Математические основы информационной безопасности

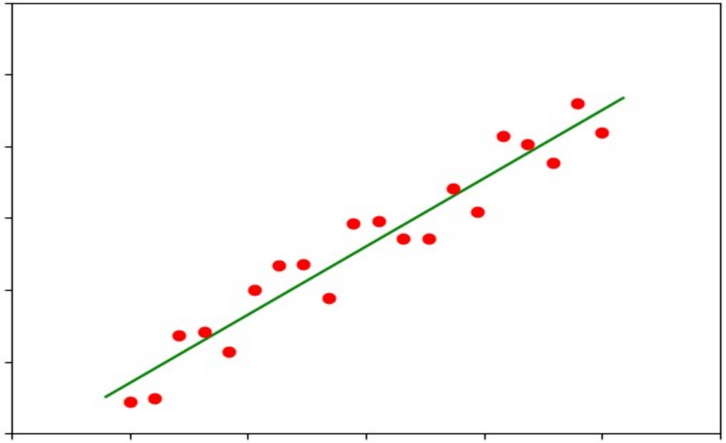
Груздев Дмитрий Николаевич

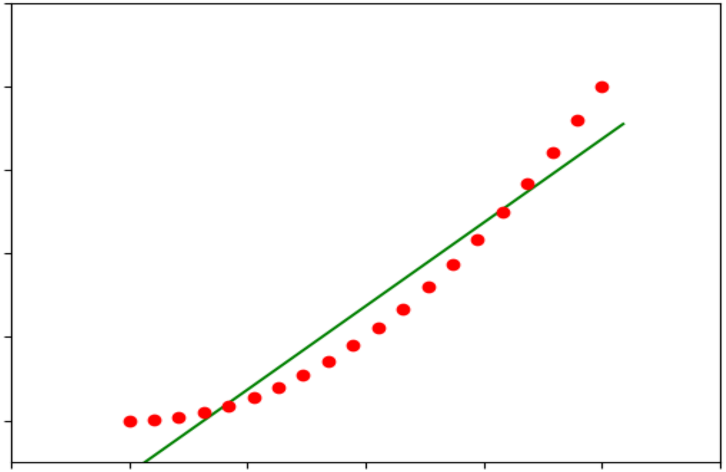
Переобучение

Ошибки естественны

Причины ошибок:

