

Теоретический минимум

1. Что такое объект, целевая переменная, признак, модель, функционал ошибки и обучение?

Объекты – абстрактные сущности (но компьютеры работают только с числами).

Целевая переменная - ?

Обучающая выборка (training set) – конечный набор объектов, для которых известны значения целевой переменной. Пример: набор ресторанов, открытых более года назад, для которых известна их прибыль за первый год.

Признаки, факторы (features) – количественные характеристики объекта.

Для решения задачи обучения по прецедентам в первую очередь фиксируется модель восстанавливаемой зависимости.

Функционал ошибки – функционал, измеряющий качество работы алгоритма.

Признак (feature) объекта – это результат измерения некоторой характеристики объекта. Формально признаком называется отображение $f : X \rightarrow D_f$, где D_f – множество допустимых значений признака.

Функция потерь (loss function) – это неотрицательная функция $L(a, x)$, характеризующая величину ошибки алгоритма a на объекте x . Если $L(a, x) = 0$, то ответ $a(x)$ называется корректным.

Процесс подбора оптимального параметра модели θ по обучающей выборке X^l называют настройкой (fitting) или обучением (training, learning) алгоритма $a \in A$.

Моделью алгоритмов называется параметрическое семейство отображений $A = \{g(x, \theta) | \theta \in \Theta\}$, где $g : X \times \theta \rightarrow Y$ – некоторая фиксированная функция, Θ – множество допустимых значений параметра θ , называемое пространством параметров или пространством поиска (search space).

2. Что такое переобучение и недообучение?

Минимизацию эмпирического риска следует применять с известной долей осторожности. Если минимум функционала ошибки достигается на алгоритме a , то это ещё не гарантирует, что a будет хорошо приближать целевую зависимость на произвольной контрольной выборке. Когда качество работы алгоритма на новых объектах, не вошедших в состав обучения, оказывается существенно хуже, чем на обучающей выборке, говорят об эффекте переобучения (overtraining) или переподгонки (overfitting). При решении практических задач с этим явлением приходится сталкиваться очень часто.

Переобучение, переподгонка (overtraining, overfitting) – нежелательное явление, возникающее при решении задач обучения по прецедентам, когда вероятность ошибки обученного алгоритма на объектах тестовой выборки оказывается существенно выше, чем средняя ошибка на обучающей выборке. Переобучение возникает при использовании избыточно сложных моделей.

Недообучение – нежелательное явление, возникающее при решении задач обучения по прецедентам, когда алгоритм обучения не обеспечивает достаточно малой величины средней ошибки на обучающей выборке. Недообучение возникает при использовании недостаточно сложных моделей.

3. Задачи обучения с учителем и без учителя - определения и примеры

Если нам известны значения целевой переменной, то есть алгоритм обучается так, чтобы правильно предсказывать целевую переменную – это обучение с учителем. Сюда относят классификацию, регрессию и ранжирование.

Если нам неизвестны значения целевой переменной или целевая переменная вообще отсутствует, то есть алгоритм обучается только по признакам объектов, то это обучение без учителя. Примерами обучения с учителем являются кластеризация, понижение размерности и др.

Кластеризация (обучение без учителя) отличается от классификации (обучения с учителем) тем, что метки исходных объектов y_i изначально не заданы, и даже может быть неизвестно само множество Y .

4. Запишите формулы для линейной модели регрессии и для средне-квадратичной ошибки.

$a(x) = \sum_{j=1}^n w_j x_j + w_0 = (w, x)$
 $Q(a, x) = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2 = \frac{1}{l} \sum_{i=1}^l ((w, x_i) - y_i)^2 \rightarrow \min_w$, где l - количество объектов.

5. Что такое градиент? Какое его свойство используется при минимизации функций?

Градиент – это вектор, в направлении которого функция быстрее всего растёт. Антиградиент (вектор, противоположный градиенту) – вектор, в направлении которого функция быстрее всего убывает. На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь.

6. Запишите формулу для одного шага градиентного спуска. Какие способы оценивания градиента вы знаете? Зачем они нужны?

$$w^{(k)} = w^{(k-1)} - \eta \nabla Q(w^{(k-1)})$$

Методы оценивания градиента:

1) Stochastic gradient descent (SGD): на каждом шаге выбираем один случайный объект и сдвигаемся в сторону антиградиента по этому объекту:

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla q_{i_k}(w^{(k-1)})$$

Скорость сходимости:

$$\mathbb{E}(Q(w^{(k)}) - Q(w^*)) = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$$

Менее трудоёмкий метод, но медленнее сходится.

2) Stochastic average gradient (SAG): инициализируем веса w_j и инициализируем вспомогательные переменные $z^{(1)}, z^{(2)}, \dots, z^{(i)} = \nabla q_i(w)$. Далее на каждом шаге выбираем случайный объект и обновляем градиент по нему (все остальные градиенты остаются такими же, как на предыдущем шаге):

$$z_i^{(k)} = \begin{cases} \nabla q_i(w^{(k-1)}), & i = i_k \\ z_i^{(k-1)}, & \text{иначе.} \end{cases}$$

Формула градиентного шага:

$$w^{(k)} = w^{(k-1)} - \eta_k \sum_{i=1}^l z_i^{(k)}$$

Скорость сходимости:

$$\mathbb{E}(Q(w^{(k)}) - Q(w^*)) = \mathcal{O}\left(\frac{1}{k}\right)$$

3) Метод моментов.

Вектор инерции (усреднение градиента по предыдущим шагам):

$$h_0 = 0$$

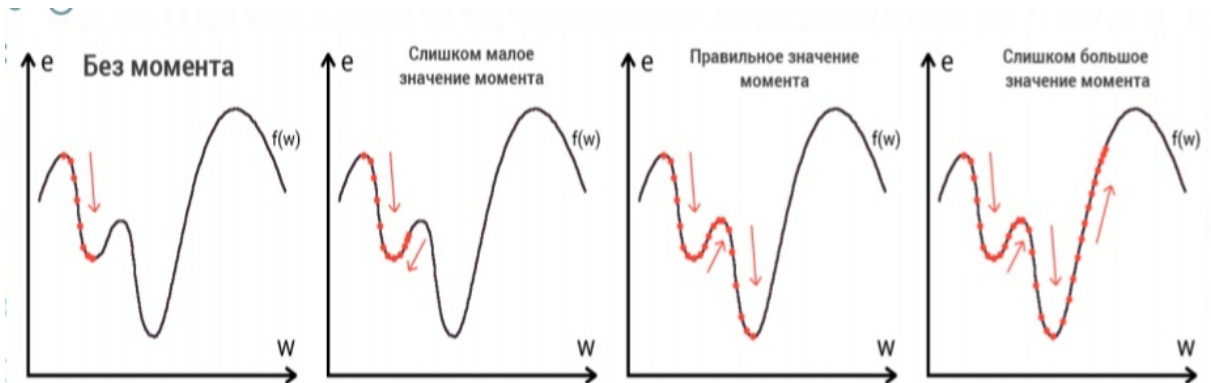
$$h_k = \alpha h_{k-1} + \eta_k \nabla Q(w^{(k-1)})$$

Формула метода моментов:

$$w^{(k)} = w^{(k-1)} - h_k$$

Подробнее:

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla Q(w^{(k-1)}) - \alpha h_{k-1}$$



4) Adagrad (Adaptive Gradient)

Сумма квадратов обновлений:

$$g_{k-1,j} = (\nabla Q(w^{(k-1)}))_j^2$$

формулы методов Adagrad:

$$G_{k,j} = G_{k-1,j} + g_{k-1,j} = G_{k-1,j} + (\nabla Q(w^{(k-1)}))_j^2$$

$$\omega_j^{(k)} = \omega_j^{(k-1)} - \frac{\eta}{\sqrt{G_{k,j} + \epsilon}} (\nabla Q(w^{(k-1)}))_j$$

Этот метод использует адаптивный шаг обучения – тем самым мы регулируем скорость сходимости метода.

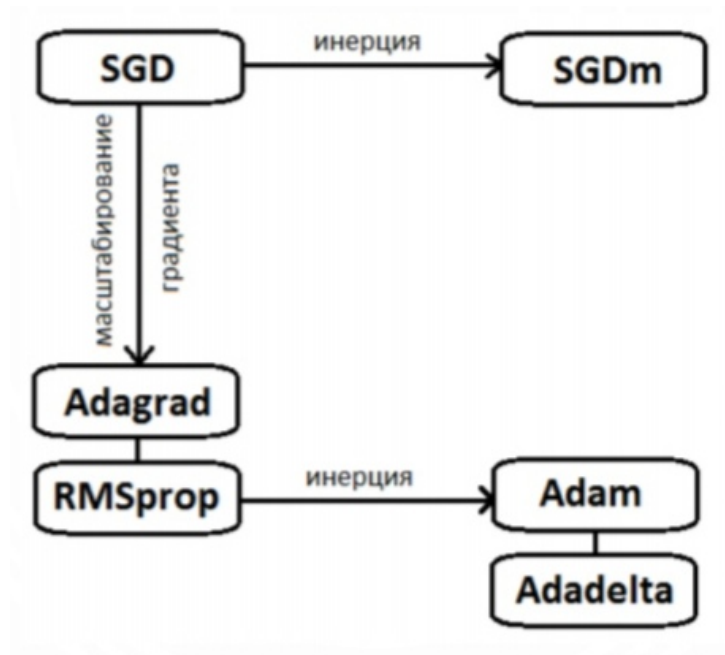
Плюс - автоматическое затухание скорости обучения. Минус - $G_{k,j}$ монотонно возрастают, поэтому шаги укорачиваются, и мы можем не успеть дойти до минимума.

5) RMSPROP (root mean square propagation)

Метод реализует экспоненциальное затухание градиентов.

