

Лекция 2

Линейные модели

Тимур Аминов. email: aminov.tv@phystech.edu , telegram: @sifonsoul

План

1. Gradient descent
2. Linear models
 - a. Linear regression
 - b. Logistic regression
 - i. Kernel trick
 - ii. Two-layer perceptron
 - c. Mini Batch Gradient Descent
 - d. Regularization
 - e. [Tensorflow's playground](#)
 - f. SVM
 - i. Separable case
 - ii. Kernel trick
 - iii. Non-separable case - smoothing
 - iv. SVM Regression

Полезные ссылки

- Линейная регрессия

<http://www.machinelearning.ru/wiki/index.php?title=%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%B0%D1%8F%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F%D1%80%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%29>

- Статья про SVM <https://habr.com/ru/post/428503/>

- Playground от гугла

<https://playground.tensorflow.org/#activation=tanh&batchSize=10&dataset=circle®Dataset=reg-plane&learningRate=0.03®ularizationRate=0&noise=0&networkShape=4.2&seed=0.50213&showTestData=false&discretize=false&percTrainData=50&x=true&y=true&xTimesY=false&xSquared=false&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=classification&initZero=false&hideText=false>

Повторение

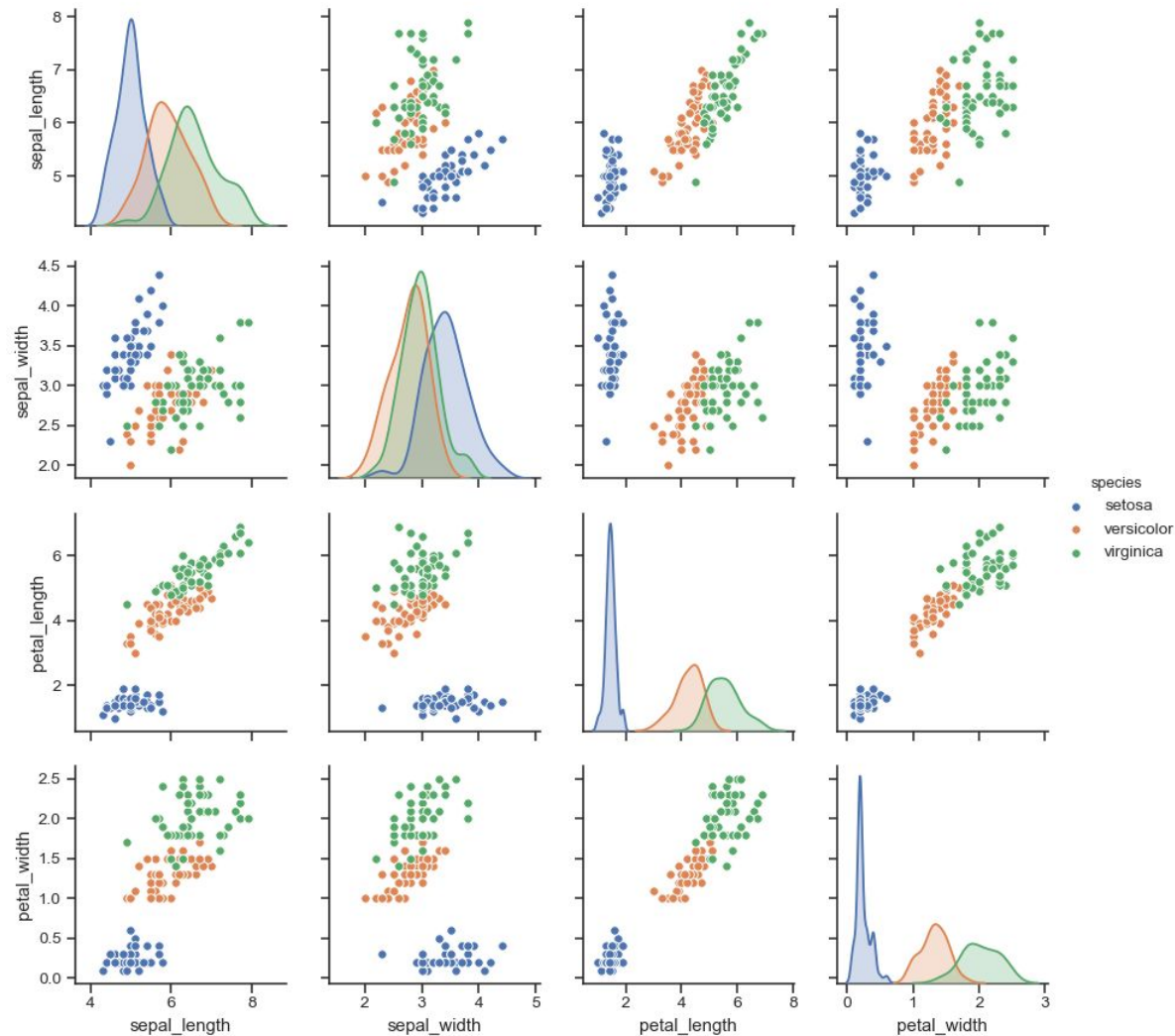
Данные

X y^* features

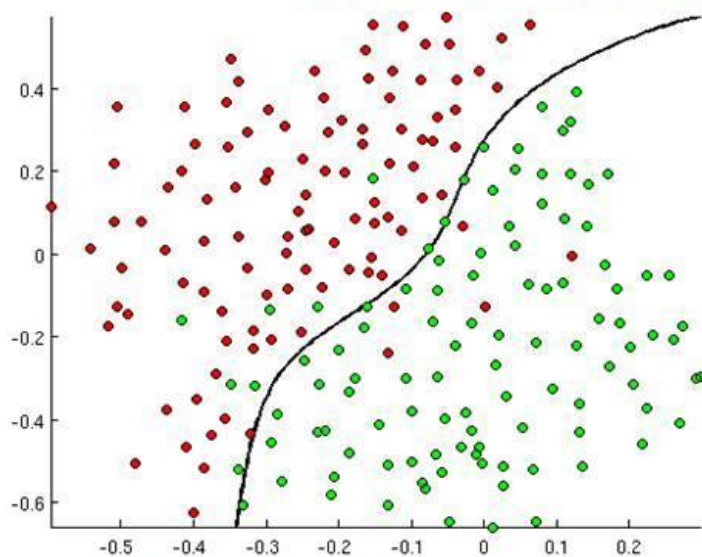
PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 310128
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450
6	0	3	Moran, Mr. James	male		0	0	330877
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909

Признаки

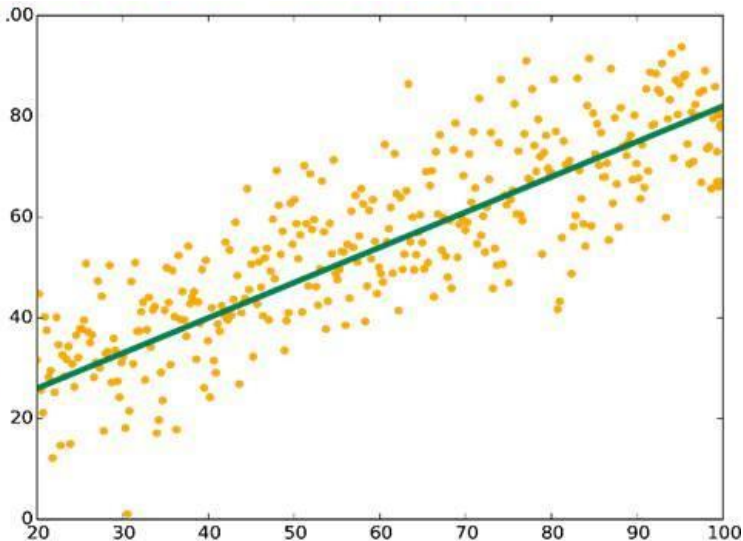
- Числовые
- Бинарные
- Категориальные



Классификация и регрессия

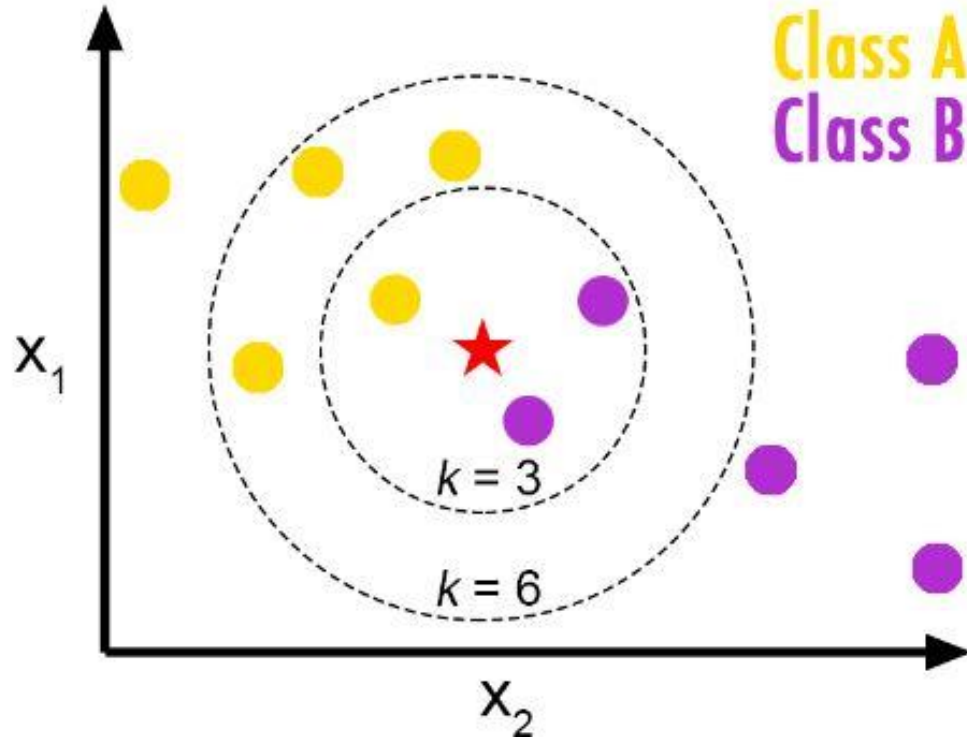


Classification

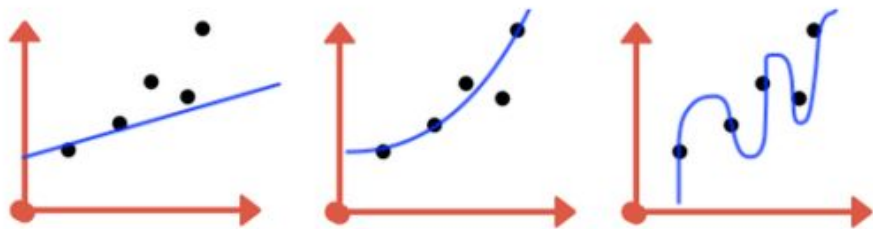


Regression

K Nearest Neighbors

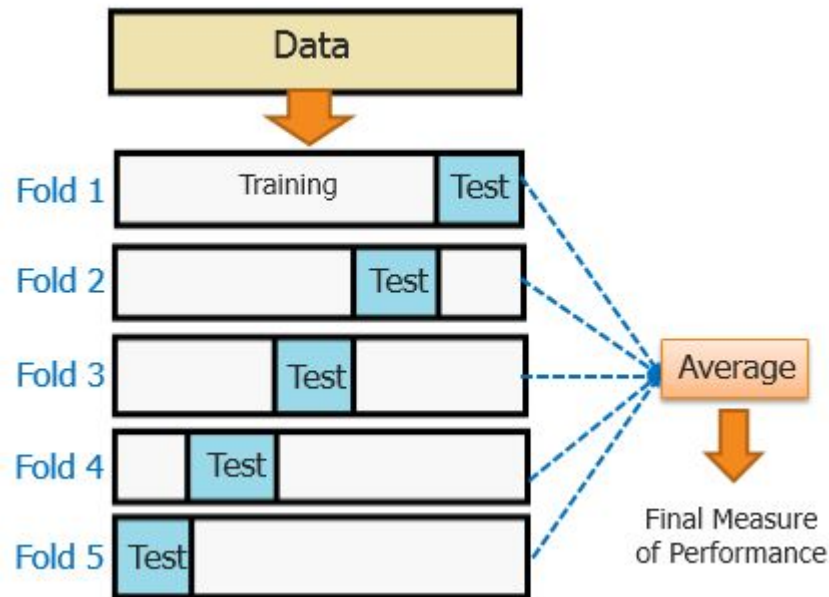


Cross-validation



Чтобы получить несмещенную оценку на всей выборке, используют кросс-валидацию.

