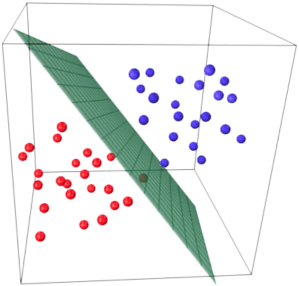
Линейная классификация

Виктор Китов

victorkitov.github.io



Курс поддержан фондом 'Интеллект'

Победитель конкурса VK среди курсов по IT

Содержание

Многоклассовый классификатор

- Определим дискриминантные функции $g_c(x)$ для классов $c = \overline{1, C}$.
- Классификация - предсказывается класс с максимальным рейтингом:
$$\widehat{y}(x) = \arg\max_c g_c(x)$$
- Граница между классами i и j:
$$\{x : g_i(x) = g_j(x)\}$$
- Отступ измеряет качество классификации:
$$M(x, y) = g_y(x) - \max_{c \neq y} g_c(x)$$

Бинарный классификатор

- $y \in \{+1, -1\}$, поэтому есть только $g_{+1}(x)$ и $g_{-1}(x)$.
- Предпочтительность класса $+1$ относительно класса -1:

$$g(x) = g_{+1}(x) - g_{-1}(x)$$

- Бинарный классификатор:

$$\widehat{y}(x) = \arg\max_{c \in \{+1,-1\}} g_c(x) = \operatorname{sign}\left(g_{+1}(x) - g_{-1}(x)\right) = \operatorname{sign}\left(g(x)\right)$$

- Отступ:

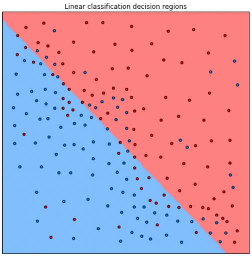
$$M(x, y) = g_y(x) - g_{-y}(x) = y\left(g_{+1}(x) - g_{-1}(x)\right) = yg(x)$$

Линейный классификатор

- Включим константу 1 в признаки для учета смещения: $x = [1, x^2, x^3, ...x^D]$.
- Линейный классификатор - все его дискриминантные функции линейны:

$$g_c(x) = w_c^T x, \quad c = \overline{1, C}.$$

- Граница между классами i и j линейна: $\left\{x : w_i^T x = w_j^T x\right\}$



Бинарный линейный классификатор

- Дискриминантные функции: $w_{+1}^T x, w_{-1}^T x$.
- Определим $w = w_{+1} - w_{-1}$.
- Бинарный линейный классификатор:

$$\widehat{y}(x) = \underset{c \in \{+1, -1\}}{\arg\max}\; w_c^T x = \operatorname{sign}\left(w_{+1}^T x - w_{-1}^T x\right) = \operatorname{sign}\left(w^T x\right)$$

- Отступ:

$$M(x, y) = w_y^T x - w_{-y}^T x = y\left(w_{+1}^T x - w_{-1}^T x\right) = y w^T x = w^T x y$$

Содержание

Вектор, ортогональный гиперплоскости

Теорема 1

Вектор w ортогонален гиперплоскости $w^T x + w_0 = 0$

Доказательство. Рассмотрим произвольные $x_A, x_B \in \{x : \; w^T x + w_0 = 0\}$:

$$w^T x_A + w_0 = 0 \tag{1}$$

$$w^T x_B + w_0 = 0 \tag{2}$$

$\square$

Вычитая (2) из (1), получим $w^T (x_A - x_B) = 0$, поэтому w ортогонален гиперплоскости.

Расстояние от точки до гиперплоскости

Теорема 2

Расстояние от точки x до гиперплоскости $w^T x + w_0 = 0$ равно $\frac{w^T x + w_0}{\|w_0\|}$.

