Методы машинного обучения

6. Линейные методы классификации (метод стохастического градиента)

Методы классификации

- **Метрические методы классификации**
- **Логические методы классификации**
- **Линейные методы**: метод стохастического градиента
- Линейные методы: метод опорных векторов
- Методы восстановления регрессии
- Нелинейная и непараметрическая регрессия
- Байесовские методы классификации
- Поиск ассоциативных правил
- Нейронные сети и бустинг
- Прогнозирование временных рядов

Видеолекции: http://shad.yandex.ru/lectures

Презентации и текст: http://www.machinelearning.ru/wiki

Машинное обучение (курс лекций, К.В.Воронцов)

План лекции

Градиентные методы обучения

- Минимизация эмпирического риска
- Линейный классификатор
- Метод стохастического градиента SG

Метрические методы. Понятие отступа

Рассмотрим классификатор $a\colon X \to Y$ вида

аналогия легче рассуждений

задача: отобрать оптимальное число объектов обучающей выборки, а остальные – выкинуть

$$a(u) = \arg\max_{y \in Y} \Gamma_y(u), \quad u \in X.$$

$\Gamma_y(u)$* показывает, насколько важен объект обучающей выборки, насколько *u* близок к классу *y

Определение

Отступом (margin) объекта $x_i \in X^\ell$ относительно классификатора $a(u)$ называется величина

смотрим, насколько оценка правильного класса превышает оценку за другие классы; если *M>>0*, x_i является эталоном

отступ – суть отдаленность от другого класса

$$M(x_i) = \Gamma_{y_i}(x_i) - \max_{y \in Y \setminus y_i} \Gamma_y(x_i).$$

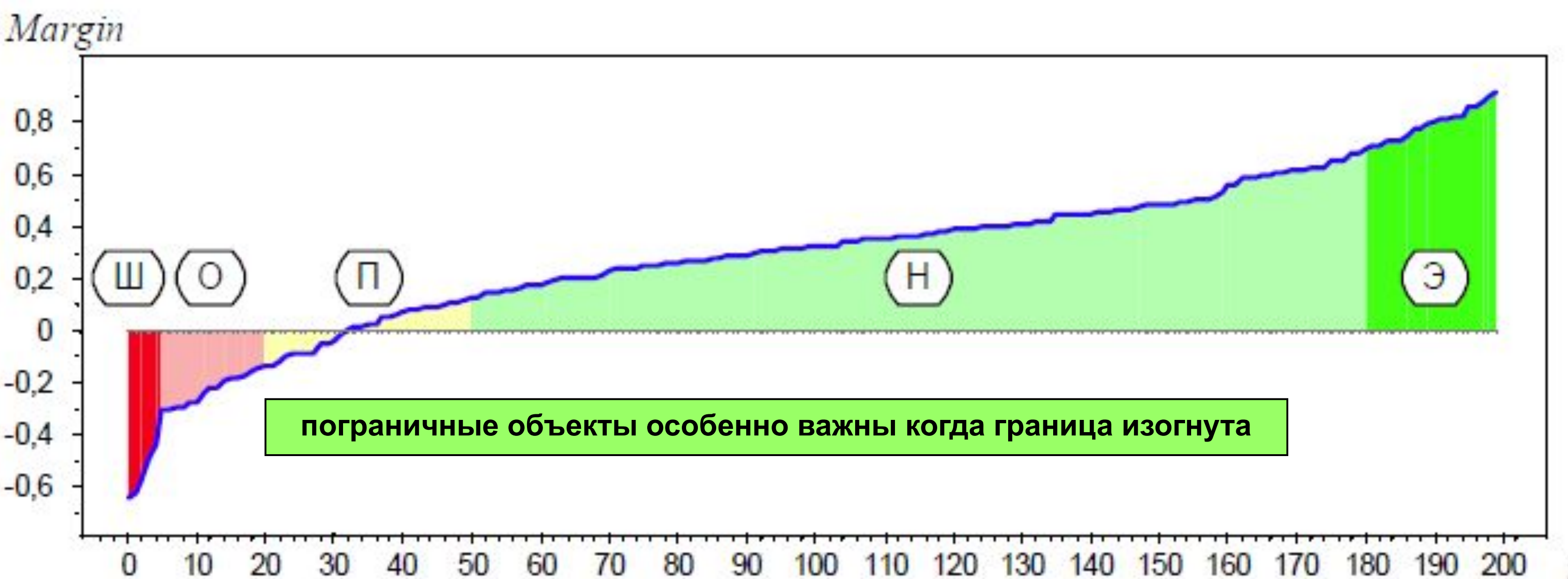
- Отступ показывает *степень типичности* объекта: чем больше $M(x_i)$, тем «глубже» x_i в своём классе;
- $M(x_i) < 0 \Leftrightarrow a(x_i) \neq y_i$;

на объекте произошла ошибка

Метрические методы. Типы объектов в зависимости от отступа

- Э — *эталонные* (можно оставить только их);
- Н — *неинформативные* (можно удалить из выборки);
- П — *пограничные* (их классификация неустойчива);
- О — *ошибочные* (причина ошибки — плохая модель);
- Ш — *шумовые* (причина ошибки — плохие данные).

