Введение в искусственный интеллект. Машинное обучение

Лекция 5. Линейные классификаторы и стохастический градиентный спуск

Бабин Д.Н., Иванов И.Е., Петюшко А.А.

кафедра Математической Теории Интеллектуальных Систем

03 ноября 2020 г.

План лекции

1. Эмпирический риск и его минимизация

План лекции

1. Эмпирический риск и его минимизация
2. Разделяющая поверхность

План лекции

1. Эмпирический риск и его минимизация
2. Разделяющая поверхность
3. GD и SGD

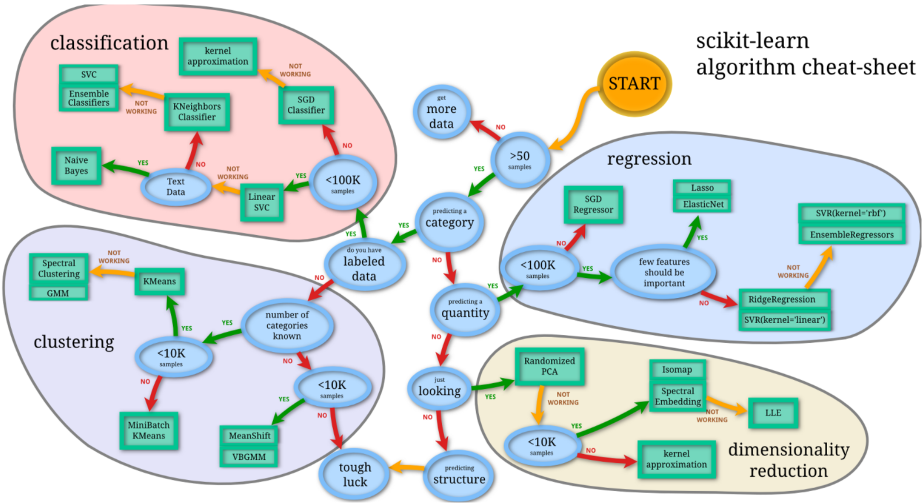
План лекции

1. Эмпирический риск и его минимизация
2. Разделяющая поверхность
3. GD и SGD
4. Теорема Новикова

План лекции

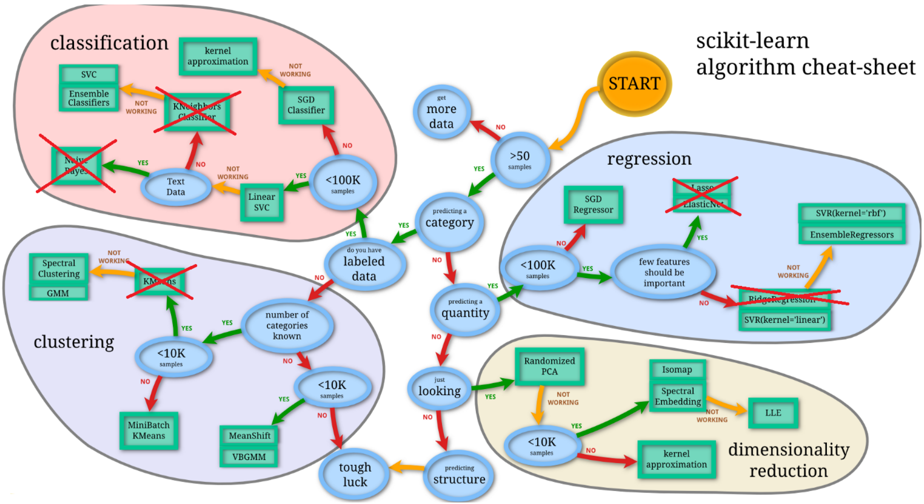
1. Эмпирический риск и его минимизация
2. Разделяющая поверхность
3. GD и SGD
4. Теорема Новикова
5. Линейный классификатор и примеры (перцептрон, логистическая регрессия, оптимальный байесовский классификатор)

Дорожная карта Scikit-Learn[1]



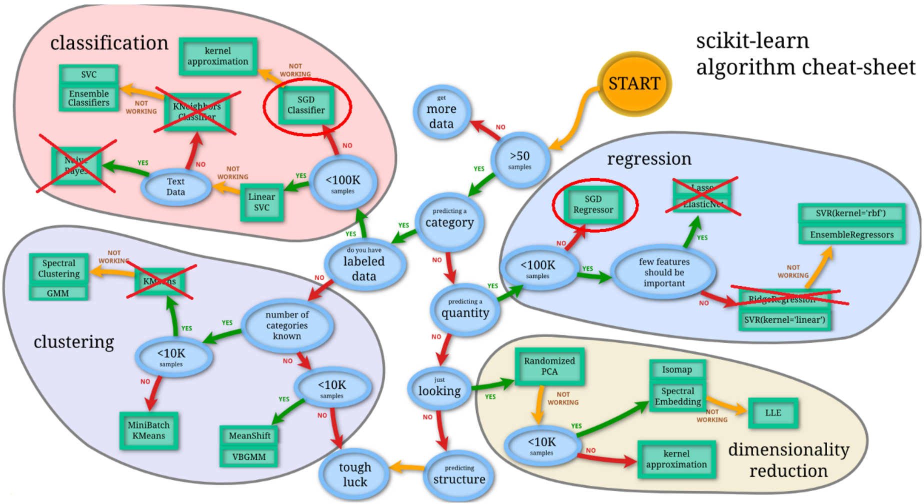
[1] https://scikit-learn.org/stable/tutorial/machine_learning_map/

Дорожная карта Scikit-Learn[1]