Формулы метода RMSprop (усредненный по истории квадрат градиента):

$$G_{k,j} = \alpha G_{k-1,j} + (1 - \alpha) g_{k-1,j}$$
$$\omega_j^{(k)} = \omega_j^{(k-1)} - \frac{\eta}{\sqrt{G_{k,j} + \epsilon}} (\nabla Q(w^{(k-1)}))_j$$



7. Что такое кросс-валидация? На что влияет количество блоков в кросс-валидации?

Кросс-валидация (cross-validation, CV) — процедура эмпирического оценивания обобщающей способности алгоритмов, обучаемых по прецедентам.

Фиксируется некоторое множество разбиений исходной выборки на две подвыборки: обучающую и контрольную. Для каждого разбиения выполняется настройка алгоритма по обучающей подвыборке, затем оценивается его средняя ошибка на объектах контрольной подвыборки. Оценкой скользящего контроля называется средняя по всем разбиениям величина ошибки на контрольных подвыборках.

1) Разбиваем объекты на тренировку (train) и валидацию (validation) несколько раз (при разбиении k раз получаем k -fold кросс-валидацию).

2) Для каждого разбиения вычисляем качество на валидационной части.

3) Усредняем полученные результаты.

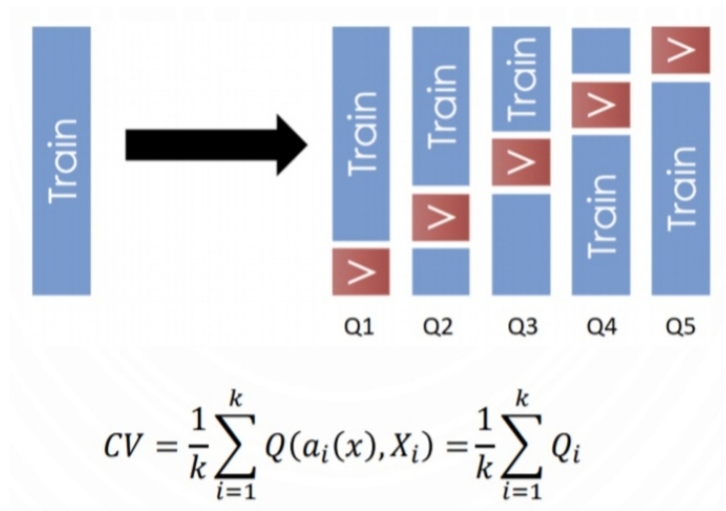
Виды кросс-валидации:

1) k -fold cross-validation – разбиваем данные на k блоков, каждый из которых по очереди становится контрольным (валидационным)

2) Complete cross-validation – перебираем ВСЕ разбиения

3) Leave-one-out cross-validation – каждый блок состоит из одного объекта (число блоков = числу объектов)

Маленькое k – оценка может быть пессимистично занижена из-за маленького размера тренировочной части. Большое k – оценка может быть неустойчивой из-за маленького размера валидационной части.



8. Что такое регуляризация? Почему L_1 -регуляризация отбирает признаки?

Утверждение. Если в выборке есть линейно-зависимые признаки, то задача оптимизации функционала ошибки бесконечное число решений.

Большие значения параметров (весов) модели w – признак переобучения.

Решение проблемы – регуляризация. Будем минимизировать регуляризованный функционал ошибки:

$$Q_\alpha(w) = Q(w) + \alpha R(w) \rightarrow \min_w$$

Регуляризация штрафует за слишком большие веса.

Наиболее используемые регуляризаторы:

$$R_2(w) = \|w\|_2 = \sum_{i=1}^d w_i^2$$

$$R_1(w) = \|w\|_1 = \sum_{i=1}^d |w_i|$$

Перед регуляризатором ставится коэффициент регуляризации α . Аналитическое решение в матричной форме:

$$Q(w) = (y - Xw)^T(y - Xw) + \alpha w^T I w \rightarrow \min_w$$

$$w = (X^T X + \alpha I)^{-1} X^T y$$

Матрица $X^T X + \alpha I$ положительно определена, следовательно, её можно обратить. Эта задача имеет единственное решение.

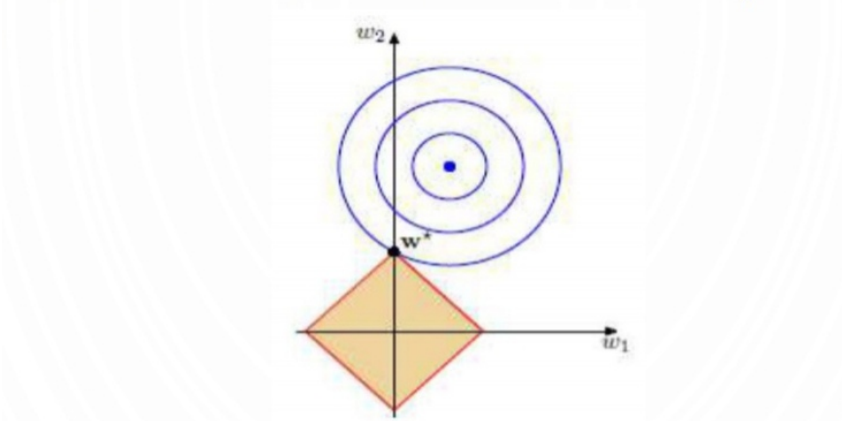
Некоторые признаки могут не иметь отношения к задаче, т.е. они не нужны. Если есть ограничения на скорость получения предсказаний, то чем меньше признаков, тем быстрее. Если признаков больше, чем объектов, то решение задачи будет неоднозначным. Поэтому в таких случаях надо делать отбор признаков, то есть убирать некоторые признаки.

Утверждение. В результате обучения модели с L_1 -регуляризатором происходит зануление некоторых весов, т.е. отбор признаков. Показать, что задачи эквивалентны:

$$Q(w) + \alpha \|w\|_1 \rightarrow \min_w$$

$$\begin{cases} Q(w) \rightarrow \min_w, \\ \|w\|_1 \leq C \end{cases}$$

Нарисуем линии уровня $Q(w)$ и область $\|w\|_1 \leq C$:



Если признак незначимый, то соответствующий вес близок к 0. Отсюда получим, что в большинстве случаев решение нашей задачи попадает в вершину ромба, т.е. обнуляет незначимый признак. Модели, в которых часть весов равна 0, называются разреженными моделями. L_1 -регуляризация зануляет часть весов, то есть делает модель разреженной.

9. Запишите формулу для линейной модели классификации. Что такое отступ?

$$a(x, w) = \text{sgn}\left(\sum_{j=1}^l w_j x_j\right)$$

Обучение - минимизация доли ошибок классификатора:

$$Q(a, x) = \frac{1}{n} \sum_{i=1}^n [a(x_i) \neq y_i] \rightarrow \min$$

где $[a(x_i) \neq y_i]$ предсказание на объекте неверное, и 0 иначе.

Обозначим $M_i = y_i(w, x_i)$ - отступ на i -м объекте. . Решение задачи (*) эквивалентно решению задачи:

$$Q(a, X) = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

Знак отступа $M_i = y_i(w, x_i)$ говорит о корректности классификации на объекте.

1) Случаи неверной классификации (предсказание не совпадает с правильным ответом):

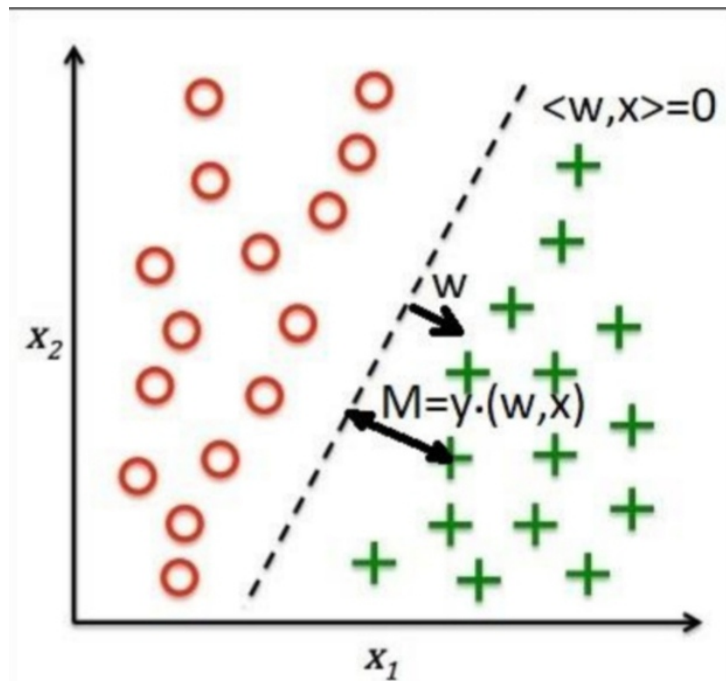
Если $(w, x) > 0$ (то есть объект отнесён к классу +1), а $y = -1$, то $M_i = y_i(w, x_i) < 0$.

Аналогично, если $(w, x) < 0$ (то есть объект отнесён к классу -1), а $y = +1$, то $M_i = y_i(w, x_i) < 0$.

2) Случаи верной классификации:

Если $(w, x) > 0$ и $y = -1$ (или наоборот), то $M_i = y_i(w, x_i) > 0$.

Абсолютная величина отступа M обозначает степень уверенности классификатора в ответе (чем ближе M к нулю, тем меньше уверенность в ответе).