полезно строить такие картинки – показывают, сколько каких объектов находится в выборке



Логические методы. В каком виде ищут закономерности?

3. *Полуплоскость* — линейная пороговая функция:

снова используется небольшое число признаков *j* (некое подпространство)

$$R(x) = \Big[\sum_{j\in J} w_j f_j(x) \geqslant w_0\Big].$$

получаем линейную комбинацию признаков (*будут рассмотрены далее*), а не $\wedge$, но здесь складываются «*км* с *кг*»

4. *Шар* — пороговая функция близости:

если вокруг точки x_0 описали шар радиусом w_0, в котором много объектов одного класса (а других – мало), то это закономерность

$$R(x) = \big[r(x, x_0) \leqslant w_0\big],$$

метрика *r*, аналог того, что было в метрических методах (эталонность сравнения)

АВО — алгоритмы вычисления оценок [Ю. И. Журавлёв, 1971]:

способ вычисления оценки

$$r(x, x_0) = \max_{j\in J} w_j \big|f_j(x) - f_j(x_0)\big|.$$

SCM — машины покрывающих множеств [M. Marchand, 2001]:

способ вычисления оценки

$$r(x, x_0) = \sum_{j\in J} w_j \big|f_j(x) - f_j(x_0)\big|^{\gamma}.$$

используется прецедентная логика в проверке и интерпретации результата

Параметры J, w_j, w_0, x_0 настраиваются по обучающей выборке путём оптимизации *критерия информативности*.

Задача построения разделяющей поверхности

- Задача классификации с двумя классами, $Y = \{-1, +1\}$: по обучающей выборке $X^\ell = (x_i, y_i)_{i=1}^\ell$ построить алгоритм классификации $a(x, w) = \operatorname{sign} f(x, w)$, где
  $f(x, w)$ — *разделяющая (дискриминантная) функция*,
  w — вектор параметров.
- $f(x, w) = 0$ — разделяющая поверхность;
  $M_i(w) = y_i f(x_i, w)$ — *отступ* (margin) объекта x_i;
  $M_i(w) < 0 \iff$ алгоритм $a(x, w)$ ошибается на x_i.

хотим, чтобы классификатор был основан на принципе разделения

от поверхности

x **– признаковое описание объекта,**
w **– вектор параметров**

если функция *f* возвратила на объекте x_i:
значение > 0, то относим x_i в класс +1,
значение < 0, то относим x_i в класс –1,
значение = 0, то… относим x_i, например, в класс +1

преимущество таких классификаторов: вводится понятие «*надёжность классификации*», которое связано с тем, насколько далеко объект находится от границы между классами (если объект лежит близко к границе, то небольшое изменение в условиях задачи способно менять его классовую принадлежность)

если *y* и *f* <u>одного знака</u>, то <u>ошибки нет</u>, и чем больше абсолютное значение величины $M_i(w)$, тем надёжнее классификация; если *y* и *f* <u>разных знаков</u>, то <u>ошибка</u>, и если большое абсолютное значение $M_i(w)$, то это однозначно выброс

Задача построения разделяющей поверхности

понятие отступа позволяет записать функционал числа ошибок на обучающей выборке (эмпирический риск)

- $f(x, w) = 0$ — разделяющая поверхность;
  $M_i(w) = y_i f(x_i, w)$ — *отступ* (margin) объекта x_i;
  $M_i(w) < 0 \iff$ алгоритм $a(x, w)$ ошибается на x_i.
- *Минимизация эмпирического риска:*

замена пороговой функции потерь на непрерывную

$$Q(w) = \sum_{i=1}^{\ell} [M_i(w) < 0] \leqslant \widetilde{Q}(w) = \sum_{i=1}^{\ell} \mathscr{L}(M_i(w)) \to \min_w;$$

огрубление характеристики – ошибка или не ошибка – теряется информация о надёжности *i*-ого объекта

сделаем так, чтобы функционал непрерывным образом зависел бы от отступов

функция потерь $\mathscr{L}(M)$ невозрастающая, неотрицательная.

<u>преимущества функции потерь *L(M)*</u>: 1. более тонкая характеризация надёжности классификации, 2. получаем инструмент, который позволит применять градиентные методы оптимизации

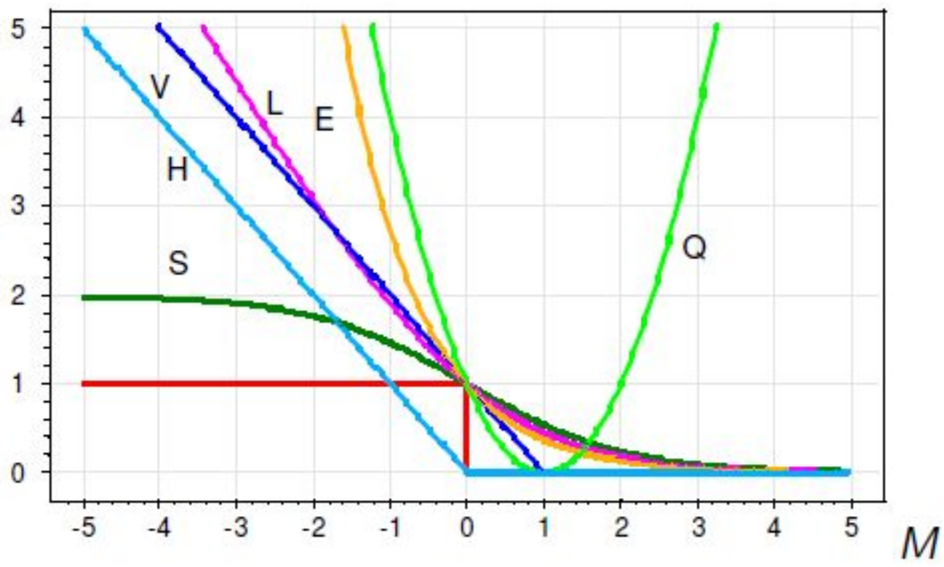
подбираем *L(M)* так, чтобы она сверху аппроксимировала пороговую функцию потерь, а т.к. *L(M)* мы минимизируем, то минимизируется и исходный функционал; если решать первую задачу, то это тяжёлая задача комбинаторной оптимизации, которая имеет бесконечно много решений

