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

Гриненко Юрий

24 декабря 2023 г.

Содержание

1 Классификация

- Имеется случайный вектор признаков $\xi \in \mathbb{R}^p$, дискретная случайная величина $\eta \in \mathcal{G} = \{G_k\}_{k=1}^K$, метка класса.
- y - вектор меток класса, соответствующий матрице признаков X;
- Задача — построить классификатор $f : \mathbb{R}^p \to \mathcal{G}$ по обучающей выборке $X_{train}; y_{train}$, предсказывающий метку класса на контрольной выборке.

Имеем N реализаций случайного вектора (ξ, η). При переходе к выборкам случайные величины заменяются на матрицу наблюдений и на вектор классовой принадлежности соответственно. Строим классификатор

$$f : \mathbb{R}^p \to \mathcal{G}. \tag{1}$$

Запишем в форме

$$f(x_i, w) = sign(\langle x_i, w\rangle - w_o) = sign \sum_{i=1}^{n} w_i x_i - w_o, \tag{2}$$

где $w = (w_1, ..., w_n)$ —- вектор весов, $\langle x_i, w\rangle$ — скалярное произведение признакового описания объекта на вектор весов.

Вводим несколько ключевых понятий:

- Величина отступа (margin) — $M_i(w, w_0) = (\langle x_i, w\rangle - w_0)y_i$; чем меньше значение отступа, тем ближе объект к границе классов, соответственно, выше вероятность ошибки (справедливо и обратное);
- Функция потерь $\mathcal{L}(a, x)$ — неотрицательная функция, характеризующая величину ошибки предсказания на объекте x_i. Задачу классификации можно свести к минимизации функции потерь;
- Функционал качества алгоритма $Q(a, X^n) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(a, x_i)$ на выборке X^n, также называемый эмпирическим риском.

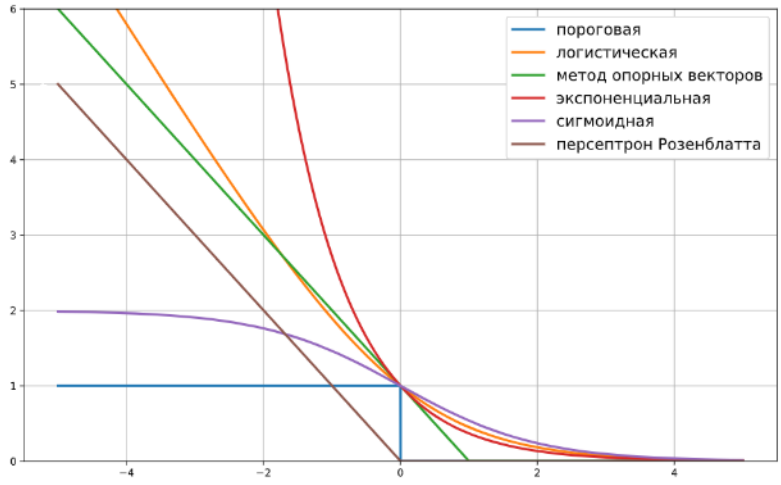

Рис. 1: Непрерывные аппроксимации пороговой функции потерь

Так, вид некоторых Margin-based loss functions:

- $Q(M) = (1 - M)^2$, квадратичная;
- $V(M) = (1 - M)$, кусочно-линейная;
- $S(M) = 2(1 + e^M)^-1$, сигмоидная;
- $L(M) = log_2(1 + e^{-M})$, логистичесая;
- $E(M) = e^-M$, экспоненциальная.

1.1 Качество классификации, метрики

Матрица ошибок — способ разбить объекты на четыре категории в зависимости от комбинации истинного ответа и ответа алгоритма.

		True Class		
		Positive	Negative	Total
Predicted	Positive	TP	FP	$TP + FP$
	Negative	FN	TN	$FN + TN$
	Total	$TP + FN$	$FP + TN$	N

Доля правильных ответов для двух классов:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3}$$

Может показывать плохо интерпретируемые результаты в случае, если предсказываемые классы в неравномерны, или если стоит задача корректно предсказать только один класс. Лучше интерпретируем метрики, которые показывают точность предсказания одного из классов:

$$Precision = \frac{TP}{TP + FP}, \tag{4}$$

Precision — доля правильно предсказанных объектов класса среди всех, отнесенных алгоритмом к этому классу. Интерпретируется как способность отличать этот класс от других классов.

$$Recall = \frac{TP}{TP + FN}, \tag{5}$$

Recall — доля объектов класса, которую алгоритм классифицировал правильно. Показывает способность алгоритма обнаруживать данный класс вообще.

2 Дискриминантный анализ

2.1 Общий подход к классификации

Строим классифицирующие функции f_i, индивид относится к классу с максимальным значением классифицирующей функции на нем.

$$f(x) = \arg\max_{Y \in \mathcal{G}} P(Y|\xi = x). \tag{6}$$

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

$$P(Y = i|X = x) = \frac{p_i(x)\pi_i}{\sum_{i=1}^{K} p_i(x)\pi_i}, \tag{7}$$

В качестве классифицирующих функций берем

$$f_i(x) = \frac{p_i(x)\pi_i}{\sum_{j=1}^{K} p_j(x)\pi_j}. \tag{8}$$

Знаменатель будет одинаков у всех таких функций, поэтому можем оставить только значимую часть (числитель).