- Неточность измерений
- Неточность выбранной модели

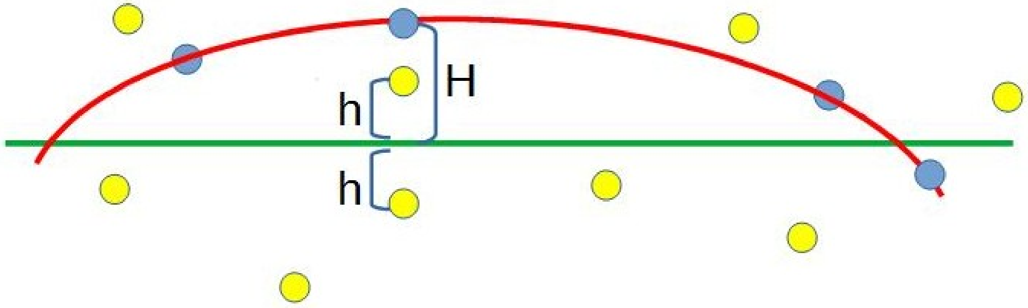


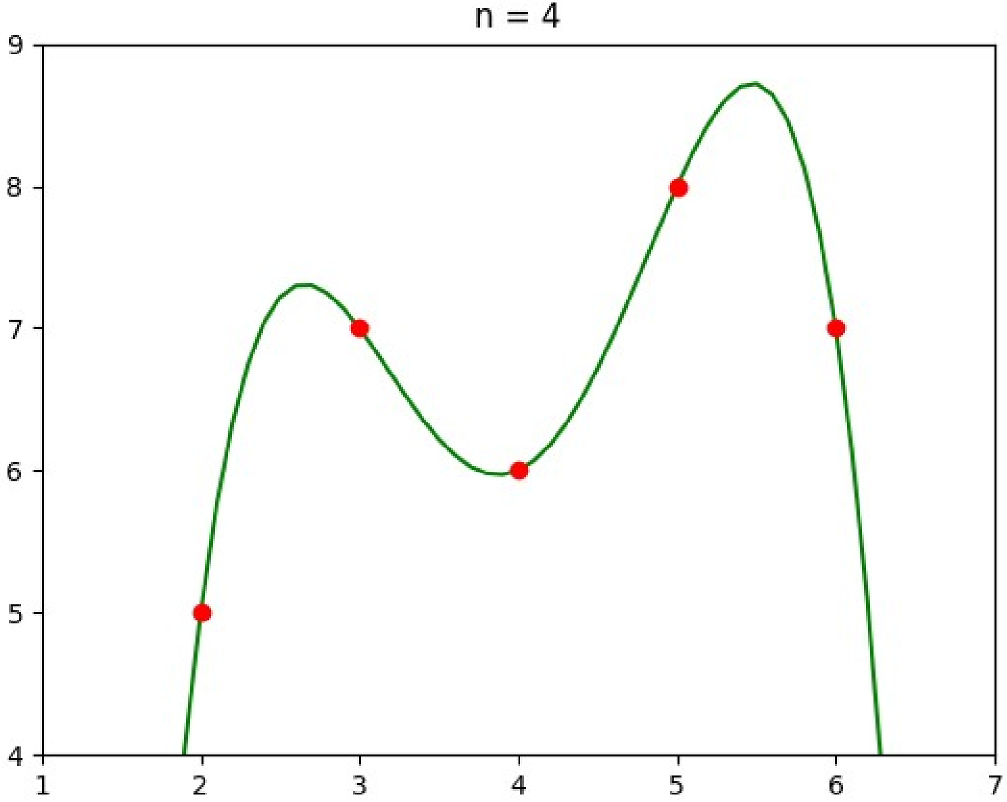


Неточность модели

$$y(x) = \frac{1}{1+25x^2};\qquad \mathrm{A(x)}\ \text{- полиномиальная регрессия}\ n = 38$$

Неточность измерений





E = h2 + h2 = 2*h2

E = (H-h)2 + (H+h)2 = 2*H2 + 2*h2

ΔE = 2*H2

Мультиколлинеарность

$y = \Theta_1 * x^{(1)} + \Theta_2 * x^{(2)} + \ldots + \Theta_n * x^{(n)}$

$a * x^{(a)} + b * x^{(b)} + c * x^{(c)} = 0$

$y = \Theta_1 * x^{(1)} + \Theta_2 * x^{(2)} + \ldots + \Theta_n * x^{(n)} + k * a * x^{(a)} + k * b * x^{(b)} + k * c * x^{(c)}$

$y = \Theta_1 * x^{(1)} + \ldots + (\Theta_a + ka) * x^{(a)} + (\Theta_b + kb) * x^{(b)} + (\Theta_c + kc) * x^{(c)} + \ldots + \Theta_n * x^{(n)}$

При обучении могут получиться любые значения весов из семейства.

При больших значениях весов алгоритм менее стабилен.

Регуляризация весов

$E = 0.5 * \Sigma(A(x_i) – y_i)^2$

$E = 0.5 * \Sigma(A(x_i) – y_i)^2 + \lambda * \Sigma_{1 \le i \le n}|\Theta_i|$ – L1 регуляризация

$E = 0.5 * \Sigma(A(x_i) – y_i)^2 + \lambda * \Sigma_{1 \le i \le n}\Theta_i^2$ – L2 регуляризация

Θ_0 не штрафуется

Изменение формулы для градиентного спуска:

- $\Delta\Theta_i = -\alpha * dE/d\Theta_i = -\alpha * dE_0/d\Theta_i \pm \alpha * \lambda$ – для L1
- $\Delta\Theta_i = -\alpha * dE/d\Theta_i = -\alpha * dE_0/d\Theta_i – 2 * \alpha * \lambda * \Theta_i$ – для L2

Ограничение весов

$\Sigma|w_{ij}^{(p)}| < \lambda$ - ограничение суммы весов

$|w_{ij}^{(p)}| < \lambda$ - ограничение каждого веса

Особенности:

- большая наглядность
- не изменяется формула ошибки

Выборки

$(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ – данные

Разделим данные на 3 части:

- $(x_1, y_1), \dots, (x_{tr}, y_{tr})$ – обучающая выборка
- $(x_{tr+1}, y_{tr+1}), \dots, (x_{cv}, y_{cv})$ – валидационная выборка
- $(x_{cv+1}, y_{cv+1}), \dots, (x_m, y_m)$ – контрольной выборка