Модель при этом обучается несколько раз.



Параметры модели

- Внешние параметры или гиперпараметры
 - Задаются перед обучением вручную
 - Например, число соседей в K Nearest Neighbors
 - Подбираются с помощью кросс-валидации
- Внутренние, обучаемые параметры
 - Задаются во время обучения самим алгоритмом обучения
 - Например, веса для линейной регрессии

Gradient Descent

Задача оптимизации

Нахождение минимума (максимума) некоторой функции - важная подзадача во многих методах машинного обучения, в том числе deep learning.

$$\min_{\theta_1, \theta_2, \dots, \theta_n} F(\theta_1, \theta_2, \dots, \theta_n)$$

Функция F зависит от многих параметров (иногда миллионы) и дифференцируема по каждому из них.

Нужен быстрый способ найти минимум. Хотя бы локальный.

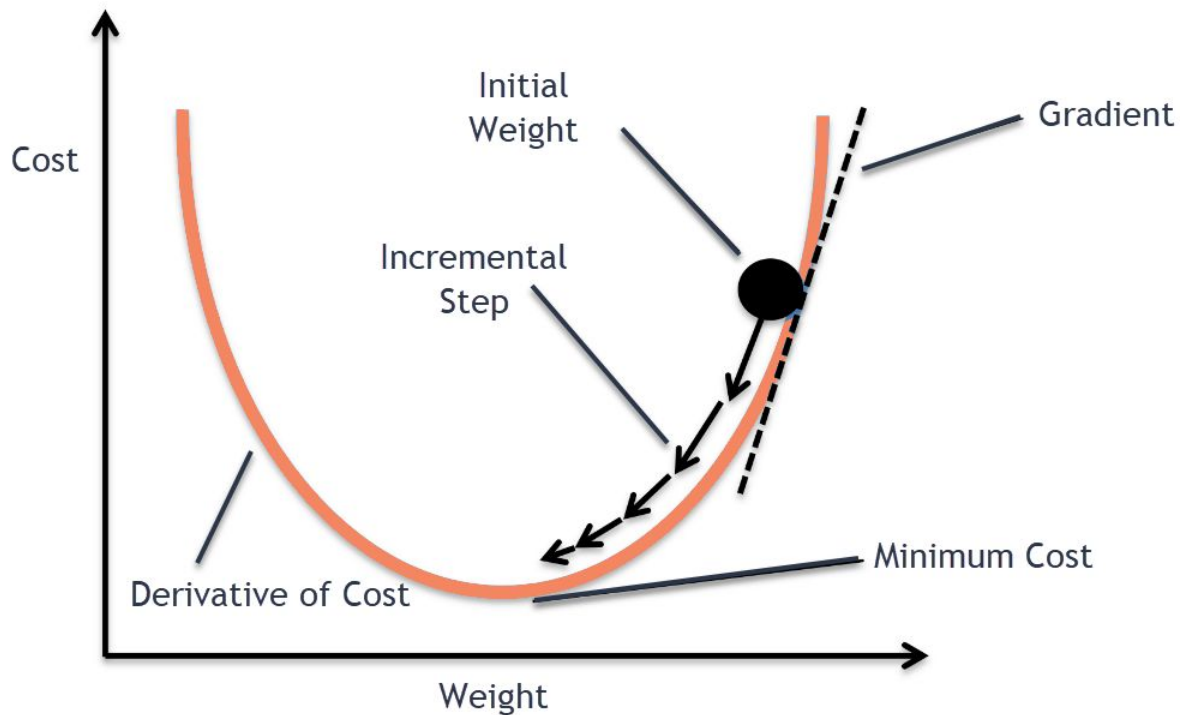
Градиентный спуск

$$\theta'_i = \theta_i - \alpha \frac{\delta F(\theta_i)}{\delta \theta_i}$$

α - learning rate

Векторная запись:

$$\vec{\theta}' = \vec{\theta} - \alpha \vec{\nabla}_{\theta} F$$



Interviewer: What's your biggest strength?

Me: I'm an expert in machine learning.

Interviewer: What's $9 + 10$?

Me: Its 3.

Interviewer: Not even close. It's 19.

Me: It's 16.

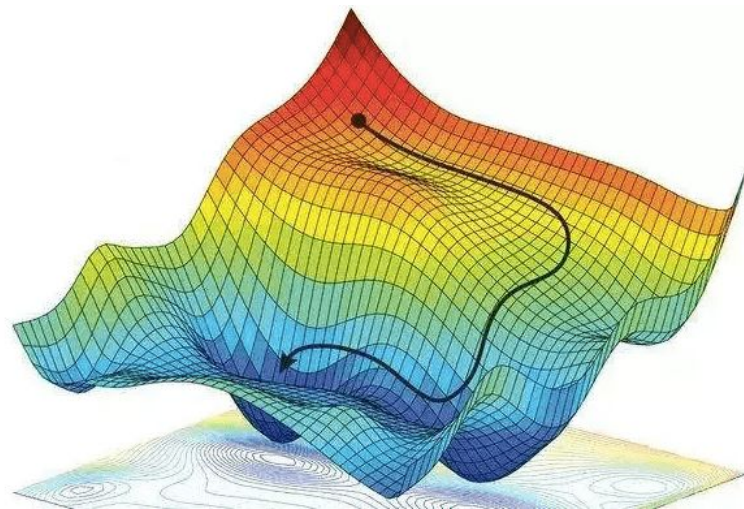
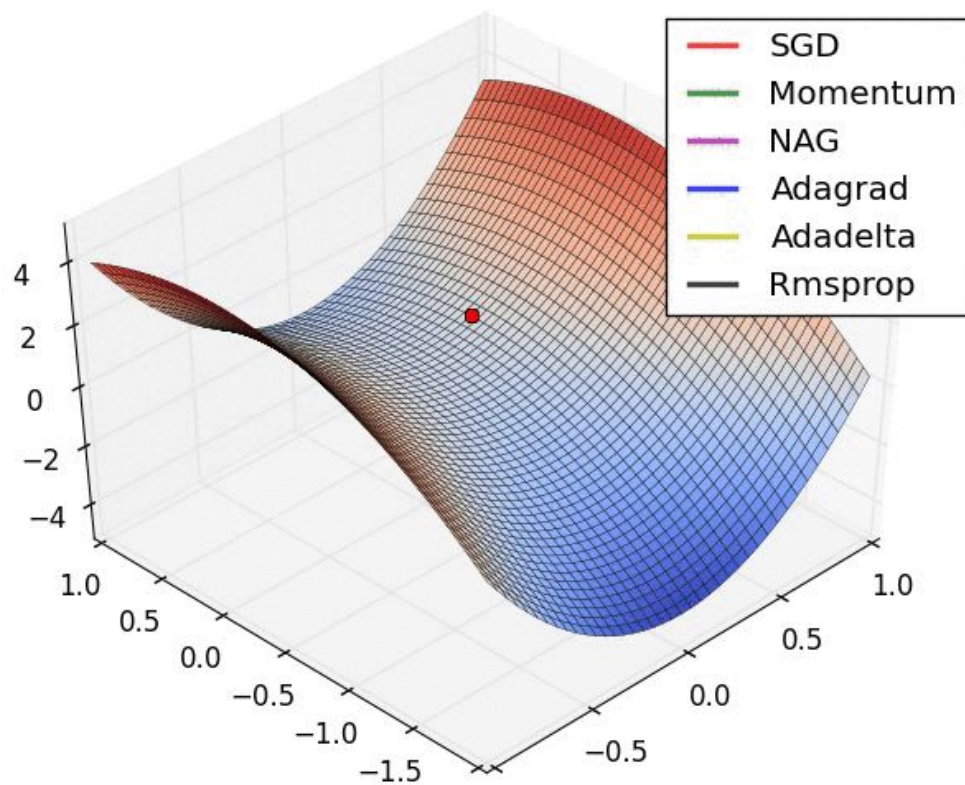
Interviewer: Wrong. Its still 19.

Me: It's 18.

Interviewer: No, it's 19.

Me: it's 19.

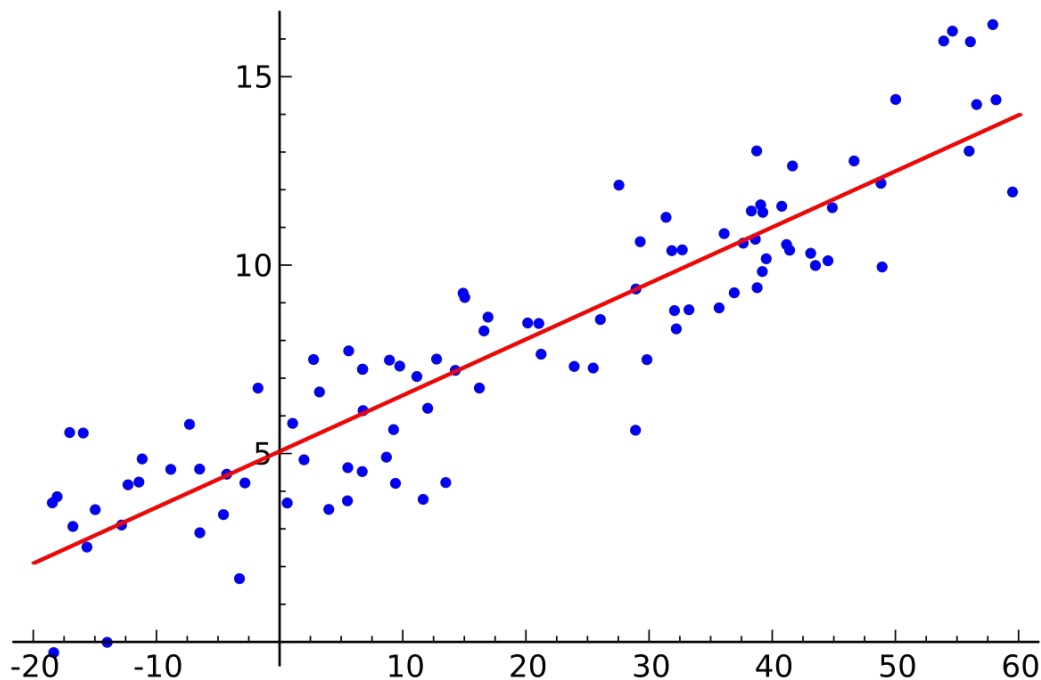
Interviewer: You're hired



Linear models

Linear regression

Линейная регрессия. Также известна как МНК - метод наименьших квадратов.