Априорные вероятности выбираем в зависимости от наших предположений/задачи. С помощью априорных вероятностей формально задавать важность ошибочных классификаций для разных классов. Несколько вариантов выбора:

- Равномерно, $\forall i \in 1:k$; $\pi = 1/k$;
- По соотношению в обучающей выборке $\pi_i = n_i / \sum_{j=1}^{k} n_j$;
- На основе дополнительной информации (месте проведения исследования, социальных факторах и прочем).

2.2 Линейный дискриминантный анализ

В LDA предполагаем нормальное распределении классов с одинаковой ковариационной матрицей. Классифицирующие функции имеют вид

$$f_i(x) = \frac{\pi_i}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^{\mathsf{T}}\Sigma^{-1}(x-\mu_i)\right), \tag{9}$$

и приводятся к виду

$$h_i(x) = -\frac{1}{2}\mu_i^{\mathsf{T}}\Sigma^{-1}\mu_i + \mu_i^{\mathsf{T}}\Sigma^{-1}x + log\ \pi_i. \tag{10}$$

2.3 Канонические переменные

Канонические переменные — признаки, наилучшим образом разделяющие классы. Задача состоит в нахождении линейного преобразования $\mathbf{Z} = A^T\mathbf{X}$.

В англоязычной литературе имеет название CDA (Canonical Discriminant Analysis). Результат должен быть примерно как на изображении, сравниваем с PCA.

Вычислим внутриклассовую ковариационную матрицу:

$$\mathbf{E} = \frac{1}{n-K}\sum_{i=1}^{K}\sum_{j:y_j\prime=Y_i}(x_j-\hat{\mu}_i)^{\mathrm{T}}(x_j-\hat{\mu}_i) \tag{11}$$

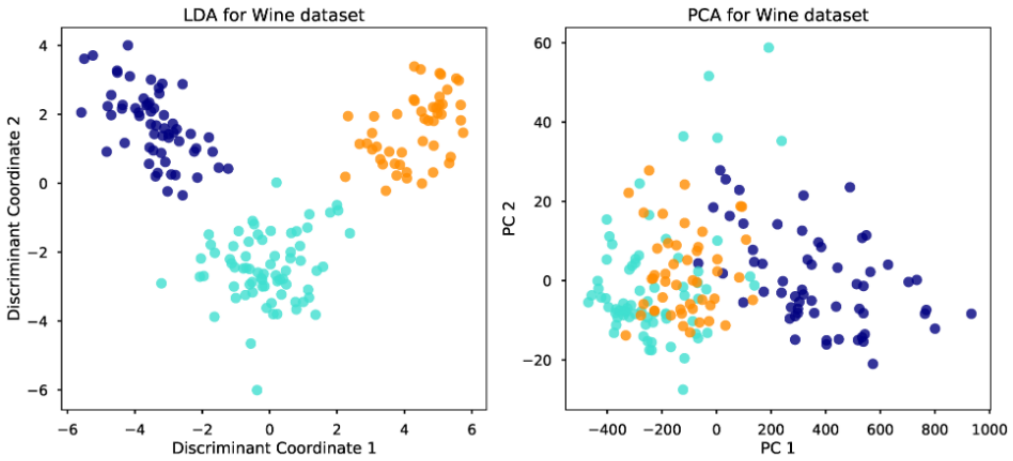

Рис. 2: Canonical Discriminant Analysis and PCA on data

Вычисляем межклассовую ковариационную матрицу:

$$\mathbf{H} = \sum_{i=1}^{K} n_i (\hat{\mu}_i - \hat{\mu})^{\mathrm{T}} (\hat{\mu}_i - \mu) \tag{12}$$

$\zeta = A\xi$ — новый признак.

На выборочном языке новые признаки $\mathbf{Z} = A^{\mathrm{T}}\mathbf{X}$. Выборочная ковариационная матрица новых признаков:

$$A^{\mathrm{T}}\mathbf{T}A = A^{\mathrm{T}}(\mathbf{E} + \mathbf{H})A = A^{\mathrm{T}}\mathbf{E}A + A^{\mathrm{T}}\mathbf{H}A, \tag{13}$$

$\mathbf{T}$ — total covariance matrix, первое слагаемое — оценка внутригрупповых отклонений, второе — межгрупповых. Переходим к обобщенной задаче на собственные числа и собственные вектора:

$$\frac{A^{\mathrm{T}}\mathbf{H}A}{A^{\mathrm{T}}\mathbf{E}A} \to \max_{A}. \tag{14}$$

$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$ — собственные числа матрицы $\mathbf{E}^{-1}\mathbf{H}$, $A_1, \ldots, A_d$ — соответствующие им собственные вектора. Тогда максимум выше равен λ_1 и достигается на A_1. В этих обозначениях, канонические коэффициенты являются собственными векторами матрицы $\mathrm{E}^{-1}\mathrm{H}$, новые признаки Z_i — канонические переменные, Z_i ортогональны.