Соотношение длин выборок ~ 3 : 1 : 1

Ошибки на выборках:

- $E_{train} = 0.5 * \Sigma(A(x_i) - y_i)^2$ – на обучающей выборке
- $E_{cv} = 0.5 * \Sigma(A(x_i) - y_i)^2$ – на валидационной выборке
- $E_{test} = 0.5 * \Sigma(A(x_i) - y_i)^2$ – на контрольной выборке

Подбор параметров алгоритма

- $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$ – данные, $x \in \mathbb{R}$, $y \in \mathbb{R}$

  A – полиномиальная регрессия.

  Требуется подобрать наилучшую степень многочлена.

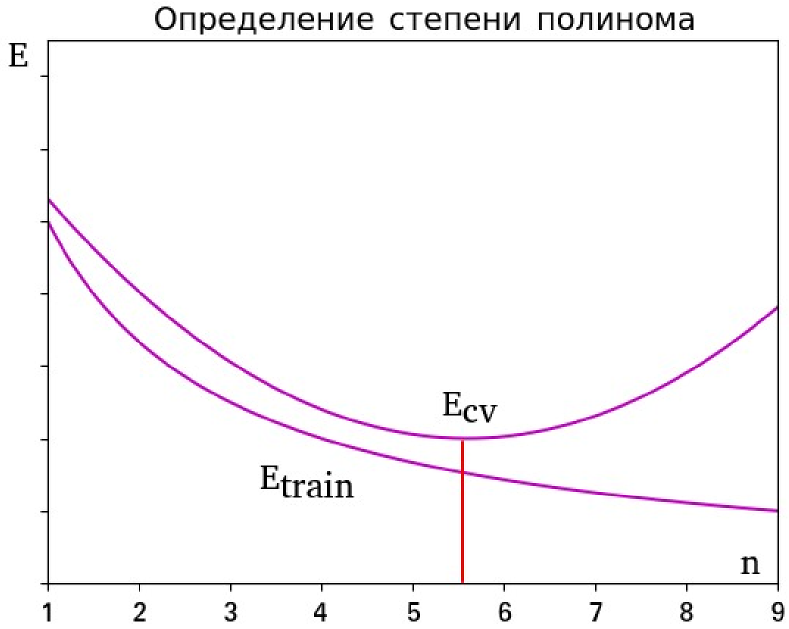
  $A_1(x) = \Theta_0 + \Theta_1 * x$
  $A_2(x) = \Theta_0 + \Theta_1 * x + \Theta_2 * x^2$
  $A_3(x) = \Theta_0 + \Theta_1 * x + \Theta_2 * x^2 + \Theta_3 * x^3$
  ...
  $A_9(x) = \Theta_0 + \Theta_1 * x + \ldots + \Theta_9 * x^9$

  По обучающей выборке находим Θ_i, минимизируя E_{train}.

  Для каждого варианта вычисляем $E_{cv}^{(i)}$.

  В качестве итогового выбираем $A_j(x)$ с наименьшей E_{cv}.

  Оценкой качества выбранного алгоритма является его E_{test}.



Недообучение и переобучение

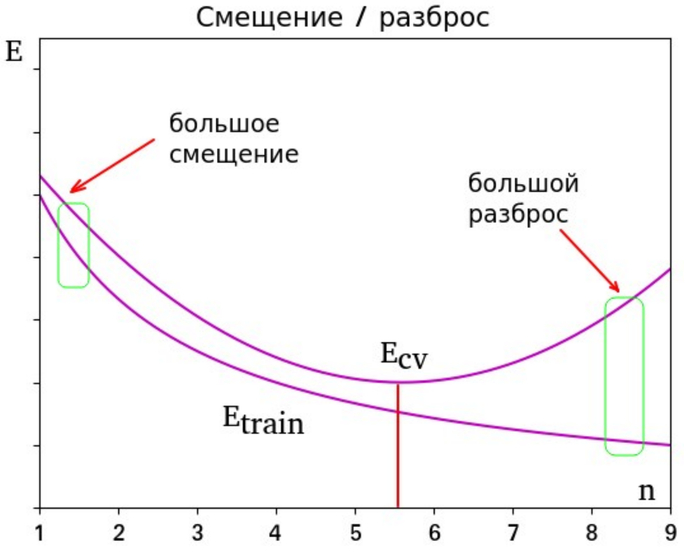
Недообучение (большое **смещение**):

