ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА

Лекции по курсу «Нейронные сети и машинное обучение»

Для студентов направления 090401

Очное обучение 2024-2025 учебный год

Лектор: Солдатова Ольга Петровна, к.т.н., доцент

Математические модели машинного обучения

Лекция 2

Методы экспериментальной оценки качества алгоритмов.

Математические модели машинного обучения.

Основы теории вероятностей.

Наивный байесовский классификатор.

Линейная регрессия.

Логистическая регрессия.

Метод «k-ближайших соседей».

Машинное обучение и признаки.

Метод главных компонент.

Дискриминантный анализ.

Линейный дискриминант Фишера.

Деревья решений.

Построение решающего дерева.

Случайный лес.

Теория Вапника-Червоненкиса.

Метод опорных векторов.

Литература по курсу

1. Куприянов А.В. «Искусственный интеллект и машинное обучение» https://do.ssau.ru/moodle/course/view.php?id=1459
2. Осовский С. Нейронные сети для обработки информации / Пер. с пол. И.Д. Рудинского. – М.: Финансы и статистика, 2002. – 344 с.: ил.
3. Горбаченко, В. И. Интеллектуальные системы: нечеткие системы и сети : учебное пособие для вузов / В. И. Горбаченко, Б. С. Ахметов, О. Ю. Кузнецова. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2018. — 105 с. — (Университеты России). — ISBN 978-5-534-08359-0. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/424887 – Режим доступа: https://urait.ru/bcode/424887
4. 4. Гафаров Ф.М. Искусственные нейронные сети и приложения: учеб. пособие / Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Изд-во Казан. ун-та, 2018. – 121 с. – Текст : электронный. – Режим доступа: https://repository.kpfu.ru/?p_id=187099
5. Хайкин С. Нейронные сети: Полный курс: Пер. с англ. - 2-е изд. – М.: Вильямс, 2006. – 1104 с.: ил.
6. Борисов В.В., Круглов В.В., Федулов А.С. Нечёткие модели и сети. – М.: Горячая линия– Телеком, 2007. -284 с.: ил.
7. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечёткие системы: Пер. с польск. И.Д.Рудинского, - М.: Горячая линия – Телеком, 2007. – 452 с. ил.
8. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. — СПб.: Питер, 2018. — 480 с.: ил. — (Серия ≪Библиотека программиста≫).
9. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение / пер. с анг. А. А. Слинкина. – 2-е изд., испр. – М.: ДМК Пресс, 2018. – 652 с.: цв. ил.

Методы экспериментальной оценки качества алгоритмов обучения

- Теоретические оценки общего риска:
  - Нетривиально рассчитываются для конкретных алгоритмов.
  - Сильно завышены.
  - Как следствие, для практической задачи малоинформативны.
- Для конкретной задачи, желательно получить точные количественные оценки качества работы, поэтому используются экспериментальные методы, называемые **методами валидации**:
  - удерживание;
  - скользящий контроль;
  - и др.

Удерживание

- Вспомним, что такое общий риск:

$$R(a, X) = P_{X^l}\left(a(x) \neq y\right) = \int_X P(x)\left[a(x) \neq y\right]dx$$

  - Его минимизация для нас является основной целью.
  - Однако, напрямую его посчитать невозможно (требует вычислений на неограниченном множестве).

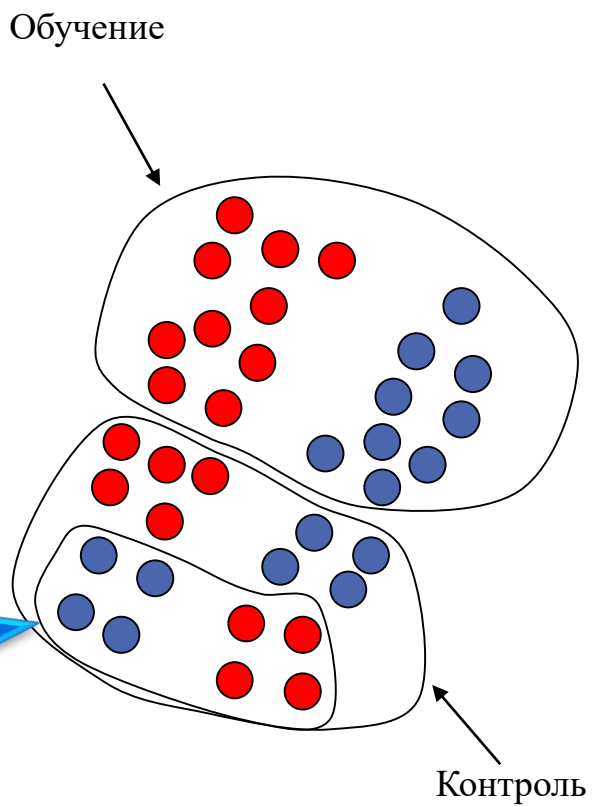
Поэтому оценим общий риск ошибкой на некотором конечном подмножестве *X* не пересекающимся с обучающей выборкой - это и есть удерживание.

- Пусть, имеется набор данных с известными ответами $X^k = \{x_1, \ldots, x_k\}$
- Разобьём его на 2 подмножества: $X^l \bigcup X^c = X^k : X^l \bigcap X^c = 0$
- Будем использовать для обучения X^l, а для контроля X^c.
- Оценим общий риск ошибкой на подмножестве X^c, не пересекающимся с обучающей выборкой:

$$R(a, X) \sim P\left(a(x) \neq y \mid X^c\right) = \frac{1}{c}\sum_{j=1}^{c}\left[a(x_j) \neq y_j\right]$$

Недостатки удерживания

- Быстро и просто рассчитывается.
- Некоторые «сложные» прецеденты могут полностью попасть в только одну из выборок и тогда оценка ошибки будет смещённой.
- Можно повторить несколько раз и усреднить результат. Это позволяет частично избавиться от «сложных» прецедентов.



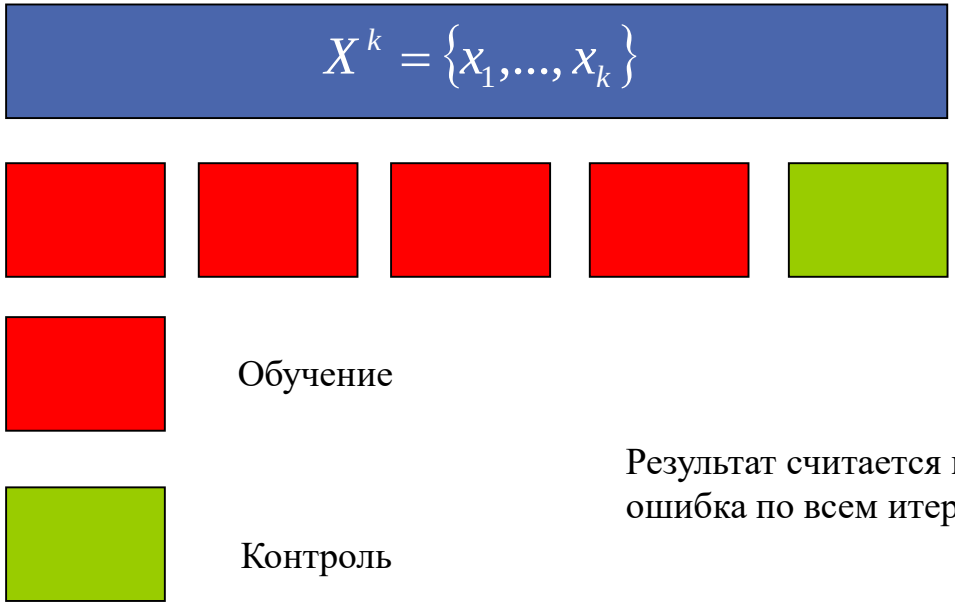
Ошибка произойдёт
не по вине классификатора,
а из-за разбиения!

*возможна и обратная
ситуация

Скользящий контроль

- Разделим выборку на множество непересекающихся подмножеств и будем поочерёдно использовать одно из них для контроля, а остальные для тренировки.
  - Разбиваем: $\{X^i\}_1^f : X^i \cap X^j = 0, i \neq j$

    $\bigcup_{i=1}^{f} X^i = X^k$
  - Приближаем риск: $P(\mu(X^k) \neq y^*) \approx \frac{1}{f} \sum_{i=1}^{f} P(\mu(X^i) \neq y^* \mid \bigcup_{j \neq i} X^i)$



Результат считается как средняя ошибка по всем итерациям

Свойства скользящего контроля

- ✓ В пределе равен общему риску.
- ✓ Каждый прецедент будет один раз присутствовать в контрольной выборке.
- ✓ Обучающие выборки будут сильно перекрываться (чем больше сегментов, тем больше перекрытие):
  - ➢ Если один группа «сложных прецедентов» попала полностью в один сегмент, то оценка будет смещенной.

k-блочная кросс-валидация и Scikit-learn

В библиотеке Scikit-learn такой вид валидации реализован при помощи KFold.
Применение k-блочной кросс-валидации для двух итераций:

```
import numpy as np
import sklearn
from sklearn.model_selection import KFold
```

```
kf = KFold(n_splits = 2)
```

Эвристические алгоритмы повышения точности оценок

Бустинг (boosting)

- ✓ Базовые алгоритмы строятся последовательно, для каждого базового алгоритма, начиная со второго, веса обучающих объектов пересчитываются так, чтобы он точнее настраивался на тех объектах, на которых чаще ошибались все предыдущие алгоритмы. Веса алгоритмов также вычисляются исходя из числа допущенных ими ошибок.
- ✓ Бустинг представляет собой жадный алгоритм построения композиции алгоритмов.
- ✓ Наиболее известен алгоритм AdBoost предложенный в 1995 г. Шапиро (Schapire R. The boosting approach to machine learning: An overview // MSRI Workshop on Nonlinear Estimation and Classification, Berkeley, CA. - 2001.)

