Обучение с учителем. Классификация. Дискриминантный анализ. Логистическая регрессия. Метод опорных векторов. Выбор модели с помощью кросс-валидации. Метод стохастического градиента.

Белкова Анна, Редкокош Кирилл, Лобанова Полина, гр. 21.М03-мм

11 декабря 2022 г.

Содержание

1 Классификация

1.1 Постановка задачи

Дано:

- X – матрица признаков, где $\boldsymbol{x}_i \in \mathbb{R}^p$ – i вектор этой матрицы (признаки i индивида).
- $\mathcal{Y}$ – конечное множество номеров (имён, меток) классов, где $y_i \in \mathcal{Y}$, а $\boldsymbol{y}$ — вектор меток классов для матрицы X.

Задача:

- По выборке $(X_{train}, \boldsymbol{y}_{train})$, построить классификатор $f : \mathbb{R}^p \to \mathcal{Y}$, который по выборке $(X_{test}, \boldsymbol{y}_{test})$ предскажет метку класса.
- Хотим, чтобы на f достигается минимальная ошибка классификации в некотором смысле.

На генеральном языке:

- $\boldsymbol{\xi} \in \mathbb{R}^p$ – случайный вектор признаков.
- $\eta \in \mathcal{Y}$ – дискретная случайная величина, метка класса.
- $P(\boldsymbol{\xi}, \eta)$ – их совместное распределение.

Дано:
Выборка $(X_{train}, \boldsymbol{y}_{train})$ – N реализаций случайного вектора $(\boldsymbol{\xi}, \eta)$, по выборке необходимо построить классификатор

$$f : \mathbb{R}^p \to \mathcal{Y}.$$

Линейная модель классификации в общем случае: $f(\boldsymbol{x}_i, \boldsymbol{w}) = sign\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle$, где $\boldsymbol{w}$ вектор весов признаков, а w_0 некоторый сдвиг, который позволяет нам не получить 0 значение.
Мы будем рассматривать классификатор в следующей форме, добавив вектор из единиц:

$$f(\boldsymbol{x}_i, \boldsymbol{w}) = sign\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle.$$

Отступ (margin): $M_i = \langle (\boldsymbol{w}, \boldsymbol{x}_i), \boldsymbol{x}_i \rangle y_i$ — "расстояние"между реальным и предсказанным значением . В случае отрицательного значения, считаем, что объект не принадлежит классу.
Функцию потерь $\mathcal{L}(M)$— неотрицательная функция, характеризующая величину ошибки предсказания. Пороговая функция потерь: $[M(x_i) < 0]$.
Тогда задача классификации можно свести к минимизации функции потерь:

$$\mathcal{L}(M) \to \min$$

Далее будем рассматривать отступ, со знаком минус, как штраф за неверную классификацию.

1.2 Метрики качества классификации

Часто возникает необходимость в изучении различных аспектов качества уже обученного классификатора. Обсудим подробнее распространённые подходы к измерению качества моделей.

1.2.1 Матрица ошибок

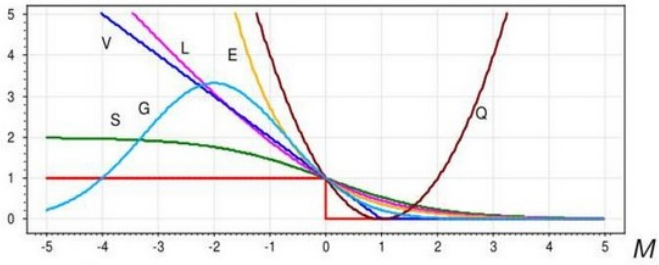
Перед переходом к самим метрикам необходимо ввести важную концепцию для описания этих метрик в терминах ошибок классификации — confusion matrix (матрица ошибок).

Допустим, что у нас есть два класса и алгоритм, предсказывающий принадлежность каждого объекта одному из классов, тогда матрица ошибок классификации будет выглядеть следующим образом:

	y=1	y=0
$\hat{y} = 1$	True Positive (TP)	False Positive (FP)
$\hat{y} = 0$	False Negative (FN)	True Negative (TN)

Где $\hat{y}$ — это ответ алгоритма на объекте, а y — истинная метка класса на этом объекте. Таким образом, ошибки классификации бывают двух видов: False Negative (FN) и False Positive (FP).

Функции потерь $\mathscr{L}(M)$ в задачах классификации на два класса



$E(M) = e^{-M}$ — экспоненциальная (AdaBoost);
$L(M) = \log_2(1 + e^{-M})$ — логарифмическая (LogitBoost);
$G(M) = \exp(-cM(M + s))$ — гауссовская (BrownBoost);
$Q(M) = (1 - M)^2$ — квадратичная;
$S(M) = 2(1 + e^{M})^{-1}$ — сигмоидная;
$V(M) = (1 - M)_+$ — кусочно-линейная (SVM);

1.2.2 Accuracy

Интуитивно понятной, очевидной и почти неиспользуемой метрикой является accuracy — доля правильных ответов алгоритма:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Эта метрика бесполезна в задачах с неравными классами, и это легко показать на примере.

Допустим, мы хотим оценить работу спам-фильтра почты. У нас есть 100 не-спам писем, 90 из которых наш классификатор определил верно (True Negative = 90, False Positive = 10), и 10 спам-писем, 5 из которых классификатор также определил верно (True Positive = 5, False Negative = 5). Тогда accuracy:

$$accuracy = \frac{5 + 90}{5 + 90 + 10 + 5} = 86,4\%$$

Однако если мы просто будем предсказывать все письма как не-спам, то получим более высокую accuracy:

$$accuracy = \frac{0 + 100}{0 + 100 + 0 + 10} = 90,9\%$$

При этом, наша модель совершенно не обладает никакой предсказательной силой, так как изначально мы хотели определять письма со спамом. Преодолеть это нам поможет переход с общей для всех классов метрики к отдельным показателям качества классов.

1.2.3 Precision, recall и F-мера

Для оценки качества работы алгоритма на каждом из классов по отдельности введем метрики precision (точность) и recall (полнота).

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

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

Precision можно интерпретировать как долю объектов, названных классификатором положительными и при этом действительно являющимися положительными, а recall показывает, какую долю объектов положительного класса из всех объектов положительного класса нашел алгоритм.

Именно введение precision не позволяет нам записывать все объекты в один класс, так как в этом случае мы получаем рост уровня False Positive. Recall демонстрирует способность алгоритма обнаруживать данный класс вообще, а precision — способность отличать этот класс от других классов.

Precision и recall не зависят, в отличие от accuracy, от соотношения классов и потому применимы в условиях несбалансированных выборок.