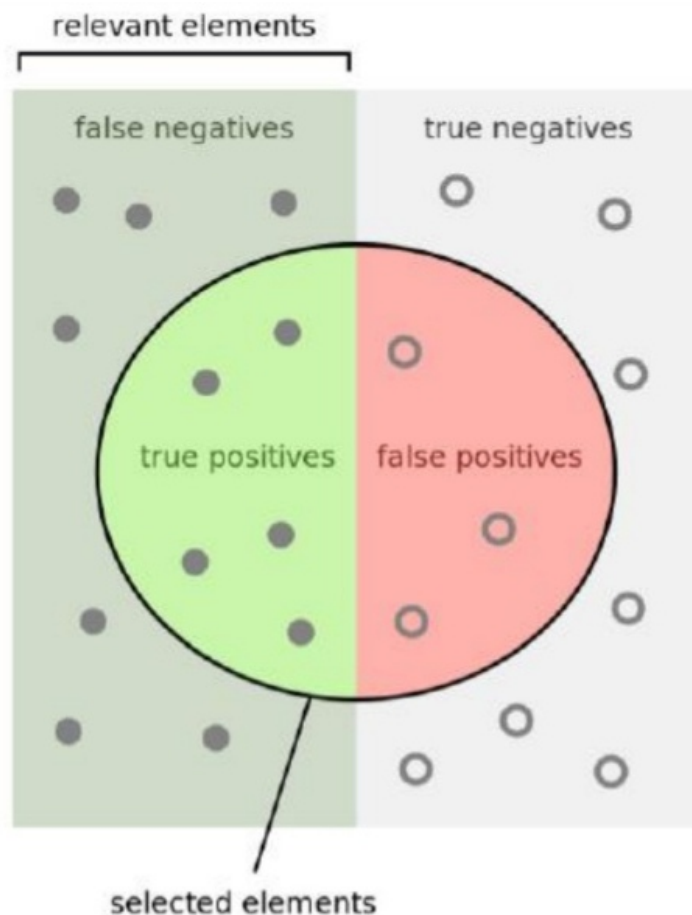
10. Что такое точность и полнота?

Precision (точность) Показывает, насколько можно доверять классификатору при $a(x) = +1$:

$$Precision(a, X) = \frac{TP}{TP + FP}$$

Recall (полнота) показывает, как много объектов положительного класса находит классификатор:

$$Recall(a, X) = \frac{TP}{TP + FN}$$



11. Что такое AUC-ROC? Как построить ROC-кривую?

Площадь под ROC-кривой AUC (Area Under Curve) является агрегированной характеристикой качества классификации, не зависящей от соотношения цен ошибок. Чем больше значение AUC, тем «лучше» модель классификации. Данный показатель часто используется для сравнительного анализа нескольких моделей классификации.

Хотим измерить качество всего семейства классификаторов независимо от выбранного порога. $AUC \in [0, 1]$.

ROC-КРИВАЯ

Для каждого значения порога t вычислим:

- **False Positive Rate** (доля неверно принятых объектов отрицательного класса):

$$FPR = \frac{FP}{FP + TN} = \frac{\sum_i [y_i = -1][a(x_i) = +1]}{\sum_i [y_i = -1]}$$

- **True Positive Rate** (доля верно принятых объектов положительного класса):

$$TPR = \frac{TP}{TP + FN} = \frac{\sum_i [y_i = +1][a(x_i) = +1]}{\sum_i [y_i = +1]}$$

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по

убыванию предсказаний:

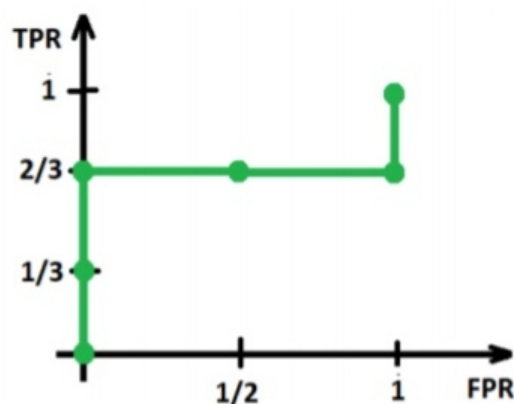
(0.7, 0.4, 0.2, 0.1, 0.05)

5 шаг: $t = 0$, то есть

$$a(x) = [b(x) > 0]$$

$$TPR = \frac{3}{3+0} = 1,$$

$$FPR = \frac{2}{2+0} = 1.$$



12. Запишите функционал логистической регрессии. Как он связан с

методом максимума правдоподобия?

$$a(x, w) = \sigma(w^T x), \text{ где } \sigma(z) = \frac{1}{1 + e^{-z}}$$

ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ - ЭТО ЛИНЕЙНЫЙ КЛАССИФИКАТОР!

Функции потерь подходят:

- 1) Квадратичная.
- 2) Логистическая.

$$L(y, z) = [y = +1] \log b(x, w) + [y = -1] \log(1 - b(x, w))$$

Вероятности, которые выдает алгоритм $b(x)$ должны согласовываться с выборкой. Вероятность того, что в выборке встретится объект x с классом y :

$$b(x)^{[y=+1]}(1 - b(x))^{[y=-1]}$$

Правдоподобие выборки:

$$(b, X) = \prod_{n=1}^l b(x_i)^{[y_i=+1]}(1 - b(x_i))^{[y_i=-1]} \rightarrow \max_b$$

Прологарифмируем правдоподобие и поставим перед ним минус, получим следующую эквивалентную задачу (это и есть log-loss):

$$- \sum_{i=1}^l ([y_i = +1] \log b(x_i) + [y_i = -1] \log(1 - b(x_i))) \rightarrow \min_b$$

13. Запишите задачу метода опорных векторов для линейно неразделимого случая. Как функционал этой задачи связан с отступом классификатора?

Линейно неразделимая выборка: существует хотя бы один объект $x \in X$, что $y_i((w, x_i) + w_0) < 1$.

Смягчим ограничения, введя штрафы $\xi_i \geq 0$: $y_i((w, x_i) + w_0) \geq 1 - \xi_i$, $i = 1, \dots, l$. Хотим:

- 1) Минимизировать штрафы $\sum_{i=1}^l \xi_i$.
- 2) Максимизировать отступ $\frac{1}{\|w\|}$.

Задача оптимизации:

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, \quad i = 1, \dots, l \\ \xi_i \geq 0, \quad i = 1, \dots, l \end{cases}$$

Эта задача является выпуклой. Покажем это:

На задачу оптимизации SVM можно смотреть как на оптимизацию функции по-

СВЕДЕНИЕ К БЕЗУСЛОВНОЙ ЗАДАЧЕ

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} (1) \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, i = 1, \dots, l (2) \\ \xi_i \geq 0, i = 1, \dots, l (3) \end{cases}$$

- Перепишем (2) и (3):

$$\begin{cases} \xi_i \geq 1 - y_i((w, x_i) + w_0) = 1 - M_i \\ \xi_i \geq 0 \end{cases}$$

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} (1) \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, i = 1, \dots, l (2) \\ \xi_i \geq 0, i = 1, \dots, l (3) \end{cases}$$

- Перепишем (2) и (3):

$$\begin{cases} \xi_i \geq 1 - y_i((w, x_i) + w_0) \\ \xi_i \geq 0 \end{cases} \Rightarrow \xi_i = \max(0, 1 - y_i((w, x_i) + w_0))$$

Получаем безусловную задачу оптимизации:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

теперь $L(M) = \max(0, 1 - M) = (1 - M)_+$ с регуляризацией:

$$Q(a, X) = \sum_{i=1}^l (1 - M_i(w, w_0))_+ + \frac{1}{2C} \|w\|^2 \rightarrow \min_{w, w_0}$$

14. В чём заключаются one-vs-all и all-vs-all подходы в многоклассовой классификации?

1) one-vs-all

Решаем задачу классификации на K классов.

Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых ре-

шает задачу: принадлежит объекту x классу k_i или не принадлежит?

Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора.

$$a(x) = \operatorname{argmax}_{K \in \{1, \dots, K\}} ((w_k, x) + w_{0k})$$

Предсказания классификаторов могут иметь разные масштабы, поэтому сравнивать их некорректно.


2) all-vs-all

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

15. В чём заключается преобразование категориальных признаков в вещественные с помощью счётчиков?

Счётчик (mean target encoding) – это вероятность получить значение целевой переменной для данного значения категориального признака.

	feature	target
0	Moscow	0
1	Moscow	1
2	Moscow	1
3	Moscow	0
4	Moscow	0
5	Tver	1
6	Tver	1
7	Tver	1
8	Tver	0
9	Klin	0
10	Klin	0
11	Tver	1



	feature	feature_mean	target
0	Moscow	0.4	0
1	Moscow	0.4	1
2	Moscow	0.4	1
3	Moscow	0.4	0
4	Moscow	0.4	0
5	Tver	0.8	1
6	Tver	0.8	1
7	Tver	0.8	1
8	Tver	0.8	0
9	Klin	0.0	0
10	Klin	0.0	0
11	Tver	0.8	1

16. Что такое ядро?

Ядро - это функция $K(x, z)$, представимая в виде скалярного произведения $K(x, z) = (\phi(x), \phi(z))$, где $\phi : X \rightarrow H$ - отображение из исходного признакового пространства X в некоторое спрямляющее пространство признаков H .

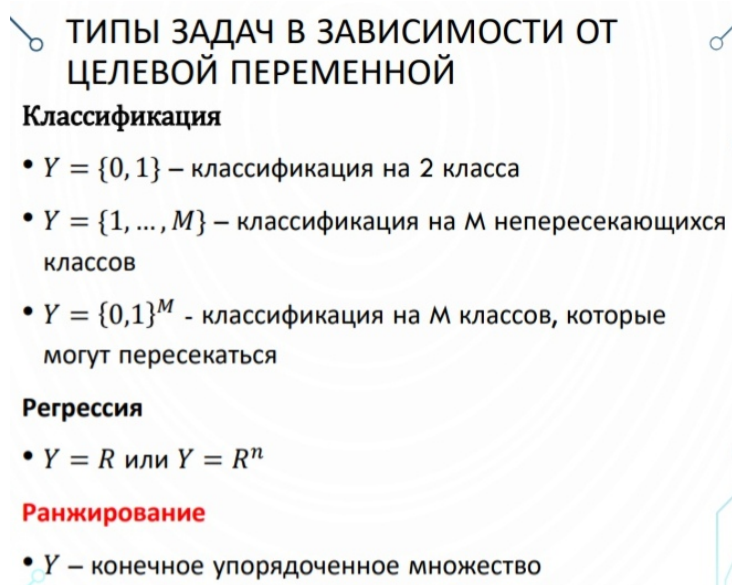
17. Запишите определения полиномиального и гауссовского ядра.