Баггинг (bagging — «**b**ootstrap **agg**regation»)

- ➢ В отличие от *бустинга* все элементарные классификаторы обучаются и работают параллельно (независимо друг от друга). Идея заключается в том, что классификаторы не исправляют ошибки друг друга, а компенсируют их при голосовании.
- ➢ Предложен Л. Брейманом (Breiman L. Arcing classifiers // The Annals of Statistics. – 1998. – Vol. 26, no. 3. – Pp. 801−849.), производится взвешенное голосование базовых алгоритмов, обученных на различных подвыборках данных, либо на различных частях признакового описания объектов.
- ➢ Базовые классификаторы должны быть независимыми, это могут быть классификаторы основанные на разных группах методов или же обученные на независимых наборах данных. Во втором случае можно использовать один и тот же метод.
- ➢ Из множества объектов обучающей выборки формируются *бутстреп- выборки* – случайные подмножества объектов, как правило с повторами.

Математические модели машинного обучения

Придумано большое количество моделей машинного обучения, простых и сложных, точных и не очень, легко вычисляемых и требующих огромных вычислительных мощностей.

Поиск лучшей модели — это сложный эксперимент, надо перебрать разные модели, разные варианты моделей, а все это требует времени и средств.

Нельзя заранее сказать какая модель будет лучше или хуже, бывает так, что простая модель окажется лучше сложной или наоборот. В первую очередь это зависит от данных, на которых модель обучается.

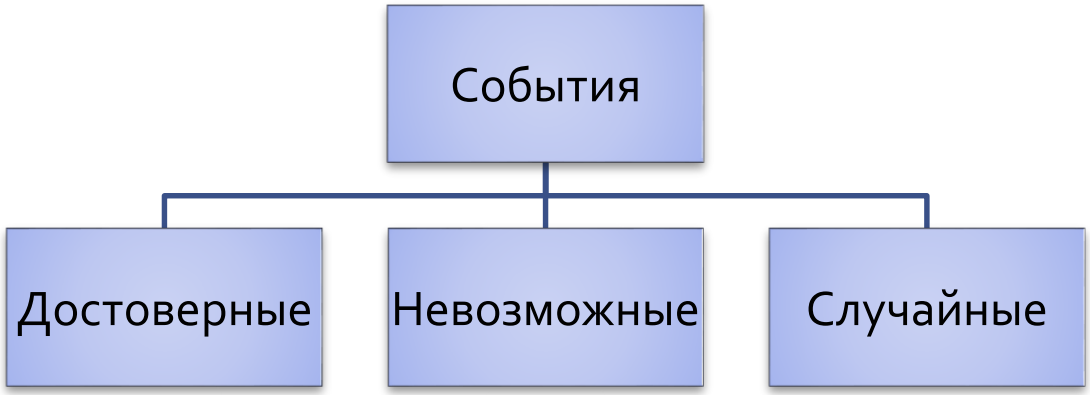
Есть множество простых задач, которые уже многократно успешно решались, но можно изменить условия и модель перестанет работать. Например, обучили модель распознавать автомобили, но изменилось освещение, пошел дождь, образовался туман, и модель не работает.

Модели машинного обучения

Основные математические модели машинного обучения:

- деревья решений и случайный лес;
- наивная байесовская классификация;
- линейная регрессия и метод наименьших квадратов;
- логистическая регрессия;
- метод опорных векторов;
- линейный дискриминант Фишера;
- ансамблевые методы: бустинг, беггинг, конкретно xgboost, catboost;
- метод «к-средних» Kmeans;
- метод «к ближайших соседей» KNN;
- метод главных компонент PCA;
- нейронные сети.

Виды событий

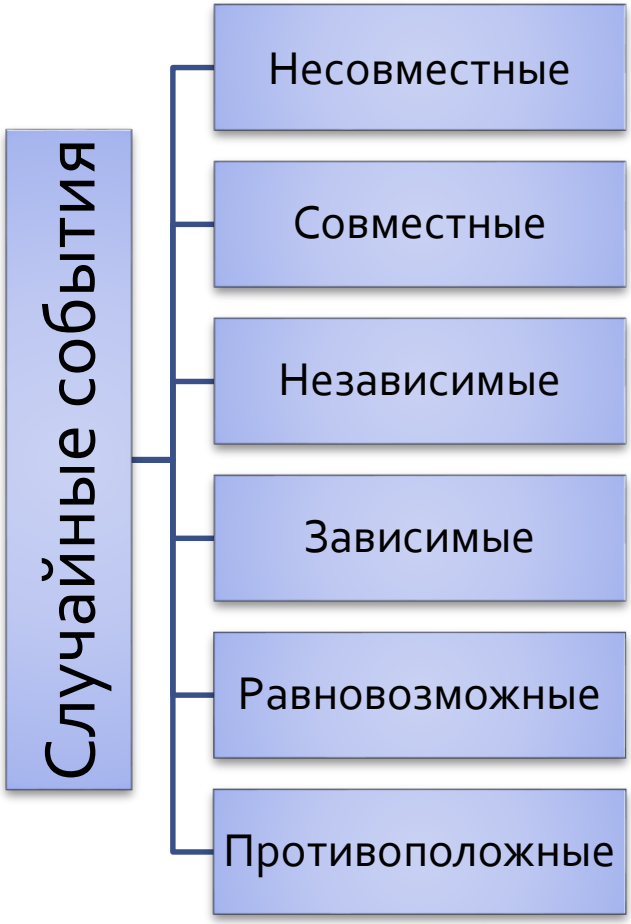


Достоверные события **всегда** происходят при осуществлении данной совокупности условий.

Невозможные события **никогда не** происходят при осуществлении данной совокупности условий.

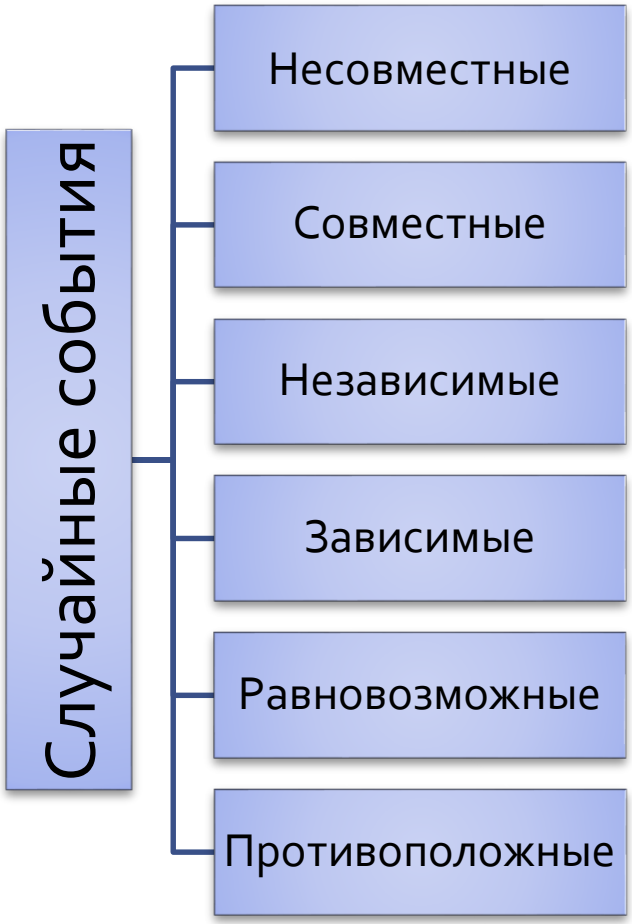
Случайные события **могут произойти** или **не произойти** при осуществлении данной совокупности условий.

Случайные события



- **Несовместными** называются события, которые не могут одновременно произойти в одном испытании
- Совокупность случайных событий A_1, A_2, A_3,... A_n называется **полной группой** для данного испытания, если в результате испытания обязательно происходит только одно из событий этой совокупности
- Два события (A и $\bar{A}$) называются **противоположными**, если появление одного из них равносильно не появлению другого

Случайные события



- **Совместными** называются события, которые могут одновременно произойти в одном испытании
- События называются **независимыми**, если появление одного из них не изменяет вероятности появления второго.
- События называются **зависимыми** если появление одного из них зависит от появления другого
- **Равновозможными** называются события, если ни у одного из них нет объективного преимущества перед другим

Классическое определение вероятности

Вероятностью события A называют отношение числа благоприятствующих этому событию элементарных событий *(m)* к общему числу всех равновозможных несовместных элементарных событий *(n)*, образующих полную группу:

$$P(A) = \frac{m}{n}$$

Чтобы рассчитать классическую вероятность необходимо до проведения испытаний теоретически подсчитать:

- общее число всех **равновозможных несовместных** элементарных событий *(n)*
- число **благоприятствующих** этому событию равновозможных несовместных элементарных событий *(m)*

- **Вероятность** достоверного **события** $P = 1$
- **Вероятность** невозможного **события** $P = 0$
- **Вероятность** случайного **события** $0 < P < 1$

Условная вероятность

Определение.
Пусть $P(A)>0$.

Условной вероятностью $P(B/A)$ события B при условии, что событие A наступило, называется число

$$P(B/A)=\frac{P(AB)}{P(A)}$$

Обозначения:

$$P(B/A)=P_A(B)$$

Условная вероятность удовлетворяет всем аксиомам вероятности.

В частности, $0 \le P(B/A) \le 1$, $P(A/A)=1$

Независимые события. Полная группа событий

Определение.

- События A и B называются **независимыми**, если $P(AB) = P(A)P(B)$

Определение. Пусть $P(A)>0$ и $P(B)>0$.

- **Событие *A* не зависит от *B***, если $P(A/B) = P(A)$

Следствие.

- Если событие A не зависит от B, то и событие B не зависит от A.