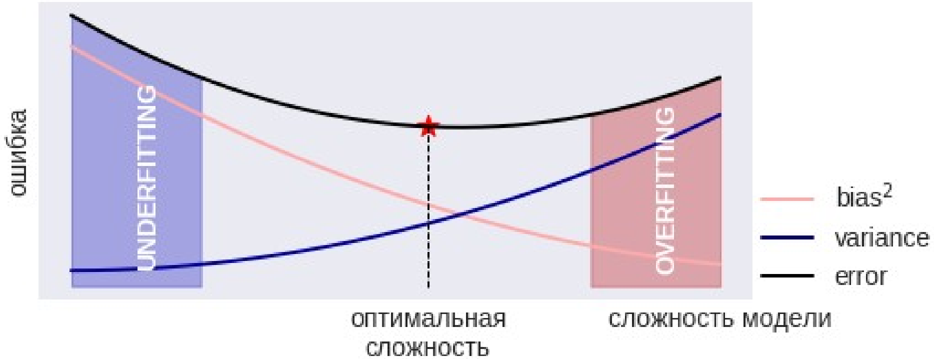
- Etrain – большая
- Ecv ≈ Etrain

Переобучение (большой **разброс**):

- Etrain – маленькая
- Ecv >> Etrain

Сложность модели алгоритмов – способность семейства алгоритмов настраиваться на выборки (разнообразие семейства алгоритмов).

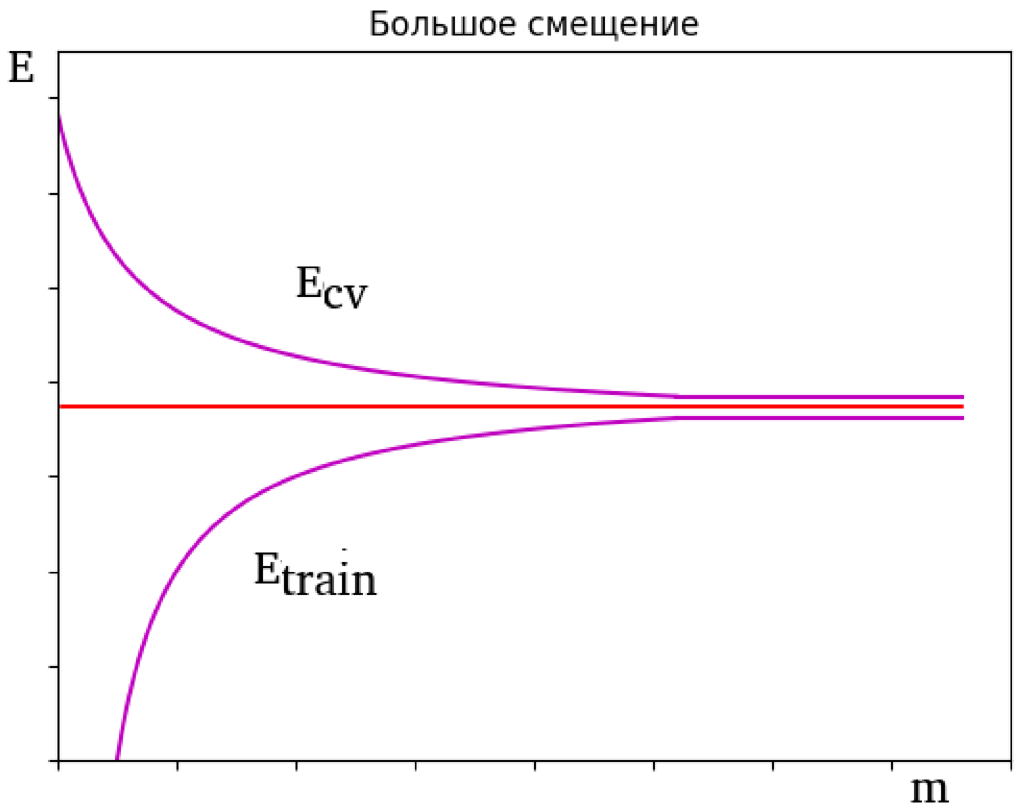




Большое смещение

Особенности:

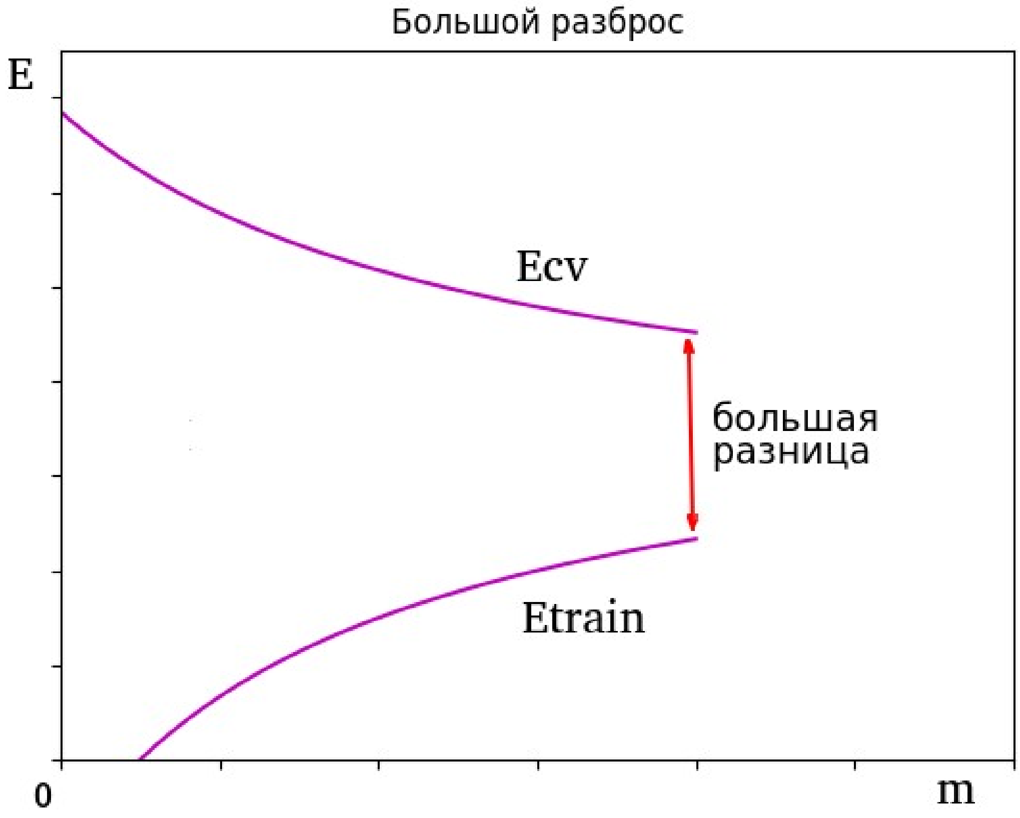
- увеличение обучающей выборки не улучшит качество алгоритма;
- желательно увеличить сложность алгоритма (добавить параметров);
- желательно уменьшить коэффициенты регуляризации алгоритма.



Большой разброс

Особенности:

- желательно увеличить размер обучающей выборки;
- желательно уменьшить сложность алгоритма (отбросить некоторые параметры);
- желательно увеличить коэффициены регуляризации алгоритма.

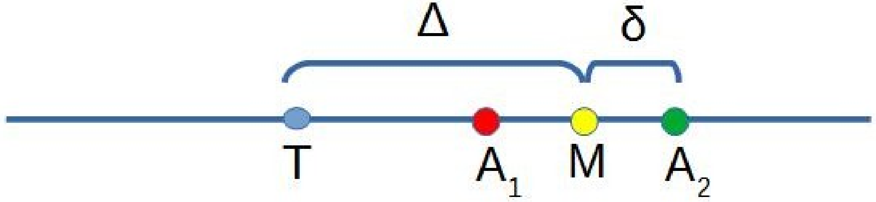


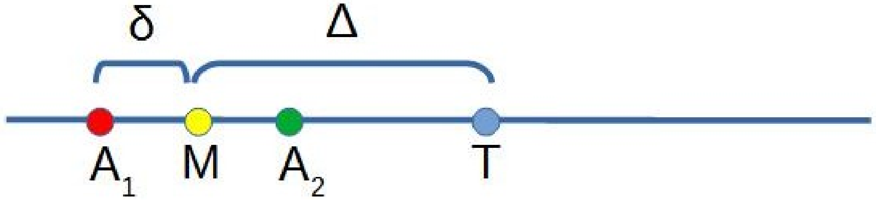
Усреднение результатов

A_1 и A_2 – два алгоритма для решения одной задачи.