Существует несколько различных способов объединить precision и recall в агрегированный критерий качества. F-мера (в общем случае F_β) — среднее гармоническое precision и recall :

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

β в данном случае определяет вес точности в метрике, и при $\beta = 1$ это среднее гармоническое (с множителем 2, чтобы в случае precision = 1 и recall = 1 иметь $F_1 = 1$) F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

1.2.4 AUC-ROC и AUC-PR

Одним из способов оценить модель, является AUC-ROC (или ROC AUC) — площадь (Area Under Curve) под кривой ошибок (Receiver Operating Characteristic curve). Данная кривая представляет из себя линию от (0,0) до (1,1) в координатах True Positive Rate (TPR) и False Positive Rate (FPR):

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

$$FPR = \frac{FP}{FP + TN}$$

TPR– это полнота, а FPR показывает, какую долю из объектов negative класса алгоритм предсказал неверно. В идеальном случае, когда классификатор не делает ошибок (FPR = 0, TPR = 1) мы получим площадь под кривой, равную единице; в противном случае, когда классификатор случайно выдает вероятности классов, AUC-ROC будет стремиться к 0.5, так как классификатор будет выдавать одинаковое количество TP и FP.

Каждая точка на графике соответствует выбору некоторого порога. Площадь под кривой в данном случае показывает качество алгоритма.

Precision и recall также используют для построения кривой и, аналогично AUC-ROC, находят площадь под ней:

1.3 Модификации датасета для выравнивания соотношения классов

Одним из распространенных способов решения проблемы несбалансированных данных является избыточная выборка. Чрезмерная выборка относится к различным методам, которые направлены на увеличение количества экземпляров из недопредставленного класса в наборе данных.

1.3.1 Случайная наивная избыточная выборка

Самый простой способ сделать это - случайным образом выбрать наблюдения из класса меньшинства и добавить их в набор данных, пока мы не достигнем баланса между большинством и классом меньшинства.

Одна проблема со случайной наивной избыточной выборкой заключается в том, что она просто дублирует уже существующие данные. Поэтому, хотя алгоритмы классификации подвергаются большему количеству наблюдений из класса меньшинства, они не узнают больше о том, как отличить наблюдения одного класса от другого. Новые данные не содержат больше информации о характеристиках класса, чем старые данные.

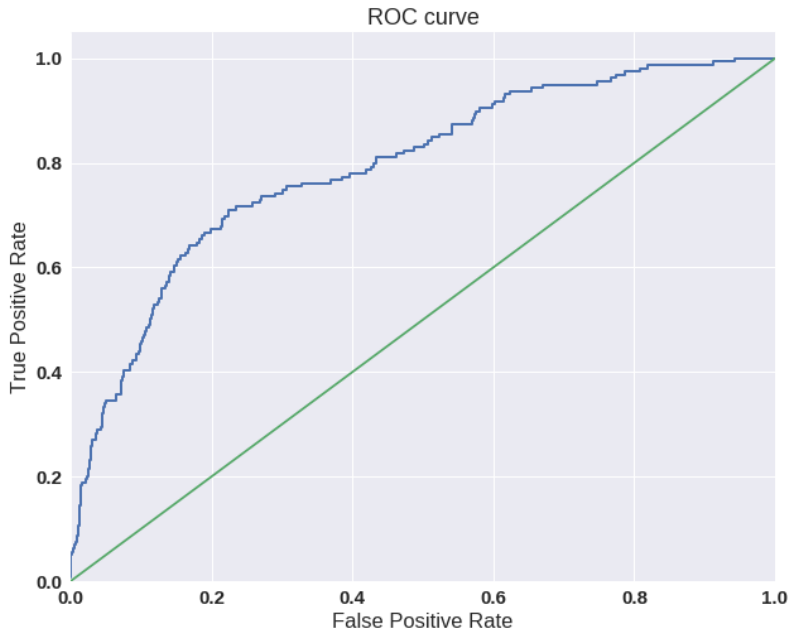


Рис. 1: ROC-кривая

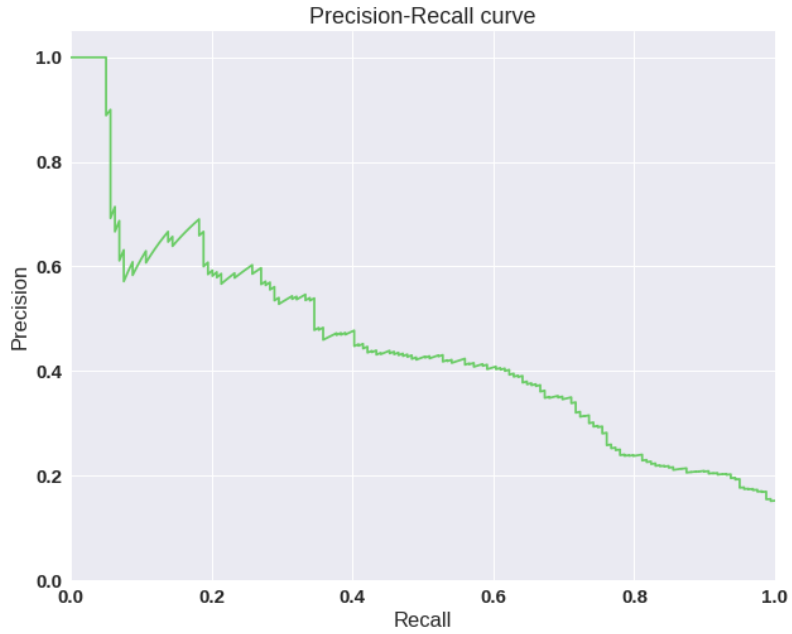


Рис. 2: PR-кривая

1.3.2 SMOTE

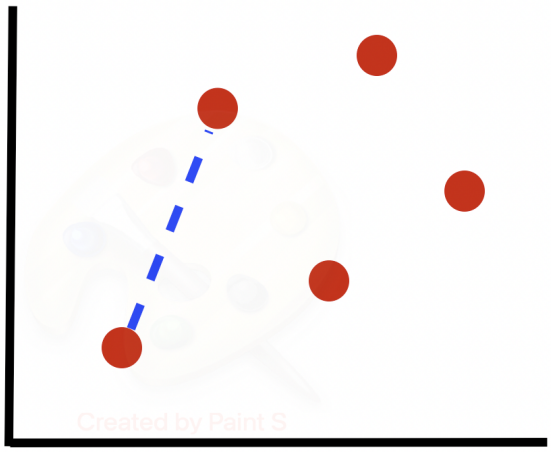
Метод увеличения числа примеров миноритарного класса (Synthetic Minority Over-sampling Technique, SMOTE) — это алгоритм предварительной обработки данных, используемый для устранения дисбаланса классов в наборе данных.