- Доказательство. $P(AB) = P(A/B)P(B) = P(A)P(B)$

На практике из физической независимости событий делают вывод о теоретико-вероятностной независимости.

$$P(B/A) = \frac{P(AB)}{P(A)} = \frac{P(A)P(B)}{P(A)} = P(B)$$

- События **образуют полную группу**, если они:

1) попарно несовместны

2) в результате эксперимента обязательно какое- либо одно из них наступит

$P(H_i H_j) = 0,\ i \neq j$ $H_1 + H_2 + ,...,+ H_n = \Omega$ H_i **- гипотезы**

- **Пример.**
- В стохастическом эксперименте рассмотрим события
- Они образуют полную группу. A и $\overline{A}$

Формула полной вероятности

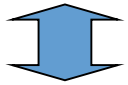
Теорема.

- Если события $H_1, H_2, ..., H_n$

образуют **полную группу**,

то для любого события A справедлива формула

$$P(A) = P(H_1)P(A/H_1) + ... + P(H_n)P(A/H_n)$$



$$P(A) = \sum_{i=1}^{n} P(H_i)P(A/H_i)$$

Формула Байеса

Теорема.

- Пусть события $H_1, H_2, ..., H_n$

образуют **полную группу.**

Пусть событие ***A*** наступило ***(P(A)>0)***.

Тогда вероятность того,

что при этом была реализована гипотеза (наступило событие) H_k

вычисляется по формуле

$$P(H_k/A) = \frac{P(H_k)P(A/H_k)}{P(A)} = \frac{P(H_k)P(A/H_k)}{\sum_{i=1}^{n} P(H_i)P(A/H_i)}$$

Формула Байеса позволяет переоценить вероятности гипотез после того, как проведено испытание, в результате которого произошло событие *A*.

Формула Байеса. Частный случай

- Рассмотрим события H и $\overline{H}$

они образуют **полную группу.**

Пусть событие A наступило $(P(A)>0)$.

Тогда вероятность того,

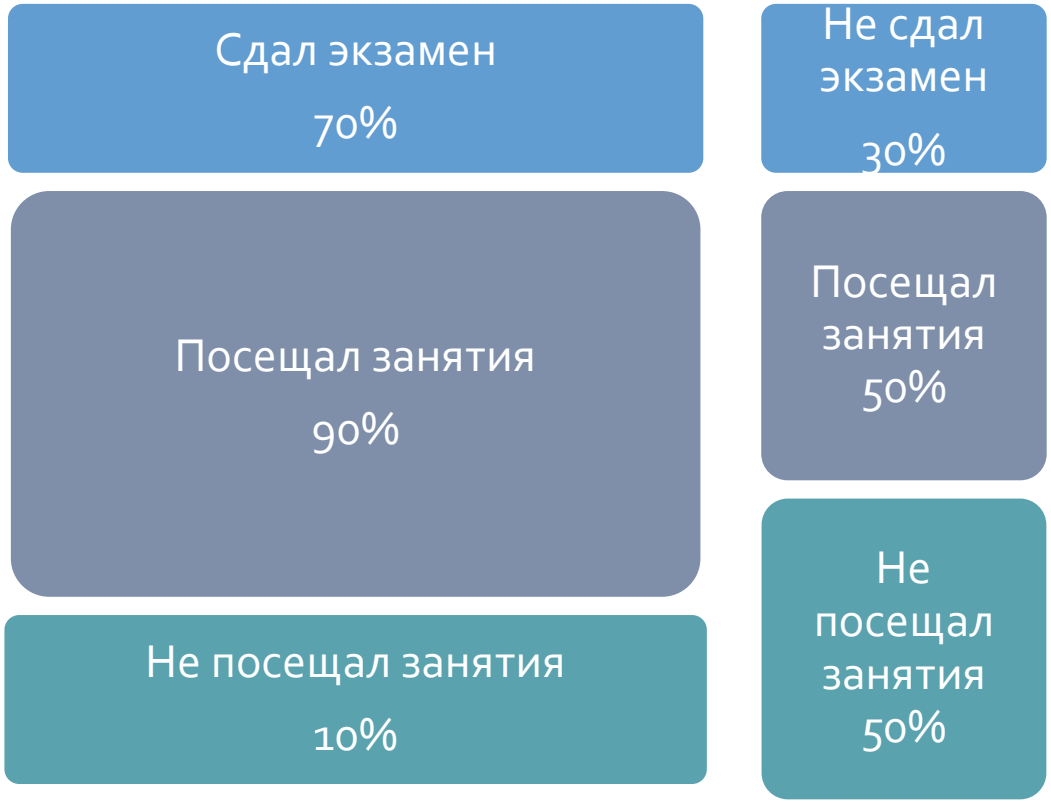
что при этом была реализована гипотеза H

вычисляется по формуле

$$P(H/A)=\frac{P(H)P(A/H)}{P(A)}$$

$$P(H/A)=\frac{P(H)P(A/H)}{P(H)P(A/H)+P(\overline{H})P(A/\overline{H})}$$

Пример расчёта вероятности



$$P(\textit{сдал экзамен} \mid \textit{посещал занятия}) = ?$$

Пример расчёта вероятности

$$P(\text{сдал}) = 0{,}7 \qquad P(\text{не сдал}) = 0{,}3$$

$$P(\text{посещал} \mid \text{сдал}) = 0{,}9$$

$$P(\text{посещал} \mid \text{не сдал}) = 0{,}5$$

$$P(\text{не посещал} \mid \text{сдал}) = 0{,}1$$

$$P(\text{не посещал} \mid \text{не сдал}) = 0{,}5$$

$$P(\text{сдал экзамен} \mid \text{посещал занятия}) =$$

$$= P(\text{сдал})P(\text{посещал} \mid \text{сдал}) / P(\text{посещал}) =$$

$$= 0{,}7 * 0{,}9 / 0{,}78 = 0{,}81$$

$$P(\text{посещал}) = P(\text{сдал})P(\text{посещал} \mid \text{сдал}) +$$

$$+ P(\text{не сдал})P(\text{посещал} \mid \text{не сдал}) =$$

$$= 0{,}9 * 0{,}7 + 0{,}5 * 0{,}3 = 0{,}63 + 0{,}15 = 0{,}78$$

Вероятностная формулировка задачи машинного обучения

- Эмпирический риск:

$$R_{Emp}(a, X^l) = P\left(a(x) \neq y \mid X^l\right) = \frac{1}{l}\sum_{i=1}^{l}[a(x_i) \neq y_i]$$

- Общий риск:

$$R(a, X) = P\left(a(x) \neq y \mid X\right) = \int_X P(x)\left[a(x) \neq y\right]dx$$

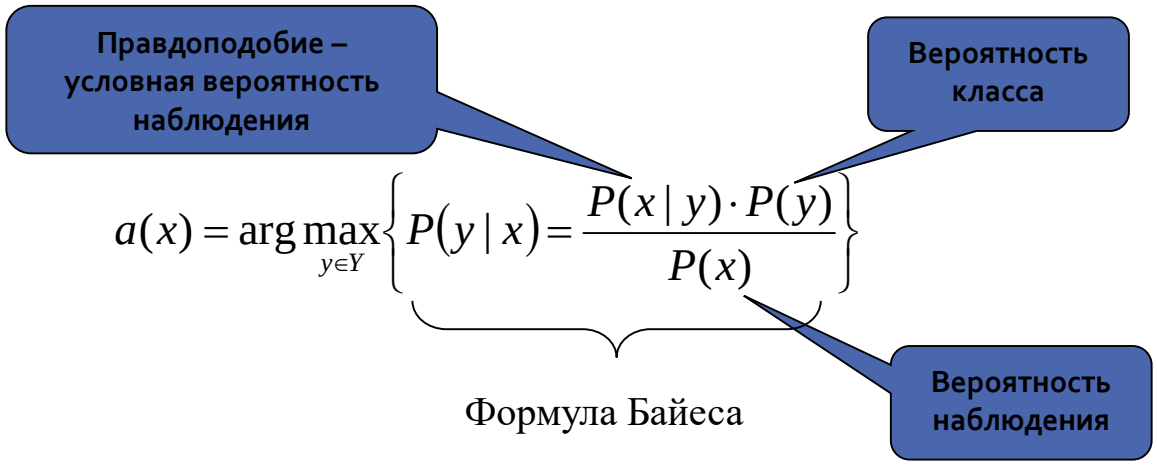
  - рассчитать невозможно
  - требуется минимизировать

- Модель алгоритма и метод обучения определяются так же, как и раньше.

Наивный байесовский классификатор

Наивный байесовский классификатор — простой вероятностный классификатор, основанный на применении Теоремы Байеса со строгими (наивными) предположениями о независимости.