[1] https://scikit-learn.org/stable/tutorial/machine_learning_map/

Дорожная карта Scikit-Learn[1]



[1] https://scikit-learn.org/stable/tutorial/machine_learning_map/

Обучение по прецедентам

- X – множество описаний объектов, Y – множество допустимых ответов
- Неизвестная целевая зависимость: отображение $y^* : X \to Y$
- Конечная обучающая выборка: $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$, т.ч. $y_i = y^*(x_i)$

Обучение по прецедентам

- X – множество описаний объектов, Y – множество допустимых ответов
- Неизвестная целевая зависимость: отображение $y^* : X \to Y$
- Конечная обучающая выборка: $X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, т.ч. $y_i = y^*(x_i)$
- **Задача обучения по прецедентам** состоит в том, чтобы построить алгоритм $a : X \to Y$, который приближал бы целевую зависимость y^* как на обучающей выборке X^m, так и на всём множестве X

Обучение по прецедентам

- X – множество описаний объектов, Y – множество допустимых ответов
- Неизвестная целевая зависимость: отображение $y^* : X \to Y$
- Конечная обучающая выборка: $X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, т.ч. $y_i = y^*(x_i)$
- **Задача обучения по прецедентам** состоит в том, чтобы построить алгоритм $a : X \to Y$, который приближал бы целевую зависимость y^* как на обучающей выборке X^m, так и на всём множестве X
- **Эмпирический риск** – это средняя величина ошибки a на X^m

Обучение по прецедентам

- X – множество описаний объектов, Y – множество допустимых ответов
- Неизвестная целевая зависимость: отображение $y^* : X \to Y$
- Конечная обучающая выборка: $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$, т.ч. $y_i = y^*(x_i)$
- **Задача обучения по прецедентам** состоит в том, чтобы построить алгоритм $a : X \to Y$, который приближал бы целевую зависимость y^* как на обучающей выборке X^m, так и на всём множестве X
- **Эмпирический риск** – это средняя величина ошибки a на X^m
- **Метод минимизации эмпирического риска** – это общий подход к решению широкого класса задач обучения по прецедентам (задачи классификации и регрессии)

Эмпирический риск – определения

Функция потерь $L(y, y')$

Характеризует величину отклонения ответа $y = a(x)$ от правильного ответа $y' = y^*(x)$ на объекте $x \in X$

Эмпирический риск – определения

Функция потерь $L(y, y')$

Характеризует величину отклонения ответа $y = a(x)$ от правильного ответа $y' = y^*(x)$ на объекте $x \in X$

Множество алгоритмов $A = \{a : X \to Y\}$

В этом множестве будет вестись поиск отображения, приближающего неизвестную целевую зависимость

Эмпирический риск – определения

Функция потерь $L(y, y')$

Характеризует величину отклонения ответа $y = a(x)$ от правильного ответа $y' = y^*(x)$ на объекте $x \in X$

Множество алгоритмов $A = \{a : X \to Y\}$

В этом множестве будет вестись поиск отображения, приближающего неизвестную целевую зависимость

Эмпирический риск

Функционал качества, характеризующий среднюю ошибку алгоритма a на выборке X^m: $R(a, X^m) = \frac{1}{m} \sum_{i=1}^{m} L(a(x_i), y^*(x_i))$

Минимизация эмпирического риска

Минимизация эмпирического риска (М.Э.Р.)

В заданном множестве алгоритмов A найти алгоритм, минимизирующий эмпирический риск:

$$a = \arg\min_{a \in A} R(a, X^m)$$

Минимизация эмпирического риска (М.Э.Р.)

В заданном множестве алгоритмов A найти алгоритм, минимизирующий эмпирический риск:

$$a = \arg\min_{a \in A} R(a, X^m)$$

Достоинство М.Э.Р.

Конструктивный и универсальный подход, позволяющий сводить задачу обучения к задачам численной оптимизации

Минимизация эмпирического риска (М.Э.Р.)

В заданном множестве алгоритмов A найти алгоритм, минимизирующий эмпирический риск:

$$a = \arg\min_{a \in A} R(a, X^m)$$

Достоинство М.Э.Р.

Конструктивный и универсальный подход, позволяющий сводить задачу обучения к задачам численной оптимизации

Недостаток М.Э.Р.

Явление переобучения, которое возникает практически всегда при использовании метода м.э.р.

Примеры функций потерь

Задача классификации

- Пороговая функция $L(y, y') = [y \neq y']$
- Функция разрывна $\Rightarrow$ минимизация эмпирического риска – это задача комбинаторной оптимизации $\Rightarrow$ во многих практически важных случаях сводится к поиску максимальной совместной подсистемы в системе неравенств (число неравенств совпадает с число объектов обучения m) и является NP-полной

Примеры функций потерь

Задача классификации

- Пороговая функция $L(y, y') = [y \neq y']$
- Функция разрывна $\Rightarrow$ минимизация эмпирического риска – это задача комбинаторной оптимизации $\Rightarrow$ во многих практически важных случаях сводится к поиску максимальной совместной подсистемы в системе неравенств (число неравенств совпадает с число объектов обучения m) и является NP-полной

Задача регрессии

Квадратичная функция потерь $L(y, y') = (y - y')^2$

Разделяющая поверхность

- Рассмотрим задачу бинарной классификации: $X \to Y$, $Y = \{+1, -1\}$ на обучающей выборке $X^m = (x_i, y_i)_{i=1}^m$
- Будем алгоритм искать в виде $a(x, w) = \operatorname{sign} g(x, w)$, где $g(x, w)$ - дискриминантная функция, а w - вектор параметров
- $g(x, w) = 0$ - разделяющая поверхность;
  $M_i(w) = y_i g(x_i, w)$ - отступ объекта x_i;
  $M_i(w) < 0 \Leftrightarrow a(x_i, w) \neq y_i$.