Linear regression

Линейная зависимость:

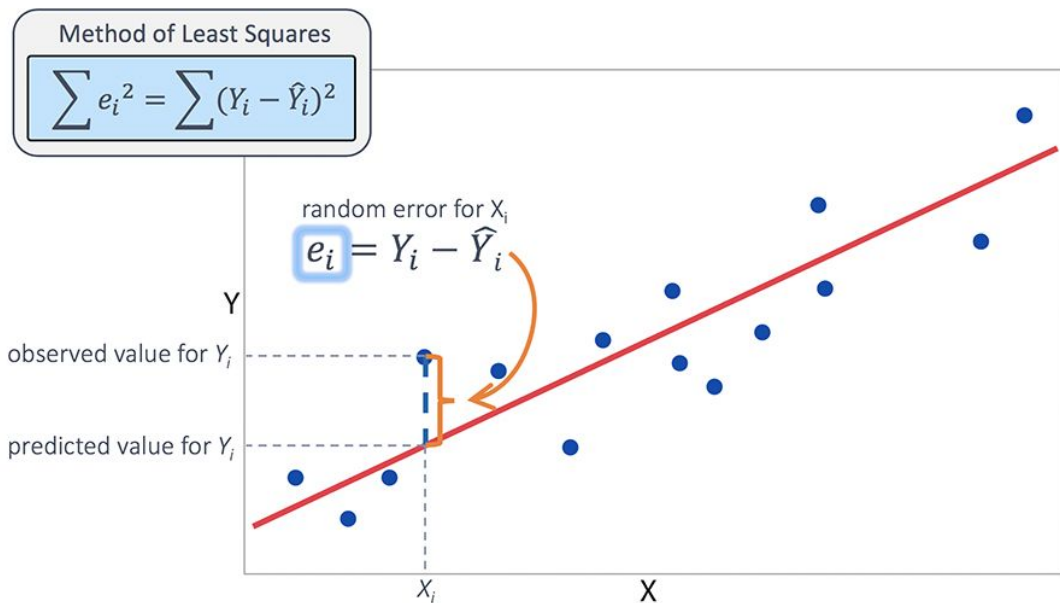
$$\hat{y}_i = wx_i + b$$

Где w и b нужно подобрать.

Среднеквадратичная ошибка:

$$\mathcal{L} = \sum_i (wx_i + b - y_i)^2$$

Будем называть это loss, или “функция потерь”



Точное решение

В случае многих переменных:

$$\hat{y}_i = W\vec{x}_i + b$$

$$\mathcal{L} = \sum_i (W\vec{x}_i + b - y_i)^2$$

Целевая переменная - все равно скаляр

Нахождение минимума сводится к решению системы линейных уравнений:

$$\frac{\delta \mathcal{L}}{\delta W_j} = \frac{\delta \mathcal{L}}{\delta b} = 0$$

Или:

$$\sum_i \vec{x}_i (W\vec{x}_i + b - y_i) = 0$$


Перепишем:

$$\sum_i \vec{x}_i (\vec{x}_i^T W - y_i) = 0$$

Еще перепишем:

$$X^T X W - X^T Y = 0$$

$$W = (X^T X)^{-1} X^T Y$$



Тогда решение: (штрихи опущены)

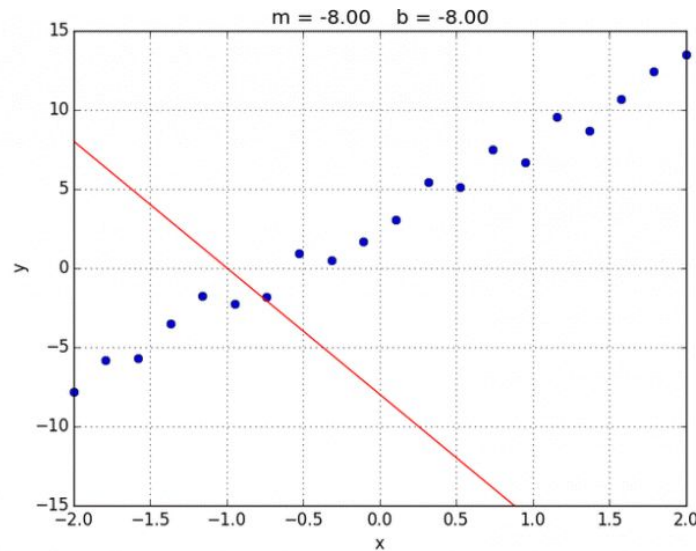
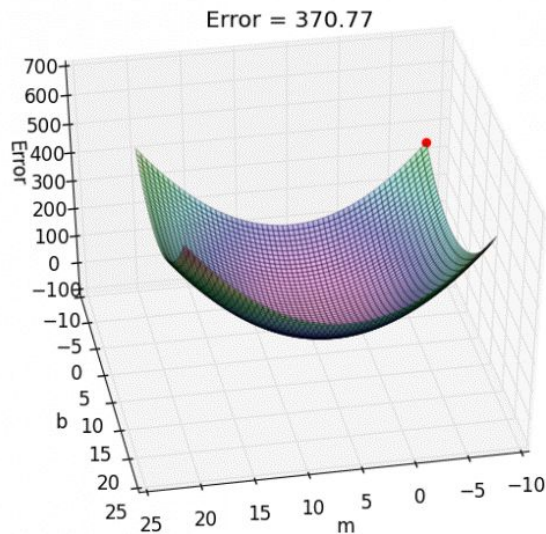
$$\mathcal{L} = \sum_i (W \vec{x}_i + b - y_i)^2$$

Градиентный спуск

Точное решение существует и единственно.

Однако при большом кол-ве параметров и данных искать его аналитически становится невыгодно.

Градиентный спуск часто оказывается быстрее.



начальные значения весов выбираются случайно

Другие loss'ы

Среднеквадратичная ошибка имеет свои минусы, и иногда не подходит для конкретной задачи

- MSE чувствительна к выбросам
- MAE не гладкая
- Huber их объединяет, но имеет внешний параметр

Не для каждой функции потерь есть аналитическое решение

Градиентный спуск работает всегда, когда функция хотя бы кусочно дифференцируема

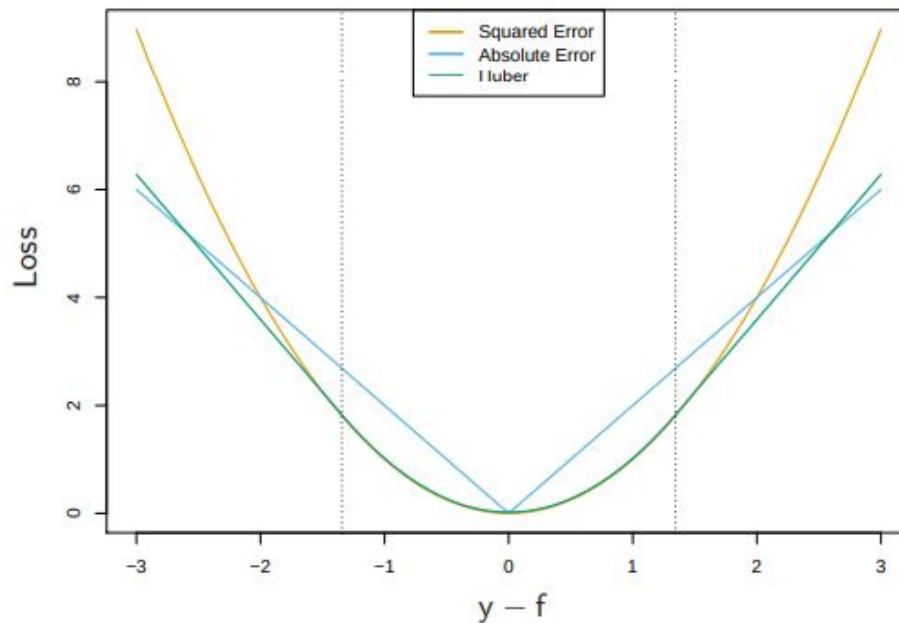


FIGURE 10.5. A comparison of three loss functions for regression, plotted as a function of the margin $y - f$. The Huber loss function combines the good properties of squared-error loss near zero and absolute error loss when $|y - f|$ is large.

Отличие loss от метрики

- loss используется для подбора параметров модели на тренировочном наборе
- метрика используется для оценки модели на тестовом наборе
- они могут быть как одинаковыми функциями, так и разными
- метрика не обязательно должна быть дифференцируема
- метрик может быть несколько, а loss у модели один

метрики для регрессии

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

$$\text{SMAPE} = \frac{1}{n} \sum_{i=1}^n \frac{2 \cdot |y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|}$$

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

Регуляризация

$$\mathcal{L}' = \mathcal{L} + \lambda \cdot ||w||_{L_2}$$

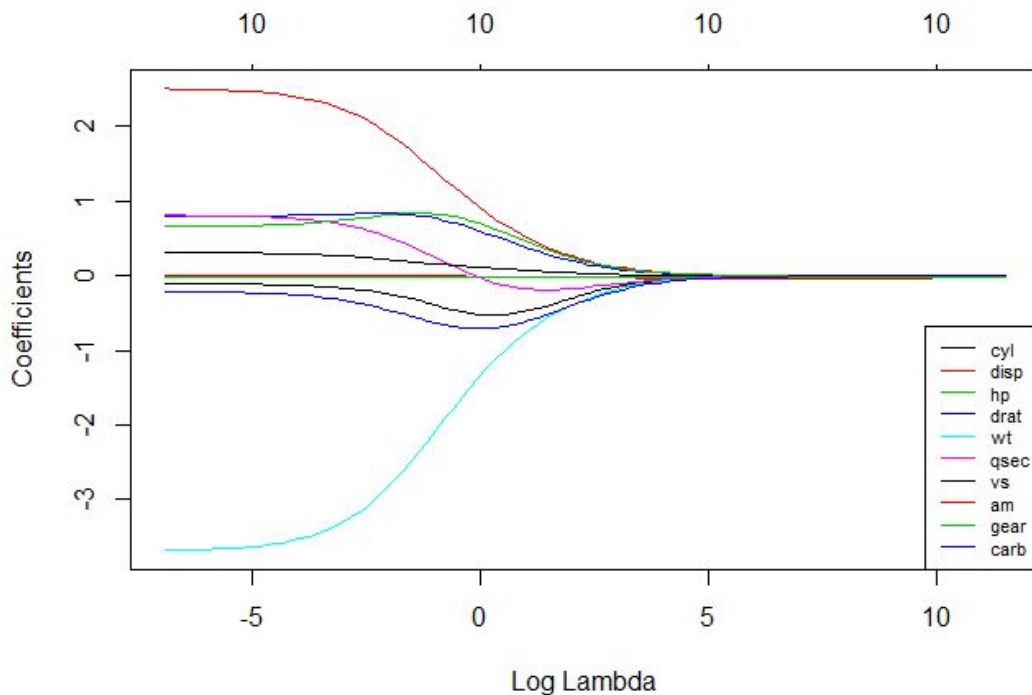
L2 regularization

Проблемы линейных моделей:

- Некоторые признаки линейно зависимы
- По некоторым признакам статистика представлена мало

Можно добавить веса к функции потерь, и тогда модель будет получать штраф за большие веса