- Предположения:
  - Известна функция правдоподобия: $P(x \mid y)$
  - Известны априорные вероятности: $P(y), P(x)$
- Принцип максимума апостериорной вероятности:



Формула Байеса

Наивный байесовский классификатор

Алгоритм:

1. Для каждой гипотезы вычислить апостериорную вероятность.
2. Выбрать гипотезу с максимальной апостериорной вероятностью

Эмпирический риск: $R_{emp}(a, X) = P(a(x) \neq y \mid X)$

Особенности наивного байесовского классификатора:

- *Нужно знать функцию правдоподобия и априорные вероятности.*
- *Отсутствуют априорные причины верить, что одна из гипотез более вероятна чем другая (наивность).*
- *Отвечает на вопрос – какова наиболее вероятная гипотеза при имеющихся данных?*
- *Надо ответить на вопрос – какова наиболее вероятная классификации нового примера при имеющихся данных?*

Разбиение пространства, как задача классификации

- Задача классификации: определить вектор x в один из K классов Y .
- В итоге всё пространство разобьётся на эти классы.
- То есть при классификации надо построить разделяющую поверхность.
- Рассмотрим линейную дискриминантную функцию:

$$y(x) = w^T x + w_0$$

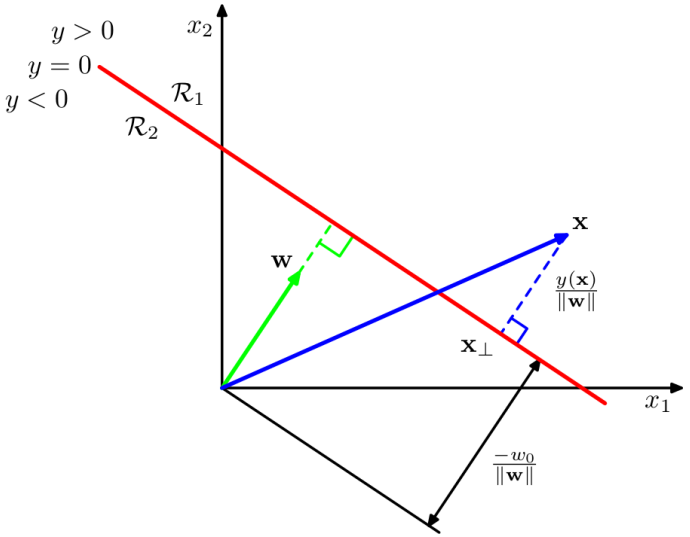
Разбиение пространства и классификация

Задача классификации: определить вектор $\boldsymbol{x}$ в один из классов $\boldsymbol{Y}$.

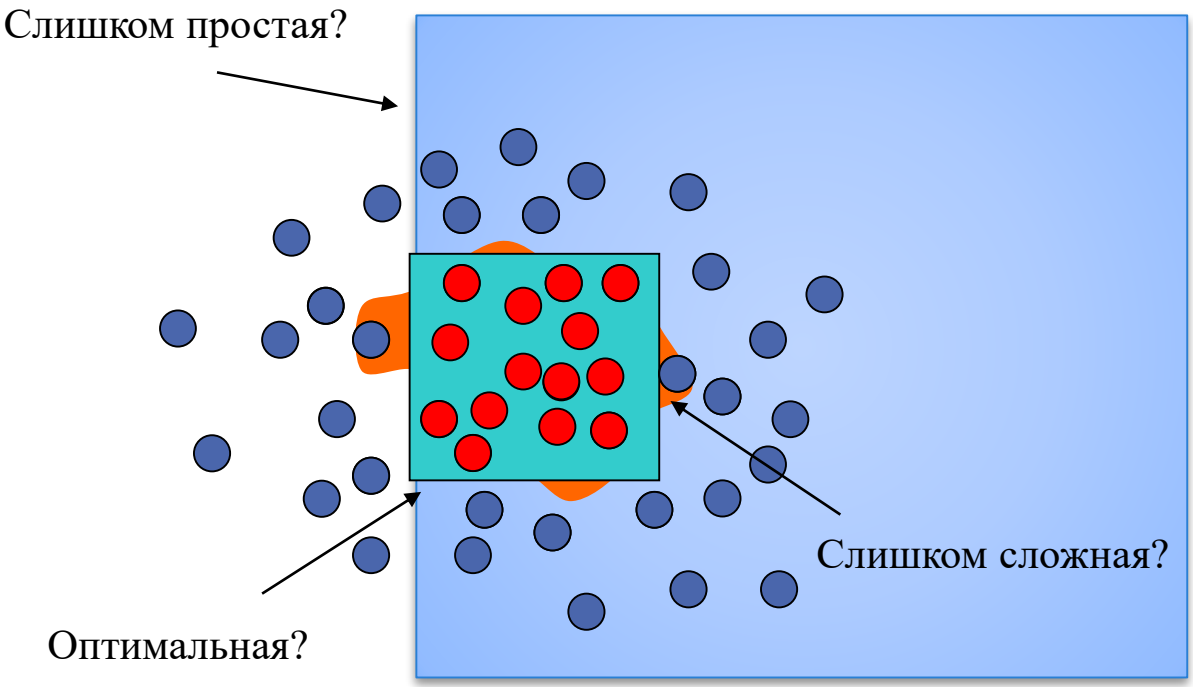
Рассматриваем линейную дискриминантную функцию:

$$y(x) = w^T x + w_0$$

В линейном случае мы спроецировали все точки в размерность 1 (на нормаль разделяющей гиперплоскости) так, чтобы в этой размерности 1 они «хорошо» разделялись.



Пример

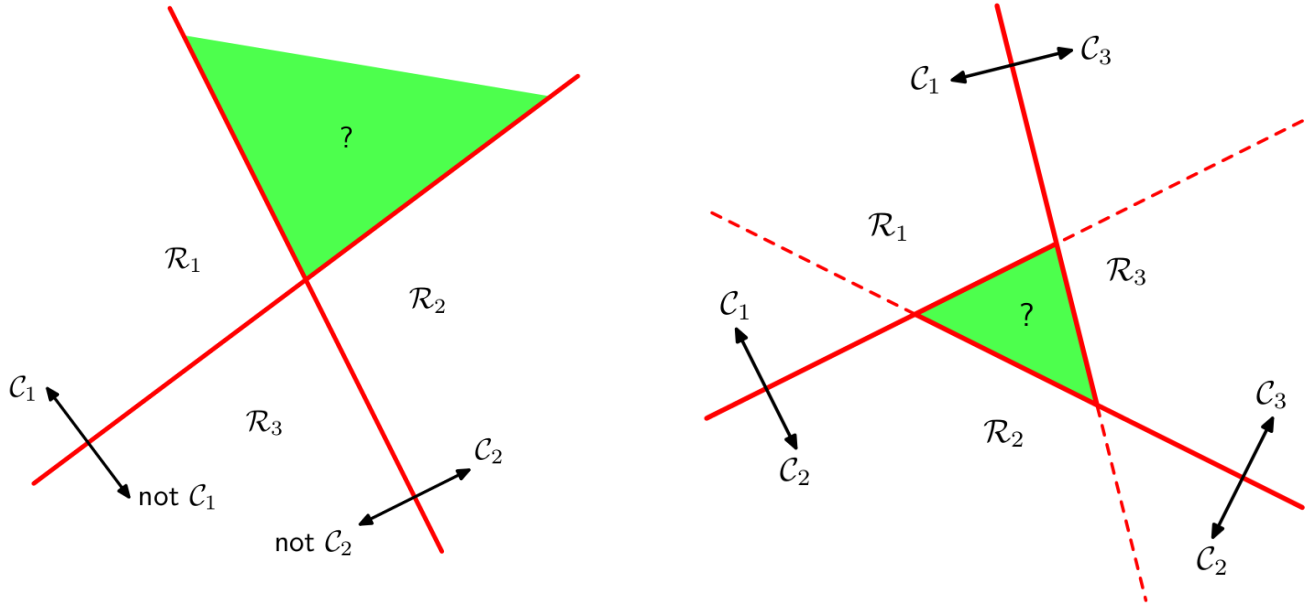


Разделение на несколько классов

- Можно рассмотреть поверхности вида «один против всех».
- Можно рассмотреть поверхности вида «каждый против каждого».
- Можно рассмотреть единый дискриминант из k линейных функций вида

$$y_k(x) = w_k^T x + w_{k0} \quad .$$

- Классифицируем в Y_k если соответствующий y_k – максимален.

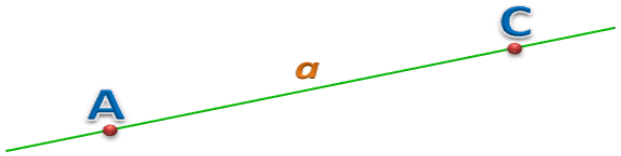


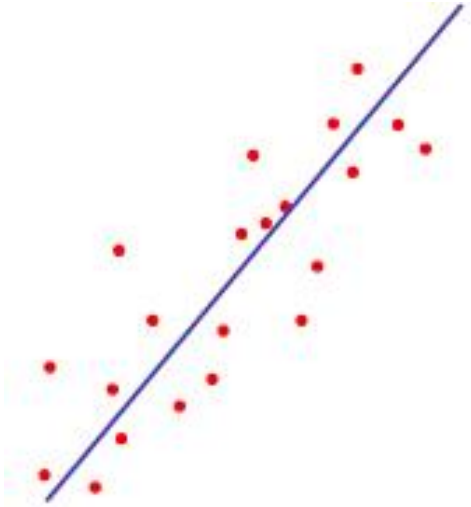
Задача линейной регрессии

- **Нужно найти функцию, которая отображает зависимость одних переменных или данных от других.**
- Зависимые данные называются **зависимыми переменными, выходами** или **ответами.**
- Независимые данные называются **независимыми переменными, входами или предсказателями.**
- Обычно в регрессии присутствует одна непрерывная и неограниченная зависимая переменная.
- Входные переменные могут быть неограниченными, дискретными или категорическими данными.

Задача линейной регрессии

Через две точки на плоскости можно провести только одну прямую.





Что делать, если точек на плоскости – три и более?