Разделяющая поверхность

- Рассмотрим задачу бинарной классификации: $X \to Y$, $Y = \{+1, -1\}$ на обучающей выборке $X^m = (x_i, y_i)_{i=1}^m$
- Будем алгоритм искать в виде $a(x, w) = \operatorname{sign} g(x, w)$, где $g(x, w)$ - дискриминантная функция, а w - вектор параметров
- $g(x, w) = 0$ - разделяющая поверхность;
  $M_i(w) = y_i g(x_i, w)$ - отступ объекта x_i;
  $M_i(w) < 0 \Leftrightarrow a(x_i, w) \neq y_i$.
- М.Э.Р.: $R(a, X^m) = \frac{1}{m}\sum_{i=1}^m [M_i(w) < 0] \leq \tilde{R}(a, X^m) = \frac{1}{m}\sum_{i=1}^m L(M_i(w))$, где новая функция потерь $L(M)$ - невозрастающая и неотрицательная аппроксимация функции $[M < 0]$, т.ч.: $L(M) \geq [M < 0]$

Разделяющая поверхность

- Рассмотрим задачу бинарной классификации: $X \to Y$, $Y = \{+1, -1\}$ на обучающей выборке $X^m = (x_i, y_i)_{i=1}^m$
- Будем алгоритм искать в виде $a(x, w) = \operatorname{sign} g(x, w)$, где $g(x, w)$ - дискриминантная функция, а w - вектор параметров
- $g(x, w) = 0$ - разделяющая поверхность;
  $M_i(w) = y_i g(x_i, w)$ - отступ объекта x_i;
  $M_i(w) < 0 \Leftrightarrow a(x_i, w) \neq y_i$.
- М.Э.Р.: $R(a, X^m) = \frac{1}{m} \sum_{i=1}^m [M_i(w) < 0] \leq \tilde{R}(a, X^m) = \frac{1}{m} \sum_{i=1}^m L(M_i(w))$, где новая функция потерь $L(M)$ - невозрастающая и неотрицательная аппроксимация функции $[M < 0]$, т.ч.: $L(M) \geq [M < 0]$

Замечание. В дальнейшем будем предполагать, что мы работаем сразу с аппроксимаций Э.Р. $\tilde{R}$, поэтому знак $\tilde{}$ будем опускать.

Вероятностный смысл минимизации аппроксимированного Э.Р.

Рассмотрим принцип максимизации правдоподобия, или MLE (Maximum Likelihood Estimation).

- Параметрическая модель плотности распределения $p(x, y|w)$
- Максимизация логарифма правдоподобия

$$L(w, X^m) = \ln \prod_{i=1}^{m} p(x_i, y_i|w) = \sum_{i=1}^{m} \ln p(x_i, y_i|w) \to \max_{w}$$

Вероятностный смысл минимизации аппроксимированного Э.Р.

Рассмотрим принцип максимизации правдоподобия, или MLE (Maximum Likelihood Estimation).

- Параметрическая модель плотности распределения $p(x, y|w)$
- Максимизация логарифма правдоподобия

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

- Минимизация аппроксимированного Э.Р.

$$R(w, X^m) = \frac{1}{m} \sum_{i=1}^{m} L(M_i(w)) \to \min_w$$

Вероятностный смысл минимизации аппроксимированного Э.Р.

Рассмотрим принцип максимизации правдоподобия, или MLE (Maximum Likelihood Estimation).

- Параметрическая модель плотности распределения $p(x, y|w)$
- Максимизация логарифма правдоподобия

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

- Минимизация аппроксимированного Э.Р.

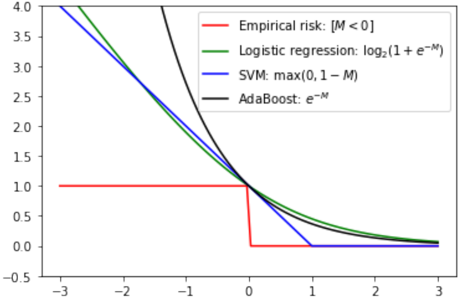
$$R(w, X^m) = \frac{1}{m} \sum_{i=1}^{m} L(M_i(w)) \to \min_w$$

Вывод. Эти два принципа эквивалентны при $L(M_i(w)) = -\ln p(x_i, y_i|w)$ (коэффициент $\frac{1}{m}$ не влияет на вывод).

Об аппроксимации

- Рассмотрим аппроксимацию функции ошибки на обучающем примере: $L(M) \geq [M < 0]$
- В дальнейшем будем рассматривать в основном достаточно гладкие (непрерывные и дифференцируемые) функции $L(M)$
- Некоторые аппроксимации способны улучшать обобщающую способность классификатора
- Непрерывные аппроксимации позволяют применять известные численные методы оптимизации для настройки весов w (например, градиентные методы и методы выпуклого программирования)

Примеры аппроксимации функции $[M < 0]$:



Классический градиентный спуск

Задача: минимизировать аппроксимированный Э.Р. (выбор алгоритма осуществляется по w):

$$R(w) = \frac{1}{m}\sum_{i=1}^{m} L(M_i(w)) = \frac{1}{m}\sum_{i=1}^{m} L_i(w) \to \min_{w}$$

Классический градиентный спуск

Задача: минимизировать аппроксимированный Э.Р. (выбор алгоритма осуществляется по w):

$$R(w) = \frac{1}{m}\sum_{i=1}^{m} L(M_i(w)) = \frac{1}{m}\sum_{i=1}^{m} L_i(w) \to \min_{w}$$