Непрерывные аппроксимации пороговой функции потерь

Часто используемые непрерывные функции потерь $\mathscr{L}(M)$:

градиентные методы – численные методы решения с помощью градиента задач, сводящихся к нахождению экстремумов функции

градиент показывает направление самого быстрого возрастания функции



<u>современный принцип</u>: можно как угодно менять функции потерь и получать тот или иной по качеству метод, потому что решение сильно зависит от *L(M)* – зависит от того, как мы штрафуем за ошибки

$V(M) = (1 - M)_+$ — кусочно-линейная (SVM);

$H(M) = (-M)_+$ — кусочно-линейная (Hebb's rule);

$L(M) = \log_2(1 + e^{-M})$ — логарифмическая (LR);

$Q(M) = (1 - M)^2$ — квадратичная (FLD);

$S(M) = 2(1 + e^{M})^{-1}$ — сигмоидная (ANN);

$E(M) = e^{-M}$ — экспоненциальная (AdaBoost);

$[M < 0]$ — пороговая функция потерь.

Линейный классификатор

$f_j: X \to \mathbb{R}$, $j = 1, \dots, n$ — числовые признаки;
Линейный алгоритм классификации:

возьмем вместо непонятной дискриминантной функции *f* линейную функцию

будем считать, что объекты заданы векторами из R^n, т.е. имеется *n* числовых признаков $f_1 \dots f_n$ и мы составляем их линейную комбинацию с весами *w*

$$a(x, w) = \operatorname{sign}\left(\sum_{j=1}^{n} w_j f_j(x) - w_0\right),$$

где $w_0, w_1, \dots, w_n \in \mathbb{R}$ — коэффициенты (веса признаков);
Введём константный признак $f_0 \equiv -1$.

технический прием для сокращения записи

Векторная запись:

$$a(x, w) = \operatorname{sign}\bigl(\langle w, x\rangle\bigr).$$

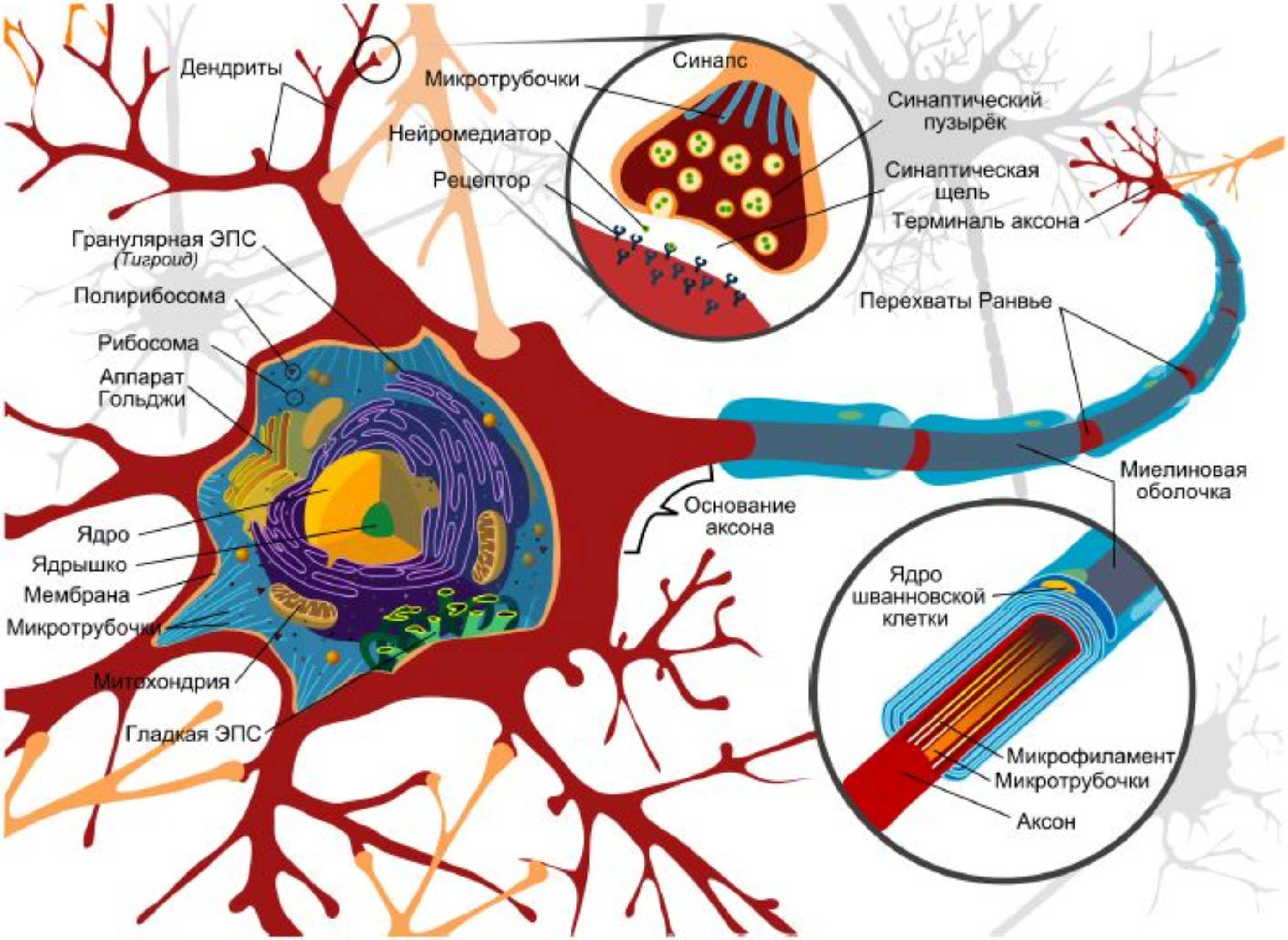
нас интересует знак скалярного произведения *w* на *x*