Метод наименьших квадратов (МНК) состоит в том, чтобы найти такие коэффициенты регрессии, при которых достигается минимум следующего функционала качества на заданной обучающей выборки:

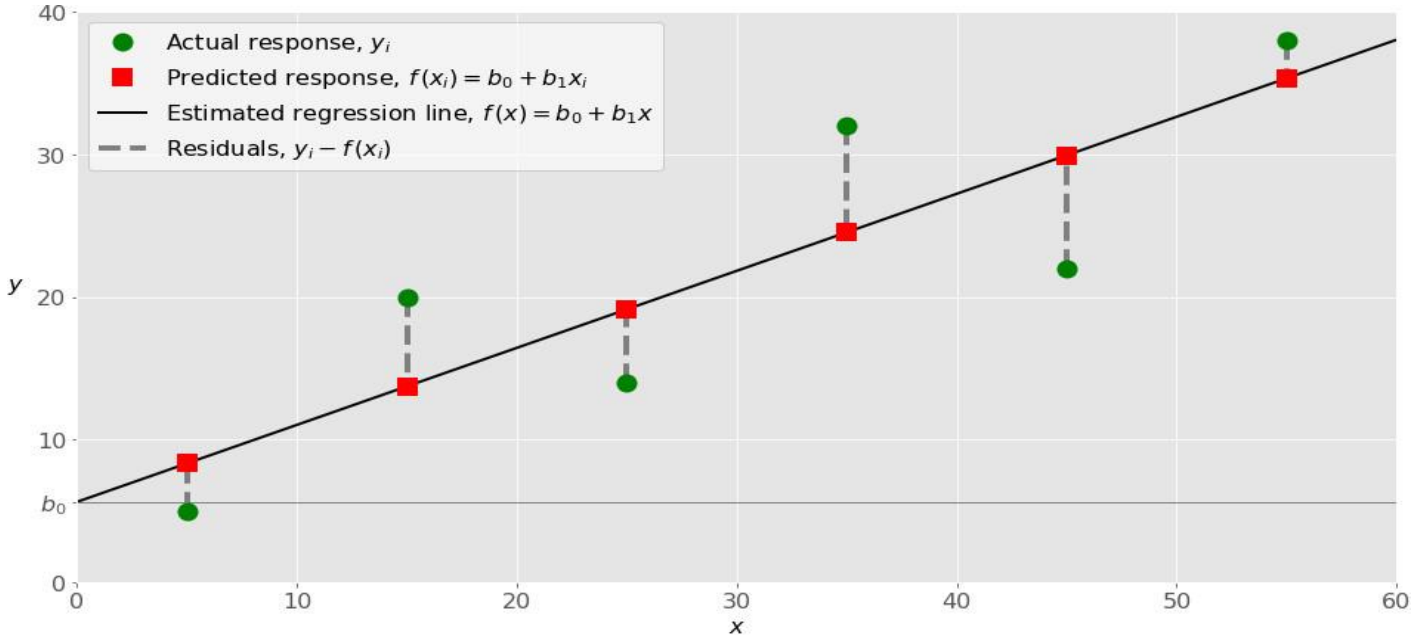
$$y_i' = y_k'(x_i) = w_k^T x_i + w_k^0$$

$$\sum_i (y_i - y_i')^2 = \sum_i (y_i - w_k^T x_i + w_k^0)^2 \xrightarrow{k} \min$$

Библиотека Scikit-learn

- Библиотека Scikit-learn — самый распространённый выбор для решения задач классического машинного обучения.
- Scikit-learn специализируется на алгоритмах машинного обучения для решения задач
  - обучения с учителем:
    - ***классификации*** *(предсказание признака, множество допустимых значений которого ограничено)*
    - ***регрессии*** *(предсказание признака с вещественными значениями)*
  - обучения без учителя:
    - ***кластеризации*** *(разбиение данных по классам, которые модель определит сама)*,
    - ***понижения размерности*** *(представление данных в пространстве меньшей размерности с минимальными потерями полезной информации)*
    - ***детектирования аномалий***.

Линейная регрессия в библиотеке Scikit-learn



- Вход-выход (x-y) (зелёные круги) – результаты наблюдений.
- Оценочная функция регрессии (чёрная линия) выражается уравнением $f(x) = b_0 + b_1 x$.
- Предсказанные ответы (красные квадраты) – точки линии регрессии, соответствующие входным значениям.
- Остатки (вертикальные пунктирные серые линии) – при реализации линейной регрессии минимизируется сумма квадратов расстояний.

Логистическая регрессия

Логистическая регрессия — это метод построения линейных классификаторов. Логистическая регрессия, несмотря на свое название, представляет собой скорее линейную модель классификации, чем регрессию. Логистическая регрессия также известна в литературе как логит-регрессия, классификация максимальной энтропии (MaxEnt) или лог-линейный классификатор. В этой модели вероятности, описывающие возможные результаты одного испытания, моделируются с использованием логистической функции.

Логистическая регрессия реализована в **LogisticRegression.** Эта реализация может соответствовать бинарной, однозначной или полиномиальной логистической регрессии с необязательной L1, L2, Elastic Net регуляризацией.

Рассмотрим постановку задачи. Пусть имеется обучающая выборка. Это пара объект-ответ. Объекты описываются n-вещественными признаками, а ответы — это числа (−1) и (+1), то есть рассматривается бинарная задача классификации. Линейная модель классификации — это скалярное произведение вектора объектов на вектор весов. Вектор весов — это направляющий вектор разделяющей гиперплоскости. Если скалярное произведение положительное, то объект относится к классу (+1), если отрицательное, то к классу (−1).

Логистическая регрессия

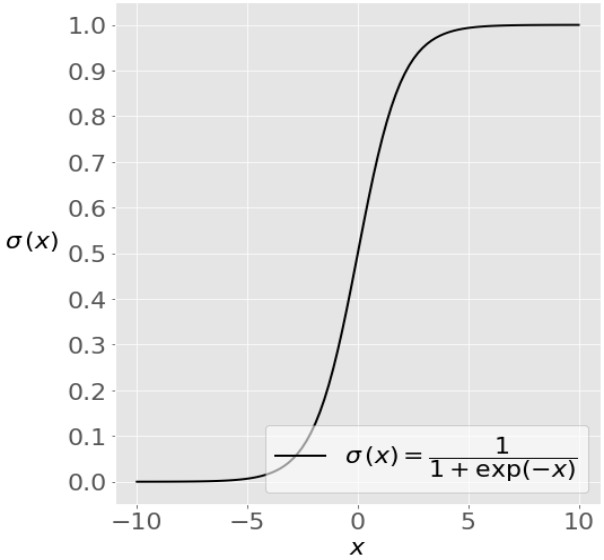
Рассмотрим непрерывную аппроксимацию бинарной функции ошибки. Если в качестве ошибки рассматривается бинарная величина, то просто подсчитывается число ошибок классификатора на обучающей выборке.

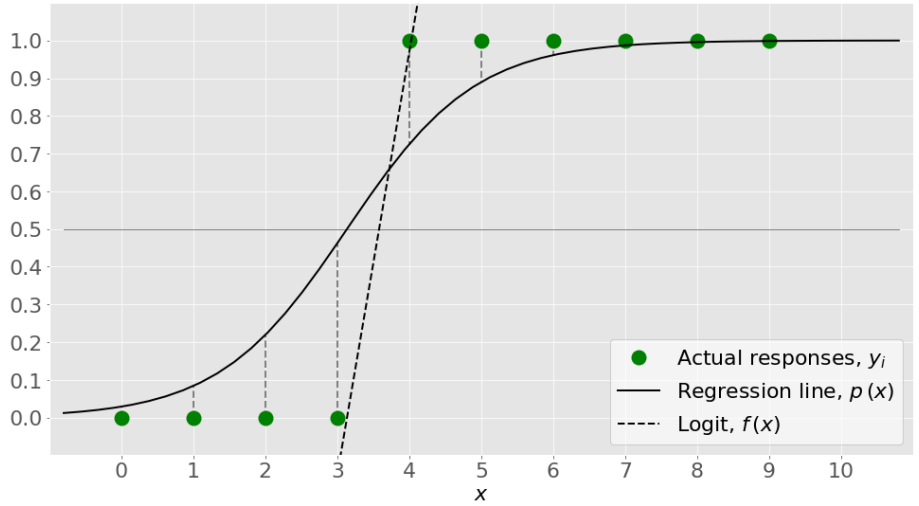
Если же используется непрерывная функция ошибки, которая мажорирует сверху бинарную функцию ошибки, то получаем функционал, который гораздо удобнее минимизировать, потому что он является непрерывным или даже гладким.

Но поскольку функционал мажорирует сверху, то, минимизируя этот функционал, также будем минимизировать и исходное число ошибок. Будем рассматривать конкретную функцию ошибки, которая называется логарифмической, и ее вид показан на рисунке.

Логарифмическая функция ошибки похожа на функцию ошибки метода опорных векторов, однако в методе опорных векторов функция ошибки кусочно-линейная, а здесь гладкая. Логарифмическая функция приобретает очень интересный вероятностный смысл, если предположить, что имеется вероятностная модель порождения данных, то есть что выборка данных является выборкой независимых наблюдений из одного и того же параметрического семейства распределений.

Логистическая регрессия





$$f(z) = \frac{1}{1+e^{-z}}$$ – логистическая функция

$$z = \sum_n \theta_n x_n$$ – регрессия

Логистическая регрессия

Это совместная плотность распределений иксов и игреков с каким-то параметром w.

Поскольку выборка независимая, то, исходя из принципа максимума правдоподобия, можно определить параметр w, а независимость дает возможность представить правдоподобие в виде суммы логарифмов совместных плотностей объектов и ответов. Совместную плотность p(x, y) можно представить по формуле условной вероятности в виде произведения условной вероятности ответа y при условии x и безусловной плотности p(x). Предположим, что плотность p(x) не зависит от параметра модели, а параметр модели используется только для описания условной или, апостериорной вероятности класса для данного объекта x. Оказывается, что если апостериорная вероятность класса yi-тое для объекта xi-тое задается сигмоидальной функцией, то принцип максимума правдоподобия даст тот же функционал, который был введён из чисто эвристических соображений. Получается, что данный функционал — это минимум аппроксимированного эмпирического риска, и функционал правдоподобия — максимум логарифма правдоподобия, они вот при этом оказываются эквивалентны.

.

Логистическая регрессия

Оказывается, что если оптимизировать такой функционал, то можно оценить и апостериорную вероятность класса для каждого объекта, который требуется классифицировать. Это и есть основное отличие логистической регрессии от других методов линейной классификации. Она позволяет оценивать вероятности классов. Как же производится оптимизация этого функционала? Есть 2 подхода. Первый подход — это методы первого порядка. В частности, можно использовать метод стохастического градиентного спуска. И если расписать формулу стохастического градиента, то окажется, что в нее тоже войдет вероятность правильной классификации — апостериорная вероятность класса yi-тое (правильного класса) на объекте xi-тое.