Численная оптимизация методом градиентного спуска

- $w^{(0)} :=$ начальное приближение
- $w^{(t+1)} := w^{(t)} - \eta \cdot \nabla R(w^{(t)})$ - итерация алгоритма
- η - градиентный шаг

Классический градиентный спуск

Задача: минимизировать аппроксимированный Э.Р. (выбор алгоритма осуществляется по w):

$$R(w) = \frac{1}{m}\sum_{i=1}^{m} L(M_i(w)) = \frac{1}{m}\sum_{i=1}^{m} L_i(w) \to \min_w$$

Численная оптимизация методом градиентного спуска

- $w^{(0)} :=$ начальное приближение
- $w^{(t+1)} := w^{(t)} - \eta \cdot \nabla R(w^{(t)})$ - итерация алгоритма
- η - градиентный шаг

Проблема: сложно считать в условиях большого количества объектов в обучающей выборке.

Стохастический градиентный спуск

Алгоритм стохастического градиентного спуска

- Инициализация весов w
- Инициализация Э.Р. $R := \frac{1}{m}\sum_{i=1}^{m} L_i(w)$

Итерации

- Выбор объекта $x_i \in X^m$ (например, случайным образом)
- Вычисление ошибки на данном объекте: $\varepsilon_i = L_i(w)$
- Шаг градиентного спуска: $w := w - \eta \cdot \nabla L_i(w)$
- Вычисление сглаженного Э.Р.: $R := (1-\lambda)R + \lambda\varepsilon_i$

Замечание: параметр сглаживания $\lambda \in [0,1]$ (можно использовать, например, 0.1).

Вариативность SGD

Инициализация

- $w_j = 0 \quad \forall j = 1, \dots, n$ (где n - число весов)
- $w_j = rand(-\frac{1}{2n}, \frac{1}{2n})$
- Предобучение на другой обучающей выборке

Вариативность SGD

Инициализация

- $w_j = 0 \quad \forall j = 1, \dots, n$ (где n - число весов)
- $w_j = rand(-\frac{1}{2n}, \frac{1}{2n})$
- Предобучение на другой обучающей выборке

Порядок выбора объектов x_i

- Случайная перетасовка: попеременно брать объекты разных классов
- Чаще брать объекты с большой ошибкой (маленькое значение M_i)
- Чаще брать объекты с большой неуверенностью (маленькое значение $|M_i|$)

Вариативность SGD

Инициализация

- $w_j = 0 \quad \forall j = 1, \ldots, n$ (где n - число весов)
- $w_j = rand(-\frac{1}{2n}, \frac{1}{2n})$
- Предобучение на другой обучающей выборке

Порядок выбора объектов x_i

- Случайная перетасовка: попеременно брать объекты разных классов
- Чаще брать объекты с большой ошибкой (маленькое значение M_i)
- Чаще брать объекты с большой неуверенностью (маленькое значение $|M_i|$)

Критерий остановки

- Исчерпали лимит по числу шагов
- Значение Э.Р. либо весов перестало меняться

Пакетный (mini-batch) SGD

Пакетный SGD

Идея: на каждом шаге использовать более надежную оценку градиента не на одном примере, а на нескольких

Итерации

- Выбор подмножества объектов мощности $1 < k < m$: $J = \{i_1, \ldots, i_k\}$
- Вычисление ошибки на этих объектах: $L_{i_1}(w^{(t)}), \ldots, L_{i_k}(w^{(t)})$
- Шаг градиентного спуска: $w^{(t+1)} := w^{(t)} - \eta \cdot \frac{1}{k} \sum\limits_{j=1}^{k} \nabla_w L_{i_j}(w^{(t)})$

Выбор шага SGD

- Сходимость гарантируется[2] для выпуклых функций Э.Р. и ограниченного градиента при
  $\eta_t \to 0, \sum_{t=0}^{\infty} \eta_t = \infty, \sum_{t=0}^{\infty} \eta_t^2 < \infty$ (например, подходит $\eta_t = \frac{1}{t}$)
  - При этом скорость сходимости будет также $O(\frac{1}{t})$
- Метод скорейшего градиентного спуска
  $R(w - \eta \nabla R(w)) \to min_\eta$ позволяет найти оптимальный η^*
- Время от времени бывает полезно делать большие шаги для "выпрыгивания" из локальных минимумов

[2] Robbins, H. and Monro, S. (1951). "A stochastic approximation method"

Плюсы и минусы SGD

Плюсы

- Легко реализуется на практике;
- Легко обобщается на любые алгоритмы и функции потерь;
- Возможно онлайн до-обучение (для нового x_i);
- Необязательно использовать все объекты x_i.

Плюсы и минусы SGD

Плюсы

- Легко реализуется на практике;
- Легко обобщается на любые алгоритмы и функции потерь;
- Возможно онлайн до-обучение (для нового x_i);
- Необязательно использовать все объекты x_i.

Минусы

- На практике возможны расходимость / медленная сходимость;
- Локальные минимумы!!!
- Подбор шага градиента, условия остановки неочевидны;
- Переобучение.