Полиномиальное ядро - $((x, z) + R)^d, R > 0$.

Гауссовское (радиальное) ядро - $\exp\{-\gamma \|x - y\|^2\}$

Основные билеты

1. Типы задач в машинном обучении (с примерами), обучение с учителем и без учителя, обучение модели, оценка качества модели, алгоритм решения задачи анализа данных.



Задачи классификации:

- 1) Задачи медицинской диагностики (пациент здоров или болен)
- 2) Задачи кредитного скоринга (выдаст банк кредит данному клиенту или нет)
- 3) Задача предсказания оттока клиентов (уйдет клиент в следующем месяце или нет)
- 4) Предсказание поведения пользователя (кликнет пользователь по данному баннеру или нет)
- 5) Классификация изображений (на изображении кошка или собака)
- 6) Определение типа объекта на изображении
- 7) Определение наиболее подходящей профессии для данного кандидата

Задачи регрессии:

- 1) Предсказание стоимости недвижимости (стоимость квартиры в Москве)
- 2) Предсказание прибыли ресторана
- 3) Предсказание поведения временного ряда в будущем (стоимость акций)
- 4) Предсказание зарплаты выпускника вуза по его оценкам

Задачи ранжирования:

- 1) Вывести подходящие запросу документы в порядке уменьшения релевантности
- 2) Вывести кандидатов на должность в порядке уменьшения релевантности

Задачи кластеризации:

- 1) Разбить пользователей на группы, внутри каждой из которых будут похожие пользователи
- 2) Разбить текстовые документы на группы по схожести документов

Если нам известны значения целевой переменной, то есть алгоритм обучается так, чтобы правильно предсказывать целевую переменную – это обучение с учителем. Сюда относят классификацию, регрессию и ранжирование.

Если нам неизвестны значения целевой переменной или целевая переменная вообще отсутствует, то есть алгоритм обучается только по признакам объектов, то это



обучение без учителя. Примерами обучения с учителем являются кластеризация, понижение размерности и др.

Процесс поиска оптимального алгоритма (оптимального набора параметров или весов) называется обучением.

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются метрики качества. Среднеквадратичная ошибка – для регрессии. Доля правильных ответов – для классификации.

Алгоритм решения задачи анализа данных:

1. Постановка задачи
2. Выделение признаков
3. Формирование выборки
4. Выбор функции потерь и метрики качества
5. Предобработка данных
6. Построение модели
7. Оценивание качества модели

2. Линейная модель регрессии. Аналитическое решение для среднеквадратичной ошибки (с выводом). Связь метода максимального правдоподобия с методом наименьших квадратов.

$$a(x) = \sum_{j=1}^n w_j x_j + w_0 = (w, x)$$

$Q(a, x) = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2 = \frac{1}{l} \sum_{i=1}^l ((w, x_i) - y_i)^2 \rightarrow \min_w$, где l - количество объектов.

$$\frac{1}{l} \|Xw - y\|^2 \rightarrow \min_w$$

Точное решение: $w = (X^T X)^{-1} X^T y$.

Доказательство:

$$Q(w) = (y - Xw)^T (y - Xw) = y^T y - y^T Xw - w^T X^T y + w^T X^T X w$$

$$\frac{\partial Q(w)}{\partial w} = 0 - X^T y - X^T y + (X^T X + X^T X)w = 0$$

$$-2X^T y + 2X^T X w = 0$$

$$X^T X w = X^T y$$

$$w = (X^T X)^{-1} X^T y$$

Условия второго порядка:

$$\nabla_w^2 Q(w) = 0 + 2X^T X = 2X^T X$$

$X^T X$ положительно определена тогда и только тогда, когда $\forall z \neq 0 \quad z^T X^T X z > 0$.

$$z^T X^T X z = (Xz)^T (Xz) = \|Xz\|^2$$

Надо, чтобы определитель под нормой не был равен 0, или не было линейно зависимых столбцов. Объектов больше или равно чем признаков.

Если эти условия выполнены, то условия второго порядка выполнены. Тем более, по критерию Сильвестра все главные миноры положительны. В частности, определитель больше нуля, то есть матрица обратима.

Модель данных с некоррелированным гауссовским шумом:

$$y_i = (w, x_i) + \varepsilon_i, \varepsilon_i \sim N(0, \sigma^2), i = 1, \dots, l$$

Метод максимума правдоподобия (ММП):

$$L(\varepsilon_1, \dots, \varepsilon_l | w) = \prod_{i=1}^l \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} \varepsilon_i^2\right) \rightarrow \max_w$$

$$-\ln L(\varepsilon_1, \dots, \varepsilon_l | w) = \text{const} + \frac{1}{2\sigma^2} \sum_{i=1}^l ((w, x_i) - y_i)^2 \rightarrow \min_w$$

В данном случае ММП совпадает с МНК.

3. Линейная регрессия. Градиентный спуск. Градиентное обучение линейной регрессии

$$a(x) = \sum_{j=1}^n w_j x_j + w_0 = (w, x)$$

$Q(a, x) = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2 = \frac{1}{l} \sum_{i=1}^l ((w, x_i) - y_i)^2 \rightarrow \min_w$, где l - количество объектов.

4. Стохастический градиентный спуск, его модификации.

1) Stochastic gradient descent (SGD): на каждом шаге выбираем один случайный объект и сдвигаемся в сторону антиградиента по этому объекту:

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla q_{i_k}(w^{(k-1)})$$

ГРАДИЕНТНЫЙ СПУСК

Градиент функции Q вычисляется как сумма градиентов функции потерь $q_i(w)$ по всем объектам:

$$\nabla Q(w) = \sum_{i=1}^l \nabla q_i(w)$$

Градиентный спуск:

$$w^{(k)} = w^{(k-1)} - \eta \sum_{i=1}^l \nabla q_i(w^{(k-1)})$$

Скорость сходимости: $Q(w^{(k)}) - Q(w^*) = \mathcal{O}\left(\frac{1}{k}\right)$

Скорость сходимости:

$$\mathbb{E}(Q(w^{(k)}) - Q(w^*)) = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$$

Менее трудоёмкий метод, но медленнее сходится.

2) Stochastic average gradient (SAG): инициализируем веса w_j и инициализируем вспомогательные переменные $z^{(1)}, z^{(2)}, \dots: z^{(i)} = \nabla q_i(w)$. Далее на каждом шаге выбираем случайный объект и обновляем градиент по нему (все остальные градиенты остаются такими же, как на предыдущем шаге):

$$z_i^{(k)} = \begin{cases} \nabla q_i(w^{(k-1)}), & i = i_k \\ z_i^{(k-1)}, & \text{иначе.} \end{cases}$$

Формула градиентного шага:

$$w^{(k)} = w^{(k-1)} - \eta_k \sum_{i=1}^l z_i^{(k)}$$

Скорость сходимости:

$$\mathbb{E}(Q(w^{(k)}) - Q(w^*)) = \mathcal{O}\left(\frac{1}{k}\right)$$

3) Метод моментов.

Вектор инерции (усреднение градиента по предыдущим шагам):

$$h_0 = 0$$

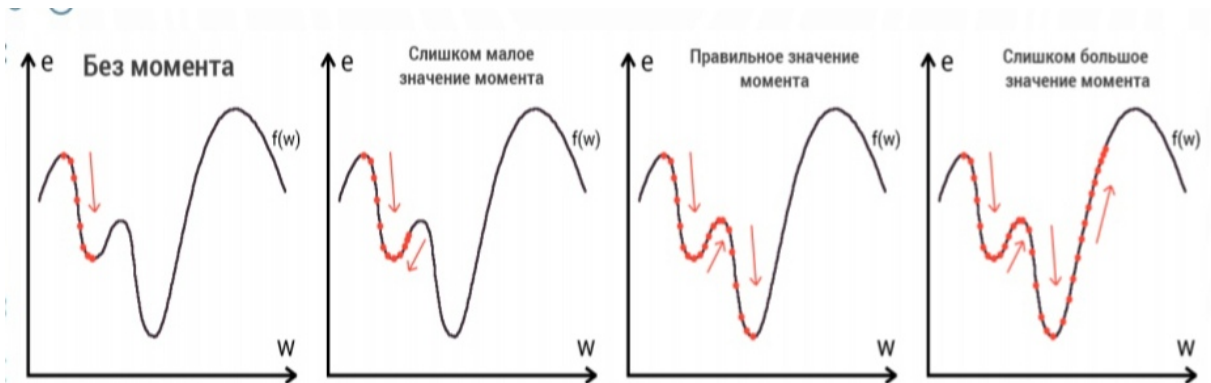
$$h_k = \alpha h_{k-1} + \eta_k \nabla Q(w^{(k-1)})$$

Формула метода моментов:

$$w^{(k)} = w^{(k-1)} - h_k$$

Подробнее:

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla Q(w^{(k-1)}) - \alpha h_{k-1}$$



4) Adagrad (Adaptive Gradient)

Сумма квадратов обновлений:

$$g_{k-1,j} = (\nabla Q(w^{(k-1)}))_j^2$$

формулы методов Adagrad:

$$G_{k,j} = G_{k-1,j} + g_{k-1,j} = G_{k-1,j} + (\nabla Q(w^{(k-1)}))_j^2$$

$$\omega_j^{(k)} = \omega_j^{(k-1)} - \frac{\eta}{\sqrt{G_{k,j} + \epsilon}} (\nabla Q(w^{(k-1)}))_j$$

Этот метод использует адаптивный шаг обучения – тем самым мы регулируем скорость сходимости метода.

Плюс - автоматическое затухание скорости обучения. Минус - $G_{k,j}$ монотонно возрастают, поэтому шаги укорачиваются, и мы можем не успеть дойти до минимума.

5) RMSPROP (root mean square propagation)

Метод реализует экспоненциальное затухание градиентов.