Есть другой подход. Методы второго порядка также применяются для решения этой задачи. Это приведет к очень известному методу, который чаще всего и используется на практике для построения логистической регрессии, — это итеративно перевзвешиваемый метод наименьших квадратов. На каждой итерации этого метода решается линейная регрессионная задача, и это еще один повод называть данный метод регрессией, несмотря на то, что все-таки решается задача классификации.

Логистическая регрессия

Это линейный классификатор, оценивающий апостериорные вероятности классов $P(y/x)$, необходимые в задачах оценивания рисков.

Регуляризация улучшает обобщающую способность логистической регрессии :

- *L2 – регуляризация сокращает веса линейно зависимых признаков;*
- *L1 – регуляризация для отбора признаков.*

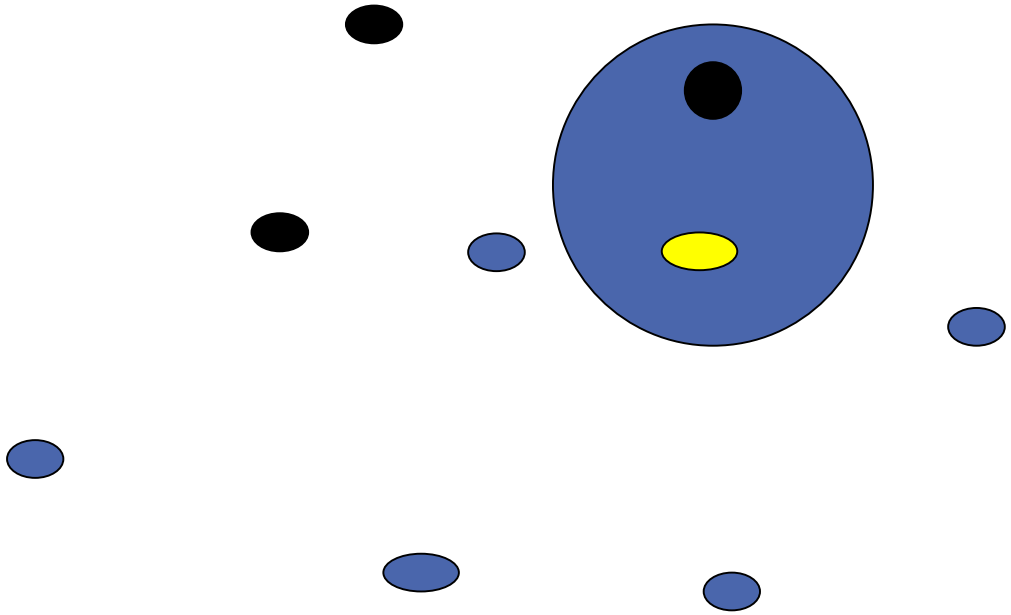
Для оценки коэффициентов регрессии обычно применяется метод оценки максимального правдоподобия

Метод «k-ближайших соседей»

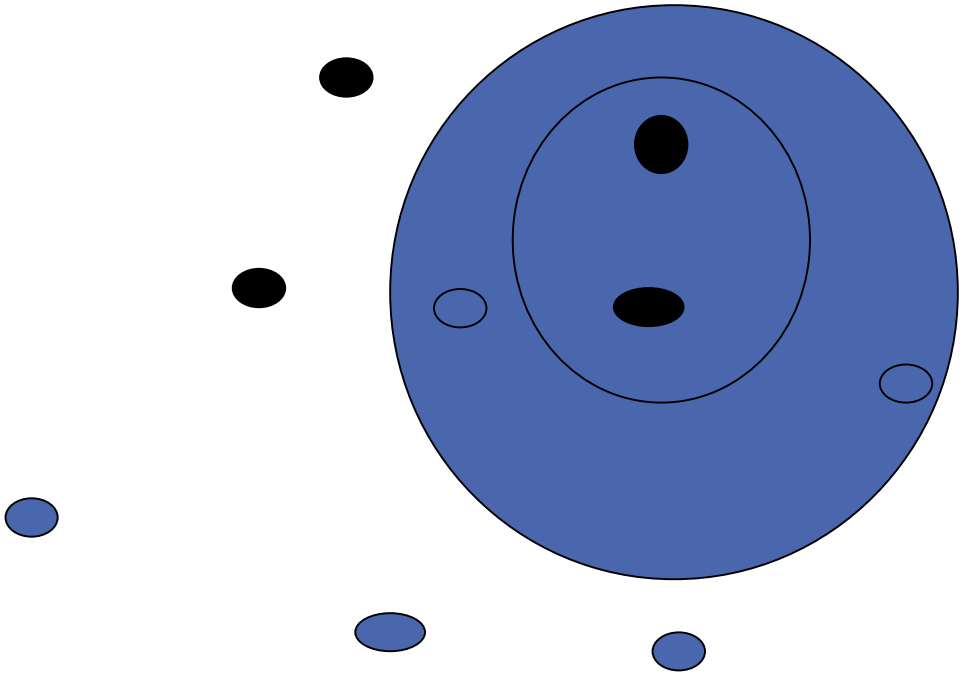
K-nearest neighbor – kNN

- Метод решения задачи классификации, который относит объекты к классу, которому принадлежит большинство из *k* его ближайших соседей в многомерном пространстве признаков.
- Число *k* – это количество соседних объектов в пространстве признаков, которое сравнивается с классифицируемым объектом.
- Использование только одного ближайшего соседа (1NN) ведёт к ошибкам из-за:
  - нетипичных примеров.
  - ошибок в ручной привязке единственного обучающего примера.
- Более устойчивой альтернативой является *k* наиболее похожих примеров и определение большинства
- Величина *k* типично нечётная: 3, 5

1-ближайший сосед



3-ближайших соседа



Нормализация и вычисление расстояния

- Евклидово расстояние $\rho(x_1, x_2) = (x_1 - x_2)^T (x_1 - x_2)$
- Минимаксная нормализация: $x^* = \frac{x - x_{min}}{x_{max} - x_{min}}$
- Нормализация с помощью стандартного отклонения: $x^* = \frac{x - x_{mean}}{\sigma_x}$
- где σ_x - стандартное отклонение

Достоинства метода kNN

Достоинства:

- ✓ Программная реализация относительно проста.
- ✓ Возможность модификации алгоритма.
- ✓ Алгоритм устойчив к аномальным выбросам.
- ✓ Возможность интерпретации результатов работы алгоритма.

Недостатки:

- ➢ Набор данных, используемый для алгоритма, должен быть репрезентативным.
- ➢ Необходимость хранить обучающую выборку целиком.
- ➢ В простейших случаях метрические алгоритмы имеют крайне бедный набор параметров, что исключает возможность настройки алгоритма по данным.
- ➢ Затраты в производительности велики, поскольку необходимо вычислить расстояния между каждым экземпляром и всеми пробными экземплярами.

Пример: Ирисы Фишера

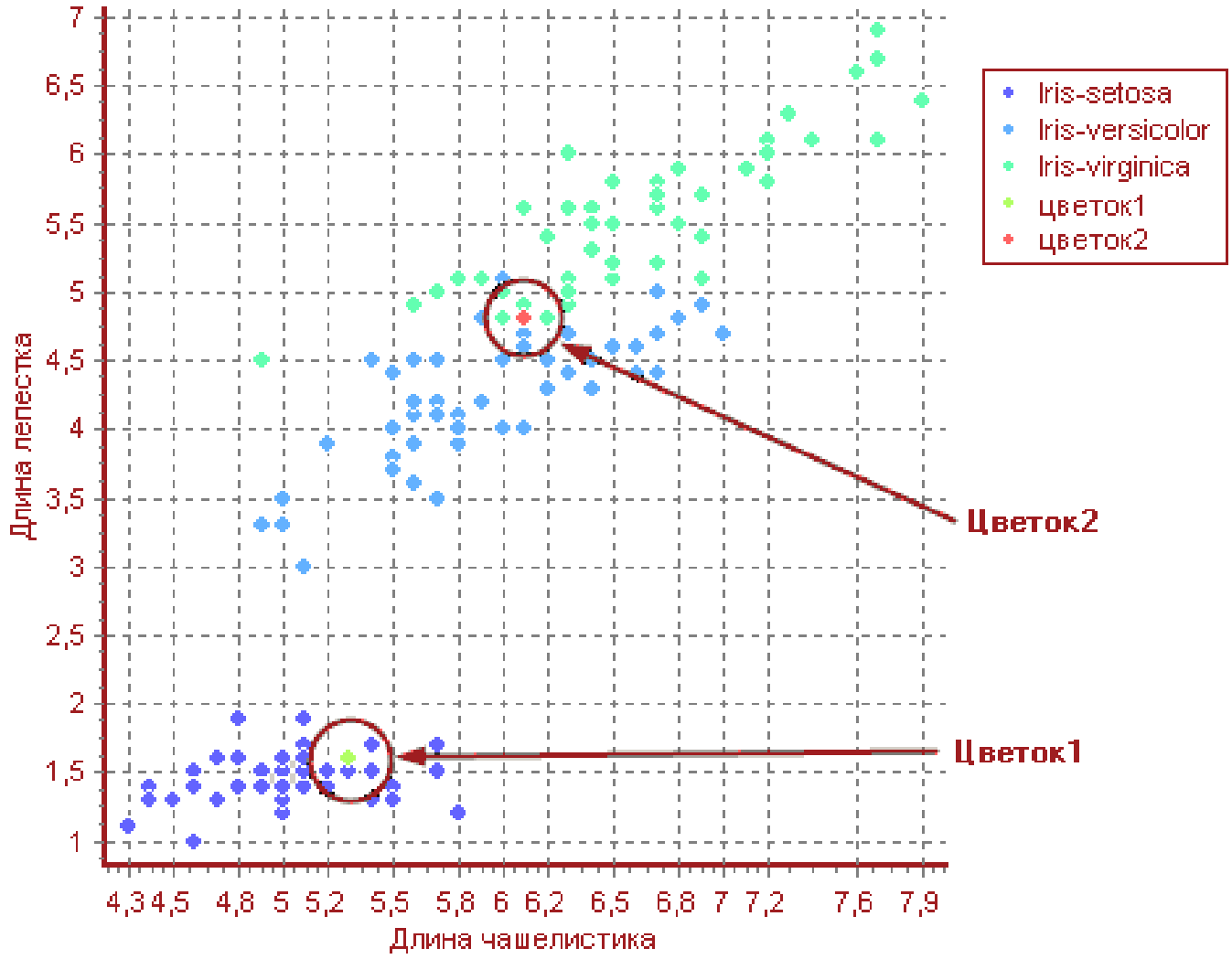
150 цветков трех классов:



Два параметра: длина чашелистика и длина лепестка.