Понятие линейной классификации

Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

Понятие линейной классификации

Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

- В случае **двух** классов разделяющей поверхностью является **гиперплоскость**, которая делит пространство признаков на два полупространства

Понятие линейной классификации

Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

- В случае **двух** классов разделяющей поверхностью является **гиперплоскость**, которая делит пространство признаков на два полупространства
- В случае числа классов **больше двух** разделяющая поверхность **кусочно-линейна**

Линейная классификация: определения

Два класса

Дискриминантная функция:
$g(x, w) = \sum_{j=1}^{n} w_j f_j - w_0$, где
$f_j : X \to \mathbb{R}$ – числовые признаки.
Алгоритм классификации
$a(x, w) = \mathrm{sign}(\sum_{j=1}^{n} w_j f_j - w_0)$.
Если ввести константный признак $f_0 \equiv -1$, то
$x = (f_0(x), \dots, f_n(x))$,
и алгоритм в векторной записи:
$a(x, w) = \mathrm{sign}(\langle w, x \rangle)$.
Отступ объекта x_i: $M_i = \langle w, x_i \rangle y_i$

Два класса

Дискриминантная функция:
$g(x, w) = \sum_{j=1}^{n} w_j f_j - w_0$, где
$f_j : X \to \mathbb{R}$ – числовые признаки.
Алгоритм классификации
$a(x, w) = \mathrm{sign}(\sum_{j=1}^{n} w_j f_j - w_0)$.
Если ввести константный признак $f_0 \equiv -1$, то
$x = (f_0(x), \dots, f_n(x))$,
и алгоритм в векторной записи:
$a(x, w) = \mathrm{sign}(\langle w, x \rangle)$.
Отступ объекта x_i: $M_i = \langle w, x_i \rangle y_i$

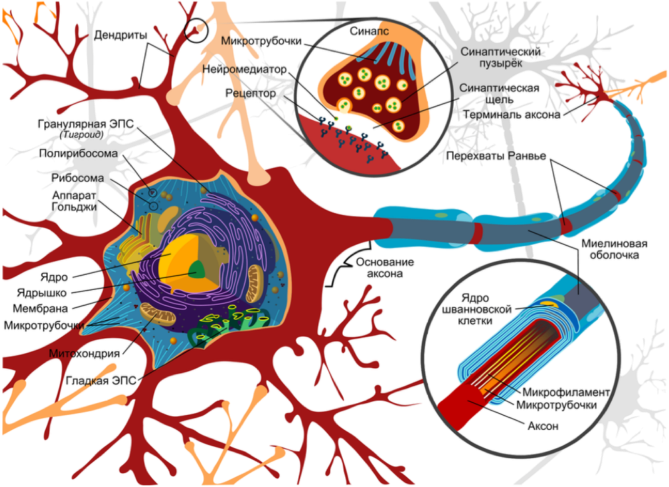
Произвольное число классов

У каждого класса $c \in Y$ свой вектор весов: $w^c = (w_0^c, \dots, w_n^c)$.
Линейный классификатор:
$a(x, w) = \arg\max_{c \in Y} \sum_{j=0}^{n} w_j^c f_j(x) = \arg\max_{c \in Y} \langle w^c, x \rangle$.
Отступ:
$M_i(w) = \langle x_i, w^{y_i} \rangle - \max_{c \in Y, c \neq y_i} \langle x_i, w^c \rangle$

Замечание. Обратите внимание на разницу со случаем двух классов!

Биологический нейрон

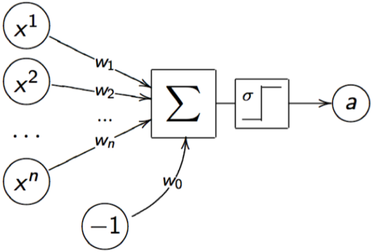
- Кора головного мозга содержит 10^{11} нейронов
- Каждый нейрон связан синапсами с $10^3 - 10^4$ другими нейронами
- Скорость распространения импульсов 100 м/с
- Входы (много) - дендриты
- Выход (один) - аксон



Предложена МакКалоком и Питтсом в 1943 году[3].

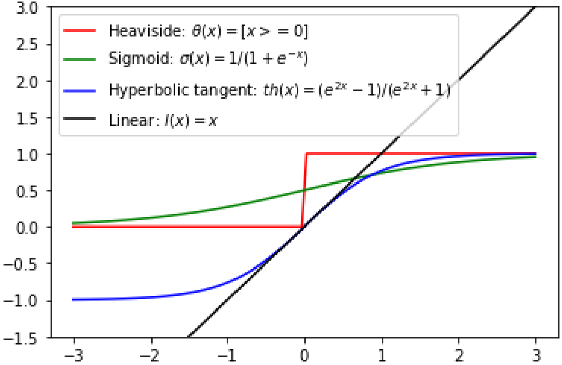
$$a(x, w) = \sigma(\langle w, x\rangle) = \sigma(\sum_{j=1}^{n} w_j f_j - w_0)$$

где $\sigma(x)$ - некоторая функция активации (например, sign).



[3] McCulloch, W. S. and Pitts, W. (1943). "A logical calculus of the ideas immanent in nervous activity"

Примеры функций активаций



SGD для линейной регрессии: ADALINE

В задаче линейной регрессии функция потерь:

$$L(a(x_i, w), y_i) = (a(x_i, w) - y_i)^2$$

Адаптивный линейный нейрон (ADAptive Linear NEuron) ADALINE предложен Видроу и Хоффом в 1960[4]:

$$a(x, w) = \langle w, x \rangle$$

Градиентный шаг стохастического градиентного спуска - т.н. **дельта-правило**:

$$w^{(t+1)} = w^{(t)} - \eta(\langle w^{(t)}, x_i \rangle - y_i)x_i$$

[4] Widrow, B. and Hoff, M. E. (1960). "Adaptive switching circuits"

SGD для линейного классификатора

Алгоритм стохастического градиентного спуска в общем виде для линейного классификатора:

$$L = L(\langle w, x_i \rangle y_i) \Rightarrow w^{(t+1)} = w^{(t)} - \eta \, L'(z)\big|_{z=\langle w^{(t)}, x_i \rangle y_i} x_i y_i$$

Рассмотрим подробнее некоторые частные случаи.

