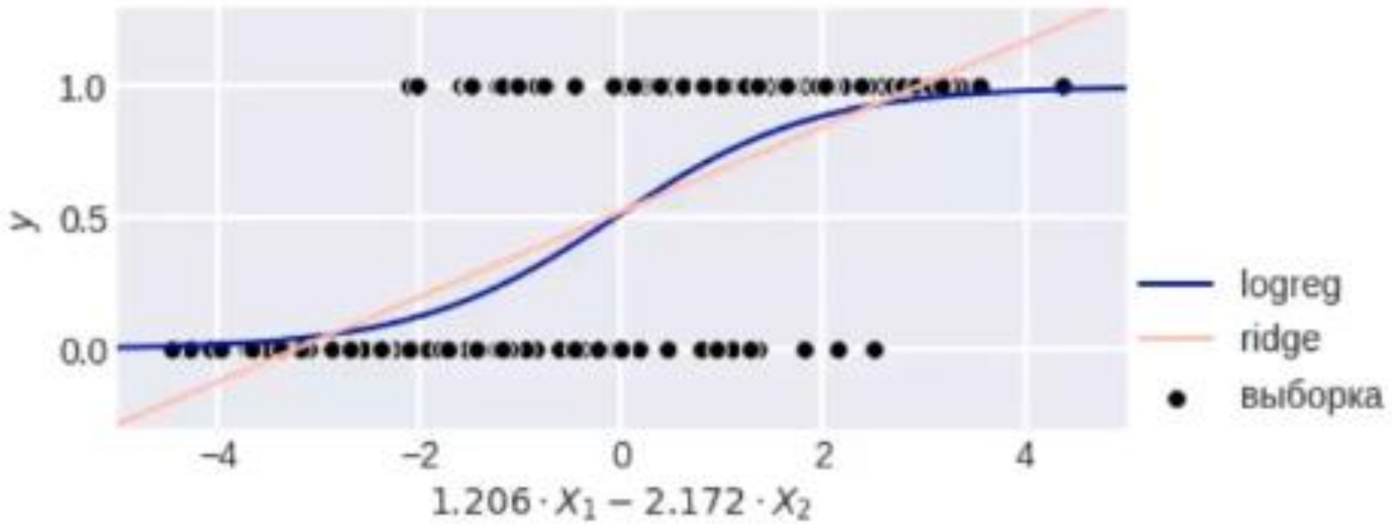
$$\nabla_w \log L = -\sum_i \nabla_w \log(1+\exp(-y_i' w^{\mathrm{T}} x_i)) =$$

$$= -\sum_i \frac{\exp(-y_i' w^{\mathrm{T}} x_i)}{1+\exp(-y_i' w^{\mathrm{T}} x_i)}(-y_i' x_i) = \sum_i \frac{y_i' x_i}{1+\exp(+y_i' w^{\mathrm{T}} x_i)} = \sum_i \sigma(-y_i' w^{\mathrm{T}} x_i) y_i' x_i$$

Качество логистической регрессии

– логарифм правдоподобия (потом будет соответствующая функция ошибки logloss)





метод SGD (запомним): $w \leftarrow w + \eta\sigma(-y_i' w^{\mathrm{T}} x_i) y_i' x_i$

Многоклассовая логистическая регрессия

Multiclass logreg / multinomial regression

Для j-го класса имеем свою функцию $w_j^{\mathrm{T}} x$
хотим такие значения превращать в распределения

$$(w_1^{\mathrm{T}} x, w_2^{\mathrm{T}} x, \ldots, w_l^{\mathrm{T}} x) \in \mathbb{R}^l \to (p_1(x), p_2(x), \ldots, p_l(x)) \in [0,1]^l_{\Sigma=1}$$

в glmnet такой «симметричный вариант»: $P(Y = k \mid x) = \dfrac{\exp(w_{0k} + w_{1k} X_1 + \ldots + w_{nk} X_n)}{\sum_{j=1}^{l} \exp(w_{0j} + w_{1j} X_1 + \ldots + w_{nj} X_n)}$

Такая функция называется softmax:

$$\mathrm{softmax}(a_1, \ldots, a_l) = \frac{1}{\exp(a_1) + \ldots + \exp(a_l)} [\exp(a_1), \ldots, \exp(a_l)]$$

Реализация в scikit-learn

```python
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(penalty='l2',
                         dual=False,
                         tol=0.0001,
                         C=1.0,
                         fit_intercept=True,
                         intercept_scaling=1,
                         class_weight=None,
                         random_state=None,
                         solver='lbfgs',
                         max_iter=100,
                         multi_class='auto',
                         verbose=0,
                         warm_start=False,
                         n_jobs=None,
                         l1_ratio=None) # если penalty='elasticnet'
clf.fit(X, y)
clf.predict_proba(X)
```

Приложения: банковский скоринг

Банковский скоринг

По описанию и истории клиента → вероятность (оценка) возврата кредита

Нужна логистическая регрессия

- есть возможность получать вещественное число в виде ответа
- есть более мощные методы (на решающих деревьях), но здесь полезна интерпретация

Все категориальные признаки – OHE-перекодировка

Банковский скоринг

Если решение сводится к $a(x) = 1/(1+\exp(-(w_0 + w_1X_1 + \ldots + w_nX_n))$

где все признаки бинарные, то мы составляем скоринговую карту

Показатель	Значение показателя	Скоринг-балл
Возраст	До 30 лет	0
	От 30 до 50 лет	35
	Старше 50 лет	28
Образование	Среднее	0
	Среднее специальное	29
	Высшее	35
Состоит ли в браке	Да	25
	Нет	0
Брал ли кредит ранее	Да	41
	Нет	0
Трудовой стаж	Менее 1 года	0
	От 1 до 5 лет	19
	От 5 до 10 лет	24
	Более 10 лет	31

wiki.loginom.ru/articles/scorecard

Спасибо за внимание!

Запорожцев Иван Федорович
zaporozhtsev.if.work@gmail.com