В общих чертах этот алгоритм можно описать следующим образом. Он находит разность между данным образцом и его ближайшим соседом. Эта разность умножается на случайное число в интер-

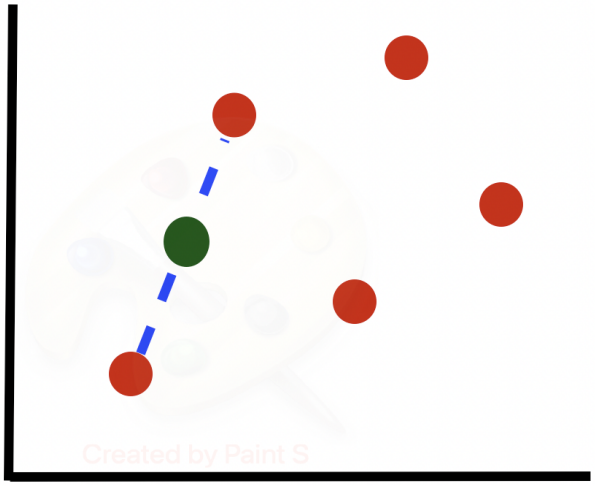
вале от 0 до 1. Полученное значение добавляется к данному образцу для формирования нового синтезированного образца в пространстве признаков. Подобные действия продолжаются со следующим ближайшим соседом, до заданного пользователем количества образцов.

Проиллюстрируем алгоритм более подробно. Предположим, у нас есть несбалансированный набор данных (индивидов одного класса гораздо больше, чем другого).

Берем индивида и вычисляем k-ближайших соседей. Затем выбираем случайного ближайшего соседа из k-ближайших соседей.



Вычисляем разность между двумя точками и умножаем ее на случайное число от 0 до 1. Получаем синтезированный образец вдоль линии между двумя точками.

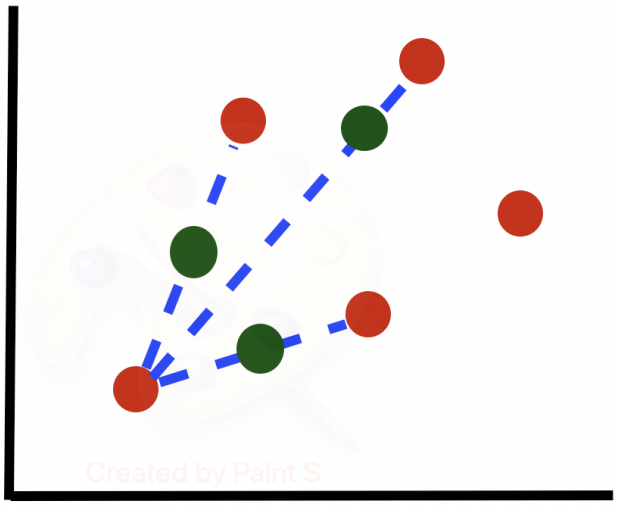


Выбираем m соседей для этого индивида и повторяем процедуру дублирования. m подбирается из соотношения классов.

Для каждого из исходных индивидов повторяем весь алгоритм для достижения равного количества индивидов в классах.

SMOTE Расширения

Как и в большинстве алгоритмов, есть несколько расширений SMOTE. Они нацелены на улучшение SMOTE путем добавления его функциональности или уменьшения его слабых сторон. Примеры расширений SMOTE, которые можно найти в imblearn, включают:



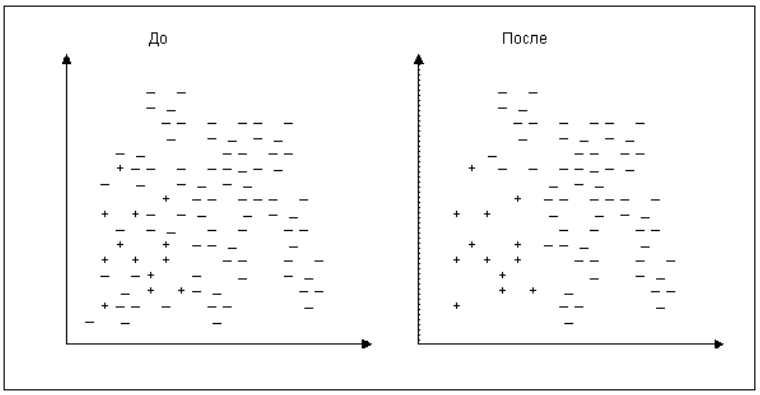
- BorderlineSMOTE: Вместо избыточной выборки между всеми наблюдениями меньшинств, BorderlineSMOTE стремится увеличить количество наблюдений меньшинств, которые граничат с наблюдениями большинства. Цель здесь - дать классификатору возможность более четко различать эти пограничные наблюдения.
- SVMSMOTE: SVMSMOTE, как следует из его названия, использует алгоритм машины опорных векторов для генерации новых наблюдений меньшинства вблизи границы между классами большинства и меньшинства.

1.3.3 Tomek Links

Пусть индивиды E_i и E_j принадлежат к различным классам, $d(E_i, E_j)$ – расстояние между указанными примерами. Пара (E_i, E_j) называется связью Томека, если не найдется ни одного примера E_l такого, что будет справедлива совокупность неравенств:

$$\begin{cases} d(E_i, E_l) < d(E_i, E_j), \\ d(E_j, E_l) < d(E_i, E_j) \end{cases}$$

Согласно данному подходу, все индивиды из большей группы, входящие в связи Томека, должны быть удалены из набора данных. Этот способ хорошо удаляет записи, которые можно рассматривать в качестве «зашумляющих». Далее визуально показан набор данных в двумерном пространстве признаков до и после применения поиска связей Томека.



2 Дискриминантный анализ

Суть дискриминантного анализа заключается в том, чтобы смоделировать распределение X в каждом из классов отдельно, а затем использовать теорему Байеса, чтобы получить $P(Y = i \mid X = x)$

2.1 Байесовский классификатор

В качестве меры ошибки предсказания введем функцию потерь. Рассмотрим матрицу $\mathbf{L}$ размера $K \times K$, где $K = card(\mathcal{Y})$. На диагонали $\mathbf{L}$ стоят нули, а $\mathbf{L}(i, j) = \lambda_{ij}$ – цена ошибки отнесения элемента класса Y_i к классу Y_j. Часто используется 0-1 функция потерь, где каждая ошибка оценивается единицей.

Математическое ожидание функции потерь (средний риск):

$$R(a) = \mathbb{E}(\mathbf{L}(\eta, a(\xi))) = \mathbb{E}_\xi \sum_{k=1}^{K} L(Y_i, a(\xi))P(Y_i \mid \xi).$$

Отсюда получаем функцию классификации:

$$f(x) = \arg\min_{Y \in \mathcal{Y}} \sum_{k=1}^{K} L(Y_i, Y)P(Y_i \mid \xi = x).$$

Если подставим сюда 0-1 функцию потерь, получим

$$f(x) = \arg\min_{Y \in \mathcal{Y}} 1 - P(Y \mid \xi = x).$$

Или, что то же самое

$$f(x) = \arg\max_{Y \in \mathcal{Y}} P(Y \mid \xi = x) = \arg\max_{Y \in \mathcal{Y}} P(Y)P(\xi \mid \eta = Y).$$

Это решение называется байесовским классификатором, а такой подход – принципом максимума апостериорной вероятности.