Ошибки алгоритмов A_1 и A_2 примерно равны,
а результаты существенно различаются.

Усреднение результатов A_1 и A_2 в среднем даст меньшую ошибку при решении задачи.





$E_1^{(1)} = (\Delta - \delta)^2$ $\quad E_1^{(2)} = (\Delta + \delta)^2$
$E_2^{(1)} = (\Delta + \delta)^2$ $\quad E_2^{(2)} = (\Delta - \delta)^2$
$E_M^{(1)} = \Delta^2$ $\quad E_M^{(2)} = \Delta^2$

$E_1^{(ср)} = 2*\Delta^2 + 2*\delta^2$
$E_2^{(ср)} = 2*\Delta^2 + 2*\delta^2$
$E_M^{(ср)} = 2*\Delta^2$

$A_1 \sim (T, \sigma^2)$, $A_2 \sim (T, \sigma^2)$
$E(0.5*A_1+0.5*A_2) = 0.5*T+0.5*T = T$
$D(0.5*A_1+0.5*A_2) = 0.25*\sigma^2+0.25*\sigma^2 = 0.5*\sigma^2$
т.к. A_1 и A_2 - независимы

Получение различных алгоритмов

- Разные минимумы одной и той же модели.
- Использование других подходов к решению задачи (деревья решений, SVM …).
- Различные методы регуляризации весов.
- Разное количество слоев в нейросети и скрытых узлов в слое.
- Разные функции активации в нейроне.

Метод прореживания

Dropout

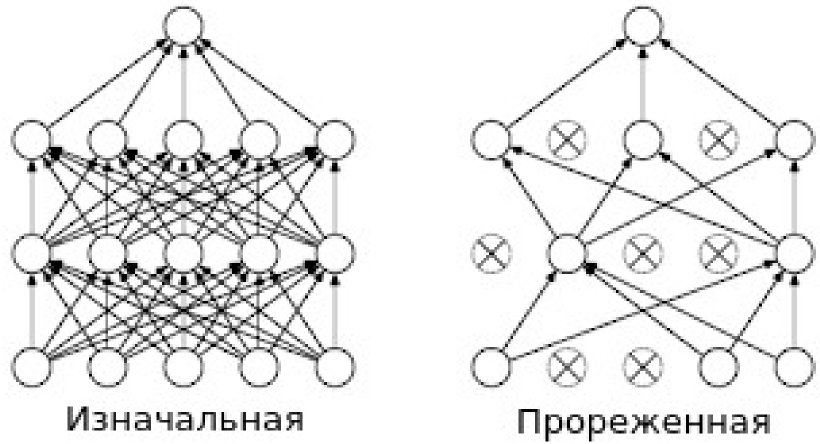
С вероятностью p нейрон временно исключается из сети (не учавствует в распространении сигнала и обучении).

Предотвращает совместную адаптацию нейронов.

Является аналогом усреднения работы 2^N сетей.

При использовании необходимо скорректировать веса модели.

В обратном варианте метода во время обучения корректируется функция активации.