Далее

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

Теперь нас интересует, какое количество таких канонических переменных брать. Проверяем гипотезу

$$H_0 : A_i, i = \ell, \ldots, d.$$

Используем статистику $\Lambda - prime$

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

Гипотеза может быть представлена как

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

Критерий:

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

Другие статистики для проверки гипотезы:

- 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: $V = trace(\mathbf{H}\mathbf{E}^{-1})$.

2.4 Квадратичный дискриминантный анализ

Применяем QDA, когда каждый класс имеет многомерное нормальное распрелеление и различные ковариационные матрицы $\Sigma_{i=1}^K$. Классифицирующие функции принимают вид

$$g_i(x) = log\ f_i(x) = log\ \pi_i - \frac{1}{2} log|\Sigma_i| - \frac{1}{2}(x^{\mathsf{T}} - \mu_i)\Sigma_i^{-1}(x - \mu_i). \qquad (15)$$

2.5 Регуляризованный дискриминантный анализ, RDA

RDA представляет собой компромисс между LDA и QDA. Стягиваем ковариации каждого класса к общей матрице

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$$

где $\hat{\Sigma}_k$ — оценка из QDA, $\hat{\Sigma}$ — оценка из LDA. $\alpha \in [0;1]$ — настраиваемый параметр, определяющий, оценивать ли ковариации отдельно $\alpha = 1$ или совместно $\alpha = 0$ (pooled).

Или к единичной матрице

$$\hat{\Sigma}_k(\gamma) = \gamma\hat{\Sigma}_k + (1-\gamma)\hat{\sigma}\mathbf{I}$$

Так как RDA это способ регуляризации, он используется при наличие мультиколлинеарности.

3 Логистическая регрессия

Вместо предсказания бинарной переменной мы предсказываем непрерывную переменную со значениями на отрезке [0,1] при любых значениях независимых переменных. Из этой ситуации можно выйти так: научить линейную модель правильно предсказывать какой-то объект, связанный с вероятностью, но с диапазоном значений $(-\infty, \infty)$, и преобразовать ответы модели в вероятность. Используем logit преобразование, логарифм отношения вероятности положительного события к отрицательному $\log(\frac{p}{1-\mathrm{p}})$.

Если ответом модели является $\log(\frac{p}{1-\mathrm{p}})$, то легко получаем, что

$$\langle w, x_i\rangle = \log(\frac{p}{1-p}),$$

$$e^{\langle w,x_i\rangle} = \frac{p}{1-p},$$

$$\mathrm{p} = \frac{1}{1+e^{-\langle w,x_i\}}}.$$

Функция в правой части – сигмоида, и обозначается

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

Сигмоида обладает следующими свойствами, которые нам интересны:

- Монотонно возрастает;
- отображает всю числовую прямую на интервал (0; 1);
- $\sigma(-x) = 1 - \sigma(x)$.

Функция потерь имеет вид (Оранжевым цветом на рис. 1)

$$\mathcal{L}(M_i) = \log(1 + e^{-M}). \tag{16}$$

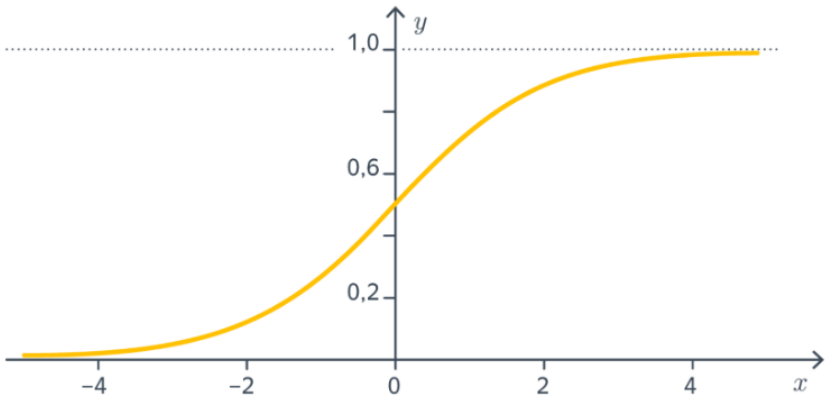

Рис. 3: Логистическая кривая (Сигмоида)

Таким образом,

$$p = \sigma(\langle w, X_i \rangle).$$

Как теперь научиться оптимизировать w так, чтобы модель как можно лучше предсказывала логиты? Нужно применить метод максимума правдоподобия для распределения Бернулли. Записываем функцию правдоподобия для распределения Бернулли:

$$p(y|X,\ w) = \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}.$$

Приводим к логарифмическому виду:

$$\ell(w, X, y) = \sum_{i=1}^{n}(y_i \log(p_i) + (1-y_i)\log(1-p_i)) =$$

$$= \sum_{i=1}^{n}(y_i \log(\sigma(\langle w, x_i \rangle)) + (1-y_i)\log(1-\sigma(\langle w,\ x_i \rangle))).$$

Т.к. справедливо

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

то переписываем функцию правдоподобия следующим образом:

$$\ell(w, X, y) = \sum_{i=1}^{n}(y_i \log(\sigma(\langle w, x_i \rangle)) + (1-y_i)\log(\sigma(-\langle w,\ x_i \rangle))).$$

Максимизация правдоподобия эквивалентна минимизации функционала, гладко аппроксимирующего эмпирический риск. Чтобы получить функцию потерь, которую мы будем минимизировать, умножаем на -1 и получаем

$$\mathcal{L}(w, X, y) = -\sum_{i=1}^{n}(y_i \log(\sigma(\langle w, x_i \rangle)) + (1-y_i)\log(\sigma(-\langle w, x_i \rangle))).$$

Методы решения задачи минимизации — метод стохастического градиента, метод Ньютона-Рафсона.