Формулы метода RMSprop (усредненный по истории квадрат градиента):

$$G_{k,j} = \alpha G_{k-1,j} + (1 - \alpha) g_{k-1,j}$$

$$\omega_j^{(k)} = \omega_j^{(k-1)} - \frac{\eta}{\sqrt{G_{k,j} + \epsilon}} (\nabla Q(w^{(k-1)}))_j$$

5. Функционалы ошибки для регрессии: MSE, MAE, коэффициент детерминации, квантильные потери, MAPE, SMAPE. В каких случаях какими из них стоит пользоваться.

1) MSE

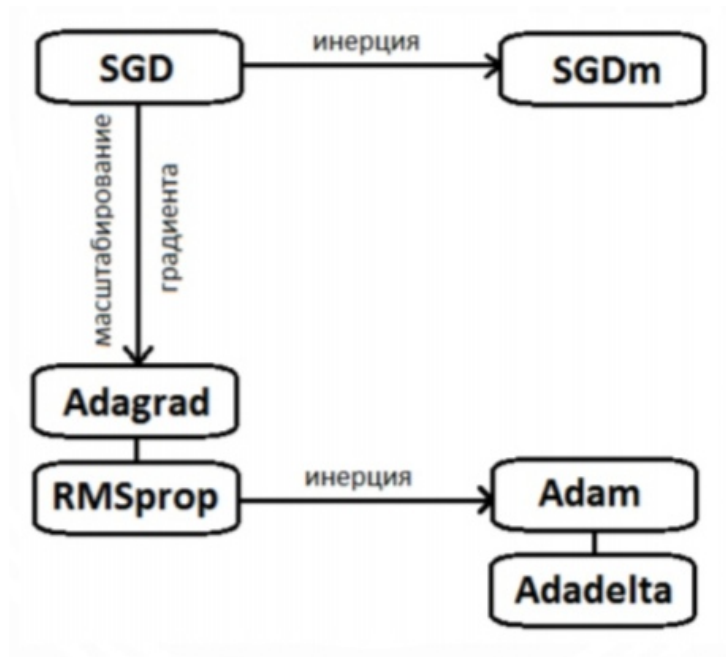
$$MSE(a, X) = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2$$

Плюсы:

Позволяет сравнивать модели

Подходит для контроля качества во время обучения

Минусы:



Плохо интерпретируется, т.к. не сохраняет единицы измерения (если целевая переменная – кг, то MSE измеряется в кг в квадрате)

Тяжело понять, насколько хорошо данная модель решает задачу, так как MSE не ограничена сверху

2) RMSE

$$RMSE(a, X) = \sqrt{\frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2}$$

Плюсы:

Все плюсы MSE

Сохраняет единицы измерения (в отличие от MSE)

Минусы:

Тяжело понять, насколько хорошо данная модель решает задачу, так как RMSE не ограничена сверху

3) $R^2(a, X)$

$$R^2(a, X) = 1 - \frac{\sum_{i=1}^l (a(x_i) - y_i)^2}{\sum_{i=1}^l (y_i - \bar{y})^2}$$

Коэффициент детерминации это доля дисперсии целевой переменной, объясняемая моделью. Чем ближе R^2 к 1, тем лучше модель объясняет данные. Чем ближе R^2 к 0, тем ближе модель к константному предсказанию.

4) MAE

$$MAE(a, X) = \frac{1}{l} \sum_{i=1}^l |a(x_i) - y_i|$$

Плюсы:

Менее чувствителен к выбросам, чем MSE

Минусы:

MAE - не дифференцируемый функционал

5) MSLE

$$MSLE(a, X) = \frac{1}{l} \sum_{i=1}^l (\log(a(x_i) + 1) - \log(y + 1))^2$$

Подходит для задач с неотрицательной целевой переменной ($y \geq 0$)

Штрафует за отклонения в порядке величин

Штрафует заниженные прогнозы сильнее, чем завышенные

6) MAPE

$$MAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|a(x_i) - y_i|}{|y_i|}$$

MAPE измеряет относительную ошибку.

Плюсы:

Ограничена: $0 \leq MAPE \leq 1$

Хорошо интерпретируема: например, MAPE=0.16 означает, что ошибка модели в среднем составляет 16% от фактических значений.

Минусы:

По-разному относится к недо- и перепрогнозу.

7) SMAPE

$$SMAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|a(x_i) - y_i|}{(|y_i| + |a(x_i)|)^2}$$

SMAPE – попытка сделать симметричным прогноз (то есть дать одинаковую ошибку для недо- и перепрогноза).

8) Квантильная регрессия

$$Q(a, X_l) = \sum_{i=1}^l \rho_\tau(y_i - a(x_i))$$

Здесь:

$$\rho_\tau(z) = (\tau - 1)[z < 0]z + \tau[z \geq 0]z = (\tau - \frac{1}{2})z + \frac{1}{2}|z|$$

Чем больше τ , тем больше мы штрафует за занижение прогноза.

Пусть в каждой точке $x \in X$. (пространство объектов) задано распределение $p(y|x)$ на ответах для данного объекта. Тогда оптимизация функции потерь $\rho_\tau(z)$ дает алгоритм $a(x)$, приближающий τ -квантиль распределения ответов в каждой точке $x \in X$.

6. Регуляризация. Аналитический вид вектора весов в линейной регрессии со среднеквадратичной ошибкой и квадратичным регуляризатором (с выводом).

Утверждение. Если в выборке есть линейно-зависимые признаки, то задача оптимизации функционала ошибки бесконечное число решений.

Большие значения параметров (весов) модели w – признак переобучения.

Решение проблемы – регуляризация. Будем минимизировать регуляризованный функционал ошибки:

$$Q_\alpha(w) = Q(w) + \alpha R(w) \rightarrow \min_w$$

Регуляризация штрафует за слишком большие веса.

Наиболее используемые регуляризаторы:

$$R_2(w) = \|w\|_2 = \sum_{i=1}^d w_i^2$$

$$R_1(w) = \|w\|_1 = \sum_{i=1}^d |w_i|$$

Перед регуляризатором ставится коэффициент регуляризации α .

Аналитическое решение в матричной форме:

$$Q(w) = (y - Xw)^T(y - Xw) + \alpha w^T I w \rightarrow \min_w$$

$$w = (X^T X + \alpha I)^{-1} X^T y$$

Матрица $X^T X + \alpha I$ положительно определена, следовательно, её можно обратить. Эта задача имеет единственное решение.

$$Q(w) = (y - Xw)^T(y - Xw) + \alpha w^T I w \rightarrow \min_w$$

$$\nabla Q(w) = -2X^T y + 2X^T X w + \alpha(I + I^T)w = 0$$

$$(X^T X + \alpha I)w = X^T y$$

$$w = (X^T X + \alpha I)^{-1} X^T y$$

7. Разреженные модели и L_1 -регуляризация. Почему использование L_1 -регуляризатора приводит к отбору признаков?

Модели, в которых часть весов равна 0, называются разреженными моделями.

Некоторые признаки могут не иметь отношения к задаче, т.е. они не нужны. Если есть ограничения на скорость получения предсказаний, то чем меньше признаков, тем быстрее. Если признаков больше, чем объектов, то решение задачи будет неоднозначным. Поэтому в таких случаях надо делать отбор признаков, то есть убирать некоторые признаки.

Утверждение. В результате обучения модели с L_1 -регуляризатором происходит зануление некоторых весов, т.е. отбор признаков. Показать, что задачи эквивалентны:

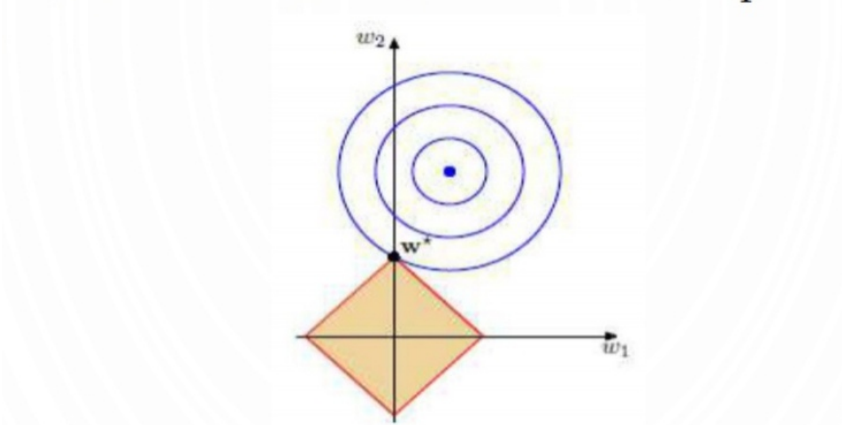
$$Q(w) + \alpha \|w\|_1 \rightarrow \min_w$$

$$\begin{cases} Q(w) \rightarrow \min_w, \\ \|w\|_1 \leq C \end{cases}$$

Если признак незначимый, то соответствующий вес близок к 0. Отсюда получим, что в большинстве случаев решение нашей задачи попадает в вершину ромба, т.е. обнуляет незначимый признак. Модели, в которых часть весов равна 0, называются разреженными моделями. L_1 -регуляризация зануляет часть весов, то есть делает модель разреженной.

8. Линейная модель классификации. Отступ. Обучение линейных клас-

Нарисуем линии уровня $Q(w)$ и область $\|w\|_1 \leq C$:



сификаторов через аппроксимацию функции $\text{sign}(\cdot)$. Примеры таких функций.