В случае линейной регрессии такая модель называется **ridge regression**

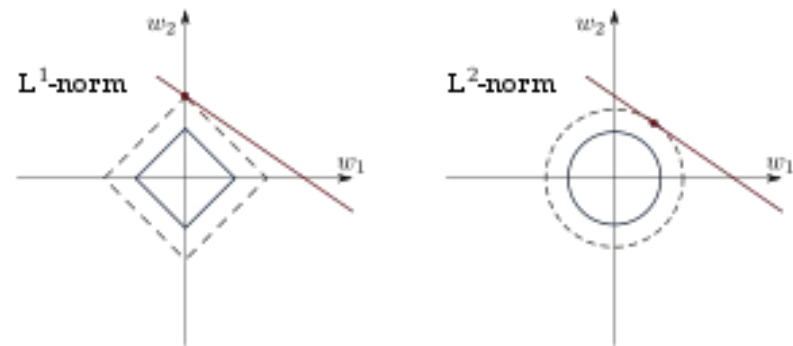


L1 regularization

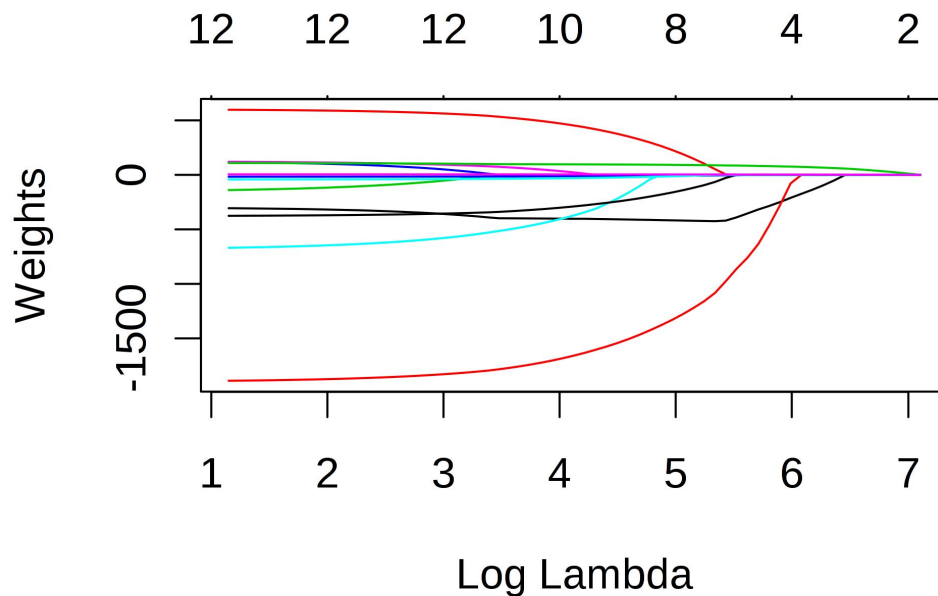
Норма L1 тоже применяется, и ведет себя немного по-другому.

Модель линейной регрессии с такой регуляризацией называется **lasso regression**

LASSO - least absolute shrinkage and selection operator

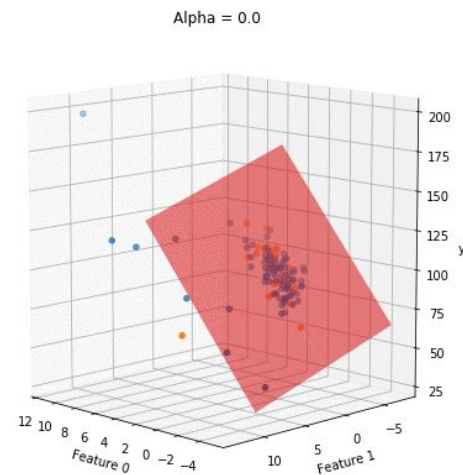
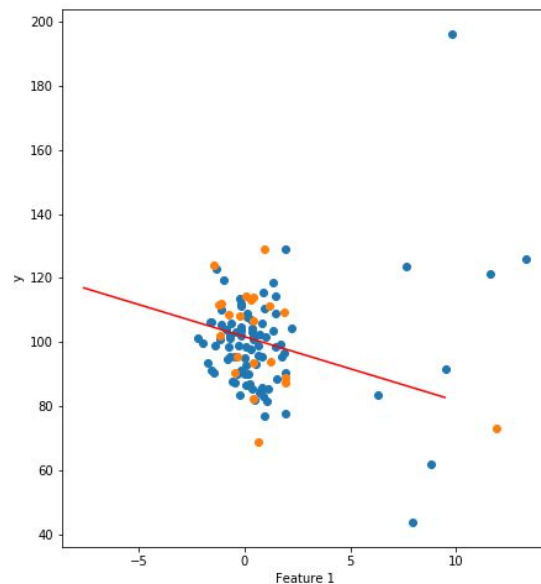
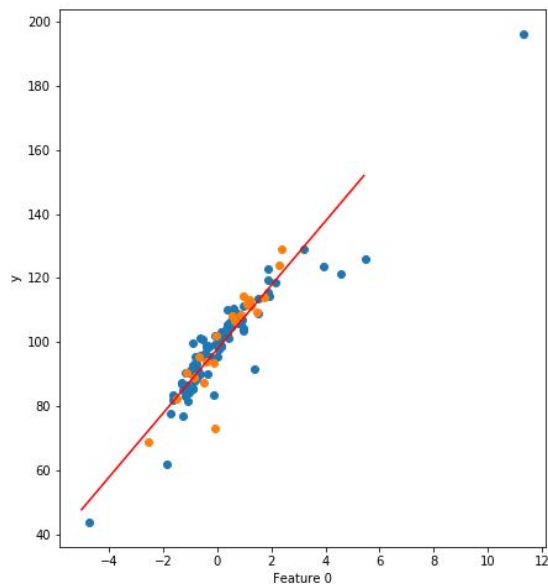


$$\mathcal{L}' = \mathcal{L} + \lambda \cdot ||w||_{L_1}$$



Регуляризация: как выглядит

$$\hat{y}_i = W \vec{x}_i + b$$



Параметр Feature 1 содержит выбросы, которые ведут к переобучению. При увеличении α ($=\lambda$), вес данного параметра стремится к нулю. Заметьте, что среднее значение y сохраняется

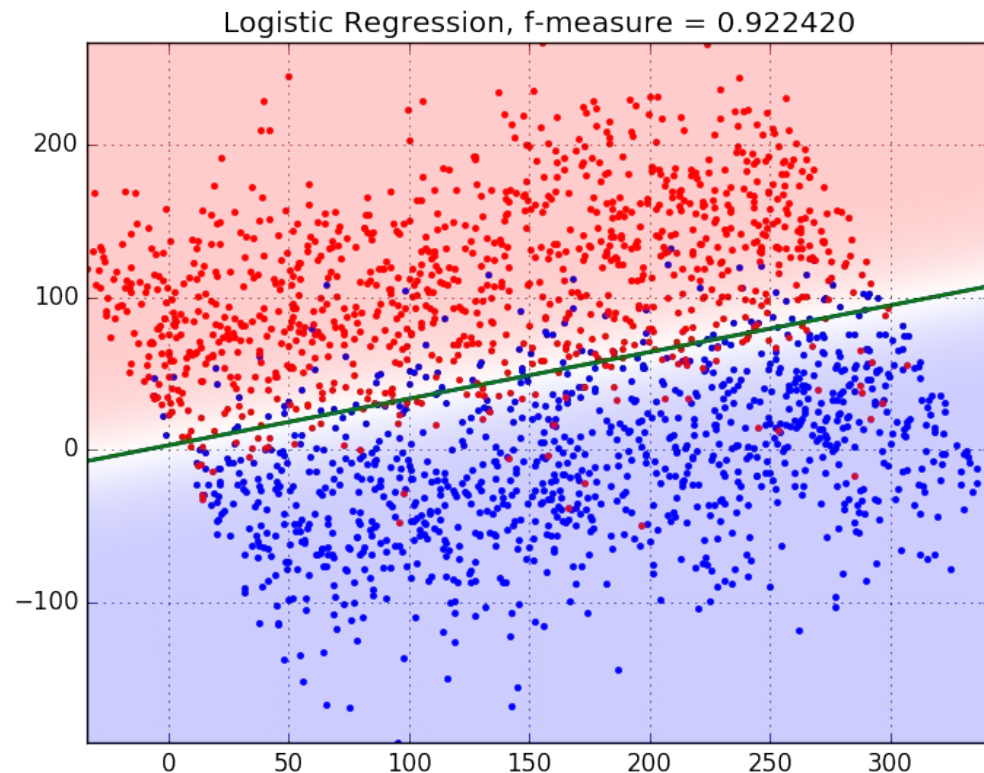
Логистическая регрессия

Logistic regression

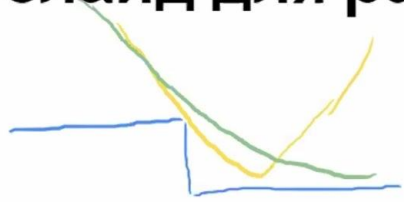
Логистическая регрессия - линейный метод **классификации**

Название исторически сложилось, т.к. этот метод предсказывает вероятность

Также называют линейным классификатором



слайд для рассуждений



Хотим $L = \sum_i [y_i \neq \hat{y}_i]$ $\mathcal{C} = \{-1, 1\}$

$$\text{Margin} = y_i f(x_i) = y_i \cdot x_i^T w$$

$$L_{\log} = (y_i - x_i^T w)^2 = \frac{(y_i^2 - y_i x_i^T w)^2}{y_i^2} = (1 - M_i)^2$$

$$P_+ = P(y=1|x_i) \in [0, 1]$$

$$\frac{P_+}{P_-} = \frac{P_+}{1-P_+} \in [0, +\infty]$$

$$\log \frac{P_+}{1-P_+} \in \mathbb{R}$$

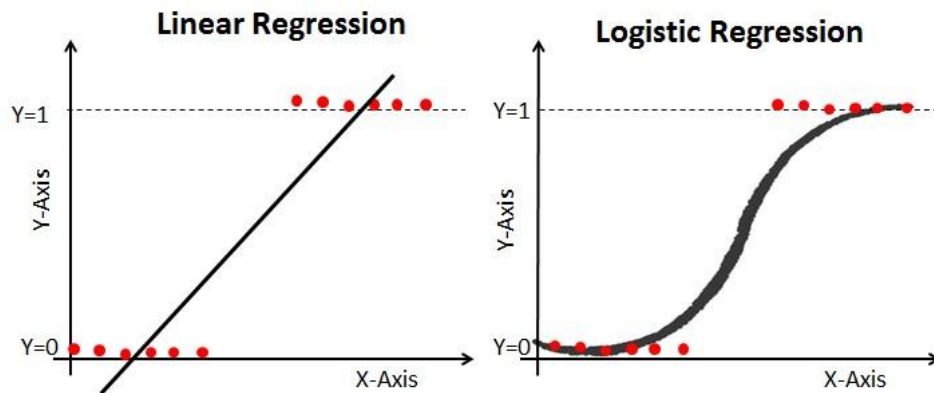
$$\frac{P_+}{1-P_+} = \exp[x^T w]$$

$$P_+ = \frac{1}{1 + \exp[x^T w]} = \sigma(x^T w)$$

1) симметрия: $1 - \sigma(x) = \sigma(-x)$

2) производная: $\sigma'(x) = \sigma(x)(1 - \sigma(x))$

Логистическая функция



$$y = \frac{1}{1 + e^{-(W\vec{x}+b)}}$$

Кросс-энтропия

В качестве функции потерь
используется т.н. кросс-энтропия

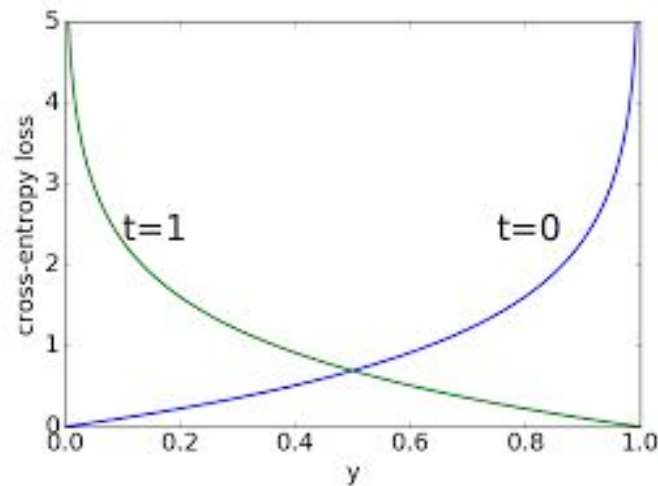
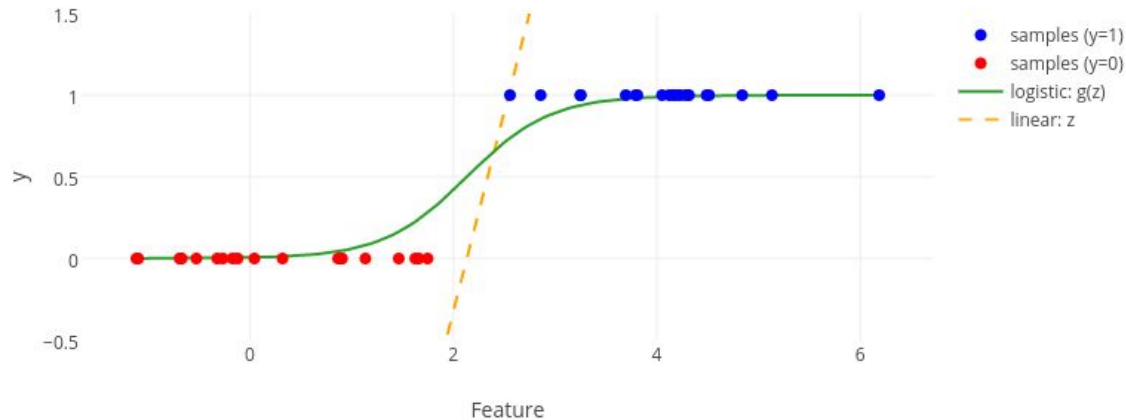
$$H(\hat{p}, p) = - \sum_i p_i \log(\hat{p}_i)$$