Два новых цветка со следующими значениями длины чашелистика и лепестка: 5,3 и 1,6 (*цветок 1*), 6,1 и 4,8 (***цветок 2***).

Ирисы Фишера: диаграмма размещения классов



Ирисы Фишера: простое голосование

Цветок 1. Зададим k=3.

Ближайшие соседи: *A* (5,3; 1,5), *B*(5,2; 1,5) и *C*(5,2; 1,5).

$$d(\text{цветок } 1, A) = \sqrt{(5{,}3 - 5{,}3)^2 + (1{,}6 - 1{,}5)^2} = 0{,}1$$

$$d(\text{цветок } 1, B) = \sqrt{(5{,}3 - 5{,}2)^2 + (1{,}6 - 1{,}5)^2} = 0{,}14$$

$$d(\text{цветок } 1, \text{C}) = \sqrt{(5{,}3 - 5{,}2)^2 + (1{,}6 - 1{,}5)^2} = 0{,}14$$

Объект	Чашелистик	Лепесток	Расстояние	Класс
Цветок 1	5,3	1,6	-	-
A	5,3	1,5	0,1	Iris Setosa
B	5,2	1,5	0,14	Iris Setosa
C	5,2	1,5	0,14	Iris Setosa

Класс *цветка 1*: *Iris Setosa*

Ирисы Фишера: простое голосование

Цветок 2. Зададим k=3 и предположим, что длина лепестка вдвое важнее длины чашелистика.

Ближайшие соседи: $A(6{,}1;\ 4{,}7)$, $B(6;\ 4{,}8)$, $C(6{,}2\ 4{,}8)$

$$d(\text{цветок } 1, A) = \sqrt{(6{,}1 - 6{,}1)^2 + 2(4{,}8 - 4{,}7)^2} = 0{,}14$$

$$d(\text{цветок } 1, B) = \sqrt{(6{,}1 - 6)^2 + 2(4{,}8 - 4{,}8)^2} = 0{,}1$$

$$d(\text{цветок } 1, C) = \sqrt{(6{,}1 - 6{,}2)^2 + 2(4{,}8 - 4{,}8)^2} = 0{,}1$$

Объект	Чашелистик	Лепесток	Расстояние	Класс
Цветок 2	6,1	4,8	-	-
A	6,1	4,7	0,14	Iris Versicolour
B	6	4,8	0,1	Iris Virginica
C	6,2	4,8	0,1	Iris Virginica

Класс *цветка 2*: *Iris Virginica*

Машинное обучение и признаки

- Существуют два множества:
  - Множество *объектов – образов X*
  - Множество *ответов Y*
- Существует целевая функция $y^* : X \to Y$ значения которой известны только на конечном подмножестве объектов $\{x_1, \ldots, x_n\} \subset X$
- Пары «объект-ответ» :
- (x_i, y_i) называются прецедентами.
- Совокупность пар
- $X^N = (x_i, y_i)_{i=1}^N$ называется обучающей выборкой.
- Требуется построить отображение (гипотезу)

  $a : X \to Y; \quad a(x) = y \in Y$
- Признак – это результат измерения некоторой отличительная характеристики объекта, качественной или количественной.
- Признаком называется отображение $f : X \to D_f$

Основные проблемы построения признаков

- Как определять признаки : размышления, эксперименты, консультации и т.п.
- Как искать закономерности : перебором подмножеств признаков.
- Как объединять закономерности в алгоритм: любым классификатором.
- Как определять информативность?

Критерий информативности

Построим предикат $R : X \to \{0,1\}$ такой, что он:

1) удовлетворяет требованию **интерпретируемости:**
   - записывается на естественном языке;
   - зависит от небольшого числа признаков.

2) удовлетворяет требованию **информативности:** выделяет больше объектов ***«своего»*** класса, по сравнению с объектами всех остальных ***«чужих»*** классов.

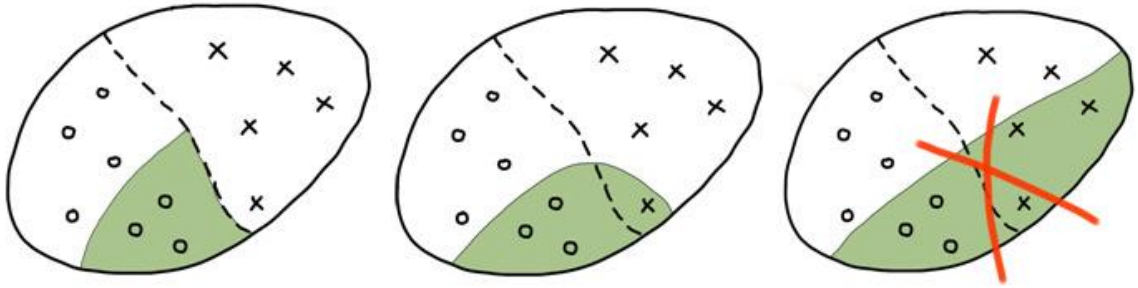
- Предикат *R* выделяет *X*, если *R(X)=1*.
- **Информативность** для некоторого класса C определяется как:

$$p_c(R) = P\{X_i : R(X_i) = 1 \wedge Y_i = C\} \to \max$$

- ***«Свои»*** объекты называются также позитивными ***(positive)***, а ***«чужие»*** – негативными ***(negative),*** поэтому для чужих верно:

$$n_c(R) = P\{X_i : R(X_i) = 1 \wedge Y_i \neq C\} \to \min$$

Критерий информативности



Как свернуть два разных критерия определения своих и чужих в один критерий информативности?

$$\begin{cases} p(R) \to \max \\ n(R) \to \min \end{cases} \Rightarrow L(p,n) \to \max$$

Наименее интересны те предикаты, которые либо выделяют слишком мало объектов, либо выделяют позитивные и негативные объекты примерно в той же пропорции, в которой они были представлены во всей выборке.

Примеры критериев информативности

- Precision $\frac{p}{p+n} \to \max$
- Accuracy $\frac{p}{p-n} \to \max$
- Cost accuracy $p - C(n) \to \max$
- Relative accuracy $\frac{p}{P} - \frac{n}{N} \to \max$
- Критерий бустинга $\sqrt{p} - \sqrt{n} \to \max$
- Нормированный критерий бустинга $\sqrt{\frac{p}{P}} - \sqrt{\frac{n}{N}} \to \max$

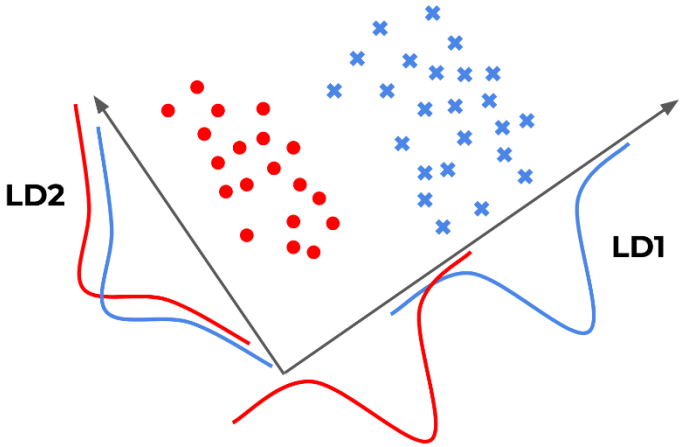
P – число “своих” во всей выборке;
N – число “чужих” во всей выборке

Признаки и разбиение пространства

- Задача классификации: определить вектор x в один из K классов Y .
- Ищем линейную дискриминантную функцию определяющую разделяющую поверхность между классами

$$y(x) = w^T x + w_0 \ .$$

- Классификация – это ***метод построения информативных признаков***.



Проблемы снижения размерности пространства признаков

Причины необходимости снижения размерности:

1. Наглядное представление данных (p = 1,2,3).
2. Лаконизм моделей, упрощение интерпретации.
3. Увеличение точности выводов, зависящей от n / (p+1).
4. Борьба с мультиколлинеарностью – взаимозависимостью.

Требования к новым показателям:

1. Максимальная информативность.
2. Взаимная некоррелированность.
3. Минимальное искажение структуры исходных данных.

Ситуации, в которых снижение размерности осуществить легко:

1. Дублирование информации (исключение).
2. Наличие неинформативных переменных (исключение).
3. Наличие однотипных переменных (агрегирование).

Метод главных компонент

Метод главных компонент – это статистическая процедура, позволяющая решить задачу оптимального уменьшения объёма исходных многомерных данных путём перехода к новым переменным (**главным компонентам или факторам**), являющимся комбинациями исходных наблюдаемых переменных.