3.1 Алгоритм Ньютона-Рафсона

Существуют различные методы решения нелинейных систем, среди которых наиболее популярным и обеспечивающим наилучшую сходимость является метод Ньютона–Рафсона. Метод предполагает выбор некоторого начального приближения решения и последовательное его улучшение в ходе выполнения ряда вычислений. Гессиан логарифма правдоподобия

$$\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^{\mathrm{T}}} = -\sum_{i=1}^{N} x_i x_i^{\mathrm{T}} p(\mathbf{x}_i; \beta)(1 - p(\mathbf{x}_i; \beta)) = 0. \tag{17}$$

Пусть β^{old} – некоторое начальное приближение вектора коэффициентов β, на каждой итерации уточняется следующим образом:

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^{\mathrm{T}}} \right)^{-1} \frac{\partial \ell(\beta)}{\partial \beta},$$

где производные вычисляются в точке β^{old}.

$\mathbf{y}$ – ответы y_i, $\mathbf{p} = (p(x_i; \beta^{old}))$, $\mathbf{W}$ – диагональная матрица размером $N \times N$ весов, где iй элемент имеет вид $p(x_i; \beta^{old})(1 - p(x_i; \beta^{old}))$. В матричных обозначениях

$$\frac{\partial \ell(\beta)}{\partial \beta} = \mathbf{X}^{\mathrm{T}}(\mathbf{y} - \mathbf{p})$$
$$\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^{\mathrm{T}}} = -\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X}.$$

Шаг алгоритма имеет вид

$$\begin{aligned} \beta^{new} &= \beta^{old} + (\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}(\mathbf{y} - \mathbf{p}) = \\ &= (\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W}(\mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})) = \\ &= (\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{z}. \end{aligned}$$

Мы переписали итерацию алгоритма как взвешенную регрессию, где в качестве ответа выступает вектор

$$\mathbf{z} = \mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p}).$$

На каждом шаге $\mathbf{p}$ меняется, а вместе с ним и $\mathbf{W}$, $\mathbf{z}$. Этот алгоритм называется iteratively reweighted least squares (IRLS) так как на каждом шаге решается задача

$$\beta^{new} = \arg\min_{\beta}(\mathbf{z} - \mathbf{X}\beta)^{\mathrm{T}}\mathbf{W}(\mathbf{z} - \mathbf{X}\beta).$$

В качестве начального приближения β^{old} можно взять оценки, полученные с помощью обычной линейной регрессии или просто $\beta^{old} = 0$. Сходимость нам не гарантируется, но обычно алгоритм сходится так как логарифм правдоподобия вогнутый.

3.2 Регуляризация

Аналогично обычной линейной регрессии, можно отбирать признаки (осуществлять feature selection) с помощью LASSO (L1) или аналога Ridge Regression (L2). Для этого максимизируем соответственно

$$\max_{\beta_0,\beta} \sum_{i=1}^{N} \Big(y_i(\beta_0 + \beta^{\mathrm{T}}\mathbf{x}_i) - \log(1 + e^{\beta_0 + \beta^{\mathrm{T}}\mathbf{x_i}}) \Big) - \lambda \sum_{j=1}^{p} |\beta_j|,$$

$$\max_{\beta_0,\beta} \sum_{i=1}^{N} \Big(y_i(\beta_0 + \beta^{\mathrm{T}}\mathbf{x}_i) - \log(1 + e^{\beta_0 + \beta^{\mathrm{T}}\mathbf{x_i}}) \Big) - \lambda \sum_{j=1}^{p} \beta_j^2.$$

Для нахождения точки максимума можно снова использовать алгоритм Ньютона-Рафсона.

4 Метод опорных векторов

Линейный классификатор, задача классификации в рамках обучения с учителем. Имеется выборка $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, $x_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$; задачей является построение классифицирующего правила $f : \mathbb{R}^p \to \{-1, 1\}$. Не накладывается ограничений на распределение.

4.1 Hard-margin SVM

Линейный классификатор:

$$f(x_i) = sign(\langle x_i, w\rangle - w_0), \tag{18}$$

Уравнение, описывающее гиперплоскость, разделяющую классы в пространстве $\mathbb{R}$:

$$\langle w, x\rangle = w_0.$$

Предположим, что присутствует линейная разделимость классов, то есть существуют такие значения параметров w, w_0, при которых функционал числа ошибок принимает нулевое значение:

$$Q(w,\ w_0) = \sum_{i=1}^{n} [y_i(\langle w,\ x_i\rangle - w_0) < 0] \tag{19}$$

Вводим критерий оптимальности: максимальное расстояние между двумя гиперплоскостями, параллельных данной и симметрично расположенных относительно неё, при котором между ними не находится ни одна из точек.