Доказательство. Пусть p - проекция x на гиперплоскость, а $h = x - p$ - ортогональное дополнение. Тогда

$$x = p + h$$

Поскольку p лежит на гиперплоскости, то

$$w^T p + w_0 = 0$$

Поскольку h ортогонально гиперплоскости по теореме 1, то

$$h = r\frac{w}{\|w\|},\ r \in \mathbb{R} \text{ - расстояние до гиперплоскости.}$$

□

Расстояние от точки до гиперплоскости

$$x = p + r\frac{w}{\|w\|}$$

После домножения равенства на w и прибавления w_0:

$$w^T x + w_0 = w^T p + w_0 + r\frac{w^T w}{\|w\|} = r\|w\|,$$

поскольку $w^T p + w_0 = 0$ и $\|w\| = \sqrt{w^T w}$. В итоге получаем

$$r = \frac{w^T x + w_0}{\|w\|}$$

Комментарии:

- С одной стороны гиперплоскости $r > 0 \Leftrightarrow w^T x + w_0 > 0$
- С другой стороны гиперплоскости $r < 0 \Leftrightarrow w^T x + w_0 < 0$.
- Расстояние от начала координат до гиперплоскости $\frac{w_0}{\|w\|}$. Поэтому w_0 отвечает за смещение.

Бинарный линейный классификатор - интерпретация

- Бинарный линейный классификатор:

$$\widehat{y}(x) = \text{sign}\left(w^T x + w_0\right)$$

разделяет классы гиперплоскостью $w^T x + w_0 = 0$.

- Т.к. расстояние до границы равно $\frac{\left|w^T x + w_0\right|}{\|w\|}$, то $\left|w^T x + w_0\right| \in [0, +\infty)$ - уверенность классификации.
  - связана с вероятностью класса
- Качество классификации, с учетом верного y:

$$M(x, y) = y\left(w^T x + w_0\right)$$

Содержание

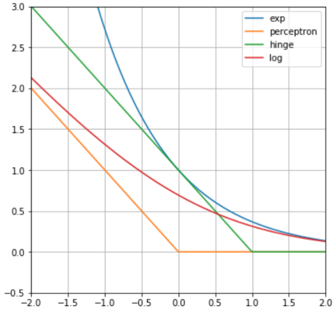
Оценка вектора весов w

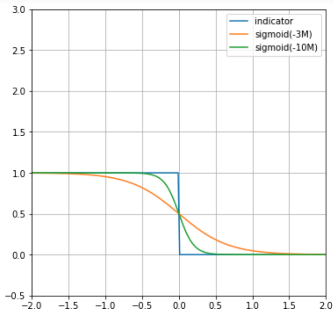
- Прямой подход: выберем w, чтобы минимизировать #ошибок:
$$\sum_{n=1}^{N} \mathbb{I}\left[w^T x_n y_n < 0\right] \to \min_w$$
- Получили кусочно-постоянный критерий, градиент=0 почти везде.
- Для получения невырожденных градиентов, применим убывающую $\mathcal{L}(\cdot)$ к отступу:
    - минимизация нового критерия аппроксимирует минимизацию старого.
$$\sum_{n=1}^{N} \mathcal{L}\left(M(x_n, y_n)\right) = \sum_{n=1}^{N} \mathcal{L}\left(w^T x_n y_n\right) \to \min_w$$

Популярные функции потерь

$$\mathcal{L}_{exp}(M) = e^{-M} \quad \mathcal{L}_{perceptron}(M) = [-M]_+$$

$$\mathcal{L}_{hinge}(M) = [1 - M]_+ \quad \mathcal{L}_{log}(M) = \log_2\left(1 + e^{-M}\right)$$





Какие из них будут выпуклыми? устойчивыми к выбросам? улучшать даже безошибочный классификатор?

Учёт матрицы цен

Если определена матрица цен (cost matrix)

$$\lambda(i,j) = \text{cost}\left\{y = i, \widehat{y} = j\right\},$$

то выгоднее оптимизировать не

$$\frac{1}{N}\sum_{n=1}^{N}\mathcal{L}\left(\widehat{y}_w\left(x_n\right), y_n\right) \to \min_w,$$

а взвешенные потери по цене

$$\frac{1}{N}\sum_{n=1}^{N}\lambda\left(\widehat{y}_w\left(x_n\right), y_n\right)\mathcal{L}\left(\widehat{y}_w\left(x_n\right), y_n\right) \to \min_w$$

Содержание

Регуляризация

- Хотим не только точную, но и простую модель.
    - простые модели обладают лучшей обобщающей способностью
    - измеряем сложность регуляризатором $R(w)$

$$\sum_{n=1}^{N}\mathcal{L}\left(M(x_n, y_n|w\right) + \lambda R(w) \to \min_{w}$$

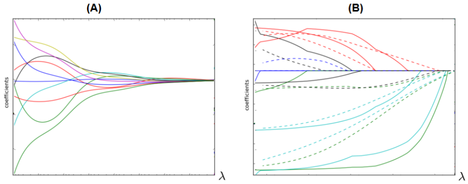
- $\lambda > 0$ - гиперпараметр[1] (насколько простота важнее точности).
- Популярные варианты $R(w)$:

$R(\beta) = \|\|w\|\|_1$	L_1 регуляризация
$R(\beta) = \|\|w\|\|_2^2$	L_2 регуляризация
$R(\beta) = \alpha \left\|w\right\|_1 + (1-\alpha)\left\|w\right\|_2^2$	ElasticNet $\alpha \in (0, 1)$

[1] Как меняется сложность модели при увеличении λ?