Счётчик (mean target encoding) – это вероятность получить значение целевой переменной для данного значения категориального признака.

	feature	target
0	Moscow	0
1	Moscow	1
2	Moscow	1
3	Moscow	0
4	Moscow	0
5	Tver	1
6	Tver	1
7	Tver	1
8	Tver	0
9	Klin	0
10	Klin	0
11	Tver	1

➔

	feature	feature_mean	target
0	Moscow	0.4	0
1	Moscow	0.4	1
2	Moscow	0.4	1
3	Moscow	0.4	0
4	Moscow	0.4	0
5	Tver	0.8	1
6	Tver	0.8	1
7	Tver	0.8	1
8	Tver	0.8	0
9	Klin	0.0	0
10	Klin	0.0	0
11	Tver	0.8	1

ONE-HOT ENCODING

$$a(x, w) = \text{sgn}\left(\sum_{j=1}^l w_j x_j\right)$$

Обучение - минимизация доли ошибок классификатора:

$$Q(a, x) = \frac{1}{n} \sum_{i=1}^n [a(x_i) \neq y_i] \rightarrow \min$$

где $[a(x_i) \neq y_i]$ предсказание на объекте неверное, и 0 иначе.

Обозначим $M_i = y_i(w, x_i)$ - отступ на i -м объекте. . Решение задачи (*) эквива-

лентно решению задачи:

$$Q(a, X) = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

Доказательство:

Обучение - минимизация доли ошибок классификатора:

$$Q(a, x) = \frac{1}{n} \sum_{i=1}^n [a(x_i) \neq y_i] = \frac{1}{n} \sum_{i=1}^n [\text{sgn}(w, x_i) \neq y_i] \rightarrow \min$$

Функционал Q можно переписать в виде:

$$Q(a, X) = \frac{1}{n} \sum_{i=1}^n [\text{sgn}(w, x_i) \neq y_i] = \frac{1}{n} \sum_{i=1}^n [y_i(w, x_i) < 0] = \frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

$M_i = y_i(w, x_i)$ - отступ.

Знак отступа $M_i = y_i(w, x_i)$ говорит о корректности классификации на объекте.

1) Случай неверной классификации (предсказание не совпадает с правильным ответом):

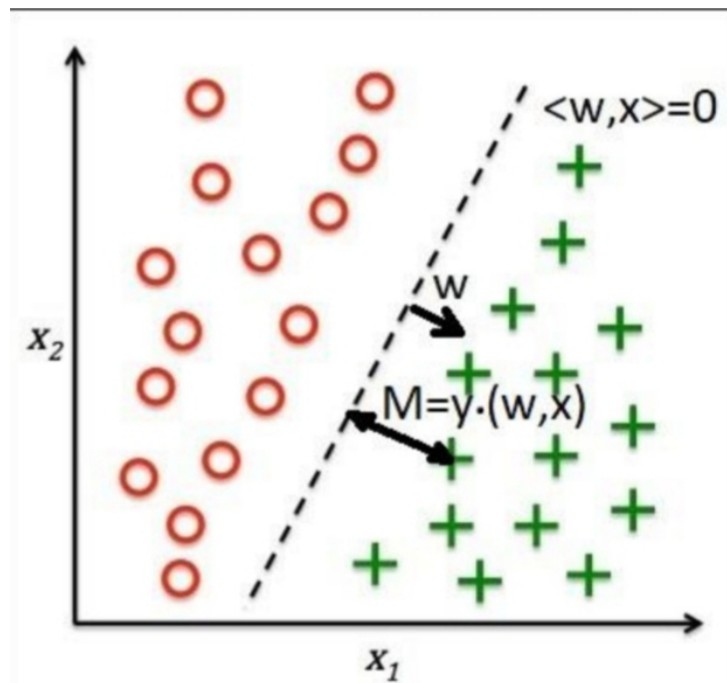
Если $(w, x) > 0$ (то есть объект отнесён к классу +1), а $y = -1$, то $M_i = y_i(w, x_i) < 0$.

Аналогично, если $(w, x) < 0$ (то есть объект отнесён к классу -1), а $y = +1$, то $M_i = y_i(w, x_i) < 0$.

2) Случай верной классификации:

Если $(w, x) > 0$ и $y = +1$ (или наоборот), то $M_i = y_i(w, x_i) > 0$.

Абсолютная величина отступа M обозначает степень уверенности классификатора в ответе (чем ближе M к нулю, тем меньше уверенность в ответе).



Ранее мы показали, что обучение классификатора – это минимизация пороговой

функции потерь:

$$\frac{1}{n} \sum_{i=1}^n [M_i < 0] \rightarrow \min$$

Пороговая функция потерь разрывна, и этот факт сильно затрудняет процесс минимизации.

Для решения этой проблемы используют другие функции потерь – непрерывные или гладкие, как правило, являющиеся верхними оценками пороговой функции.

Задача минимизации некоторой функции потерь называется минимизацией эмпирического риска (сама функция потерь – эмпирический риск).

ФУНКЦИИ ПОТЕРЬ

Минимизируя различные функции потерь, получаем разные результаты. Поэтому разные функции потерь определяют различные классификаторы.

- $L(M) = \log(1 + e^{-M})$ – логистическая функция потерь
- $V(M) = (1 - M)_+ = \max(0, 1 - M)$ – кусочно-линейная функция потерь (метод опорных векторов)
- $H(M) = (-M)_+ = \max(0, -M)$ – кусочно-линейная функция потерь (персептрон)
- $E(M) = e^{-M}$ – экспоненциальная функция потерь
- $S(M) = \frac{2}{1 + e^{-M}}$ – сигмоидная функция потерь
- $[M < 0]$ – пороговая функция потерь

9. Функционалы ошибки для классификации: матрица ошибок, ассурасу, precision, recall, F-мера.

ACCURACY

$$accuracy(a, X) = \frac{1}{n} \sum_{i=1}^n [a(x_i) = y_i]$$

Недостаток: при сильно несбалансированной выборке не отражает качество работы алгоритма.

МАТРИЦА ОШИБОК

PRECISION (точность)

Показывает, насколько можно доверять классификатору при $a(x) = +1$:

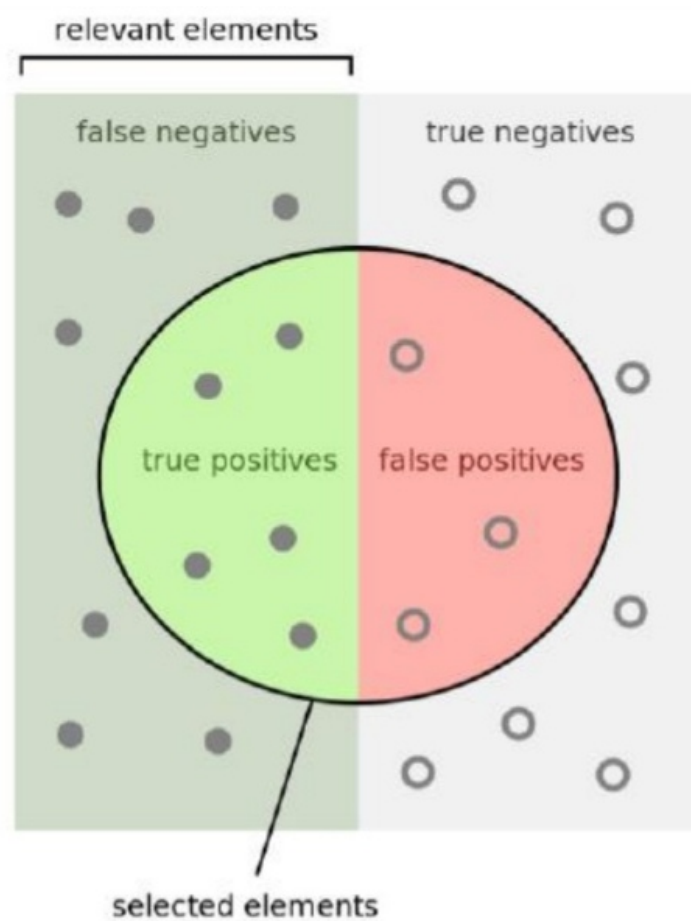
$$Precision(a, X) = \frac{TP}{TP + FP}$$

RECALL (полнота)

		Actual Value	
		positives	negatives
Predicted Value	positives	TP True Positive	FP False Positive
	negatives	FN False Negative	TN True Negative

Показывает, как много объектов положительного класса находит классификатор:

$$Recall(a, X) = \frac{TP}{TP + FN}$$





F-MEPA

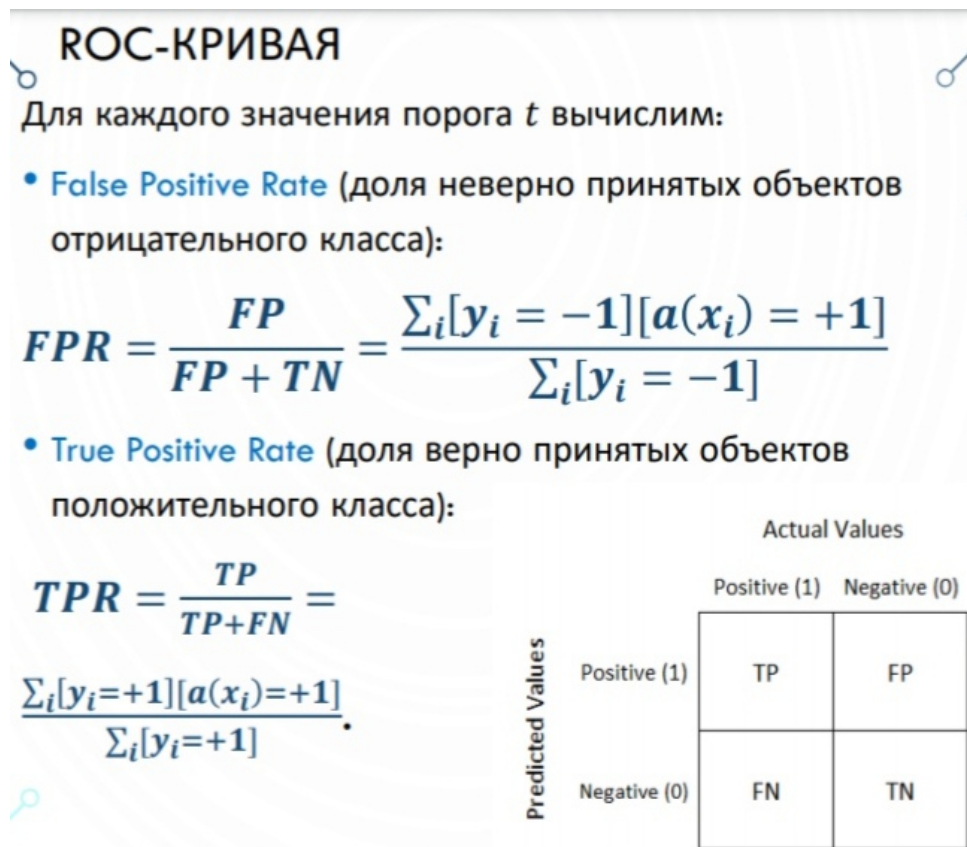
Это метрика качества, учитывающая и точность, и полноту.

$$F(a, X) = \frac{2 * Precision * Recall}{Precision + Recall}$$

10. Интегральные метрики. ROC-кривая и AUC-ROC, индекс Джини. Precision-recall кривая и AUC-PR. Пример построения ROC-кривой и PR-кривой и вычисления AUCROC, AUC-PR.