Это мера расстояния между двумя
распределениями - она минимальна,
когда одно распределение
приближается к другому

Например, лосс для одного объекта в
случае бинарной классификации:

$$\mathcal{L} = -p \cdot \log(\hat{p}) - (1 - p) \cdot \log(1 - \hat{p})$$

Logistic Regression: 1 Feature



А что если классов больше одного?

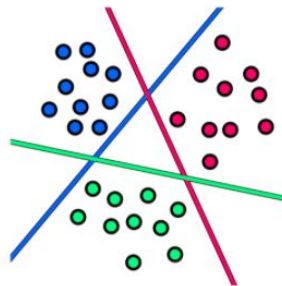
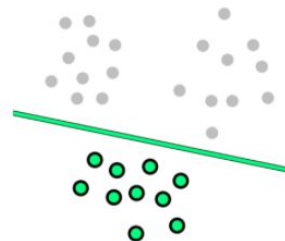
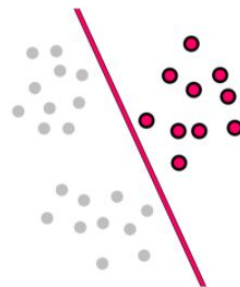
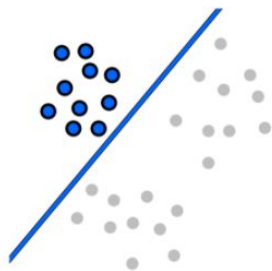
имеем две стратегии

1. OvR (One VS Rest)
2. OvO (One VS One)

OvR

имеем N
классификаторов и
хотим усреднить их
предсказания

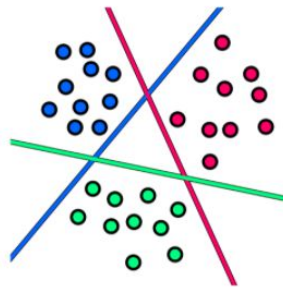
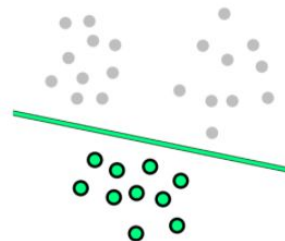
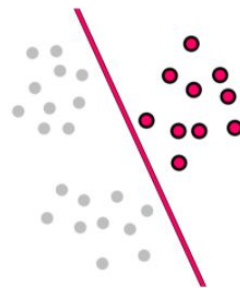
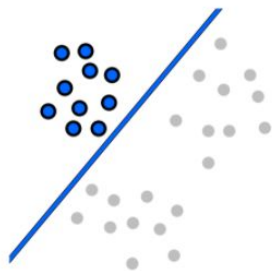
обучаемся на всех
примерах



OvO

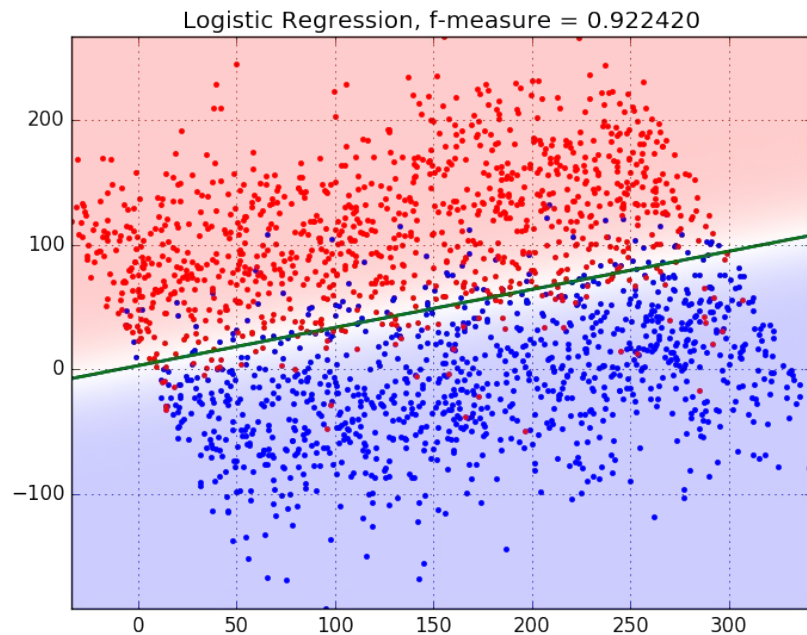
имеем $N(N-1)/2$
классификаторов и
хотим усреднить их
предсказания

обучаемся на части
датасета

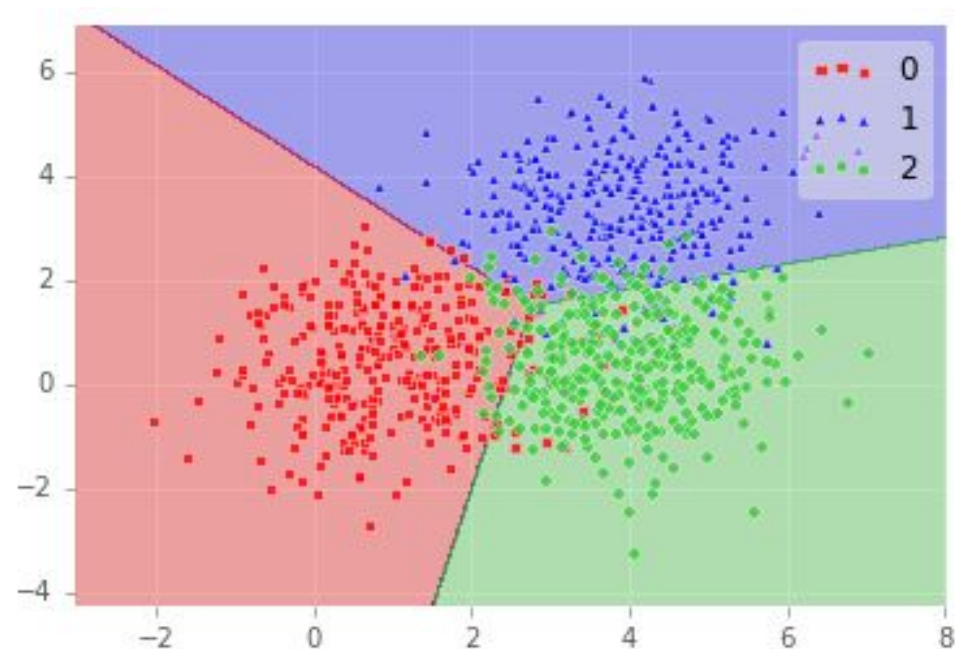


Визуализация в 2d

$$y = \text{sigmoid}(W\vec{x})$$



$$\vec{y} = \text{softmax}(W\vec{x})$$



Софтмакс

$$\vec{y} = W\vec{x}$$

В логистической регрессии предсказывается вероятность. Поэтому используется функция, которая превращает logit'ы в вероятностное распределение:

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

$$\vec{y} = \text{softmax}(W\vec{x})$$

Kernel trick

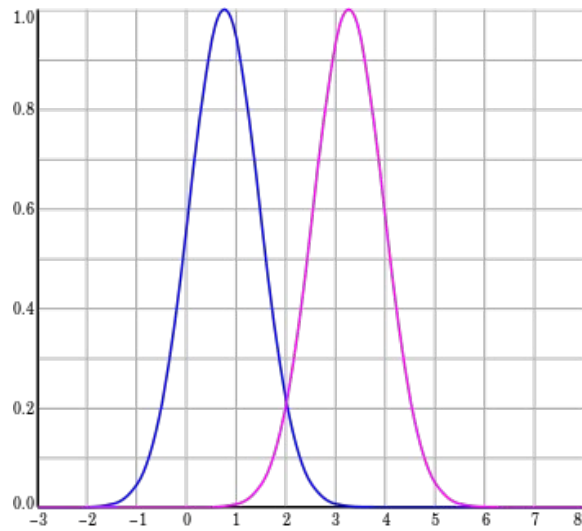
Есть простой способ сделать линейную модель нелинейной

Добавить новые признаки!

Новый признак - **нелинейная функция** от одного или нескольких базовых признаков

Например:

- Полиномы n -й степени от базовых признаков
- Радиально базисные функции



Kernel trick

Fig.3

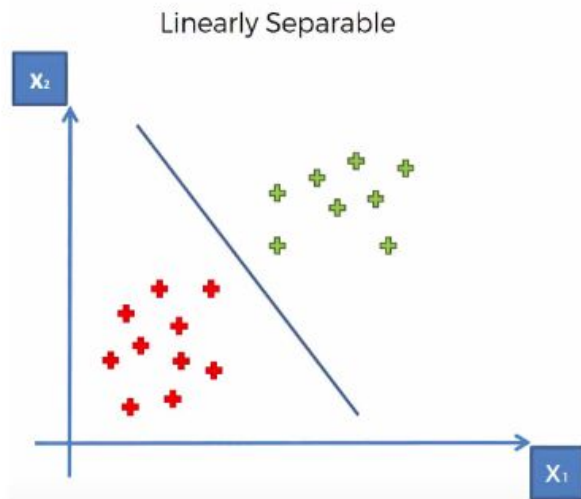
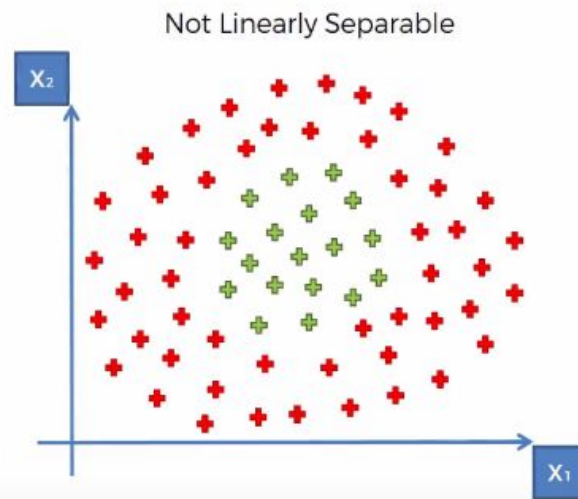
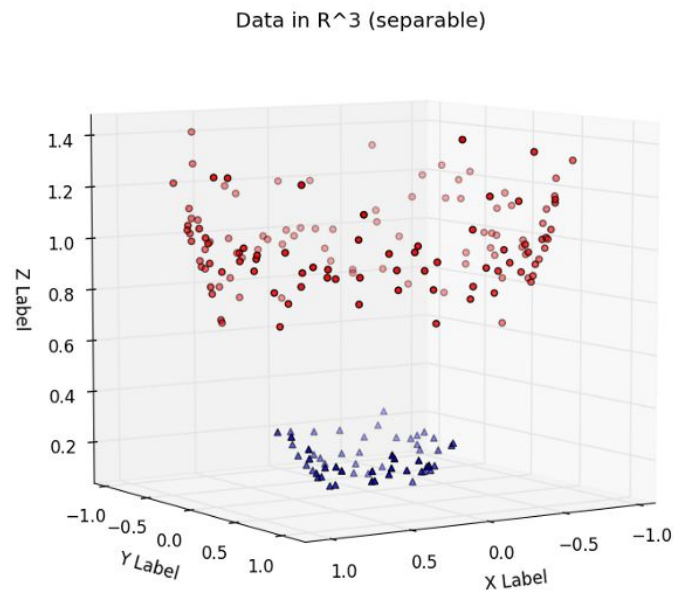
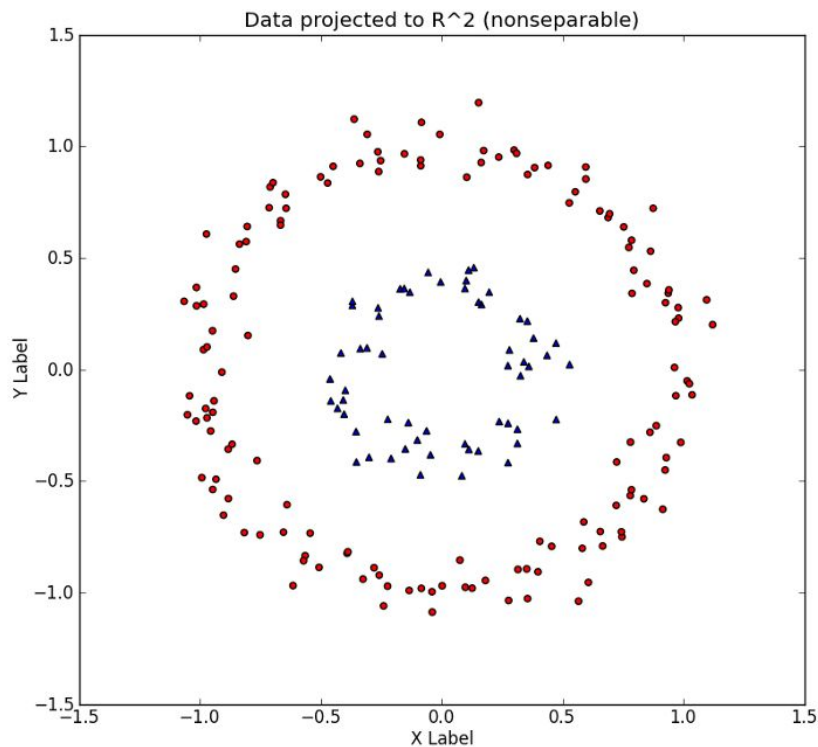


Fig.4

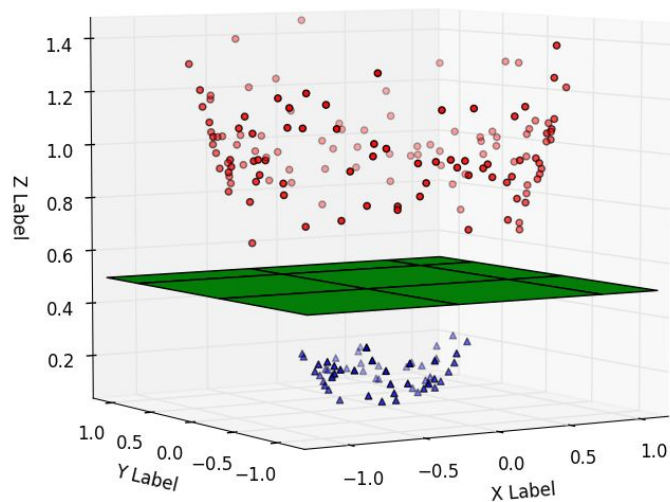


Kernel trick

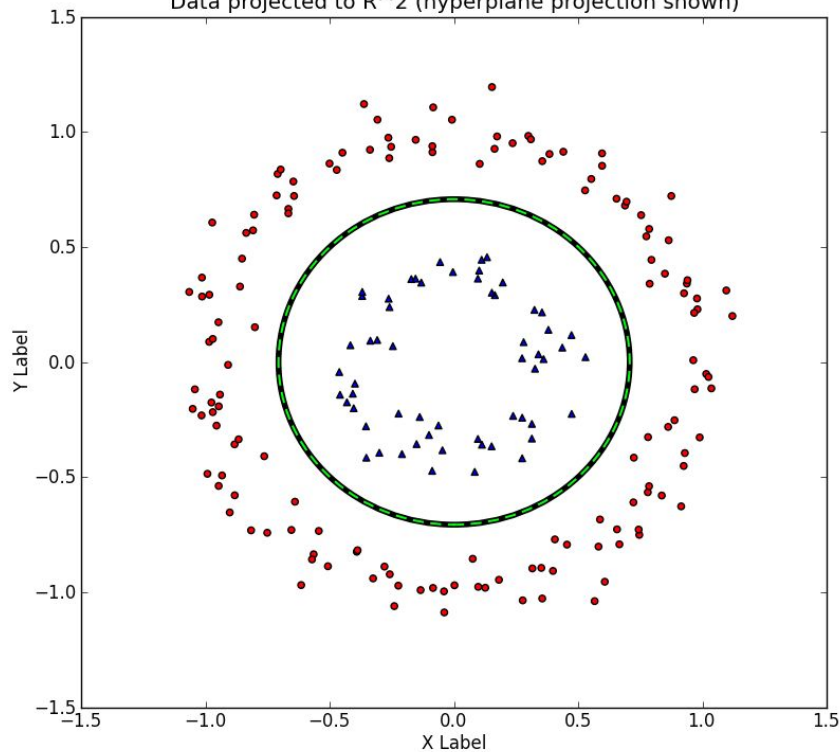


Kernel trick

Data in R^3 (separable w/ hyperplane)



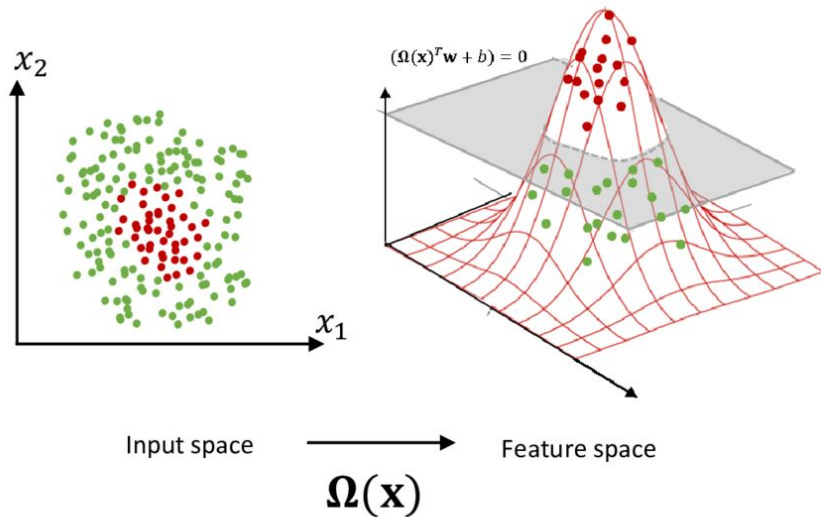
Data projected to R^2 (hyperplane projection shown)



Kernel trick

Как именно выбирается центр RBF - зависит от реализации. Например, в центрах кластеров (кластеризацию еще не проходили)

Kernel trick реализован в стандартных библиотеках



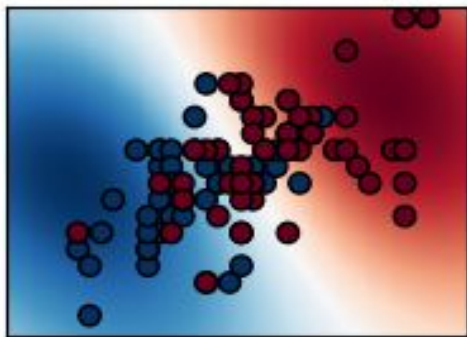
Параметр ядра

У ядра есть один параметр - γ . Он задает размер ядра.

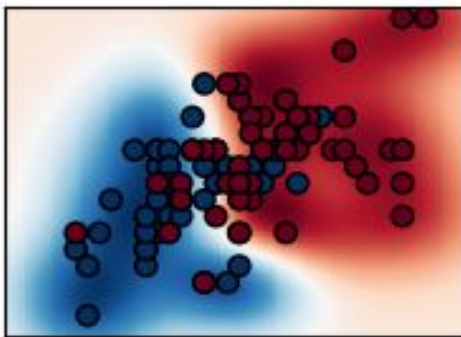
Маленькое γ - большой радиус ядра. Большое γ - малое ядро.