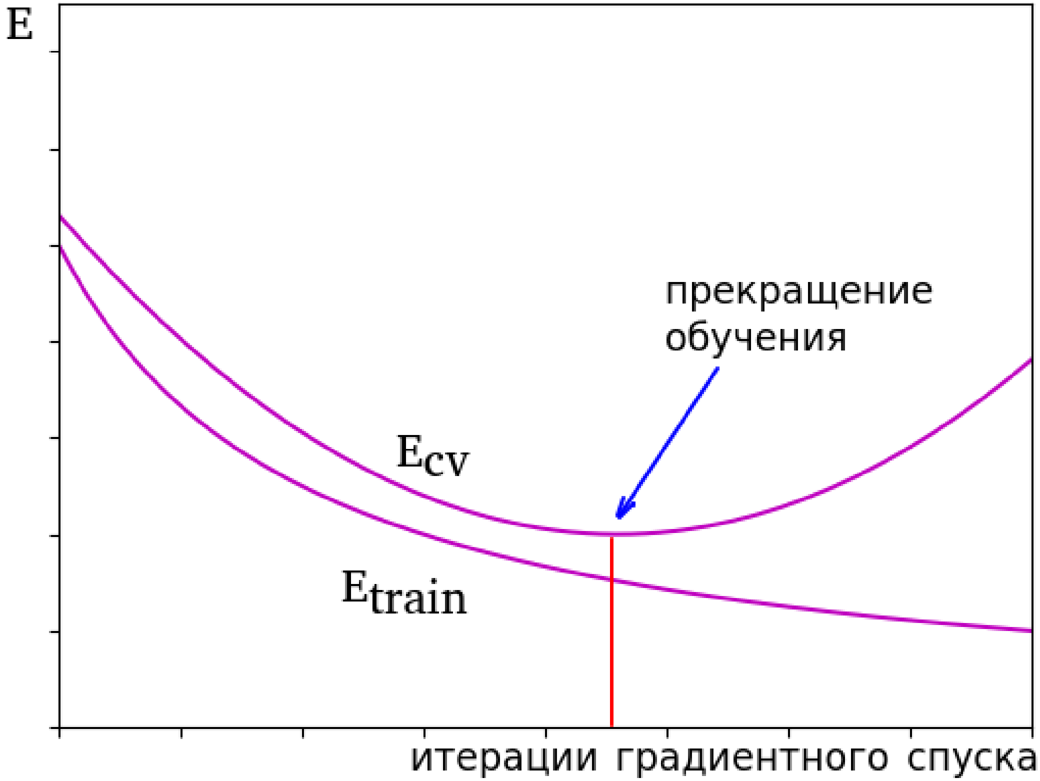


Ранняя остановка обучения

Если происходит переобучение сети, можно не доходить до минимума E_{train}.

Вместо этого:

1. Проинициализировать веса малыми значениями.
2. Проводить обучение, пока уменьшается Ecv.

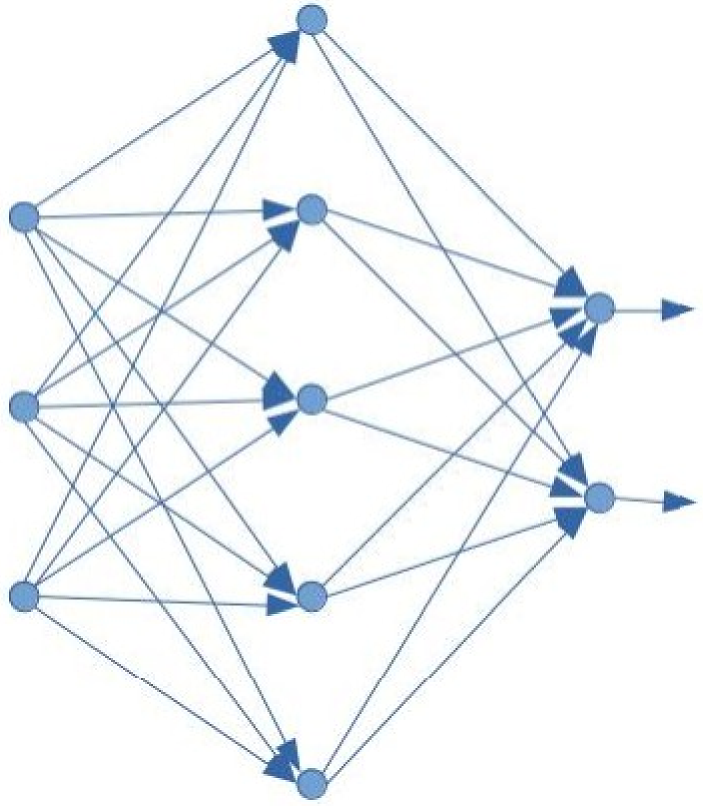


Ранняя остановка обучения

На начальном этапе веса инициализируются малыми значениями. Можно считать, что сеть является линейной. Количество весов = 3 * 2 = 6.

После обучения проявляется нелинейность функции активации. Количество весов = 25.

Ранняя остановка выбирает состояние с промежуточной сложностью.



Нормализация данных

Для многих алгоритмов обучения признаки объекта – безразмерные вещественные числа.

Если веса признаков одного порядка (при регулировании), то от признаков требуется то же условие (иначе если
$x^{(a)} \sim 1$, а $x^{(b)} \sim 10^7$, и $\Theta_a \sim \Theta_b$, то $x^{(b)}$ не участвует в обучении).