Правило Хэбба, 1949[5]

В задаче бинарной ($Y = \{-1, +1\}$) классификации линейный классификатор:
$a(x, w) = \text{sign}(\langle w, x \rangle)$
Функция потерь: $L(a(x_i, w), y_i) = [a(x_i, w) \neq y_i]$.
Шаг обновления: если $a(x_i, w^{(t)}) \neq y_i \Leftrightarrow a(x_i, w^{(t)})y_i < 0$, то $w^{(t+1)} = w^{(t)} + \eta x_i y_i$

[5] Hebb, D. O. (1949). "The organization of behavior: a neuropsychological theory."
[6] Rosenblatt, F. (1957). "The perceptron, a perceiving and recognizing automaton"

История: правила Хэбба и Розенблатта

Правило Хэбба, 1949[5]

В задаче бинарной ($Y = \{-1, +1\}$) классификации линейный классификатор:
$a(x, w) = \mathrm{sign}(\langle w, x \rangle)$
Функция потерь: $L(a(x_i, w), y_i) = [a(x_i, w) \neq y_i]$.
Шаг обновления: если $a(x_i, w^{(t)}) \neq y_i \Leftrightarrow a(x_i, w^{(t)}) y_i < 0$, то $w^{(t+1)} = w^{(t)} + \eta x_i y_i$

Правило перцептрона Розенблатта, 1957[6]

Пусть $X = \{0, 1\}^n$, $Y = \{0, +1\}$. Тогда:
если $a(x_i, w^{(t)}) \neq y_i$: $w^{(t+1)} = w^{(t)} + \eta x_i$, если $y_i = 1$, и $w^{(t+1)} = w^{(t)} - \eta x_i$, если $y_i = 0$

[5] Hebb, D. O. (1949). "The organization of behavior: a neuropsychological theory."
[6] Rosenblatt, F. (1957). "The perceptron, a perceiving and recognizing automaton"

История: правила Хэбба и Розенблатта

Правило Хэбба, 1949[5]

В задаче бинарной ($Y = \{-1, +1\}$) классификации линейный классификатор: $a(x, w) = \text{sign}(\langle w, x\rangle)$
Функция потерь: $L(a(x_i, w), y_i) = [a(x_i, w) \neq y_i]$.
Шаг обновления: если $a(x_i, w^{(t)}) \neq y_i \Leftrightarrow a(x_i, w^{(t)})y_i < 0$, то $w^{(t+1)} = w^{(t)} + \eta x_i y_i$

Правило перцептрона Розенблатта, 1957[6]

Пусть $X = \{0, 1\}^n$, $Y = \{0, +1\}$. Тогда:
если $a(x_i, w^{(t)}) \neq y_i$: $w^{(t+1)} = w^{(t)} + \eta x_i$, если $y_i = 1$, и $w^{(t+1)} = w^{(t)} - \eta x_i$, если $y_i = 0$

Правило Хэбба и правило Розенблатта - суть одно и то же, и совпадают с правилом ADALINE с заменой $a(x, w)$ на соответствующее значение из Y:

$$w^{(t+1)} = w^{(t)} - \eta(a(x_i, w^{(t)}) - y_i)x_i$$

[5] Hebb, D. O. (1949). "The organization of behavior: a neuropsychological theory."
[6] Rosenblatt, F. (1957). "The perceptron, a perceiving and recognizing automaton"

Теорема Новикова[7]

Задача бинарной классификации $X = \mathbb{R}^{n+1}$, $Y = \{-1, +1\}$.

Теорема Новикова, 1962

Пусть выборка X^m линейно разделима, т.е. $\exists \tilde{w}, ||\tilde{w}|| = 1, \exists \delta > 0 : \langle \tilde{w}, x_i \rangle y_i > \delta$ для всех $i = 1, ..., m$. Пусть начальный вектор весов $w^0 = 0$. Также в процедуре обучения каждый объект обучающей выборки появляется повторно через некоторый конечный интервал времени.

Тогда алгоритм SGD с правилом Хэбба находит вектор весов w:

- разделяющий выборку без ошибок,
- при любом шаге градиентного спуска η,
- независимо от порядка предъявления x_i,
- за конечное число исправлений вектора w: $t_{max} \leq \frac{1}{\delta^2} \max_i ||x_i||^2$

[7] Novikoff, A. (1962). "On Convergence Proofs on Perceptrons"

С одной стороны,
$\langle \tilde{w}, w^t \rangle = \langle \tilde{w}, w^{t-1} \rangle + \eta \langle \tilde{w}, x_i \rangle y_i > \langle \tilde{w}, w^{t-1} \rangle + \eta\delta > \dots > \langle \tilde{w}, w^0 \rangle + t\eta\delta = t\eta\delta$.
С другой стороны, поскольку выборка конечна, $\exists D > 0 : ||x_i|| < D$ для всех i. В силу этого $||w^t||^2 = ||w^{t-1}||^2 + \eta^2 ||x_i||^2 + 2\eta \langle w^{t-1}, x_i \rangle y_i$. Так как для применения правила Хэбба должно быть $\langle w^{t-1}, x_i \rangle y_i < 0$, то
$||w^t||^2 < ||w^{t-1}||^2 + \eta^2 D^2 < \dots < ||w^0||^2 + t\eta^2 D^2 = t\eta^2 D^2$.
По неравенству Коши-Буняковского $\langle \tilde{w}, w^t \rangle \le ||\tilde{w}|| \cdot ||w^t||$.
Объединяя эти неравенства, получаем $\eta\delta t \le \langle \tilde{w}, w^t \rangle \le \eta D \sqrt{t} \cdot ||\tilde{w}||$, или $\sqrt{t} \le \frac{D}{\delta}$.
Т.о. при $t > \frac{D^2}{\delta^2}$ не найдётся ни одного x_i, т.ч. $\langle w^t, x_i \rangle y_i < 0$, т.е. вся выборка будет правильно классифицирована. Ч.т.д.