Хотим измерить качество всего семейства классификаторов независимо от выбранного порога. Для этого будем использовать метрику AUC. AUC – Area Under ROC Curve (площадь под ROC-кривой).

Хотим измерить качество всего семейства классификаторов независимо от выбранного порога. $AUC \in [0, 1]$.



ПРИМЕР ПОСТРОЕНИЯ ROC-КРИВОЙ

- Оценки принадлежности к классу +1:

$b(x)$	0.2	0.4	0.1	0.7	0.05
y	-1	+1	-1	+1	+1

- Упорядочим объекты по

убыванию предсказаний:

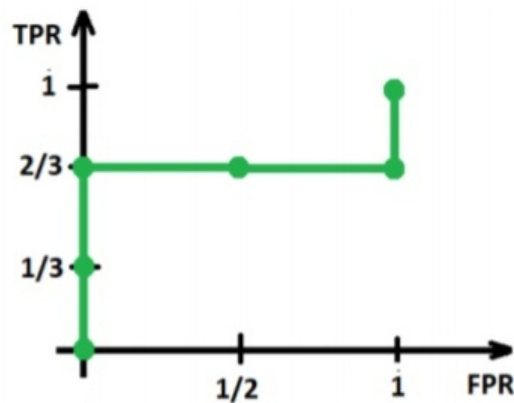
(0.7, 0.4, 0.2, 0.1, 0.05)

5 шаг: $t = 0$, то есть

$a(x) = [b(x) > 0]$

$$TPR = \frac{3}{3+0} = 1,$$

$$FPR = \frac{2}{2+0} = 1.$$

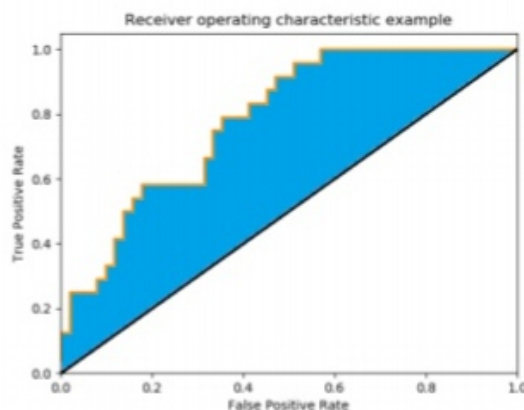


ИНДЕКС ДЖИНИ

Индекс Джини:

$$Gini = 2 \cdot AUC - 1$$

- Индекс Джини – это удвоенная площадь между главной диагональю и ROC-кривой.

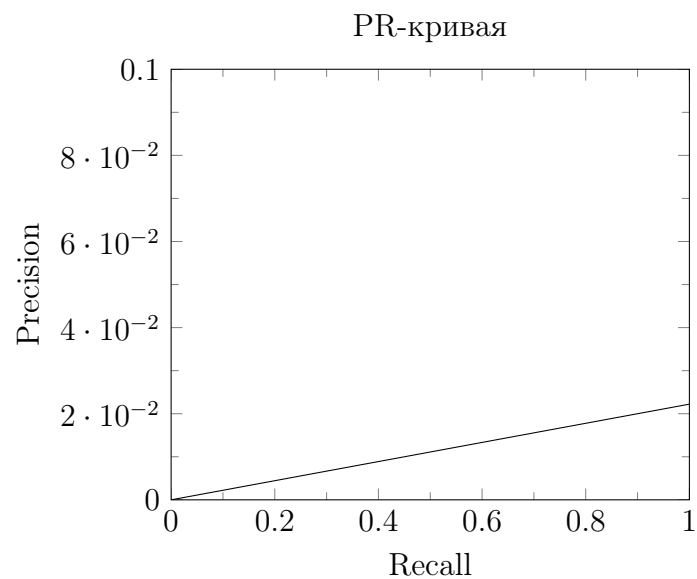
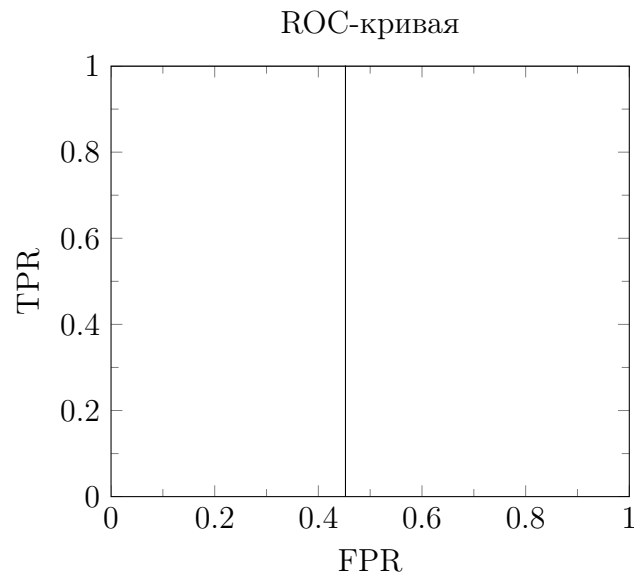


В случае малой доли объектов положительного класса AUC-ROC может давать неадекватно хороший результат.

Задача 2.6

Сначала отсортируем ответы по убыванию вероятностей для построения ROC-кривой. Проще говоря, перевернём табличку. У нас получится сначала последовательность из 4400 минус единиц, потом 100 единиц и 5500 опять минус единиц. В таком случае у нас получится следующая ROC-кривая.

$$TPR = \begin{cases} 0, FPR < \frac{4400}{9900} \\ [0, 1], FPR = \frac{4400}{9900} \\ 1, FPR > \frac{4400}{9900} \end{cases}$$



Посчитаем площадь под ROC-кривой. Для этого:

$$ROCAUC = 1 * \frac{5500}{10000 - 100} \approx 0.555555555555 \dots$$

Теперь нарисуем PR-кривую: $Precision = \frac{100}{4500} * Recall$.

Площадь под PR-кривой:

$$PRAUC = \frac{1}{2} * 1 * \frac{100}{4500} \approx 0.01111111111 \dots$$

11. Персептрон. Реализация логических функций (И, ИЛИ, НЕ) с помощью персептрона.

Персептрон – это простейшая модель классификации, при этом являющаяся предшественником нейронных сетей.

1) Задача классификации с двумя классами $y_i \in \{0, 1\}$

2) Признаки объектов – бинарные: $x_i^j \in \{0, 1\}$

Алгоритм: $a(x, w) = [w_1x_1 + \dots + w_nx_n > 0] = [(w, x) > 0]$.

12. Логистическая регрессия. Оценивание вероятностей. Вывод логистической функции потерь из метода максимального правдоподобия.

$$a(x, w) = \sigma(w^T x), \text{ где } \sigma(z) = \frac{1}{1 + e^{-z}}$$

ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ - ЭТО ЛИНЕЙНЫЙ КЛАССИФИКАТОР!

Функции потерь подходят:

1) Квадратичная.

2) Логистическая.

$$L(y, z) = [y = +1] \log b(x, w) + [y = -1] \log(1 - b(x, w))$$

Вероятности, которые выдает алгоритм $b(x)$ должны согласовываться с выборкой.

Вероятность того, что в выборке встретится объект x с классом y :

$$b(x)^{[y=+1]}(1 - b(x))^{[y=-1]}$$

Правдоподобие выборки:

$$(b, X) = \prod_{n=1}^l b(x_i)^{[y_i=+1]}(1 - b(x_i))^{[y_i=-1]} \rightarrow \max_b$$

Прологарифмируем правдоподобие и поставим перед ним минус, получим следующую эквивалентную задачу (это и есть log-loss):

$$-\sum_{i=1}^l ([y_i = +1] \log b(x_i) + [y_i = -1] \log(1 - b(x_i))) \rightarrow \min_b$$

Предположение: В каждой точке x пространства объектов задана вероятность $p(y = +1|x)$

Объекты с одинаковым признаковым описанием могут иметь разные значения целевой переменной.

Цель: построить алгоритм $b(x)$, в каждой точке x предсказывающий $p(y = +1|x)$.

Комментарий: пока что мы будем решать задачу в общем виде, то есть у нас нет ограничений на вид алгоритма $b(x)$ и на вид функции потерь $L(y, b)$.

- Пусть объект x встречается в выборке n раз с ответами $\{y_1, \dots, y_n\}$. Хотим, чтобы алгоритм выдавал вероятность положительного класса:

$$b_*(x) = \operatorname{argmin}_{b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^n L(y_i, b) \approx p(y = +1|x)$$

По закону больших чисел при $n \rightarrow \infty$ получаем

$$b_*(x) = \operatorname{argmin}_{b \in \mathbb{R}} E[L(y, b)|x]$$

Отсюда получаем *условие на функцию потерь:*

$$\operatorname{argmin} E[L(y, b)|x] = p(y = +1|x)$$

13. Метод опорных векторов. Вывод постановки задачи для разделимого случая и для неразделимого случая.