γ , как и другие гиперпараметры, влияет на сложность модели

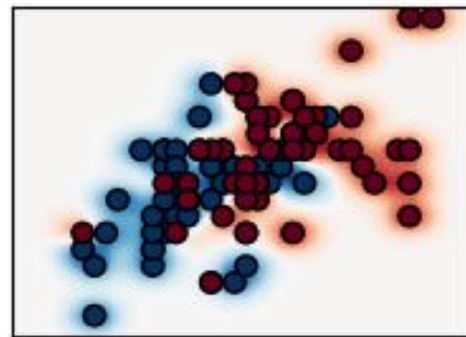
$\gamma=10^{-1}$, $C=10^0$



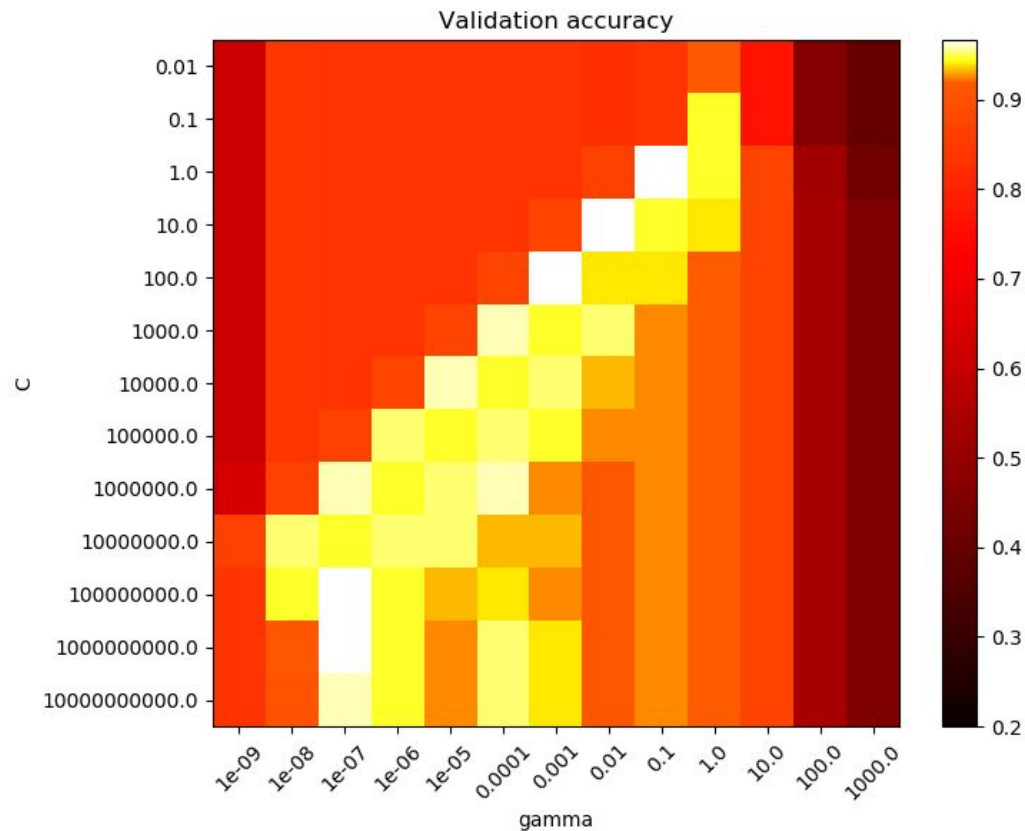
$\gamma=10^0$, $C=10^0$



$\gamma=10^1$, $C=10^0$



Параметр ядра

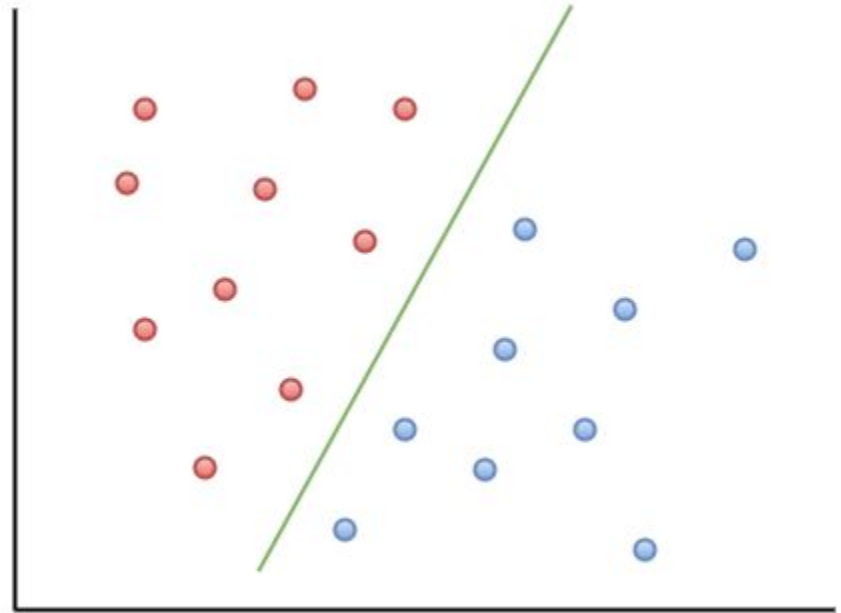


SVM

Support vector machine classifier

Предположим, что точки в пространстве можно разделить некоторой гиперплоскостью на два класса

Вопрос: как выбрать расположение гиперплоскости, чтобы максимально точно классифицировать новые точки?

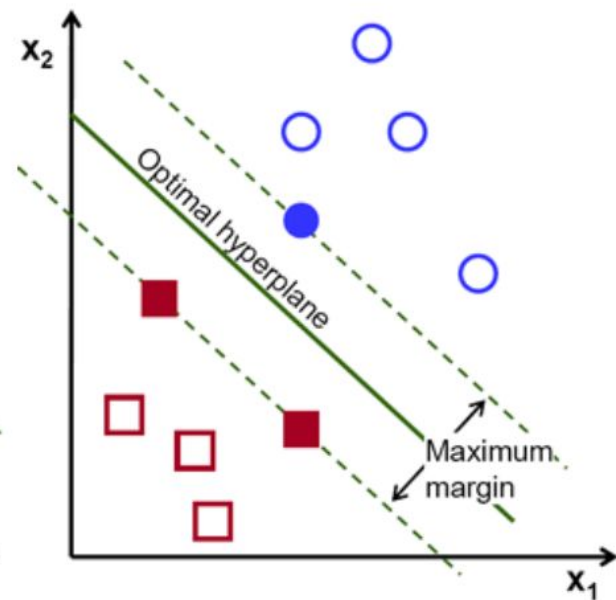
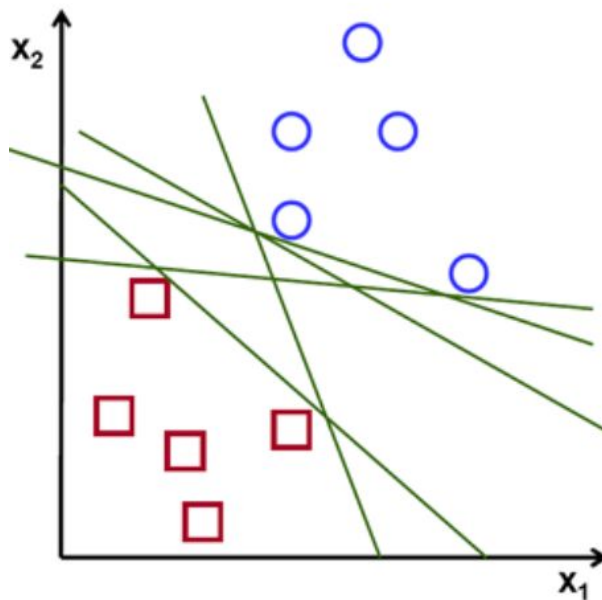


Maximum margin

Идея метода SVM: выбрать плоскость таким образом, чтобы **максимизировать зазор** между классами

Т.о. положение плоскости будет задаваться всего несколькими точками

Эти точки называются **опорными векторами**



Hyperplane

Уравнение плоскости:

$$\vec{w}\vec{x} + b = 0$$

Уравнение плоскости, смещенной в одну сторону:

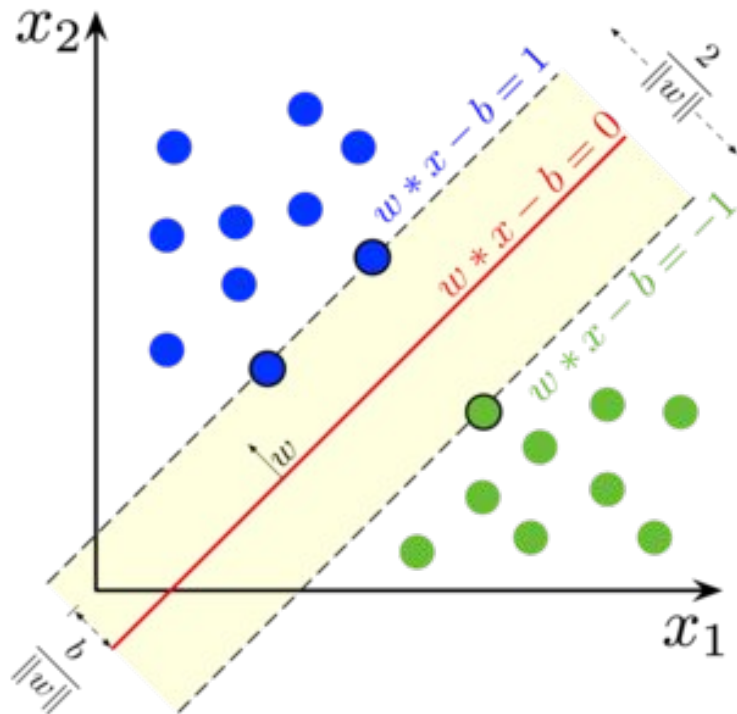
$$\vec{w}\vec{x} + b = 1$$

Уравнение плоскости, смещенной в другую сторону:

$$\vec{w}\vec{x} + b = -1$$

Величина смещения:

$$\frac{2}{\|w\|_{L_2}}$$



нужно минимизировать $\|w\|_{L_2}$

Hinge Loss

Для всех точек первого класса:

$$\vec{w}\vec{x} + b \geq 1$$

Для всех точек второго класса:

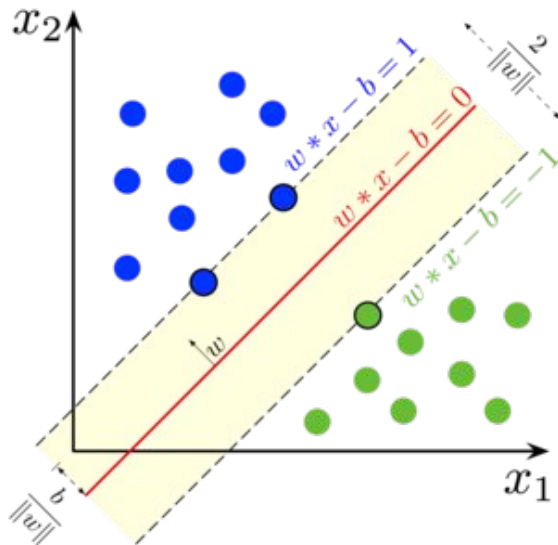
$$\vec{w}\vec{x} + b \leq -1$$

Пусть $y = +1$. Тогда условие для всех точек:

$$y(\vec{w}\vec{x} + b) \geq 1$$

И максимизировать зазор:

$$||\vec{w}||_{L_2} \rightarrow \min$$



Тогда функция потерь будет выглядеть следующим образом.

Здесь C - внешний параметр.

При больших C условие на границах становится жестким.

Можно применить градиентный спуск.

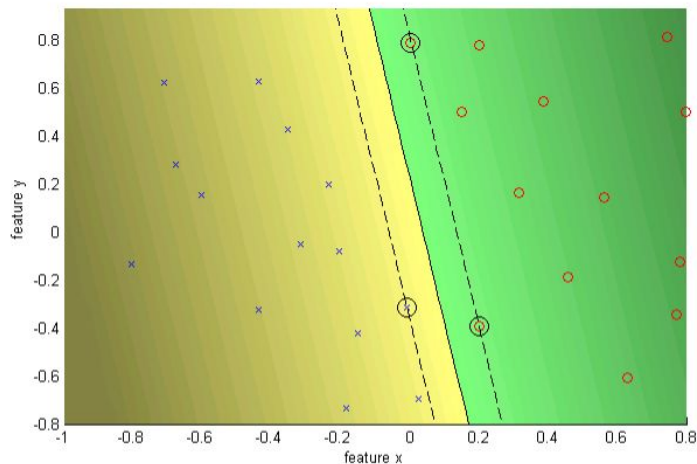
$$\mathcal{L} = ||\vec{w}||_{L_2} + C \sum_i \max(0, 1 - y_i(\vec{w}\vec{x}_i + b))$$

$$\mathcal{L} = \|\vec{w}\|_{L_2} + C \sum_i \max(0, 1 - y_i(\vec{w}\vec{x}_i + b))$$

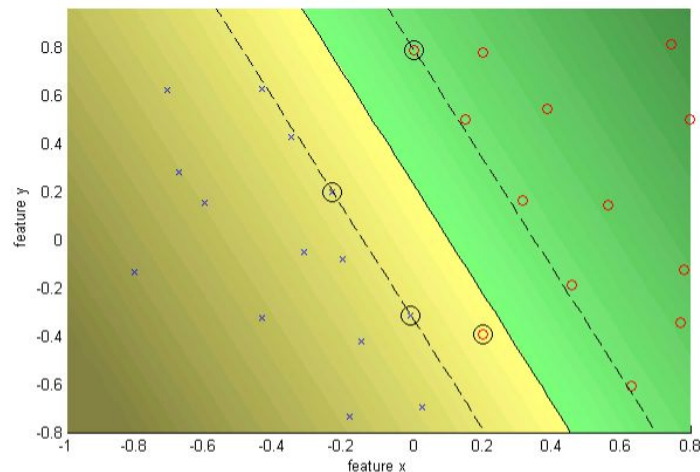
Hinge loss

Не обязательно делать C очень большим. Маленькое значение C делает классификатор устойчивым к выбросам. Кроме того, **линейная разделимость**, которую мы предположили в начале, становится **необязательным условием**.