Для построения байесовского классификатора, нам необходимо знать апостериорные вероятности $P(Y \mid \xi = x)$.

Обозначим $p_i(x) = P(\xi = x \mid \eta = Y_i)$ условные плотности классов, $\pi_i = P(\eta = Y_i)$ – априорные вероятности, $\sum_{i=1}^{K} \pi_i = 1$. По теореме Байеса получим:

$$P(Y = i \mid X = x) = \frac{p_i(x)\pi_i}{\sum_{i=1}^{K} p_i(x)\pi_i}.$$

Поэтому в качестве классифицирующих функций берут

$$f_i\,(x) = \frac{p_i(x)\pi_i}{\sum_{j=1}^{k} p_j(x)\pi_j}.$$

Так как знаменатель у всех f_i одинаковый, его можно отбросить, и итоговые классифицирующие функции будут выглядеть как $f_i\,(x) = \mathrm{P}\,(x|C_i)\,\pi_i = p_i(x)\pi_i$.

Возникает вопрос: откуда брать априорные вероятности?

1. Равномерно, $\forall i \in 1 : k\ \pi_i = 1\ /\ k$.
2. По соотношениям в обучающей выборке: $\pi_i = n_i\ /\ \sum_{j=1}^{k} n_j$.
3. На основе другой дополнительной информации о данных (результаты предыдущих исследований, etc.)

Построенный метод классификации $\mathrm{predict}(x) = \arg\max_i \pi_i p_i(x)$ минимизирует среднюю апостериорную ошибку:

$$\sum_{i=1}^{k} \pi_i \mathrm{P}(\mathrm{predict}(x) ! = i \mid Y_i).$$

2.2 Линейный дискриминантный анализ

Модель: ξ — дискретная с.в., принимающая значения $\{Y_i\}_{i=1}^k$, $\mathcal{P}(\eta \mid \xi = Y_i) = \mathcal{N}(\mu_i, \mathbf{\Sigma})$. (Предполагаем, что классы имеют нормальное распределение с одинаковой ковариационной матрицей)

Тогда плотность в точке x:

$$p_i(x) = p\left(x|\xi = Y_i\right) = \frac{1}{(2\pi)^{p/2}\left|\mathbf{\Sigma}\right|^{1/2}} \exp\left(-\frac{1}{2}\big(x^T - \mu_i\big)\mathbf{\Sigma}^{-1}\left(x - \mu_i\right)\right),$$

и классифицирующая функция $f_i\left(x\right) = \pi_i p\left(x|\xi = Y_i\right)$, где π_i — априорная вероятность наблюдения попасть в i-ю группу. Для упрощения вычислений можно переписать классифицирующую функцию через возрастающее монотонное преобразование как

$$g_i\left(x\right) = \log f_i\left(x\right) = \log \pi_i - \frac{1}{2}\log\left|\mathbf{\Sigma}\right| - \frac{1}{2}\big(x^T - \mu_i\big)\mathbf{\Sigma}^{-1}\left(x - \mu_i\right).$$

Сократив часть, не зависящую от номера класса, получаем линейные классифицирующие функции:

$$h_i(x) = -\frac{1}{2}{\mu_i}^T\mathbf{\Sigma}^{-1}\mu_i + {\mu_i}^T\mathbf{\Sigma}^{-1}x + \log \pi_i.$$

2.3 Канонические переменные

Задача: найти линейное преобразование $\mathbf{Z} = A^{\mathrm{T}}\mathbf{X}$, в результате которого получаются признаки наилучшим образом разделяющие группы. Хотелось бы, чтобы эти признаки оказались ортогональны. Далее опишем эту задачу более формально.

Вычислим внутриклассовую ковариационную матрицу:

$$\mathbf{E} = \frac{1}{n-K}\sum_{i=1}^{K}\sum_{j:y_j=Y_i}(x_j - \widehat{\mu}_i)^{\mathrm{T}}(x_j - \widehat{\mu}_i)$$

Вычисляем межклассовую ковариационную матрицу (с точностью до коэффициента):

$$\mathbf{H} = \sum_{i=1}^{K} n_i(\widehat{\mu}_i - \widehat{\mu})^{\mathrm{T}}(\widehat{\mu}_i - \widehat{\mu}).$$

Пусть $\zeta = A\xi$ – новый признак, тогда распределение $P(\zeta \mid \eta = Y_k) = \mathcal{N}_p(A^{\mathrm{T}}\mu_k, A^{\mathrm{T}}\mathbf{\Sigma}_k A)$.

На выборочном языке новые признаки $\mathbf{Z} = A^{\mathrm{T}}\mathbf{X}$. Выборочная ковариационная матрица (с точностью до коэффициента) новых признаков имеет вид:

$$A^{\mathrm{T}}\mathbf{T}A = A^{\mathrm{T}}(\mathbf{E} + \mathbf{H})A = A^{\mathrm{T}}\mathbf{E}A + A^{\mathrm{T}}\mathbf{H}A,$$

где $\mathbf{T}$ – total covariance matrix, первое слагаемое – оценка внутригрупповых отклонений, а второе – оценка межгрупповых отклонений. Воспользовавшись критерием Фишера перейдем к обобщенной задаче на собственные числа и собственные вектора:

$$\frac{A^{\mathrm{T}}\mathbf{H}A}{A^{\mathrm{T}}\mathbf{E}A} \to \max_A.$$

Пусть $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$ – собственные числа матрицы $\mathbf{E}^{-1}\mathbf{H}$, а $A_1, \ldots, A_d$ – соответствующие им собственные вектора. Тогда максимум выше равен λ_1 и достигается на A_1. При этом $A_i^{\mathrm{T}}\mathbf{E}A_j = 0$. Далее

$$\max_{A, A\perp A_1} \frac{A^{\mathrm{T}}\mathbf{H}A}{A^{\mathrm{T}}\mathbf{E}A} = \lambda_2,$$

достигается на A_2 и так далее.

Вектора A_i называют каноническими коэффициентами, а новые признаки Z_i – каноническими переменными, Z_i ортогональны.