Линейность байесовского классификатора

Из предыдущего материала известно, что оптимальный байесовский бинарный классификатор определяется как:

$a(x) = \mathrm{sign}(\lambda_+ p(y = +1|x) - \lambda_- p(y = -1|x)) = \mathrm{sign}\left(\frac{p(y=+1|x)}{p(y=-1|x)} - \frac{\lambda_-}{\lambda_+}\right)$

Теорема о линейности байесовского классификатора

Если распределения $p(y|x)$ экспонентны, параметры $d(), \delta$ не зависят от y, и среди признаков $x_1, \ldots, x_n$ есть константа, то байесовский классификатор линеен:
$a(x) = \mathrm{sign}(\langle w, x\rangle - w_0), w_0 = \ln\frac{\lambda_-}{\lambda_+}$;
при этом апостериорные вероятности классов $p(y|x) = \sigma(\langle w, x\rangle y)$, где $\sigma(z) = \frac{1}{1+e^{-z}}$ – логистическая функция (сигмоид).

Линейность байесовского классификатора

Из предыдущего материала известно, что оптимальный байесовский бинарный классификатор определяется как:

$$a(x) = \text{sign}(\lambda_+ p(y=+1|x) - \lambda_- p(y=-1|x)) = \text{sign}\left(\frac{p(y=+1|x)}{p(y=-1|x)} - \frac{\lambda_-}{\lambda_+}\right)$$

Теорема о линейности байесовского классификатора

Если распределения $p(y|x)$ экспонентны, параметры $d(), \delta$ не зависят от y, и среди признаков $x_1, \ldots, x_n$ есть константа, то байесовский классификатор линеен:

$a(x) = \text{sign}(\langle w, x\rangle - w_0), w_0 = \ln \frac{\lambda_-}{\lambda_+}$;

при этом апостериорные вероятности классов $p(y|x) = \sigma(\langle w, x\rangle y)$, где $\sigma(z) = \frac{1}{1+e^{-z}}$ – логистическая функция (сигмоид).

Определение логистической регрессии

Классификационная бинарная модель, в которой вероятность принадлежности к положительному классу задаётся сигмоидом от линейной функции по входу.

Логарифмическая функция потерь

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^{m} p(x_i, y_i) \to \max_w$

Логарифмическая функция потерь

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^{m} p(x_i, y_i) \to \max_w$

Подставим в формулу выражение для логистической регрессии $p(x, y) = p(y|x) \cdot p(x) = \sigma(\langle w, x \rangle) \cdot const(w)$:

- $L(w, X^m) = \sum_{i=1}^{m} \log \sigma(\langle w, x_i \rangle y_i) + const(w) \to \max_w$

Логарифмическая функция потерь

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^{m} p(x_i, y_i) \to \max_w$

Подставим в формулу выражение для логистической регрессии $p(x, y) = p(y|x) \cdot p(x) = \sigma(\langle w, x \rangle) \cdot const(w)$:

- $L(w, X^m) = \sum_{i=1}^{m} \log \sigma(\langle w, x_i \rangle y_i) + const(w) \to \max_w$

Максимизация L эквивалентна минимизации аппроксимированного Э.Р. R:

$$R(w, X^m) = \sum_{i=1}^{m} \log(1 + \exp(-\langle w, x_i \rangle y_i)) \to \min_w$$

Многоклассовая логистическая регрессия

Рассмотрим случай произвольного количества классов $|Y| > 2$. Тогда линейный классификатор (напоминание):

$$a(x) = \arg\max_{c \in Y} \langle w^c, x \rangle \quad x, w^c \in \mathbb{R}^n$$

Многоклассовая логистическая регрессия

Рассмотрим случай произвольного количества классов $|Y| > 2$. Тогда линейный классификатор (напоминание):

$$a(x) = \arg\max_{c \in Y} \langle w^c, x \rangle \quad x, w^c \in \mathbb{R}^n$$

Вероятность принадлежности объекта x к классу c определяется т.н. функцией SoftMax:

$$SoftMax(\langle w^c, x \rangle) = P(y = c | x, w) = \frac{\exp(\langle w^c, x \rangle)}{\sum_{z \in Y} \exp(\langle w^z, x \rangle)}$$

Т.о. функция $SoftMax : \mathbb{R}^{|Y|} \to \mathbb{R}^{|Y|}$ преобразует любой вещественнозначный вектор в вектор дискретного распределения.

О переобучении

Причины переобучения

- Маленькая обучающая выборка; большое число признаков;
- Признаки линейно зависимы;
- Неинформативные (шумовые) признаки.

О переобучении

Причины переобучения

- Маленькая обучающая выборка; большое число признаков;
- Признаки линейно зависимы;
- Неинформативные (шумовые) признаки.

Проявление переобучения

- Резкое увеличение нормы w;
- Большая разница в ошибке классификации на тестовой и обучающей выборках;

О переобучении

Причины переобучения

- Маленькая обучающая выборка; большое число признаков;
- Признаки линейно зависимы;
- Неинформативные (шумовые) признаки.

Проявление переобучения

- Резкое увеличение нормы w;
- Большая разница в ошибке классификации на тестовой и обучающей выборках;

Борьба с переобучением

- Ранняя остановка обучения;
- Уменьшение норм весов (**регуляризация**);

Вероятностный смысл простой регуляризации

Рассмотрим принцип максимума совместного правдоподобия данных и модели, или MAP (Maximum A Posteriori Probability).