Отступы объектов x_i:

$$M_i(w) = \langle w, x_i\rangle y_i.$$

***x* и *w* теперь находятся в пространстве R^{n+1}**

Похож ли нейрон на линейный классификатор?



Похож ли нейрон на линейный классификатор?

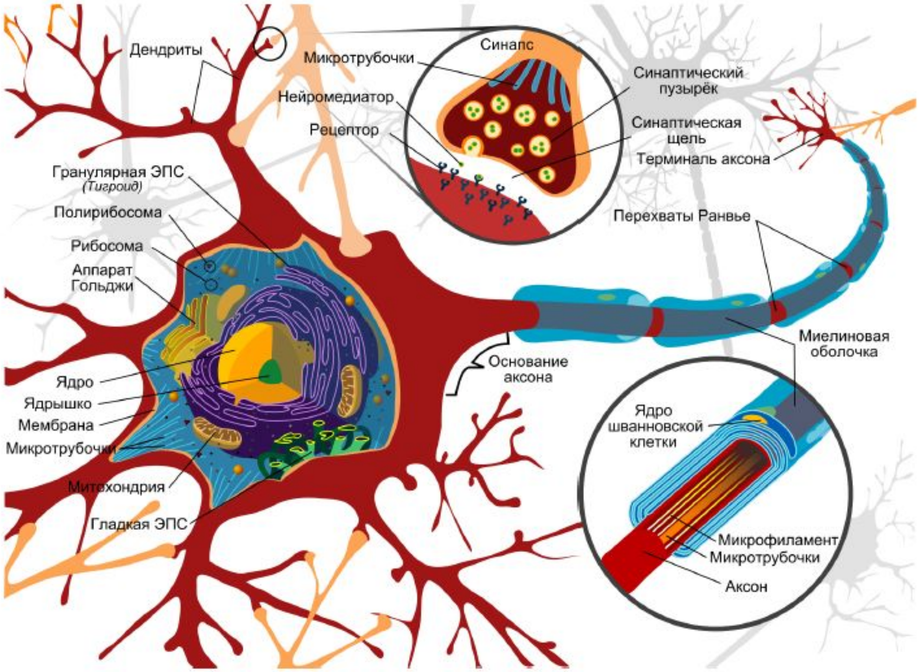
Нейрон – это структурно-функциональная единица нервной системы, представляет собой электрически возбудимую клетку, которая обрабатывает и передает информацию посредством электрических и химических сигналов.

Аксон – обычно длинный отросток нейрона, приспособленный для проведения возбуждения и информации от тела нейрона или от нейрона к исполнительному органу.

Дендриты – как правило, короткие и сильно разветвлённые отростки нейрона, служащие главным местом образования влияющих на нейрон возбуждающих и тормозных синапсов (разные нейроны имеют различное соотношение длины аксона и дендритов), и которые передают возбуждение к телу нейрона.

Нейрон может иметь несколько дендритов и обычно только один аксон. Один нейрон может иметь связи со многими (до 20 тысяч) другими нейронами.