2.4 Значимость канонических переменных

Возникает вопрос: сколько канонических переменных нам окажется достаточно взять? Другими словами, нужно проверить гипотезу:

$$H_0 : A_i, i = \ell, \ldots, d \text{ не описывают отличия.}$$

Введем статистику $\Lambda - prime$ (Wilks' Lambda):

$$\Lambda_\ell^p = \prod_{i=l}^{d} \frac{1}{1+\lambda_i}.$$

Тогда гипотезу выше можно переформулировать так

$$H_0 : \Lambda_\ell^p = 1 \Leftrightarrow \lambda_\ell = \ldots = \lambda_d = 0 \Leftrightarrow rank\mathbf{B} = \ell - 1.$$

Критерий:

$$t = \Lambda_\ell^p \sim \Lambda_{\nu_\mathbf{B}+(\ell-1),\nu_\mathbf{W}-(\ell-1)}.$$

Другими вариантами статистиками для проверки гипотезы могут являться:

- Roy's greatest root
$$r_1^2 = \frac{\lambda_1}{1+\lambda_1};$$
- Pillai's trace
$$V = trace(\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1});$$
- Hotelling-Lawley trace
$$V = trace(\mathbf{H}\mathbf{E}^{-1}).$$

2.5 Квадратичный дискриминантный анализ

Модель: ξ — дискретная с.в., принимающая значения $\{Y_i\}_{i=1}^k$, $\mathcal{P}(\eta \mid \xi = Y_i) = \mathcal{N}(\mu_i, \mathbf{\Sigma}_i)$. (Предполагаем, что каждый класс имеет многомерное нормальное распределение с различными ковариационными матрицами)

Тогда плотность в точке x:

$$p\left(x|\xi = Y_i\right) = \frac{1}{(2\pi)^{p/2}\left|\mathbf{\Sigma}_i\right|^{1/2}} \exp\left(-\frac{1}{2}\left(x^T - \mu_i\right)\mathbf{\Sigma}_i^{-1}\left(x - \mu_i\right)\right),$$

и классифицирующая функция $f_i\left(x\right) = \pi_i p\left(x|\xi = Y_i\right)$. Применяем возрастающее монотонное преобразование и оставляем в классифицирующей функции только члены, отличающиеся в разных группах:

$$g_i\left(x\right) = \log f_i\left(x\right) = \log \pi_i - \frac{1}{2}\log\left|\mathbf{\Sigma}_i\right| - \frac{1}{2}\left(x^T - \mu_i\right)\mathbf{\Sigma}_i^{-1}\left(x - \mu_i\right),$$

получаем квадратично зависящую от x классифицирующую функцию.

2.6 Оценка параметров

На практике параметры распределений классов нам не известны, поэтому предлагается использовать следующие оценки максимального правдоподобия параметров нормальных плотностей классов.

- Среднее $\overline{\mu}_i = \frac{1}{n_i}\sum\limits_{j:y_j=Y_i} x_j$,
- Ковариационная матрица класса $\widehat{\mathbf{\Sigma}}_i = \frac{1}{n_i-1}\sum\limits_{j:y_j=Y_i}(x_j - \overline{\mu}_i)^{\mathrm{T}}(x_j - \overline{\mu}_i)$,
- Pooled ковариационная матрица $\widehat{\mathbf{\Sigma}} = \sum\limits_{j=1}^{K}\frac{n_i-1}{n-K}\widehat{\mathbf{\Sigma}}_i$.

2.7 Regularized Discriminant Analysis

Оценка ковариационной матрицы $\widehat{\mathbf{\Sigma}}_i$ может оказаться выражденной или плохо обусловленной. Опишем компромис между LDA и QDA, а так же борьбу с мультиколлинеарностью.

- Regularized Discriminant Analysis. Рассматривается матрица $\widehat{\mathbf{\Sigma}}_i(\alpha) = \alpha\widehat{\mathbf{\Sigma}}_i + (1-\alpha)\widehat{\mathbf{\Sigma}}$, где $\widehat{\mathbf{\Sigma}}$ – pooled ковариационая матрица. Здесь $\alpha \in [0,1]$ порождает континуум моделей между LDA и QDA, выбирается скользящим контролем.
- Дополнительно к предыдущему методу можно похожим образом модифицировать pooled ковариационную матрицу и рассматривать $\widehat{\Sigma}(\gamma) = \gamma\widehat{\mathbf{\Sigma}}+(1-\gamma)\sigma^2\mathbf{I}_p$, где γ определяет вид ковариационной матрицы и выбирается скользящим контролем.

2.8 Наивный байесовский классификатор

Предположим, что признаки независимы внутри групп и имеют нормальное распределение:

$$p_i(x) = \prod_{j=1}^{p} p_{ij}(x_j), \quad p_{ij}(x_j) = \frac{1}{\sqrt{2\pi}\sigma_{ij}} e^{-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}}.$$

Отсюда классифицирующую функцию можно представить в виде:

$$\delta_i(x) = -\frac{1}{2}\sum_{j=1}^{p}\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2} + \log(\pi_i).$$

Аналогично подходам выше, можно подбирать ковариационную матрицу скользящим контролем в виде:

$$\widehat{\mathbf{\Sigma}}_i(\alpha) = \alpha\widehat{\mathbf{\Sigma}}_i + (1-\alpha)\ \mathrm{diag}(\sigma_{i1}^2,\ldots,\sigma_{ip}^2), \alpha \in [0,1].$$

Такой подход может быть полезен, когда признаков очень много и оценивать плотности классов оказывается сложно. Плотности p_{ki} можно оценивать по отдельности, а если признак дискретный, для этого можно использовать гистограмму.

Не смотря на такое оптимистичное предположение, наивный байесовский классификатор часто превосходит более сложные методы.

3 SVM. Метод опорных векторов.

Входные данные: $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$, $\boldsymbol{x}_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$; Задача: построение классифицирующего правила $f : \mathbb{R}^p \to \{-1, 1\}$.

Предположим, что присутствует линейная разделимость, т.е. существует гиперплоскость (определяемая уравнением $\boldsymbol{x}^T\boldsymbol{w} - w_0 = 0$ ($\boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^p; w_0 \in \mathbb{R}$).

Как и прежде у нас есть линейный классификатор:

$$f(\boldsymbol{x}_i) = sign\left(\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle - w_0\right),$$

где $\boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^p; w_0 \in \mathbb{R}$. И кусочно линейна функция потерь, где $M_i = -\langle(\boldsymbol{w}, \boldsymbol{x}_i), \boldsymbol{x}_i\rangle y_i$:

$$L(M_i) = \max\{0, 1 + M_i\}$$

3.1 SVM. Hard-margin SVM

Предположим, что присутствует линейная разделимость, т.е. существует гиперплоскость (определяемая уравнением $\boldsymbol{x}^T\boldsymbol{w} - w_0 = 0$), такая, что точки, соответствующие разным классам лежат в различных полу-пространствах относительно гиперплоскости.