Каждой из двух параллельных гиперплоскостей будет принадлежать некоторое количество точек из соответствующего класса; точки, которые принадлежат одной из гиперплоскостей — будем называть опорными векторами. Параметры линейного порогового классификатора определены с точностью до нормировки: алгоритм не изменится, если w и w_0 одновременно умножить на одну и ту же положительную константу. Удобно выбрать её таким образом, чтобы для всех пограничных (ближайших к разделяющей гиперплоскости) объектов x_i из выборки выполнялись условия. Записать можно так:

$$\langle w,\ x_i\rangle - w_0 = y_i.$$

Сделать это возможно, поскольку при оптимальном положении разделяющей гиперплоскости все пограничные объекты находятся от неё на одинаковом расстоянии. Для всех x_i справедливо

$$\langle w, x_i \rangle - w_0 \begin{cases} \leq -1, & i\ y_i = -1; \\ \geq 1, & i\ y_i = +1. \end{cases}$$

Условие $-1 < \langle w, x \rangle - w_0 < 1$ задаёт полосу, разделяющую классы. Ни одна из точек обучающей выборки не может лежать внутри этой полосы. Границами служат две параллельные гиперплоскости с направляющим вектором w. При этом сама разделяющая гиперплоскость проходит ровно по середине полосы.

Чтобы разделяющая гиперплоскость как можно дальше отстояла от точек выборки, ширина полосы должна быть максимальной. Пусть x_- и x_+ две произвольные точки классов -1 и $+1$ соответственно, лежащие на границе полосы. Тогда ширина полосы есть

$$\langle (x_+ - x_-),\ \frac{w}{\|w\|} \rangle = \frac{\langle w, x_+ \rangle - \langle w, x_- \rangle}{\|w\|} = \frac{(w_0 + 1) - (w_0 - 1)}{\|w\|} = \frac{2}{\|w\|}.$$

Ширина полосы максимальна, когда норма вектора w минимальна.

Таким образом, когда выборка линейно разделима, имеем следующую задачу — найти такие значения параметров w и w_0, при которых норма вектора w минимальна при условии выше. Это задача квадратичного программирования.

Построение оптимальной разделяющей гиперплоскости сводится к минимизации квадратичной формы

$$\begin{cases} \frac{1}{2}\|w\|^2 \to \min\limits_{\boldsymbol{w}}; \\ (\langle x_i, w \rangle - w_0)\, y_i \geq 1. \end{cases}$$

На практике линейно разделимые классы встречаются довольно редко. Поэтому постановку задачи необходимо модифицировать так, чтобы система ограничений была совместна в любой ситуации.

4.2 Soft-margin SVM

Позволим алгоритму допускать ошибки на обучающих объектах, но при этом постараемся, чтобы ошибок было поменьше. Введём набор дополнительных переменных $\xi > 0$, характеризующих величину ошибки на объектах $x_i, i = 1, ..., n$ (C задает размер штрафа за ошибки.).

$$\begin{cases} \frac{1}{2}\langle w,\ w\rangle + C\sum_{i=1}^{n}\xi_i \to \min\limits_{w,w_0,\xi}; \\ y_i(\langle w,\ x_i\rangle - w_0) \geq 1 - \xi_i,\ \forall i; \\ \xi_i \geq 0,\ \forall i. \end{cases}$$

Помним, что в задачах с двумя классами $Y = -1, +1$ отступом (margin) объекта x_i от границы классов называется величина

$$[M_i < 0] \leq (1 - M_i)_+.$$

Алгоритм (18) допускает ошибку на объекте x_i только когда отступ M_i отрицателен. Если $M_i \in (-1, +1)$, то объект x_i попадает внутрь разделяющей полосы. Если $M_i > 1$, то объект x_i классифицируется правильно, и находится на некотором удалении от разделяющей полосы. Ошибка ξ_i выражается через отступ M_i.

В силу требования минимизации суммы $\sum_{i=1}^{n}\xi_i$, можем записать, что $\xi_i = (1 - M_i)_+$. Задача оказывается эквивалентной безусловной минимизации функционала Q, не зависящего от ξ_i:

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

В силу неравенста $[M_i < 0] \leq (1 - M_i)_+$, функционал Q можно рассматривать как верхнюю оценку эмпирического риска (числа ошибочных классификаций объектов обучающей выборки), к которому добавлен регуляризатор $\|w\|^2$, умноженный на параметр регуляризаци $\dfrac{1}{2\mathrm{C}}$.

Замена пороговой функции потерь $[M < 0]$ кусочно-линейной верхней оценкой $\mathcal{L}(M) = (1 - M)_+$ делает функцию потерь чувствительной к величине ошибки. Функция потерь $\mathcal{L}(M)$ штрафует объекты за приближение к границе классов. Введение регуляризатора повышает устойчивость решения

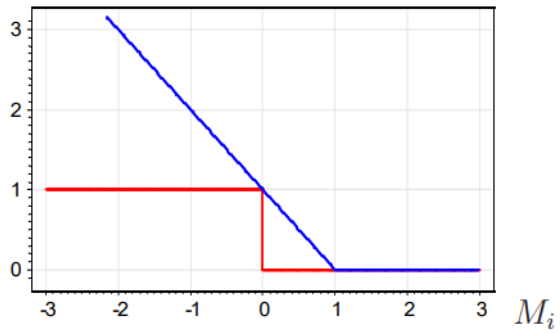

Рис. 4: Кусочно-линейная аппроксимация пороговой функции потерь: $[M_i < 0] \leq (1 - M_i)_+$