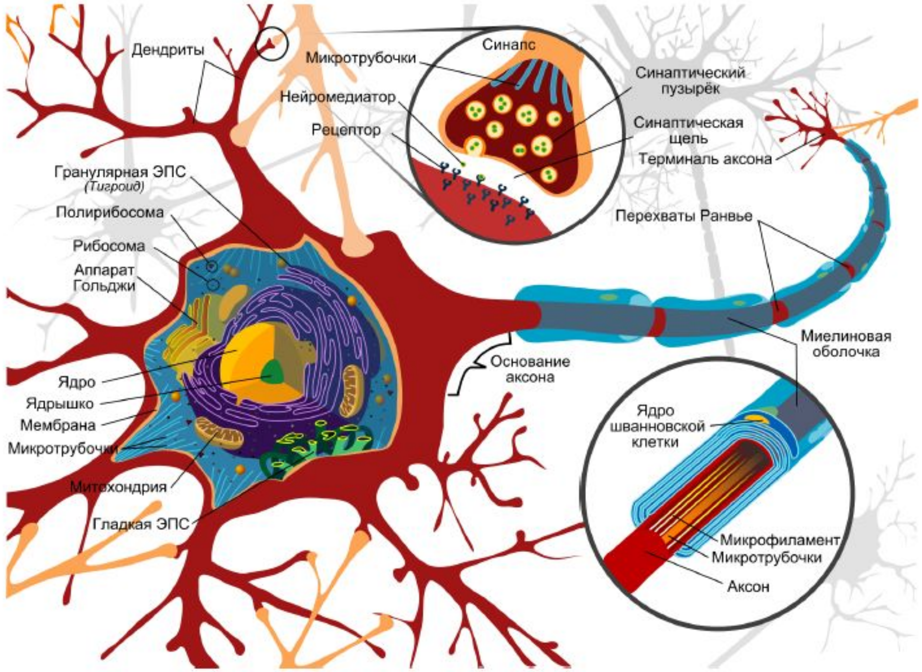
Похож ли нейрон на линейный классификатор?



в синапсах начинает концентрироваться отрицательный заряд, который затем переходит внутрь (ядра) клетки, и там, как только происходит концентрация слишком большого отрицательного заряда, который пришёл отовсюду (ото всех синапсов), клетка генерирует электрический импульс, который по аксону бежит до конца и так порождается «волна возбуждения»; если к той клетке, куда пришёл импульс, также придут импульсы от других клеток, она тоже возбудится и волна продолжится

Синапс – место контакта между двумя нейронами или между нейроном и получающей сигнал эффекторной клеткой. Служит для передачи нервного импульса между двумя клетками, причём в ходе синаптической передачи амплитуда и частота сигнала могут регулироваться.

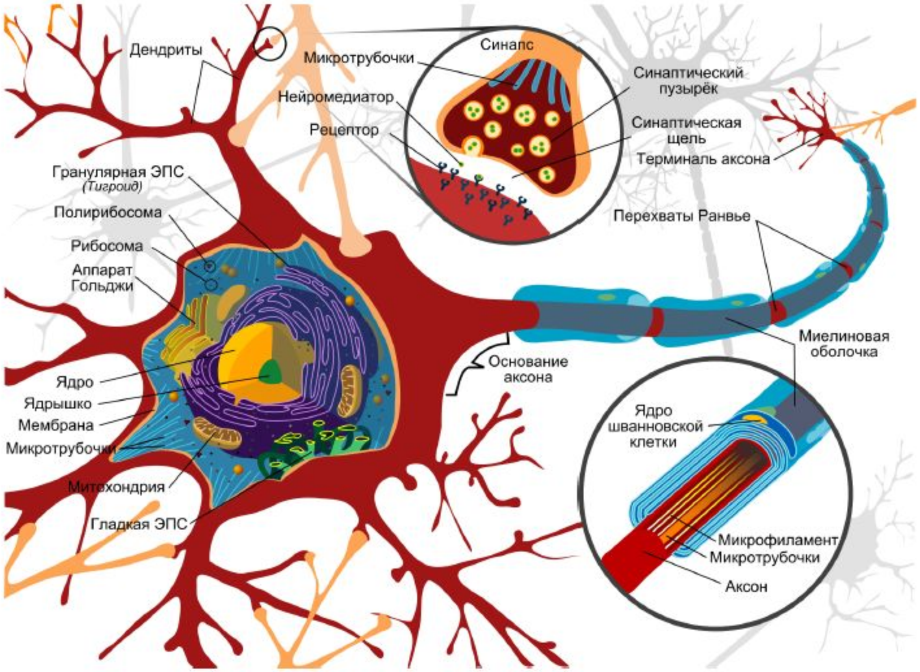
Похож ли нейрон на линейный классификатор?



клетка работает практически как дискретное устройство: после того, как она возбудилась, ей нужно некоторое время отдохнуть и она не способна генерировать импульсы; т.е. клетка – это автомат, который на входе получил заряды, суммировал их (с коэффициентами, потому что каждый синапс индивидуален и имеет свою силу связи – какую долю заряда он пропускает внутрь клетки; бывают и тормозящие синапсы, т.е. коэффициенты бывают и отрицательными)

Синапс – место контакта между двумя нейронами или между нейроном и получающей сигнал эффекторной клеткой. Служит для передачи нервного импульса между двумя клетками, причём в ходе синаптической передачи амплитуда и частота сигнала могут регулироваться.

Похож ли нейрон на линейный классификатор?