Факт принадлежности наблюдений из разных классов разным полупространствам можно (возможно, изменив знаки β, β_0) описать уравнениями:

$$\begin{cases} \boldsymbol{x}^T\boldsymbol{w} - w_0 < 0 & y_i = -1 \\ \boldsymbol{x}^T\boldsymbol{w} - w_0 > 0 & y_i = 1 \end{cases} \Leftrightarrow (\boldsymbol{x}^T\boldsymbol{w} - w_0)y_i > 0$$

В таком случае, классифицирующим правилом разумно принять $f(x) = sign(\boldsymbol{x}^T\boldsymbol{w} - w_0)$

В линейно разделимых данных может существовать более одной гиперплоскости, разделяющей данные. Введём критерий оптимальности: максимальное расстояние между двумя гиперплоскостями, параллельных данной и симметрично расположенных относительно неё, при котором между ними не находится ни одна из точек; это расстояние будем называть зазором (margin).

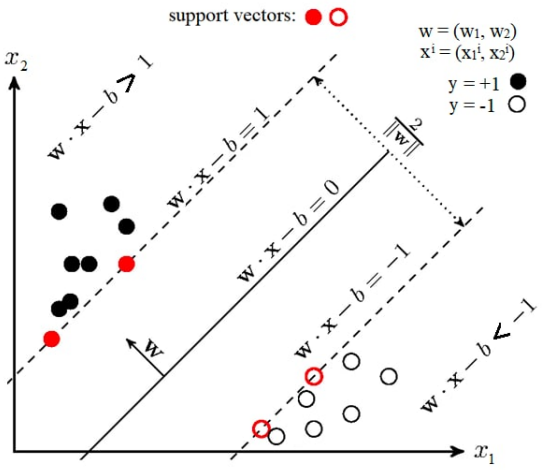
Для каждой из двух параллельных гиперплоскостей будет принадлежать некоторое количество точек из соответствующего класса (иначе, так как количество точек в выборке конечно, то расстояние между гиперплоскостями можно увеличить, сместив гиперплоскость, которой не принадлежит ни одной точки); точки, которые принадлежат одной из гиперплоскостей — будем называть опорными векторами.

С точностью до нормировки вектора $\boldsymbol{w}$ эта пара гиперплоскостей может быть описана парой уравнений:

$$\begin{cases} \boldsymbol{x}^T\boldsymbol{w} - w_0 = -1 \\ \boldsymbol{x}^T\boldsymbol{w} - w_0 = 1 \end{cases}$$

а расстояние между ними составит $\frac{2}{||\boldsymbol{w}||}$ (см. рисунок) Принадлежность точек обучающей выборки полу-пространствам описывается уравнениями

$$\begin{cases} x_i^\intercal\beta - \beta_0 \leq 1 & y_i = -1 \\ x_i^\intercal\beta - \beta_0 \geq 1 & y_i = 1 \end{cases} \Leftrightarrow (x_i^\intercal\beta - \beta_0)\, y_i \geq 1$$



Тогда задачу можно свести к задаче квадратичного программирования с линейными ограничениями:

$$\begin{cases} \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle \to \min\limits_{\boldsymbol{w}} \\ (\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle - w_0)\, y_i \geq 1 \end{cases}$$

Воспользуемся методом множителей Лагранжа:

$$\begin{cases} \inf\limits_{\boldsymbol{w}, w_0} \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle - \sum\limits_{i=1}^{n} \alpha_i \left[y_i\left(\boldsymbol{x}_i^\intercal\boldsymbol{w} + w_0\right) - 1\right] \to \max\limits_{\alpha_i} \\ \alpha_i \geq 0, \forall i \\ y_i\left(\boldsymbol{x}_i, \boldsymbol{w}\rangle - w_0\right) \geq 1 \end{cases}$$

Так как оптимизируемая функция гладкая, можно воспользоваться необходимыми условиями экстремума

$$\frac{\partial}{\partial \boldsymbol{w}} : \boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i$$

$$\frac{\partial}{\partial w_0} : 0 = \sum_{i=1}^{n} \alpha_i y_i$$

Двойственная задача Вольфа:

$$\begin{cases} \sum\limits_{i=1}^{n} \alpha_i - \frac{1}{2} \sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \to \max\limits_{\alpha_i} \\ \alpha_i \geq 0, \forall i \\ y_i \left(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle - w_0 \right) \geq 1 \\ \boldsymbol{w}_0 = \sum\limits_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \end{cases}$$

В точке оптимума выполнены условия Каруша-Куна-Такера, в частности:

$$\alpha_i \left[1 - y_i \left(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle - w_0 \right) \right] = 0 \, \forall i$$

Т.е. либо

- $\alpha_i = 0$ т.е. наблюдение не влияет на $\boldsymbol{w}, w_0$
- $\alpha_i > 0 \Rightarrow y_i \left(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle - w_0 \right) = 1$ – такое наблюдение будем называть опорным вектором

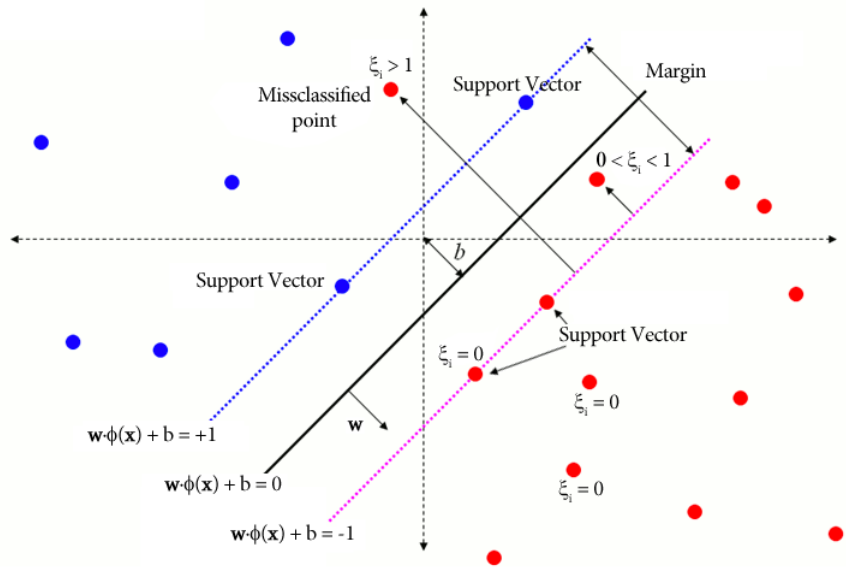
3.2 SVM.Slack variables

Поскольку в случае линейно неразделимой выборки по определению любой линейный классификатор будет ошибаться, условие $\left(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle - w_0 \right) y_i \geq 1$ не может быть выполнено для всех i. Введём ошибки $\xi \geq 0$ алгоритма и штрафы за эти ошибки в минимизируемую функцию следующим образом:

$$\begin{cases} \frac{1}{2} \langle \boldsymbol{w}, \boldsymbol{w} \rangle + C \sum\limits_{i=1}^{n} \xi_i \to \min\limits_{\boldsymbol{w}, w_0} \\ \left(\langle \boldsymbol{x}_i, \boldsymbol{w} \rangle - w_0 \right) y_i \geq 1 - \xi_i \\ , \end{cases}$$

где C задает размер штрафа за ошибки.