Линейно неразделимая выборка: существует хотя бы один объект $x \in X$, что $y_i((w, x_i) + w_0) < 1$.

Смягчим ограничения, введя штрафы $\xi_i \geq 0$: $y_i((w, x_i) + w_0) \geq 1 - \xi_i$, $i = 1, \dots, l$. Хотим:

- 1) Минимизировать штрафы $\sum_{i=1}^l \xi_i$.
- 2) Максимизировать отступ $\frac{1}{\|w\|}$.

Задача оптимизации:

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, \quad i = 1, \dots, l \\ \xi_i \geq 0, \quad i = 1, \dots, l \end{cases}$$

Эта задача является выпуклой. Покажем это:

СВЕДЕНИЕ К БЕЗУСЛОВНОЙ ЗАДАЧЕ

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} (1) \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, \quad i = 1, \dots, l (2) \\ \xi_i \geq 0, \quad i = 1, \dots, l (3) \end{cases}$$

- Перепишем (2) и (3):

$$\begin{cases} \xi_i \geq 1 - y_i((w, x_i) + w_0) = 1 - M_i \\ \xi_i \geq 0 \end{cases}$$

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \rightarrow \min_{w, w_0, \xi_i} (1) \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, \quad i = 1, \dots, l (2) \\ \xi_i \geq 0, \quad i = 1, \dots, l (3) \end{cases}$$

- Перепишем (2) и (3):

$$\begin{cases} \xi_i \geq 1 - y_i((w, x_i) + w_0) \Rightarrow \xi_i = \max(0, 1 - y_i((w, x_i) + w_0)) \\ \xi_i \geq 0 \end{cases}$$

Получаем безусловную задачу оптимизации:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \max(0, 1 - y_i((w, x_i) + w_0)) \rightarrow \min_{w, w_0}$$

На задачу оптимизации SVM можно смотреть как на оптимизацию функции по-

теперь $L(M) = \max(0, 1 - M) = (1 - M)_+$ с регуляризацией:

$$Q(a, X) = \sum_{i=1}^l (1 - M_i(w, w_0))_+ + \frac{1}{2C} \|w\|^2 \rightarrow \min_{w, w_0}$$

○ РАЗДЕЛИМЫЙ СЛУЧАЙ

- Нормируем параметры w и w_0 так, что

$$\min_{x \in X} |(w, x) + w_0| = 1$$

Тогда расстояние от точки x_0 до разделяющей гиперплоскости, задаваемой классификатором:

$$\rho(x_0, a) = \frac{|(w, x_0) + w_0|}{\|w\|}$$

- Расстояние до ближайшего объекта $x \in X$:

$$\min_{x \in X} \frac{|(w, x) + w_0|}{\|w\|} = \frac{1}{\|w\|} \min_{x \in X} |(w, x) + w_0| = \frac{1}{\|w\|}$$

Оптимизационная задача для разделимой выборки:

$$\begin{cases} \frac{1}{2} \|w\|^2 \rightarrow \min_w \\ y_i((w, x_i) + w_0) \geq 1 - \xi_i, \quad i = 1, \dots, l \end{cases}$$

14. Многоклассовая классификация: one-vs-all, all-vs-all. Какие проблемы возникают при использовании различных подходов? Метрики качества многоклассовой классификации.

1) one-vs-all

Решаем задачу классификации на K классов.

Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых решает задачу: принадлежит объекту x классу k_i или не принадлежит?

Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора.

$$a(x) = \operatorname{argmax}_{K \in \{1, \dots, K\}} ((w_k, x) + w_{0k})$$

Предсказания классификаторов могут иметь разные масштабы, поэтому сравнивать их некорректно.

2) all-vs-all

Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j . Если всего K классов, то получим $\binom{K}{2}$. каждый

такой классификатор будем обучать только на объектах классов i и j . В качестве итогового предсказания выдадим класс, который предсказало наибольшее число алгоритмов.

ПОДХОДЫ

1) Микроусреднение

Вычислим для каждого двухклассового классификатора TP, TN, FP, FN.

Усредним каждую характеристику по всем классам.

Тогда точность вычисляем по средним показателям.

2) Макроусреднение

Вычислим для каждого двухклассового классификатора TP, TN, FP, FN.

Вычислим итоговую метрику для каждого класса в отдельности precision.

Усредним итоговые метрики.

15. Многоклассовая логистическая регрессия. Вывод многоклассового лог-лосса из метода максимального правдоподобия.

- Бинарная лог.регрессия предсказывает вероятность класса 1:

$$(w, x) \rightarrow a(x) = \frac{1}{1 + e^{-(w, x)}} = \frac{e^{(w, x)}}{1 + e^{(w, x)}}$$

- Предположим, у нас есть K линейных моделей, каждая из которых дает оценку принадлежности выбранному классу: $b_k(x) = (w_k, x)$.

- Преобразуем вектор предсказаний в вектор вероятностей (softmax-преобразование):

$$\text{softmax}(b_1, \dots, b_K) = \left(\frac{\exp(b_1)}{\sum_{i=1}^K \exp(b_i)}, \frac{\exp(b_2)}{\sum_{i=1}^K \exp(b_i)}, \dots, \frac{\exp(b_K)}{\sum_{i=1}^K \exp(b_i)} \right)$$

Тогда вероятность класса k :

$$P(y = k|x, w) = \frac{\exp((w_k, x))}{\sum_{i=1}^K \exp((w_i, x))}$$

16. Работа с категориальными признаками: бинарное кодирование, хэширование, счётчики. Проблема переобучения счётчиков и борьба с ней.

ONE-HOT ENCODING

ХЭШИРОВАНИЕ

Если у категориального признака слишком много значений, скажем, миллион, то после применения one-hot кодировки мы получим миллион новых столбцов. С такой огромной матрицей тяжело работать.

Хэширование развивает идею one-hot кодирования, но позволяет получать любое заранее заданное число новых числовых столбцов после кодировки.

СЧЁТЧИК (mean target encoding) – это вероятность получить значение целевой переменной для данного значения категориального признака.

Недостаток? Когда такой способ кодирования переобучит наш алгоритм?

Ответ: если в данных много редких категорий.

$$a_j(x) = P(y = j|x, w) = \frac{\exp(b_j(x))}{\sum_{i=1}^K \exp(b_i(x))}$$

Обучение – по методу максимального правдоподобия (аналогично бинарной классификации):

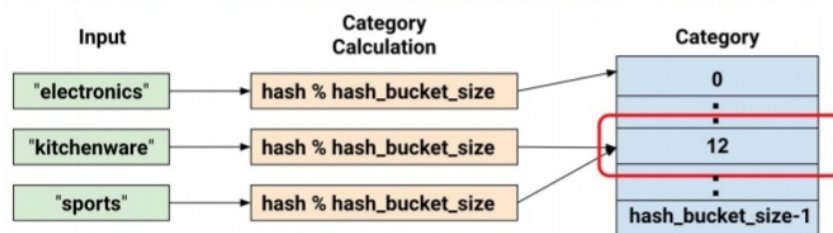
$$\begin{aligned} \Pi &= \prod_{i=1}^n a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} = \\ &= \prod_{i=1}^n \prod_{j=1}^K a_j(x_i)^{[y_i=j]} \rightarrow \max_{w_1, \dots, w_K} \end{aligned}$$

То есть в итоге обучаем одну модель (а не K моделей)

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

АЛГОРИТМ ХЭШИРОВАНИЯ

- 1) Для каждого значения признака вычисляем значение некоторой функции – хэш-функции (hash)
- 2) Задаем hash_bucket_size – итоговое количество различных значений категориального признака.
- 3) Берем остаток: hash % hash_bucket_size – тем самым кодируем каждое значение признака числом от 0 до hash_bucket_size-1.
- 4) Далее к полученным числам применяем ONE.




17. Понятия ядра и спрямляющего пространства. Ядровой трюк. Теорема Мерсера. Примеры ядер, методы построения ядер.

Ядро - это функция $K(x, z)$, представляемая в виде скалярного произведения $K(x, z) = (\phi(x), \phi(z))$, где $\phi : X \rightarrow H$ - отображение из исходного признакового пространства X в некоторое спрямляющее пространство признаков H .

При переходе к новым признакам новых признаков может потребоваться довольно много, чтобы линейно разделить выборку в новом пространстве. Поэтому сильно возрастает вычислительная сложность алгоритма, а также объем памяти, нужный для хранения всех данных.

Существует подход к решению этой проблемы под названием kernel trick (ядровой

	feature	target
0	Moscow	0
1	Moscow	1
2	Moscow	1
3	Moscow	0
4	Moscow	0
5	Tver	1
6	Tver	1
7	Tver	1
8	Tver	0
9	Klin	0
10	Klin	0
11	Tver	1



	feature	feature_mean	target
0	Moscow	0.4	0
1	Moscow	0.4	1
2	Moscow	0.4	1
3	Moscow	0.4	0
4	Moscow	0.4	0
5	Tver	0.8	1
6	Tver	0.8	1
7	Tver	0.8	1
8	Tver	0.8	0
9	Klin	0.0	0
10	Klin	0.0	0
11	Tver	0.8	1

трик). Ядровой трюк позволяет перейти в спрямляющее пространство без увеличения вычислительной сложности и требуемой памяти.

ТЕОРЕМА МЕРСЕРА

Функция $K(x, z)$ является ядром тогда и только тогда, когда:

- 1) $K(x, z) = K(z, x)$.
- 2) Для любой конечной выборки (x_1, x_2, \dots, x_l) матрица $K = (K(x_i, x_j))_{i,j=1}^l$ неотрицательно определена.

Из теоремы Мерсера следует, что ядро задаёт скалярное произведение объектов.

Идея ядрового трюка состоит в том, что некоторые модели машинного обучения (в частности, линейную регрессию и SVM) можно записать в таком виде, чтобы и модель, и функционал ошибки зависели только от скалярных произведений объектов (а не от самих объектов).

ДВОЙСТВЕННАЯ ЗАПИСЬ