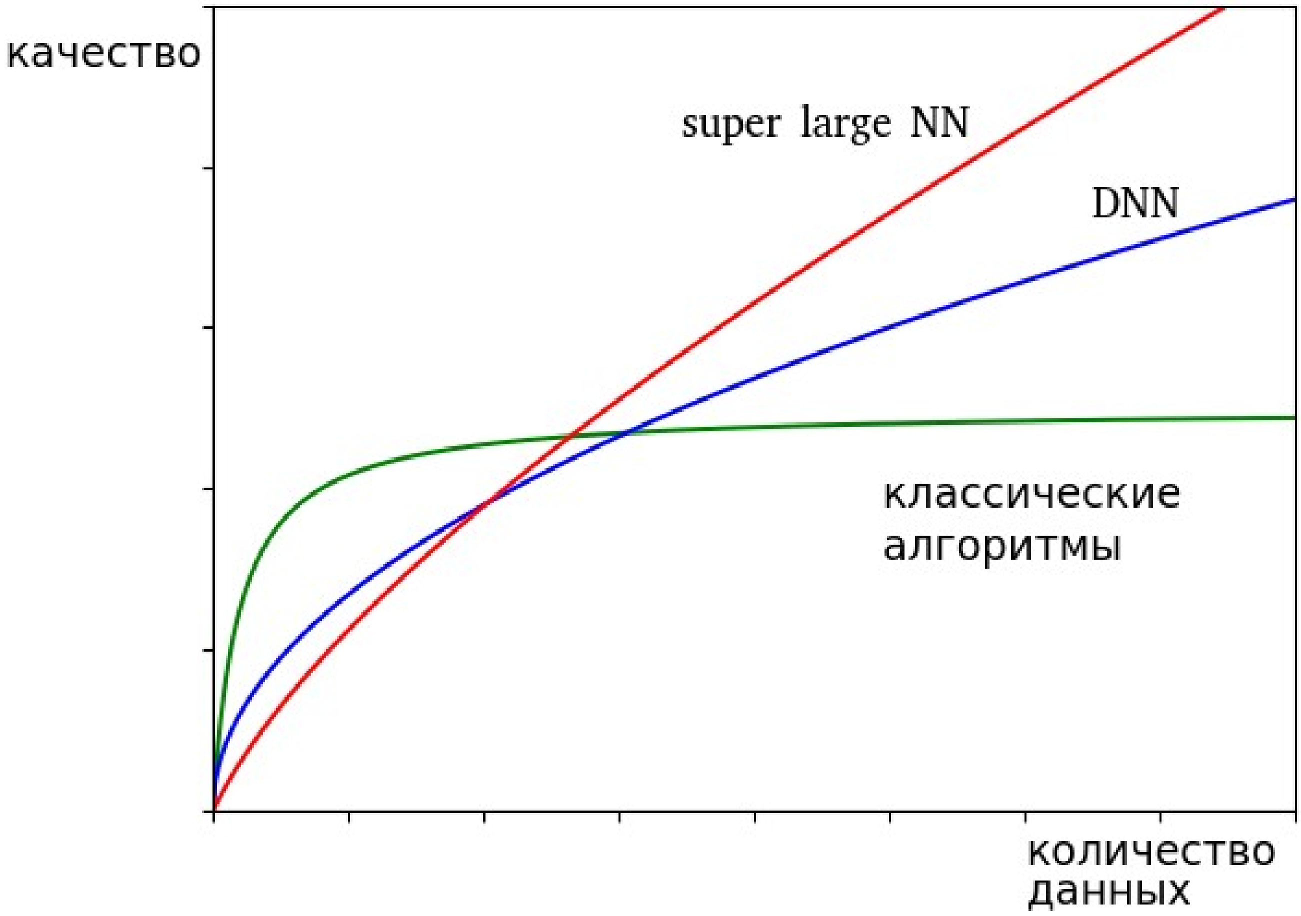
Перед обучением все данные нормализуют:

$x^{(i)*} = (x^{(i)} - x^{(i)}_{min}) / (x^{(i)}_{max} - x^{(i)}_{min})$ – **минимакс**

$x^{(i)*} = (x^{(i)} - \mu) / \sigma$ – **z-масштабирование** ($x^{(i)} \sim(\mu, \sigma^2)$)

Перед использованием алгоритма входные данные тоже должны быть нормализованы.

Развитие алгоритмов



“мертвые души”

tensorflow

https://sesc-infosec.github.io/