Мы опять получили задачу линейного программирования.



3.3 Kernel trick

Чтобы применять SVN в нелинейном случае, строилось спрямляющее пространство. В основе этого лежит очень простая и очень красивая идея: если в каком-то исходном пространстве признаков классы не являются линейно разделимыми, то может быть можно отобразить это пространство признаков в какое-то новое, в котором классы уже будут линейно разделимы.

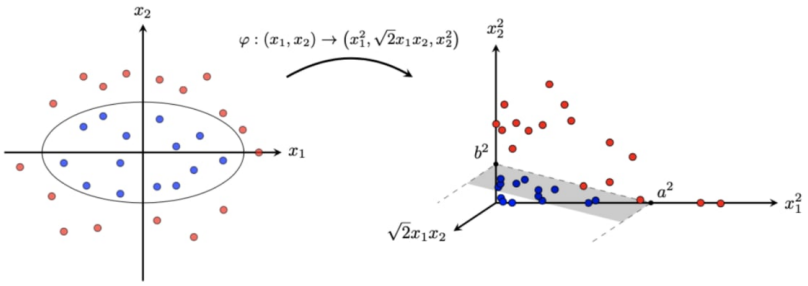
Пусть $\phi(\boldsymbol{x}_i)$ - спрямляющее отображение. Тогда, можем записать SVM в спрямляющем отображении, используя следующее скалярное произведение:

$$\boldsymbol{x}_i \to \phi(\boldsymbol{x}),\ \boldsymbol{w} \to \phi(\boldsymbol{w}),\ \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle \to \langle \phi(\boldsymbol{w}), \phi(\boldsymbol{x}_i) \rangle$$

Чтобы получить нелинейную разделимость в исходном пространстве - зададим скалярное произведение следующего вида:

$$K(\boldsymbol{w}, \boldsymbol{x}_i) = \langle \phi(\boldsymbol{w}), \phi(\boldsymbol{x}_i) \rangle$$

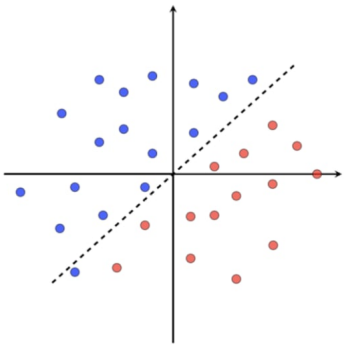
K - симметричная нелинейная функция



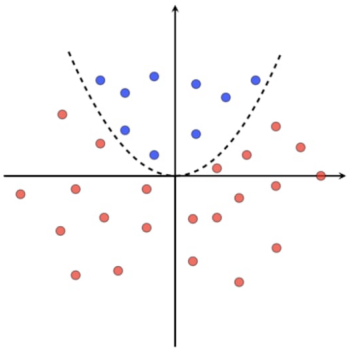
Наиболее часто используемые ядра

- Линейное ядро:
$$K(\boldsymbol{w}, \boldsymbol{x}_i) = \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle$$
- Полиномиальное ядро:
$$K(\boldsymbol{w}, \boldsymbol{x}_i) = (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + r)^d$$
- Радиальное ядро:
$$K(\boldsymbol{w}, \boldsymbol{x}_i) = exp(-\gamma ||(\boldsymbol{w} - \boldsymbol{x}_i)||^2)$$
- Сигмовидное ядро.

Примеры:



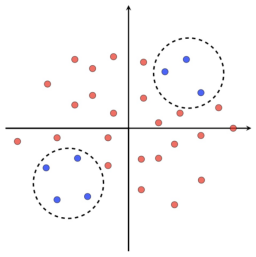
Линейное ядро



Полиномиальное ядро

3.4 Сложность SVM

Так как для SVM нужно решать задачу квадратичного программирования, то сложность варьируется между $O(p \cdot n^2)$ и $O(p \cdot n^3)$, где p – количество признаков, n – количество индивидов, зависимости от набора данных.



Радиальное ядро

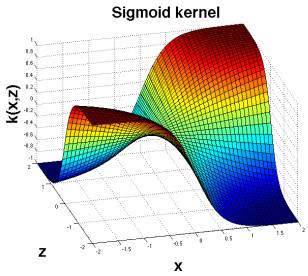


Рис. 3: Сигмовидное ядро

3.5 Мультиклассовый SVM

- Сравнение "один со многими". Строим N классифицирущих правил $f_i(\boldsymbol{x})$, кодирующих принадлежность к i-му классу за 1, -1 иначе. В качестве решающего правила используется

$$f(\boldsymbol{x}) = \underset{i}{argmax} f_i(\boldsymbol{x})$$

- Сравнение "каждый с каждым". Строим $\frac{N(N-1)}{2}$ классифицирущих правил, производящих классификацию для каждой возможной пары классов. Обозначим за N_i количество сравнений, в которых элемент x был классифицирован как принадлежный к i-му классу. В качестве решающего правила используется

$$f(\boldsymbol{x}) = \underset{i}{argmax} N_i$$

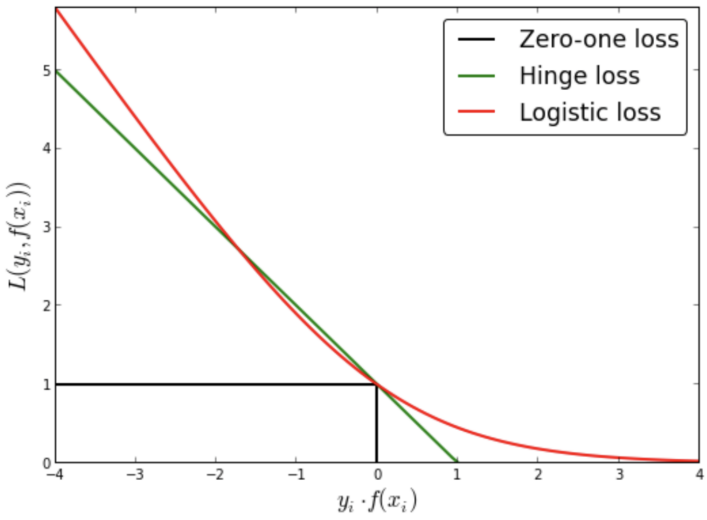
4 Логистическая регрессия

4.1 Логистическая регрессия. Подход через минимизацию функции потерь

Линейная модель классификации:

- $f(x, \theta) = \text{sign}\langle\theta, x\rangle, \quad x, \theta \in \mathbb{R}^p$
- $M = \langle\theta, x\rangle y$ — отступ.

В качестве аппроксимации пороговой функции потерь берется логарифмическая функция потерь $L(M) = \log(1 + e^{-M})$.