т.е. аналогия с линейным классификатором полная: *величина заряда*, который приходит в клетку через синапсы – это *признаки f*, синоптические связи – это веса *w*, а коэффициент w_0 – это тот порог, который необходим для того, чтобы началась генерация импульса

линейный классификатор – это, пусть грубая, но модель нервной клетки, поэтому создавая композиции таких классификаторов, есть надежда конструировать обучающиеся системы, которые обучаются также как человек (хотя видов нервных клеток позже было открыто много)

Синапс – место контакта между двумя нейронами или между нейроном и получающей сигнал эффекторной клеткой. Служит для передачи нервного импульса между двумя клетками, причём в ходе синаптической передачи амплитуда и частота сигнала могут регулироваться.

Математическая модель нейрона

Линейная модель нейрона МакКаллока-Питтса [1943]:

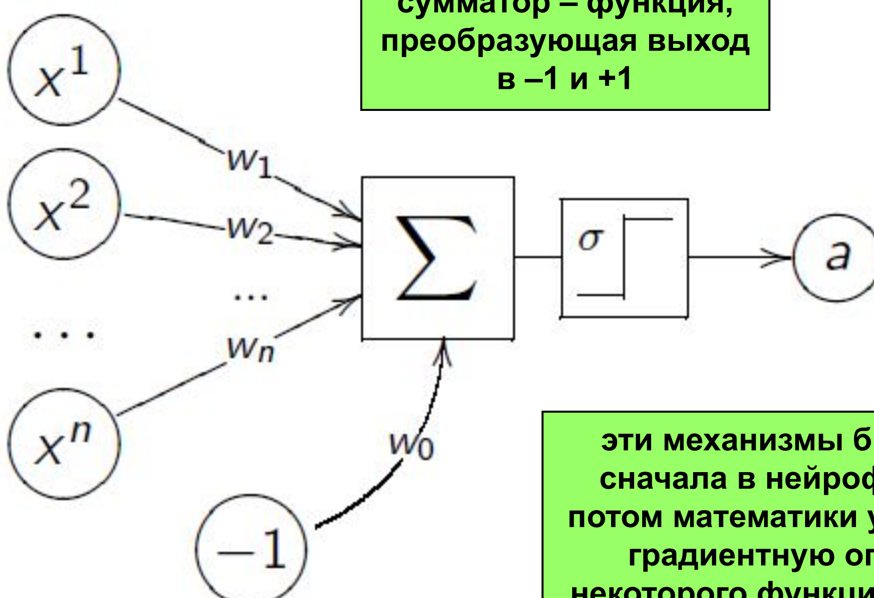
$$a(x, w) = \sigma(\langle w, x \rangle) = \sigma\left(\sum_{j=1}^{n} w_j f_j(x) - w_0 \right),$$

где $\sigma(s)$ — функция активации (в частности, sign).

в 1940-1950 годы проводилось большое число нейрофизиологических экспериментов в попытке понять, как происходит обучение в нервной клетке

<u>основной вывод</u>: запоминают синоптические связи, т.е. если две клетки последовательно возбудились, то первая правильно предугадала тот, ответ, который генерирует следующая, за это синоптическая связь награждается усилением – теперь *w* становится больше

сумматор – функция, преобразующая выход в –1 и +1



эти механизмы были открыты сначала в нейрофизиологии, а потом математики усмотрели в них градиентную оптимизацию некоторого функционала качества

пускай линейный классификатор <u>задан</u>, нервная ли это клетка или что-то ещё – не важно

Градиентный метод численной минимизации

Минимизация аппроксимированного эмпирического риска:

задан некий функционал потерь, который нужно минимизировать

$$Q(w; X^\ell) = \sum_{i=1}^{\ell} \mathscr{L}(\langle w, x_i \rangle y_i) \to \min_w .$$

в численных методах оптимизации самый простой метод – метод градиентного спуска

Численная минимизация методом *градиентного спуска*:

$w^{(0)} :=$ начальное приближение;

каждый следующий шаг – идти в направлении антиградиента

$$w^{(t+1)} := w^{(t)} - \eta \cdot \nabla Q(w^{(t)}), \qquad \nabla Q(w) = \left(\frac{\partial Q(w)}{\partial w_j} \right)_{j=0}^{n},$$

вектор *w* должен сместиться на величину *η* (эта)

где η — *градиентный шаг*, называемый также *темпом обучения*.

$$w^{(t+1)} := w^{(t)} - \eta \sum_{i=1}^{\ell} \mathscr{L}'(\langle w^{(t)}, x_i \rangle y_i) \, x_i \, y_i .$$

подставили, преобразовали и получили такую формулу

градиент показывает направление самого быстрого возрастания функции

Градиентный метод численной минимизации

$$w^{(t+1)} := w^{(t)} - \eta \sum_{i=1}^{\ell} \mathscr{L}'\big(\langle w^{(t)}, x_i\rangle y_i\big)\, x_i\, y_i.$$

<u>если условия задачи таковы, что данных очень много</u> (выборка избыточна), то формула говорит о том, что:
нужно суммировать все объекты, а потом вектор параметров w сдвинется на малый шаг,
но это весьма долго и не очень эффективно

Идея ускорения сходимости:

брать (x_i, y_i) по одному и сразу обновлять вектор весов.