Комментарии

- Зависимость w от λ для L_2 (A) и L_1 (B) регуляризации:



- L_1 может автоматически отбирать признаки.
- λ обычно находится по экспоненциальной шкале $[10^{-6}, 10^{-5}, ...10^5, 10^6]$.
    - можно уточнить по мелкой сетке в окрестности оптимума
- Использование регуляризации позволяет плавно контролировать сложности модели.

ElasticNet

- ElasticNet - линейная комбинация L_1 и L_2 регуляризации:

$$R(\beta) = \alpha||w||_1 + (1-\alpha)||w||_2^2 \to \min_w$$

$$\alpha \in [0,1] - \text{гиперпараметр.}$$

- Если два признака x^i и x^j равны:
    - Гребневая регрессия выберет оба с равным весом
        - правильно, т.к. нет априорных предпочтений
    - Лассо регрессия выберет один из них (в общем случае)
        - зато отберет лишние признаки
- ElasticNet обладает обоими преимуществами.

Учет разных признаков с разной силой

- Прогнозы обычного линейного классификатора инвариантны к масштабированию признаков:

$$g(x)=\widehat{w}_1 x^1+\widehat{w}_2 x^2+\ldots \xrightarrow{x^1 \to x^1/\alpha} (\alpha \widehat{w}_1)\left(\frac{x^1}{\alpha}\right)+\widehat{w}_2 x^2+\ldots$$

- Но не регуляризованного:

$$\sum_{n=1}^{N} \mathcal{L}\left(M(x_n, y_n|w)\right)+\lambda R(w) \to \min_{w}$$

- После изменения масштаба признаков, они будут вносить другой вклад в прогноз.
  - для большего учета признака как нужно изменить его масштаб?

Содержание

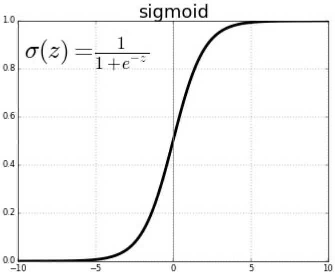
Бинарная классификация[2]

- Предпочтение $y = +1$ относительно $y = -1$ бинарного классификатора:

$$g(x) = w^T x$$

- Предположение *логистической регрессии*:

$$p(y = +1|x) = \sigma(w^T x)$$



[2] Является частным случаем GLM модели (Generalized Linear Model).

Оценка параметров[3]

Свойство сигмоиды:

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

поэтому

$$p(y=+1|x)=\sigma(w^T x)\Longrightarrow p(y=-1|x)=1-p(y=+1|x)=\sigma(-w^T$$

$$p(y|x)=\sigma(y\langle w,x\rangle)\text{ в общем случае.}$$

Оценим w методом условного максимального правдоподобия:

$$ML=P(Y|X)=\prod_{n=1}^{N}p(y_n|x_n)=\prod_{n=1}^{N}\sigma(\langle w,x_n\rangle y_n)\to\max_w$$

[3]Почему $\|w\|\to\infty$ при линейной разделимости классов? Как с этим бороться?

Минимизация эмпирического риска и максимизация правдоподобия

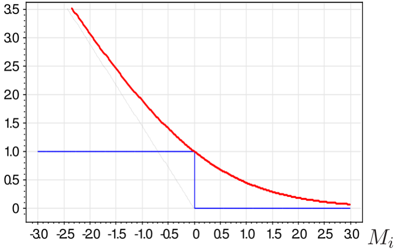
$$\prod_{n=1}^{N}\sigma(\langle w, x_n\rangle y_n) = \prod_{n=1}^{N}\frac{1}{1+e^{-\langle w,x_n\rangle y_n}} \to \max_w$$

$$\prod_{n=1}^{N}\left(1+e^{-\langle w,x_n\rangle y_n}\right) \to \min_w$$

$$\sum_{n=1}^{N}\log_2(1+e^{-\langle w,x_n\rangle y_n}) \to \min_w \quad \text{(прологарифмировали критерий)}$$