Задача 1. $\mathsf{Q}(\mathsf{X_n}, \theta) = \sum\limits_{\mathsf{i}=1}^{\mathsf{n}} \log(1 + \exp(-\mathsf{y_i}\langle\theta, \mathsf{x_i}\rangle)) \to \min\limits_{\theta}$

Методы решения задачи минимизации:

- метод стохастического градиента
- метод Ньютона-Рафсона

4.2 Логистическая регрессия. Вероятностный подход

$\mathsf{P}(\mathsf{y} = 1|\mathsf{x}, \theta) = \sigma_\theta(\mathsf{M}) = \frac{1}{1+\mathsf{e}^{-\langle \mathsf{x}, \theta\rangle \mathsf{y}}}$ — сигмоидная функция.
Свойства $\sigma(\mathsf{z})$:

- $\sigma(\mathsf{z}) \in [0, 1]$, задана на $(-\infty, +\infty)$
- $\sigma(\mathsf{z}) \to 1,\ \mathsf{z} \to +\infty$; $\sigma(\mathsf{z}) \to 0,\ \mathsf{z} \to -\infty$
- $\sigma(\mathsf{z}) + \sigma(-\mathsf{z}) = 1$
- $\sigma'(\mathsf{z}) = \sigma(\mathsf{z})\sigma(-\mathsf{z})$

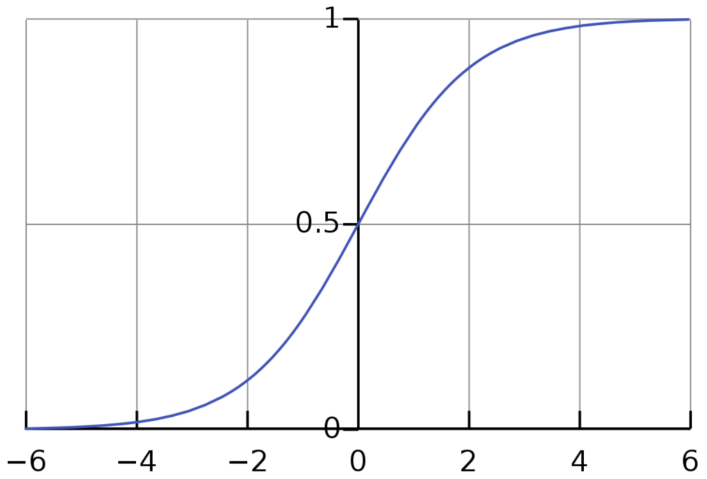


Рис. 4: Сигмоидная функция

Пусть $Y = \{0, 1\}$.

- $P(y_i = 1|x; \theta) = \sigma_\theta(x)$
- $P(y_i = 0|x; \theta) = 1 - \sigma_\theta(x)$

Тогда $P(y|x; \theta) = (\sigma_\theta(x))^y(1 - \sigma_\theta(x))^{1-y}$.
Функция правдоподобия:

$$Q(X_n, \theta) = -\log L(\theta) = -\log \prod_{i=1}^{n} (\sigma_\theta(x_i))^{y_i}(1 - \sigma_\theta(x_i))^{1-y_i} =$$

$$= -\sum_{i=1}^{n} [y_i \log(\sigma_\theta(x_i) + (1 - y_i)) \log(1 - \sigma_\theta(x_i))] \to \min_\theta$$

4.3 Линейная и логистическая регрессия

Существуют примеры данных, для которых логистическая регрессия показывает лучшие результаты, чем линейная.

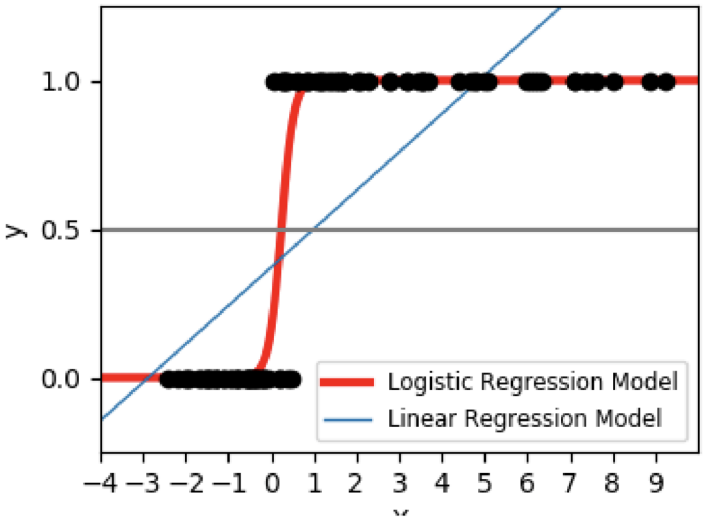


Рис. 5: Линейная и логистическая регрессия

4.4 Логистическая регрессия. Регуляризация

$Q(\theta) = -\sum_{i=1}^{n} [y_i \log(\sigma_\theta(x_i) + (1 - y_i)) \log(1 - \sigma_\theta(x_i))]$
Регуляризация в логистической регрессии:

- **L2**: $Q_\tau(\theta) = Q(\theta) + \frac{\tau}{2} \sum_{j=1}^{p} \theta_j^2 \to \min_\theta$
- **L1**: $Q_\tau(\theta) = Q(\theta) + \tau \sum_{j=1}^{p} |\theta_j| \to \min_\theta$

Параметр τ можно подбирать с помощью кросс-валидации.
Методы решения задачи минимизации:

- метод стохастического градиента
- метод Ньютона-Рафсона.

4.5 Многоклассовая логистическая регрессия

Линейный классификатор при произвольном числе классов $\mathsf{Y} = \{1, \ldots, \mathsf{K}\}$:

$$\hat{f}(x, \theta) = \arg\max_{y \in Y} \langle \theta_y, x \rangle, \ x, \theta_y \in \mathbb{R}^p$$

Вероятность того, что объект x относится к классу i:

$$\mathsf{P}(y = i | x; \theta) = \frac{\exp \langle \theta_y, x \rangle}{\sum\limits_{z \in Y} \exp \langle \theta_z, x \rangle} = \frac{e^{\theta_i^{\mathrm{T}} x}}{\sum\limits_{k=1}^{K} e^{\theta_k^{\mathrm{T}} x}}$$

Задача:

$$\mathsf{Q}(\mathsf{X_n}, \theta) = -\sum_{\mathsf{i}=1}^{\mathsf{n}} \log \mathsf{P}(\mathsf{y_i} | \mathsf{x_i}; \theta) \to \min_{\theta}$$

4.6 Логистическая регрессия. Преимущества и недостатки

Плюсы:

1. Позволяет оценить вероятности принадлежности объектов к классу
2. Достаточно быстро работает при больших объемах выборки
3. Применима в случае отсутствия линейной разделимости, если на вход подать полиномиальные признаки

Минусы:

1. Плохо работает в задачах, в которых зависимость сложная, нелинейная