<u>выход</u>: идти не по всей обучающей выборке, а по подвыборке;
а если обобщить это, то можно:
1. брать только один случайный объект – одно слагаемое этой суммы (см. формулу) и на его основании подправить вектор весов *w*;
2. брать другой случайный объект и на его основании подправить вектор весов *w* и т.д.
ЗМ. вектор весов будет метаться, но «в правильном направлении»

закон больших чисел говорит о том, что суммы можно приближенно вычислять так: взять около 30 случайных слагаемых и мы значительно приблизимся к сумме

<u>преимущество метода на больших данных</u>:
можно обучиться, не просмотрев все данные

Алгоритм SG (Stochastic Gradient)

Стохастический – умеющий угадывать

Вход:

выборка X^ℓ; темп обучения η; параметр λ;

или «градиентный шаг»

Выход:

веса $w_0, w_1, \dots, w_n$;

λ можно назначить *1/k*, где *k* – это количество усредняемых потерь ε_i

---

1: инициализировать веса w_j, $j = 0, \dots, n$;

будет отдельный слайд на тему эвристик

2: инициализировать текущую оценку функционала:
$Q := \sum_{i=1}^{\ell} \mathscr{L}\big(\langle w, x_i\rangle y_i\big)$;

текущая оценка нужна для учёта средних потерь классификатора на выборке

3: **повторять**

4: выбрать объект x_i из X^ℓ (например, случайно);

не всегда

5: вычислить потерю: $\varepsilon_i := \mathscr{L}\big(\langle w, x_i\rangle y_i\big)$;

пропустили выбранный объект через классификатор

6: градиентный шаг: $w := w - \eta \mathscr{L}'\big(\langle w, x_i\rangle y_i\big) x_i y_i$;

7: оценить значение функционала: $Q := (1 - \lambda)Q + \lambda\varepsilon_i$;

8: **пока** значение Q и/или веса w не стабилизируются;

6: примеряем формулу для выбранного объекта

7: способ грубо оценить *Q*, не пересчитывая его на всей выборке

стабилизация определяется <u>вручную</u>, когда значение *Q* выходит на ровный участок, когда видно, что в течение ряда последних итераций значение *Q* остается в неком диапазоне

Алгоритм SG: шаг 1. инициализация весов

Возможны варианты:

1. $w_j := 0$ для всех $j = 0, \dots, n$; **часто предлагается в литературе**
2. небольшие случайные значения:
   $w_j := \mathrm{random}\left(-\frac{1}{2n}, \frac{1}{2n}\right)$; **чтобы избежать «паралича нейронной сети»**
3. $w_j := \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$, $f_j = \big(f_j(x_i)\big)_{i=1}^{\ell}$ — вектор значений признака.

   Упражнение: доказать, что оценка w оптимальна, если
   1) функция потерь квадратична и
   2) признаки некоррелированы, $\langle f_j, f_k \rangle = 0$, $j \neq k$.
4. обучение по небольшой случайной подвыборке объектов;
5. многократные запуски из разных случайных начальных приближений и выбор лучшего решения.

Алгоритм SG:
шаг 4. порядок предъявления объектов

дальние объекты мало повлияют на модификацию вектора *w*, а близкие – наоборот

Возможны варианты:

1. *перетасовка объектов* (shuffling):
попеременно брать объекты из разных классов;
2. чаще брать те объекты, на которых была допущена бо́льшая ошибка **это объекты из окрестности разделяющей гиперплоскости**
(чем меньше M_i, тем больше вероятность взять объект)
(чем меньше $|M_i|$, тем больше вероятность взять объект);
3. вообще не брать «хорошие» объекты, у которых $M_i > \mu_+$
(при этом немного ускоряется сходимость);
4. вообще не брать объекты-«выбросы», у которых $M_i < \mu_-$
(при этом может улучшиться качество классификации);

отступ – это расстояние до гипер-плоскости

Параметры μ_+, μ_- придётся подбирать.

ЗМ. нарисовать рисунок подхода к увеличению скорости сходимости

Алгоритм SG:
шаг 6. выбор величины градиентного шага

откуда брать темп обучения?

Возможны варианты:

1. сходимость гарантируется (для выпуклых функций) при

если устремить к 0, но не слишком быстро и не слишком медленно

$$\eta_t \to 0, \quad \sum_{t=1}^{\infty} \eta_t = \infty, \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty,$$

подбирать шаг в конкретной задаче – это искусство

в частности можно положить $\eta_t = 1/t$;

2. *метод скорейшего градиентного спуска*:

задача одномерной оптимизации

$$Q(w - \eta \nabla Q(w)) \to \min_{\eta},$$

позволяет найти *адаптивный шаг* η^*;

Упражнение: доказать, что при квадратичной функции потерь $\eta^* = \|x_i\|^{-2}$.