Дано:

- Параметрическая модель плотности распределения $p(x, y|w)$
- Априорная информация о плотности распределения параметров модели $p(w)$
  Например, параметрическое семейство априорных распределений $p(w; h)$, где h — неизвестная и неслучайная величина (гиперпараметр).

Вероятностный смысл простой регуляризации

Рассмотрим принцип максимума совместного правдоподобия данных и модели, или MAP (Maximum A Posteriori Probability).

Дано:

- Параметрическая модель плотности распределения $p(x, y|w)$
- Априорная информация о плотности распределения параметров модели $p(w)$
  Например, параметрическое семейство априорных распределений $p(w; h)$, где h — неизвестная и неслучайная величина (гиперпараметр).

Тогда:

- Плотность $p(X^m, w; h) = p(X^m|w)p(w; h)$
- Максимизируем логарифм совместного распределения

$$L(w, X^m) = \ln p(X^m, w; h) = \sum_{i=1}^{m} \ln p(x_i, y_i|w) + \ln p(w; h) \to \max_{w,h}$$

L_2-регуляризация

Рассмотрим введение квадратичного штрафа за увеличение нормы весов в функционал Э.Р.:

$$R_\tau(w, X^m) = R(w, X^m) + \frac{\tau}{2}||w||^2 \to \min_w$$

Тогда градиент Э.Р.: $\nabla R_\tau(w, X^m) = \nabla R(w, X^m) + \tau w$,

L_2-регуляризация

Рассмотрим введение квадратичного штрафа за увеличение нормы весов в функционал Э.Р.:

$$R_\tau(w, X^m) = R(w, X^m) + \frac{\tau}{2}||w||^2 \to \min_w$$

Тогда градиент Э.Р.: $\nabla R_\tau(w, X^m) = \nabla R(w, X^m) + \tau w$,
А градиентный шаг: $w^{(t+1)} = (1 - \tau\eta)w^{(t)} - \eta\nabla R(w^{(t)}, X^m)$.

L_2-регуляризация

Рассмотрим введение квадратичного штрафа за увеличение нормы весов в функционал Э.Р.:

$$R_\tau(w, X^m) = R(w, X^m) + \frac{\tau}{2}||w||^2 \to \min_w$$

Тогда градиент Э.Р.: $\nabla R_\tau(w, X^m) = \nabla R(w, X^m) + \tau w$,
А градиентный шаг: $w^{(t+1)} = (1 - \tau\eta)w^{(t)} - \eta\nabla R(w^{(t)}, X^m)$.

Подбор параметра регуляризации τ

- Больше значение τ - больше штрафа за переобучение (но сходимость медленнее!)
- Методом скользящего контроля (cross-validation);

Takeaway notes

1. Эмпирическим риском измеряем качество классификатора

Takeaway notes

1. Эмпирическим риском измеряем качество классификатора
2. На практике используется аппроксимационный эмпирический риск

Takeaway notes

1. Эмпирическим риском измеряем качество классификатора
2. На практике используется аппроксимационный эмпирический риск
3. Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)

Takeaway notes

1. Эмпирическим риском измеряем качество классификатора
2. На практике используется аппроксимационный эмпирический риск
3. Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)
4. Градиентный спуск – алгоритм оптимизации первого порядка

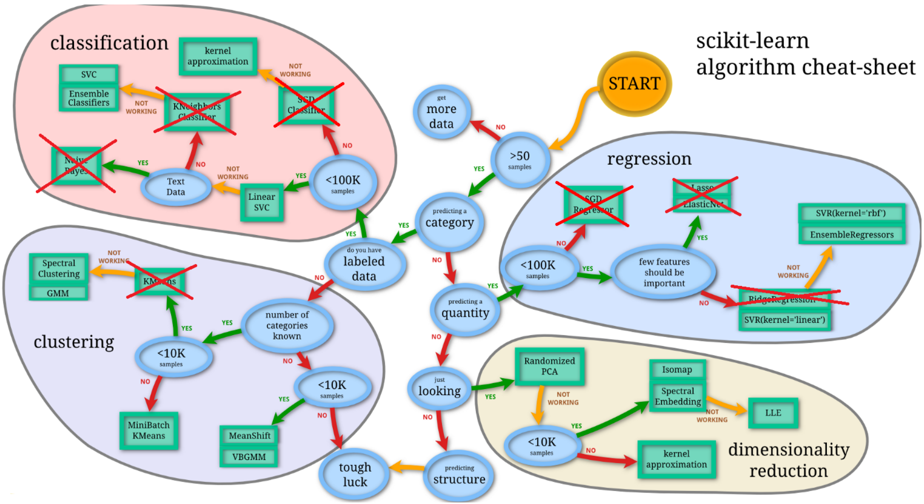
Takeaway notes

1. Эмпирическим риском измеряем качество классификатора
2. На практике используется аппроксимационный эмпирический риск
3. Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)
4. Градиентный спуск – алгоритм оптимизации первого порядка
5. SGD – практическая версия GD

Takeaway notes

1. Эмпирическим риском измеряем качество классификатора
2. На практике используется аппроксимационный эмпирический риск
3. Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)
4. Градиентный спуск – алгоритм оптимизации первого порядка
5. SGD – практическая версия GD
6. Регуляризация изменяет коэффициенты для SGD

Дорожная карта Scikit-Learn[8]



[8] https://scikit-learn.org/stable/tutorial/machine_learning_map/

Источники

На основе материалов сайта http://www.machinelearning.ru.