Иллюстрация

Обратим внимание, что логистическая ф-ция потерь - сглаженная версия ф-ции потерь персептрона



SGD для логистической регрессии

$$w := w - \varepsilon \nabla_w \mathcal{L}(w^T xy) = w - \varepsilon \frac{\partial \mathcal{L}(M)}{\partial M} \frac{\partial M}{\partial w} = w - \varepsilon \frac{\partial \mathcal{L}(M)}{\partial M} xy$$

$$\mathcal{L}(\mathrm{M}) = [-\mathrm{M}]_+ :$$

$$\frac{\partial \mathcal{L}(M)}{\partial M} = -\mathbb{I}[M < 0]$$

$$w := w + \varepsilon \mathbb{I}[M < 0] xy$$

$$\mathcal{L}(\mathrm{M}) = \log_2(1 + \mathrm{e}^{-\mathrm{M}}) :$$

$$\frac{\partial \mathcal{L}(M)}{\partial M} = \frac{1}{\log_2 e} \frac{-e^{-M}}{(1 + e^{-M})} = \frac{1}{\log_2 e} \frac{-1}{(1 + e^{M})} = -\frac{\sigma(-M)}{\log_2 e}$$

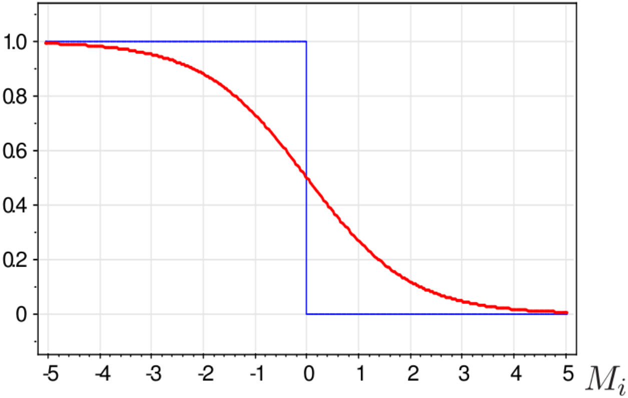
$$w := w + \varepsilon' \sigma(-M) xy$$

Получили сглаженный классификатора персептрона[4].

[4]Может ли $M(x_i, y_i)$ после обновления весов уменьшиться?

Иллюстрация

Иллюстрация:



Использование логистической регрессии

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import brier_score_loss
from sklearn.metrics import accuracy_score