Двойственная задача. Запишем функцию Лагранжа:

$$\mathcal{L}(w, w_0, \xi; \lambda, \eta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi - \sum_{i=1}^{n}\lambda_i(M_i(w, w_0) - 1 + \xi_i) - \sum_{i=1}^{n}\xi_i\eta_i =$$

$$= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n}\lambda_i(M_i(w, w_0) - 1) - \sum_{i=1}^{n}\xi_i(\lambda_i + \eta_i - C),$$

где $\lambda = (\lambda_1, \ldots, \lambda_n)$ — вектор переменных, двойственных к w; $\eta = (\eta_1, \ldots, \eta_n)$ — вектор переменных, двойственных к ξ.

Согласно теореме Куна-Таккера, задача эквивалентна двойственной задаче поиска седловой точки функции Лагранжа:

$$\begin{cases} \mathcal{L}(w, w_0, \xi; \lambda, \eta) \to \min\limits_{w, w_0, \xi}\max\limits_{\lambda, \eta}; \\ \xi_i \geq 0,\ \lambda_i \geq 0,\ \eta_i \geq 0,\ i = 1, \ldots, n; \\ \lambda_i = 0\ M_i(w,\ w_0) = 1 - \xi_i,\ i = 1, \ldots, n; \\ \eta_i = 0\ \xi_i = 0,\ i = 1, \ldots, \ell; \end{cases}$$

В последних двух строках записаны условия дополняющей нежёсткости.

Необходимым условием седловой точки функции Лагранжа является равенство нулю её производных. Отсюда получаются соотношения:

$$\begin{cases} \frac{\partial}{\partial w}: & w = \sum\limits_{i=1}^{n} \lambda_i y_i x_i \\ \frac{\partial}{\partial w_0}: & 0 = \sum\limits_{i=1}^{n} \lambda_i y_i \\ \frac{\partial}{\partial \xi_i}: & \lambda_i = C - \eta_i \end{cases}$$

Из первого равенства следует, что искомый вектор весов w является линейной комбинацией векторов обучающей выборки x_i, причём только тех, для которых $\lambda_i > 0$. Используя эти равенства, получаем

$$\begin{cases} -\mathcal{L}(\lambda) = -\sum\limits_{i=1}^{n} \lambda_i + \frac{1}{2} \sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} \lambda_i \lambda_j y_i y_k \langle x_i, x_j \rangle \to \min\limits_{\lambda} \\ 0 \leq \lambda_i \leq C, \forall i \\ \sum\limits_{i=1}^{n} \lambda_i y_i = 0 \end{cases}$$

Здесь минимизируется квадратичный функционал, имеющий неотрицательно определённую квадратичную форму, следовательно, выпуклый. Область, определяемая ограничениями неравенствами и одним равенством, также выпуклая. Следовательно, данная двойственная задача имеет единственное решение.

Константу C обычно выбирают по критерию скользящего контроля. Трудоемко, задачу приходится решать заново при каждом значении C. Если полагаем, что выборка почти линейно разделима, и лишь объекты-выбросы классифицируются неверно, то можно применить фильтрацию выбросов.

Уже показали, как вычисляеся w. Для определения порога w_0 достаточно взять произвольный опорный граничный вектор x_i и выразить w_0 из равенства $w_0 = \langle w, x_i \rangle - y_i$.

В итоге алгоритм классификации представляется в следующем виде:

$$a(x) = \text{sign} \left(\sum_{i=1}^{n} \lambda_i y_i \langle x_i,\ x \rangle - w_0 \right)$$

Суммирование идёт не по всей выборке, а только по опорным векторам, для которых $\lambda_i \neq 0$. Классификатор не изменится, если все остальные объекты исключить из выборки.

Но ненулевыми λ_I обладают не только опорные объекты, но и объекты-нарушители. Это недостаток SVM, т.к. ими часто оказываются шумовые выбросы, и построенное на них решающее правило опирается на шум.

4.3 Ядра и спрямляющие пространства (The Kernel Trick)

Подход к решению проблемы линейной неразделимости. Переход от исходного пространства признаковых описаний объектов X к новому пространству H с помощью некоторого преобразования $\psi : X \to H$. Пространство H должно быть наделено скалярным произведением, в частности, подойдёт любое евклидово, а в общем случае и гильбертово.

Функция $K : X \times X \to \mathbb{R}$ называется ядром (kernel function), если она представима в виде $K(x,\ x') = \langle \psi(x), \psi(x') \rangle$ при некотором отображении $\psi : X \to H$, где H — пространство со скалярным произведением.

Алгоритм зависит только от скалярных произведений объектов, но не от самих признаковых описаний. Скалярное произведение $\langle x, x' \rangle$ можно формально заменить ядром $K(x,\ x')$.

Функция $K(x,\ x')$ является ядром только тогда, когда она симметрична, $K(x,\ x') = K(x',\ x)$, и неотрицательно определена (теор. Мерсера) $\int_X \int_X K(x,\ x')g(x)g(x')dxdx' \geq 0$ для любой функции $g : X \to \mathbb{R}$.