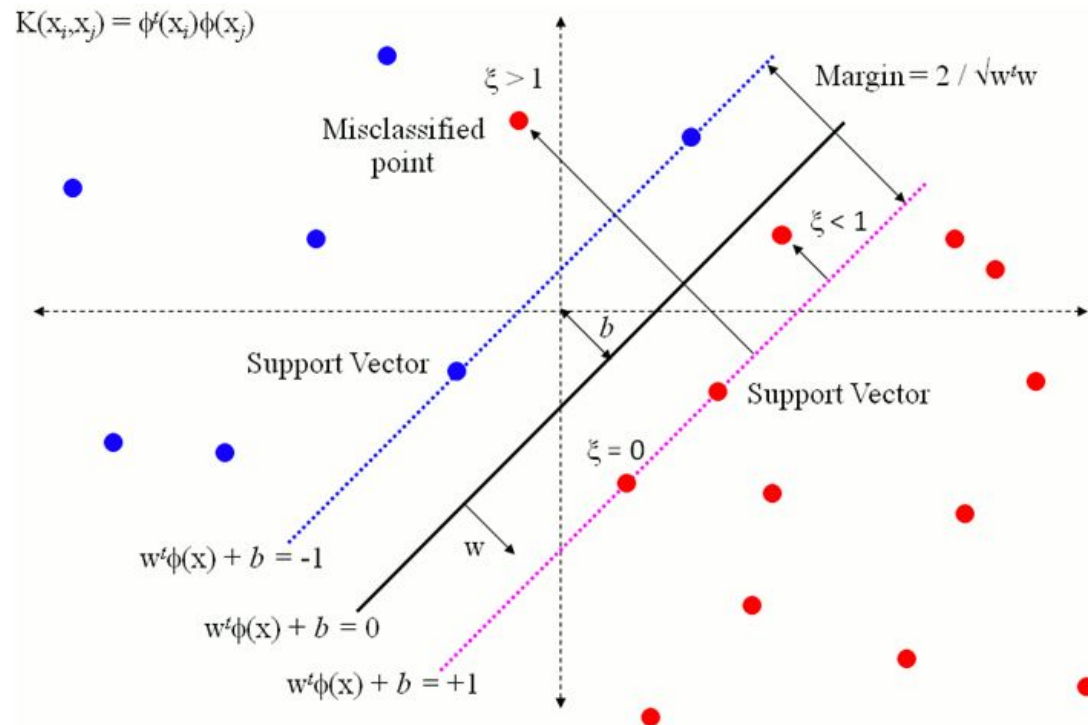
C = Infinity hard margin



C = 10 soft margin



Случай линейной неразделимости



Hinge loss

Из функций потерь, представленных справа, мы знаем кросс-энтропию (binomial deviance) и hinge loss (support vector)

Видно, что hinge loss не зависит от точек, которые уже классифицированы правильно, что делает SVM более робастным методом, чем линейная регрессия

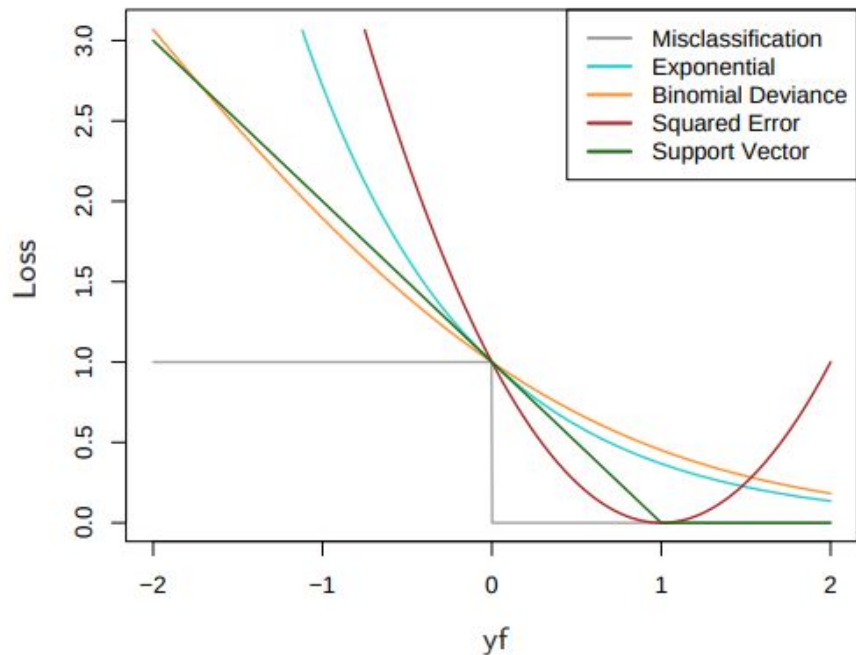


FIGURE 10.4. Loss functions for two-class classification. The response is $y = \pm 1$; the prediction is f , with class prediction $\text{sign}(f)$. The losses are misclassification: $I(\text{sign}(f) \neq y)$; exponential: $\exp(-yf)$; binomial deviance: $\log(1 + \exp(-2yf))$; squared error: $(y - f)^2$; and support vector: $(1 - yf)_+$ (see Section 12.3). Each function has been scaled so that it passes through the point $(0, 1)$.

Kernel trick

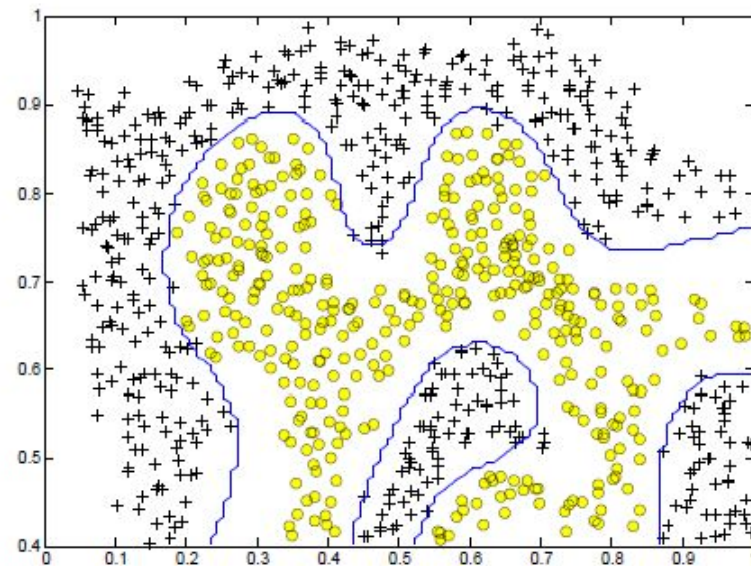
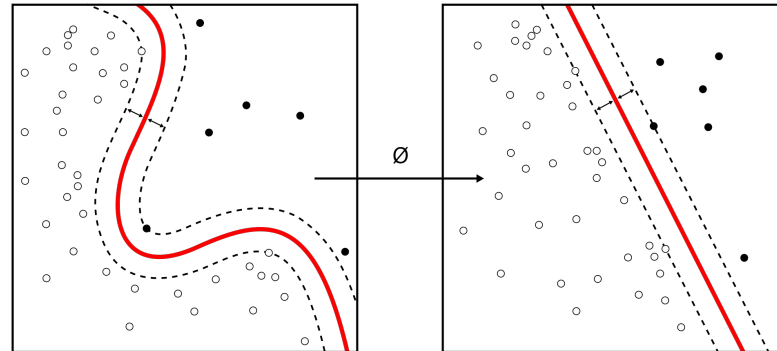
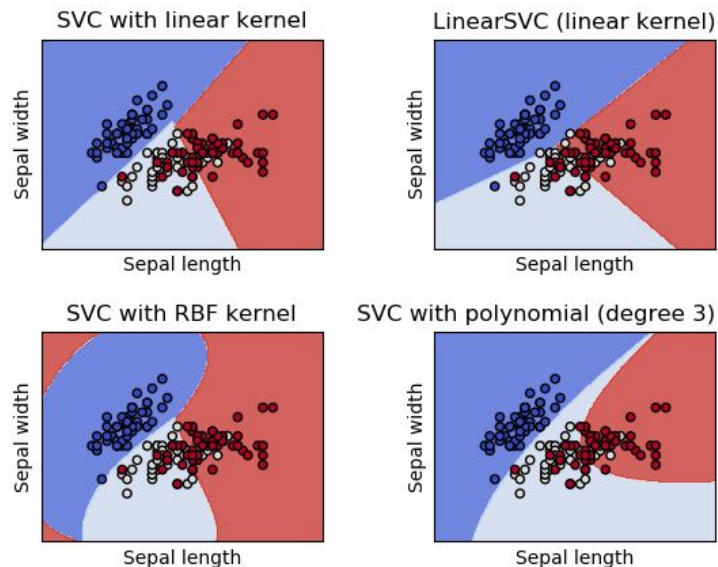
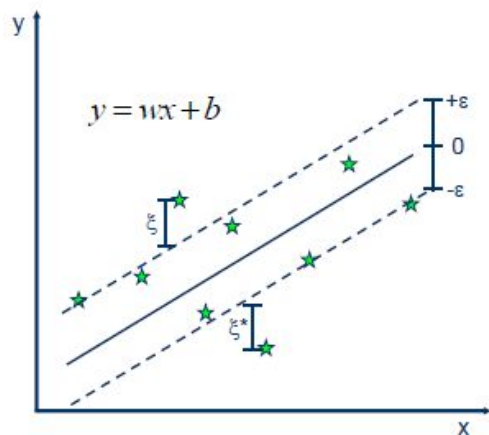


Figure 5: SVM (Gaussian Kernel) Decision Boundary (Example Dataset 2)

SVM Regression



• Minimize:

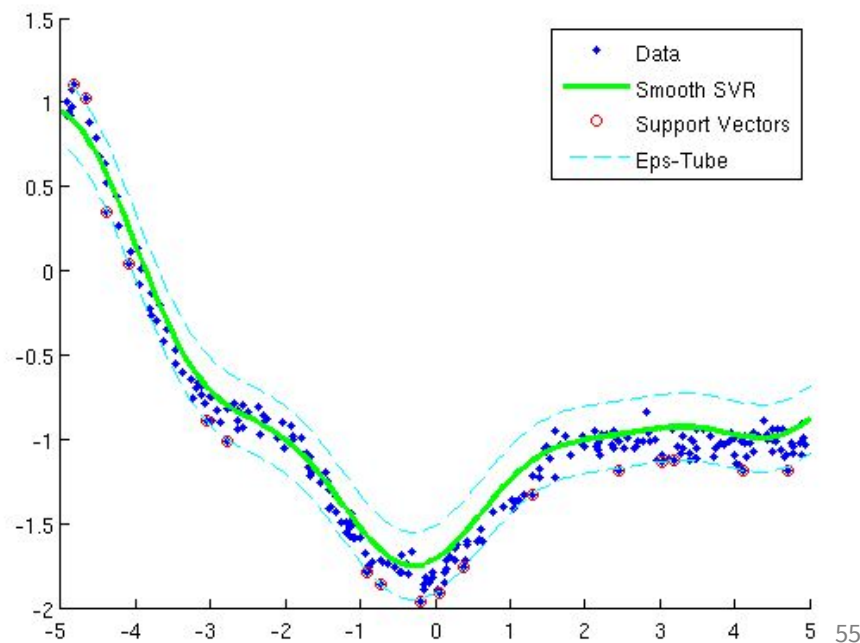
$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*)$$

• Constraints:

$$y_i - wx_i - b \leq \epsilon + \xi_i$$

$$wx_i + b - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$



Практика