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
model = LogisticRegression(C=1, penalty='l2') # иниц-
    ция
model.fit(X_train, Y_train)      # обучение модели
Y_hat = model.predict(X_test)    # построение прогнозов
print(f'Точность прогнозов: \
{100*accuracy_score(Y_test, Y_hat):.1f}%')
P_hat = model.predict_proba(X_test)  # вер-ти классов
loss = brier_score_loss(Y_test, P_hat[:,1])
print(f'Мера Бриера ошибки вероятностей: {loss:.2f}')
```

- $1/C$ - вес при регуляризаторе.
- Больше информации. Полный код.

Многоклассовая логистическая регрессия

Многоклассовая классификация:

$$\begin{array}{ll} y=1: & g_1(x)=w_1^T x \\ y=2: & g_2(x)=w_2^T x \\ \cdots & \cdots \\ y=C: & g_C(x)=w_C^T x \end{array}$$

Логистическая регрессия предполагает связь $g_1(x), ...g_C(x)$ и вероятностей классов через softmax преобразование:

$$p(y=c|x)=softmax(w_c^T x|x_1^T x, ...x_C^T x)=\frac{exp(w_c^T x)}{\sum_{i=1}^C exp(w_i^T x)}$$

SoftMax

- SoftMax преобразует уверенности исходов в их вероятности:

$$Softmax_\tau(z_1, ...z_K) =$$

$$= \left[\frac{e^{z_1/\tau}}{e^{z_1/\tau} + ... + e^{z_K/\tau}}, \frac{e^{z_2/\tau}}{e^{z_1/\tau} + ... + e^{z_K/\tau}}, ..., \frac{e^{z_K/\tau}}{e^{z_1/\tau} + ... + e^{z_K/\tau}}\right]$$

- τ - параметр температуры, контролирующий контрастность вероятностей[5].

[5] как?

Неоднозначность параметров и их оценка

- Веса $\{w_c\}$ определены с точностью до сдвига v:

$$\frac{exp((w_c - v)^T x)}{\sum_i exp((w_i - v)^T x)} = \frac{exp(-v^T x)exp(w_c^T x)}{\sum_i exp(-v^T x)exp(w_i^T x)} = \frac{exp(w_c^T x)}{\sum_i exp(w_i^T x)}$$

  Чтобы убрать неоднозначность, ограничим $w_C = 0$ (соответствует $v = w_C$)

- Параметры оценим максимизацией условного правдоподобия:

$$\begin{cases} \prod_{n=1}^{N} softmax(w_{y_n}^T x_n | x_1^T x, ... x_C^T x) \to \max_{w_1, ... w_C - 1} \\ w_C = 0 \end{cases}$$

Содержание

Многоклассовая классификация бинарными классификаторами

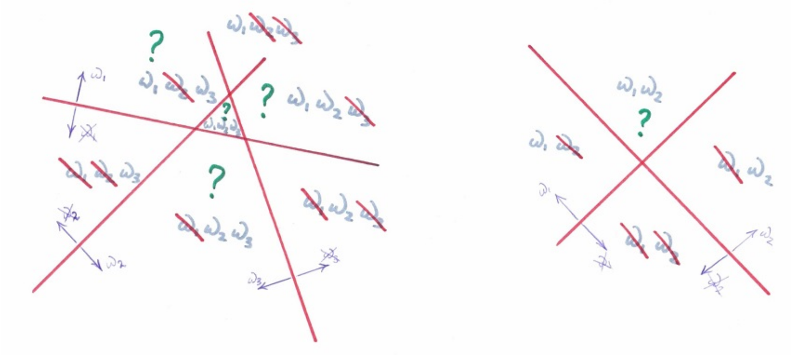
Хотим построить C-классовый классификатор по совокупности бинарных.

Подходы:

- один против всех (one-versus-all)
  - для каждого класса $c = 1, 2, ...C$ обучим бинарный классификатор на $y' = \mathbb{I}[y = c]$,
  - назначим класс, предсказанный с максимальной уверенностью (среди C классификаторов).
- один против одного (one-versus-one)
  - для каждой пары классов $i \neq j \in \{1, 2, ...C\}$ обучим бинарные классификаторы на $(x_n, y_n) : \; y_n \in \{i, j\}$.
    - назначим класс, максимально часто побеждающий среди $C(C-1)/2$ сравнений.
    - при неоднозначности - используем уверенность классификации
- коды, исправляющие ошибки (error correcting codes)

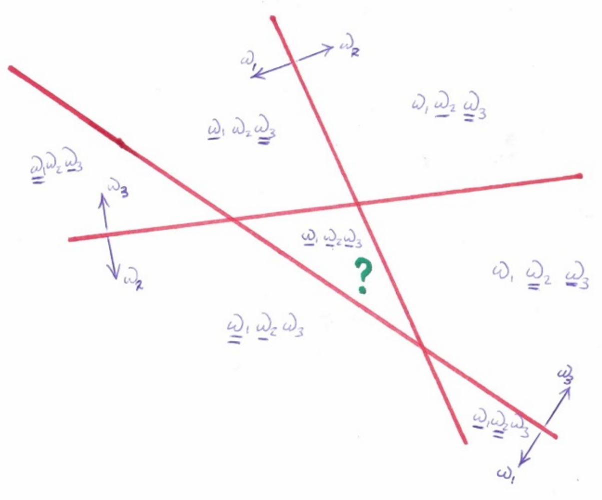
Один против всех - неоднозначность

Классификация среди 3х классов $\omega_1, \omega_2, \omega_3$:



Один против одного - неоднозначность

Классификация среди 3х классов $\omega_1, \omega_2, \omega_3$:



Коды исправляющие ошибки

- Каждый класс i кодируется бинарным представлением W_i из B бит:

$$\text{класс } i \to W_i \in \mathbb{R}^B,\;\; W_{ik} \in \{0,1\},\;\; k = 1, 2, ...B.$$

- Минимально достаточное количество бит для однозначного кодирования C классов $= \lceil \log_2 C \rceil$