Способы построения ядер:

- Произвольное скалярное произведение $K(x,\ x') = \langle x, x' \rangle$;
- Константа $K(x,\ x') = 1$ является ядром;
- Произведение ядер $K(x,\ x') = K_1(x,\ x')K_2(x,\ x')$ является ядром;
- Для любой функции $\psi : X \to \mathbb{R}$ произведение $K(x,\ x') = \psi(x)\psi(x')$ является ядром.
- Линейная комбинация ядер с неотрицательными коэффициентами $K(x,\ x') == \alpha_1 K_1(x,\ x') + \alpha_2 K_2(x,\ x')$ является ядром;
- Композиция произвольной функции $\varphi : X \to X$ и произвольного ядра K_0 является ядром: $K(x,\ x') = K_0(\varphi(x),\ \varphi(x'))$.

Наиболее часто используемые ядра: линейное ,полиномиальное, радиальное, сигмовидное.

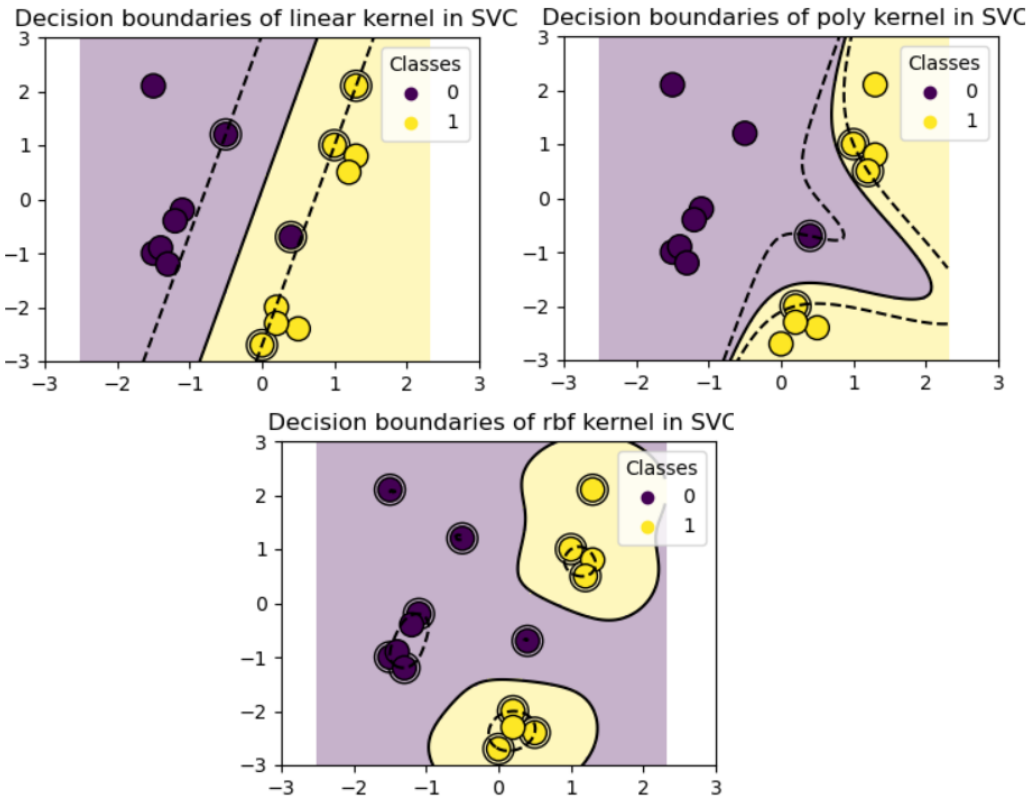

Рис. 5: Plot classification boundaries with different SVM Kernels

5 Метод стохастического градиента

Стохастический градиентный спуск - оптимизационный алгоритм, при котором градиент оптимизируемой функции считается на каждой итерации от случайно выбранного элемента выборки. Используем старые обозначения, дополнив, что $a(x, \omega)$ — семейство алгоритмов с параметром вектора весов ω.

Стохастический градиентный спуск является модификацией градиентного спуска, и в случае линейного классификатора искомый алгоритм имеет вид:

$$a(x, \omega) = \phi(\sum_{j=1}^{N} \omega_j x^j - \omega_0), \tag{20}$$

где $\phi(z)$ - функция активации.

Решаемая задача:

$$Q(\omega) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(a(x_i, \omega), y_i) \to \min_{\omega}.$$

Обновление весов:

$$\omega^{(t+1)} = \omega^{(t)} - h \nabla Q(\omega^{(t)}). \tag{21}$$

Проблема GD — чтобы определить новое приближение вектора весов необходимо вычислить градиент от каждого элемента выборки, что может сильно замедлять алгоритм. Идея ускорения заключается в использовании либо одного элемента (SGD), либо некоторой подвыборки (Batch GD) для получения нового приближения весов.

Веса, при подходе SGD, обновляются следующим образом:

$$\omega^{(t+1)} = \omega^{(t)} - h \nabla \mathcal{L}(a(x_i, \omega^{(t)}), y_i),$$

где i — случайно выбранный индекс.