3. пробные случайные шаги

— для «выбивания» из локальных минимумов;

SG: Достоинства и недостатки

Достоинства:

1. легко реализуется;
2. легко обобщается на любые f, $\mathscr{L}$;
3. возможно динамическое (потоковое) обучение;
4. на сверхбольших выборках не обязательно брать все x_i;

штраф за ошибки для того, чтобы разделяющая поверхность прошла как можно дальше от объектов

Недостатки:

1. возможна расходимость или медленная сходимость;
2. застревание в локальных минимумах;
3. подбор комплекса эвристик является искусством;
4. проблема переобучения;

приходит с опытом

SG: Проблема переобучения

Возможные причины переобучения:

1. слишком мало объектов; слишком много признаков;
2. линейная зависимость (мультиколлинеарность) признаков:
   пусть построен классификатор: $a(x, w) = \operatorname{sign}\langle w, x\rangle$;
   мультиколлинеарность: $\exists u \in \mathbb{R}^{n+1}\colon \langle u, x\rangle \equiv 0$;
   тогда $\forall\gamma \in \mathbb{R} \quad a(x, w) = \operatorname{sign}\langle w + \gamma u, x\rangle$

n > l

если в признаке (явно или неявно) заложена линейная комбинация других признаков (*доход*, *доход жены*, *доход семьи*)

Симптоматика:

1. слишком большие веса $\|w\|$;
2. неустойчивость $a(x, w)$;
3. $Q(X^\ell) \ll Q(X^k)$;

малые изменения *x* (при таких *w*) или изменение обучающей выборки могут приводить к радикальному изменению решения

мультиколлениарность – наличие сильной корреляции между признаками

Терапия:

1. сокращение весов (weight decay);
2. ранний останов (early stopping);

Анонс. Метод опорных векторов SVM (Support Vector Machine)

Задача классификации: $X = \mathbb{R}^n$, $Y = \{-1, +1\}$,
по обучающей выборке $X^\ell = (x_i, y_i)_{i=1}^\ell$
найти параметры $w \in \mathbb{R}^n$, $w_0 \in \mathbb{R}$ алгоритма классификации

строим линейный классификатор в виде конструкции как на прошлой лекции

$$a(x, w) = \operatorname{sign}\big(\langle x, w\rangle - w_0\big).$$

скалярное произведение вектора признакового описания объекта *x* и вектора весов *w*

вектор *w* – это направляющий вектор разделяющей гиперплоскости, а w_0 – это скаляр, сдвиг гиперплоскости

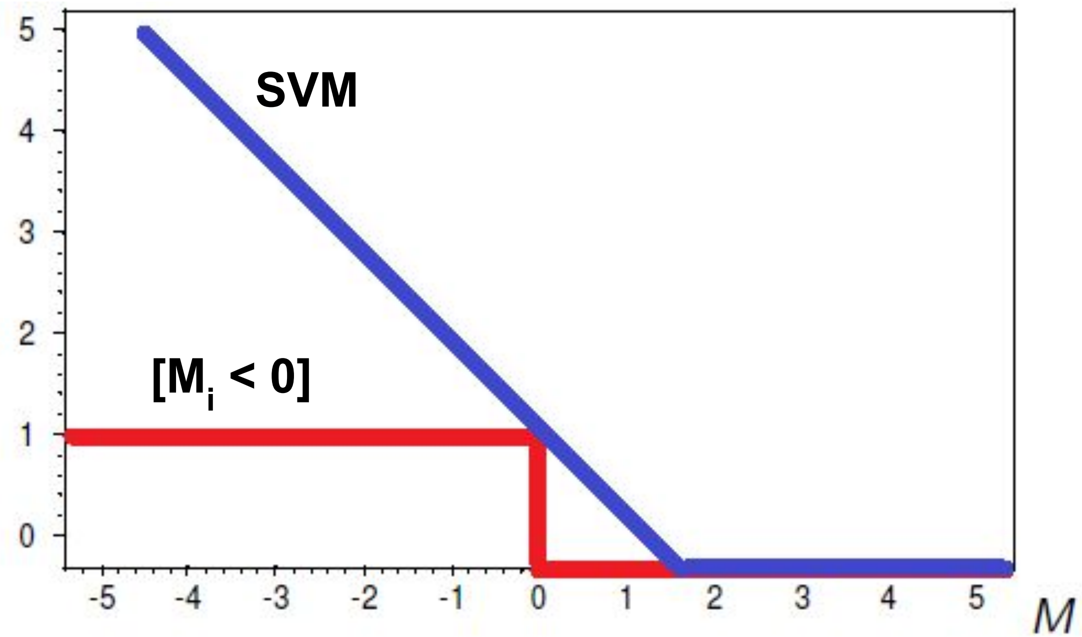
Метод минимизации аппроксимированного регуляризованного эмпирического риска:

используем аппроксимацию пороговой функции потерь

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

где $M_i(w, w_0) = y_i\big(\langle x_i, w\rangle - w_0\big)$ — *отступ* (margin) объекта x_i.

Анонс. Метод опорных векторов SVM (Support Vector Machine)



если заменить красную ступеньку чем-то непрерывным, то получаем аппроксимацию пороговой функции потерь

1. метод SVM использует кусочно-линейную аппроксимацию, изображенную на рисунке синим цветом

2. в линейных методах хорошо работает регуляризация, которая спасает от мультиколлениарности; здесь используется классическая регуляризация – сумма квадратов коэффициентов

$$\frac{1}{2C}\|w\|^2$$

регуляризация – **метод добавления некоторой дополнительной информации к условию с целью решить некорректно поставленную задачу или предотвратить переобучение**

мультиколлениарность – **это тесная корреляционная взаимосвязь между отбираемыми для анализа признаками, совместно воздействующими на общий результат, которая затрудняет оценивание параметров**