- Для заданного x, B бинарных классификаторов $\widehat{y}_1(x), ...\widehat{y}_B(x)$ предсказывают каждый бит.
- Итоговый класс прогнозируется по правилу[6]:

$$\hat{y}(x) = \arg\min_c \rho\left(W_c, [\widehat{y}_1(x), ...\widehat{y}_B(x)]\right)$$

- Обычно, $\rho(\cdot,\cdot)$ считается по L_1 норме.

[6] Какому методу будет соответствовать случай, когда y представляется one-hot кодированием?

Коды исправляющие ошибки

- Используется избыточное количество бит $B \geq \lceil \log_2 C \rceil$ через коды, исправляющие ошибки (error correcting codes)
  - ошибки отдельных классификаторов исправляются другими.
- В качестве $\widehat{y}_1(x), ...\widehat{y}_B(x)$ выдаются вероятности классов.
  - метки загрубляют информацию о неточной классификации.
- Кодовые представления классов W_i выбираются, чтобы быть максимально непохожими по расстоянию Хэмминга.

Содержание

Связь с принипом максимального правдоподобия

- $X = \{x_1, x_2, ...x_N\}$, $Y = \{y_1, y_2, ...y_N\}$ - обучающая выборка; $(x_i, y_i) \sim p(y|x, w)$
- Принцип максимума правдоподобия (при условии наблюдаемых X)

$$\widehat{w} = \arg\max_w p(Y|X, w)$$

$$\prod_{i=1}^N p(y_i|x_i, w) \to \max_w \iff \sum_{i=1}^N \ln p(y_i|x_i, w) \to \max_w$$

- Минимизация эмпирического риска:

$$\sum_{i=1}^N \mathcal{L}(g(x_i)y_i|w) \to \min_w$$

- Взаимосвязь между $\mathcal{L}(\cdot)$ и $p(\cdot)$:

$$\mathcal{L}(g(x_i)y_i|w) = -\ln p(y_i|x_i, w)$$

Связь с оценкой максимальной апостериорной вероятности

- Оценка максимальной апостериорной вероятности - англ. Maximum a prosteriori (MAP) estimation.
- Байесовский подход: w - случайная величина с априорным распределением $p(w)$
    - $p(w)$ не зависит от последующей обучающей выборки

$$p(w|Y,X)=\frac{p(Y,w|X)}{p(Y|X)}=\frac{p(Y|X,w)p(w|X)}{p(Y|X)}$$
$$\propto p(Y|X,w)p(w|X)=p(Y|X,w)p(w)$$

$$w=\arg\max_w p(w|X,Y)=\arg\max_w p(Y|X,w)p(w)$$

$$\sum_{i=1}^N \ln p(y_i|x_i,\theta)+\ln p(w)\to\max_w$$

Связь с оценкой максимальной апостериорной вероятности

$$\sum_{i=1}^{N} \ln p(y_i|x_i, \theta) + \ln p(w) \to \max_{w}$$

$$\sum_{i=1}^{N} \mathcal{L}(g(x_i)y_i|w) + \lambda R(w) \to \min_{w}$$

Взаимосвязь:

$$\mathcal{L}(g(x_i)y_i|w) = -\ln p(y_i|x_i, w), \qquad \lambda R(w) = -\ln p(w)$$

Априорные распределения для L_1 и L_2 регуляризации

- Распределение Гаусса:

$$\ln p(w,\sigma^2) = \ln\left(C_1 e^{-\frac{||w||_2^2}{2\sigma^2}}\right) = -\frac{1}{2\sigma^2}||w||_2^2 + \text{const}(w)$$

- Распределение Лапласа:

$$\ln p(w,C) = \ln\left(C_2 e^{-\frac{||w||_1}{C}}\right) = -\frac{1}{C}||w||_1 + \text{const}(w)$$

Заключение

- Линейный классификатор - классификатор с линейными дискриминантными функциями.
- Линейный бинарный классификатор:
  $\widehat{y}(x) = \text{sign}(w^T x + w_0)$, граница классов - гиперплоскость.
  - популярные методы: метод опорных векторов и логистическая регрессия.
- Регуляризация контролирует сложность модели.
  - L_1 регуляризация может отбирать признаки.
- Логистическая регрессия может оценивать вероятности классов.
- Многоклассовые классификаторы можно строить из набора бинарных классификаторов.