Новый стохастический градиент должен быть несмещённой оценкой полного градиента (сходимость SGD обеспечивается несмещенностью). Т.к. направление изменения ω будет определяться за $O(1)$, подсчет Q на каждом шаге будет слишком дорогостоящим. Для ускорения оценки, будем использовать одну из рекуррентных формул:

- Среднее арифметическое: $\overline{Q}_m = \frac{1}{m}\mathcal{L}(a(x_{i_m}, \omega^{(m)}), y_{i_m}) + (1 - \frac{1}{m})\overline{Q}_{m-1}$;
- Экспоненциальное скользящее среднее: $\overline{Q}_m = \lambda\mathcal{L}(a(x_{i_m}, \omega^{(m)}), y_{i_m}) + (1 - \lambda)\overline{Q}_{m-1}$, где λ — темп забывания "предыстории"ряда.

Так как алгоритм итеративный, нужно задать начальные значения весов оптимизируемой модели. Существует несколько способов инициализации весов, например:

- $\omega_0 = 0$;
- $\omega_0 = random(-\frac{1}{2n}, \frac{1}{2n})$;

Достоинства: метод легко реализуется, функция потерь и семейство алгоритмов могут быть любыми, подходит для задач с большими данными, иногда можно получить решение даже не обработав всю выборку;

Недостатки: проблемы с локальными экстремумами (для борьбы с этим используют технику "встряхивания коэффициентов"(jog of weights)); направление обновляется на основании получаемых в данный момент данных; алгоритм может не сходиться или сходиться слишком медленно.

Альтернативы:

- Метод импульса — запоминает $\Delta\omega$ на каждой итерации и определяет следующее изменение в виде линейной комбинации градиента и предыдущего изменения;
- AdaGrad — модификация SGD с отдельной для каждого параметра скоростью обучения;
- RMSProp — это метод, в котором скорость обучения настраивается для каждого параметра. Идея заключается в делении скорости обучения для весов на скользящие средние значения недавних градиентов для этого веса;
- Adam — ADAptive Momentum. Как правило, в этом алгоритме подбирают лишь один гиперпараметр — learning rate. Требует хранения как параметров модели, так и градиентов, накопленного импульса и нормировочных констант. Достижение более быстрой (с точки зрения количества итераций/объема рассмотренных данных) сходимости требует больших объемов памяти.

6 Выбор модели

Обозначим алгоритм классификации, аппроксимирующий целевую зависимость на всем множестве $\mathbf{X}$ по обучающей выборке $X^n = (x_i, y_i)_{i=1}^n$, как $a : X \to Y$. Задача — найти модель, наилучшую по определенному критерию. Задаем функционал средней ошибки алгоритма как

$$Q(a, X^n) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(a, x_i). \tag{22}$$

Функция потерь $\mathcal{L}$ характеризует величину ошибки алгоритма a на объекте x. Пример такой функции; $\mathcal{L}(a, x) = [a(x) \neq y*(x)]$.

6.1 Критерии выбора модели. Скользящий контроль

Критерий выбора — функционал $Q(\mu, X^n)$, характеризующий качество метода μ. Так, метод минимизации эмпирического риска строит алгоритм с минимальным значением критерия

$$\mu(X^n) = \arg\min_{a \in A} Q(a, X^n). \tag{23}$$

Скользящий контроль (cross-validation, CV) — процедура эмпирического оценивания, с помощью которой "эмулируется" наличие тестовой выборки, которая не участвует в обучении, но для которой имеются метки класса. Разбиение выборки: $X^L = X^n \cup X^k$.

$$CV(\mu, X^n) = \frac{1}{N} \sum_{n=1}^{N} Q(\mu(X_n^n), X_n^k). \tag{24}$$

Существует несколько вариантов скользящего контроля (Complete CV, leave-one-out CV, q-fold CV). Complete CV строится по всем $N = C_L^K$, вычислительно слишком сложен и применяется в теоретических исследования или если можно вывести удобную вычислительную формулу. На практике интересны другие 2 варианта.

Leave-one-out CV:

$$LOO(\mu, X^L) = \frac{1}{L} \sum_{i=1}^{L} Q(\mu(X^L \backslash \{x_i\}), \{x_i\}). \tag{25}$$

Каждый объект участвует в контроле один раз, длина обучающих подвыборок на единицу меньше длины полной выборки. Обучать приходится L раз, что тоже трудоемко.

Q-fold CV — случайное разбиение выборки на q непересекающихся блоков одинаковой длины $l_1, ..., l_n; X^L = X_1^{l_1} \cup \cdots \cup X_q^{l_q}$, $l_1 + \cdots + l_q = L$. Блок по очереди становится контрольной подвыборкой, обучение по остальным q-1 блокам. Критерий определяется как средняя по всем блокам ошибка.

$$Q_{ext}(\mu, X^L) = \frac{1}{q} \sum_{n=1}^{q} Q(\mu(X^L \backslash X_n^{l_n}), X_n^{l_n}). \tag{26}$$

Для выбора оптимального метода рекомендуется использовать несколько разные критериев. Например, отбираем несколько лучших методов по критерию, затем из них выбрать тот, для которого другой критерий принимает наименьшее значение.