Метод главных компонент – это алгебраическая процедура оптимального линейного сокращения размерности:

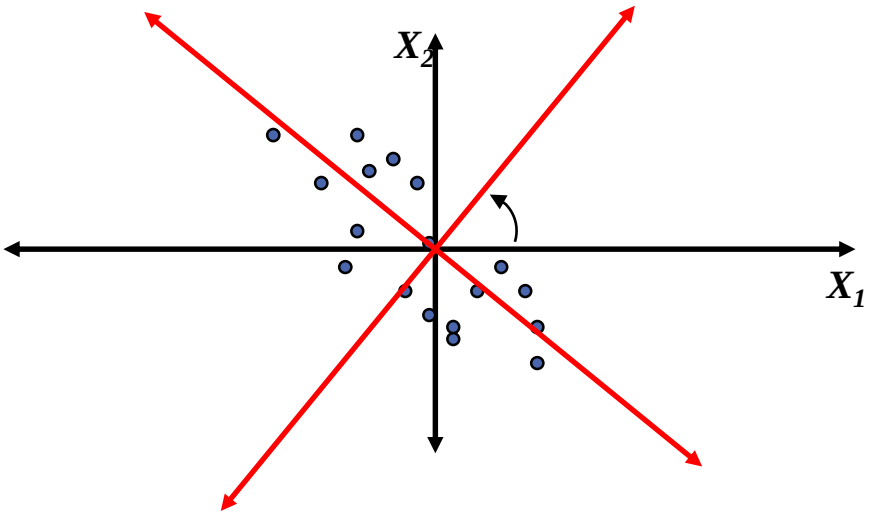
- ✓ Построение проекции (на пространство заданной размерности), в которой максимизируется дисперсия;
- ✓ Построение проекции минимальной цены (т.е. минимального суммарного расстояния до проекций всех точек).

Преобразование осей координат

Предположим, что мы имеем выборку, измеренную на случайных величинах X_1, ..., X_p.

Заметим, что эти случайные величины представляют собой оси декартовой системы координат.

Наша цель - разработать новый набор осей (линейных комбинаций исходных осей в направлениях наибольшей изменчивости:



- ✓ Преобразование исходных данных сводится к переносу и вращению системы координат в признаковом пространстве.
- ✓ Начало координат переносится в центр тяжести облака.
- ✓ Осуществляется поворот таким образом, чтобы оси многомерного эллипсоида совпали с осями координат.
- ✓ Оси эллипсоида ранжируются по длине, и та координатная ось, которая совпадает с наиболее длинной осью эллипсоида, называется первой, следующая по длине – второй и т.д.

Основы метода главных компонент

Преобразование Карунена-Лоэва – предложено независимо финским математиком Каруненом (Kari Karhunen) в 1946 и французским математиком Лоэвом (Michel Loève) в 1955. Метод известен под различными названиями, также как «разложение Карунена-Лоэва»

В дискретном случае оно диагонализирует матрицу некоторой заданной квадратичной формы K.

1. Подготовительный этап

1) Центрирование и нормирование переменных – переход к $\left(x_i^{(j)} - \bar{x}^{(j)}\right) \Big/ \sqrt{\sigma_j}$

2) Вычисление матрицы ковариаций

$$\sum = \begin{pmatrix} \hat{\sigma}_{11} & \cdots & \hat{\sigma}_{1p} \\ \cdots & \cdots & \cdots \\ \hat{\sigma}_{p1} & \cdots & \hat{\sigma}_{pp} \end{pmatrix}, \quad \hat{\sigma}_{kj} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i^{(k)} - \bar{x}^{(k)}\right)\left(x_i^{(j)} - \bar{x}^{(j)}\right) = KOBAP\left(x_1^{(k)},\ldots,x_n^{(k)};\ x_1^{(j)},\ldots,x_n^{(j)}\right).$$

2. Решение характеристического уравнения $\left|\sum - \lambda E\right| = 0$

1) Нахождение собственных чисел

2) Нахождение собственного вектора p' для каждого корня λ_k

$$\left(\sum - \lambda_k E\right) l^{(k)} = 0, \qquad \left\| l^{(k)} \right\| = 1.$$

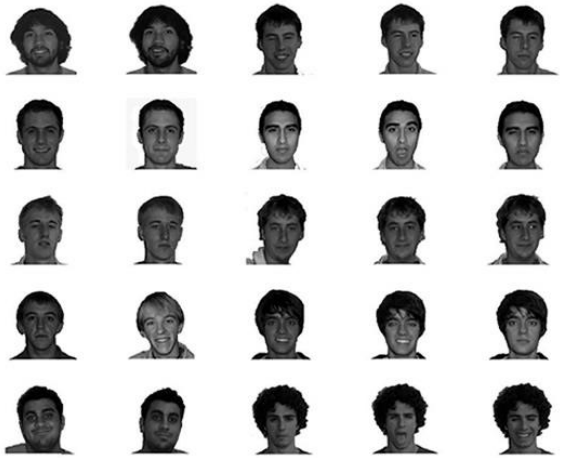
3. Переход к новым переменным

$z^{(k)} = X l^{(k)},\ k = 1,\ldots, p'$ – новые переменные, «главные компоненты»

$I_{p©} = \dfrac{\lambda_1 + \ldots + \lambda_{p'}}{\lambda_1 + \ldots + \lambda_p}$ – доля дисперсии, вносимая первыми главными компонентами

Метод собственных лиц

- ✓ Позволяет выделять характерные признаки лица и использовать их для реконструкции и восстановления;
- ✓ Главная идея этого метода состоит в представлении изображений лиц людей в виде набора главных компонент изображений, называемых «собственные лица».



The average face and first four eigenfaces

Eigenfaces 15, 100, 200, 250, 300

Дискриминантный анализ

Дискриминантный анализ – это общий термин, относящийся к нескольким тесно связанным статистическим процедурам решения задач различения (дискриминации) объектов наблюдения по набору признаков.

- Линейный ДА пытается выразить какую-либо зависимую переменную через линейную комбинацию других признаков или измерений.
- Как в дисперсионном и регрессионном анализе зависимая переменная — численная величина, является величиной номинальной (меткой класса).
- Помимо того, ЛДА имеет схожие черты с методом главных компонент, который ищет линейные комбинации величин, наилучшим образом описывающие данные.

Основной целью дискриминации является поиск линейной комбинации признаков (называемых дискриминантными признаками), которые позволили бы наилучшим образом разделить рассматриваемые группы.

Задачи дискриминантного анализа

- Определение дискриминирующих признаков (т.е. признаков, которые позволяют отнести наблюдение к той или иной группе);
- Построение дискриминирующей функции;
- Прогнозирование будущих событий, связанных с попаданием объекта в ту или иную группу на основе значений его признака (например, предсказание выживаемости пациента после операции).

Методы интерпретации межгрупповых различий (дискриминации) позволяют ответить на следующие вопросы:

- ✓ Возможно ли, используя данный набор переменных, отличить одну группу от другой?
- ✓ Насколько хорошо эти переменные позволяют провести дискриминацию?
- ✓ Какие из них наиболее информативны, а какие могут быть удалены из пространства признаков в связи с избыточностью?

Линейный дискриминант Фишера

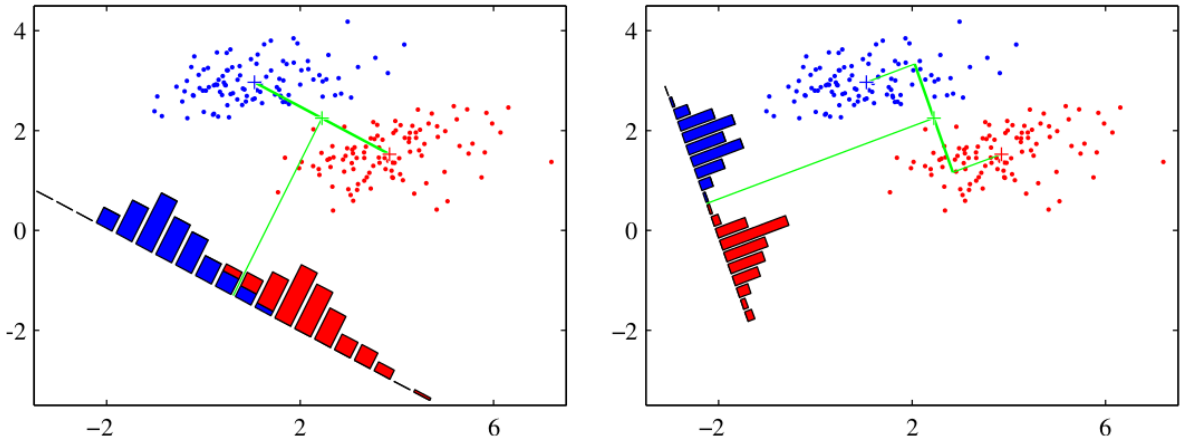
Другой взгляд на классификацию, предложенный Фишером:

- ✓ В линейном случае следует спроецировать точки в размерность 1 (на нормаль разделяющей гиперплоскости) так, чтобы в этой размерности 1 они «хорошо» разделялись.
- ✓ То есть классификация – это такой ***метод радикального сокращения размерности***.

Рассмотрим классификацию с этих позиций и попробуем добиться оптимальности в каком-то смысле.

- ➢ Рассмотрим два класса C_1 и C_2 с N_1 и N_2 точками.
- ➢ **Идея 1** – найдём серединный перпендикуляр между центрами кластеров:
- ➢ $$m_1 = \frac{1}{N_1}\sum_{C_1} x \quad , \quad m_2 = \frac{1}{N_2}\sum_{C_2} x \, .$$
- ➢ Будем максимизировать : $w^T (m_2 - m_1) \longrightarrow \max, \|w\| = 1$

Линейный дискриминант Фишера



- Чем отличается левый случай от правого?
- Слева больше дисперсия каждого кластера.
- **Идея 2:** минимизировать перекрытие классов, оптимизируя и проекцию расстояния, и дисперсию.

Линейный дискриминант Фишера

- Оценим выборочные дисперсии для проекции $y_n = w^T x_n$

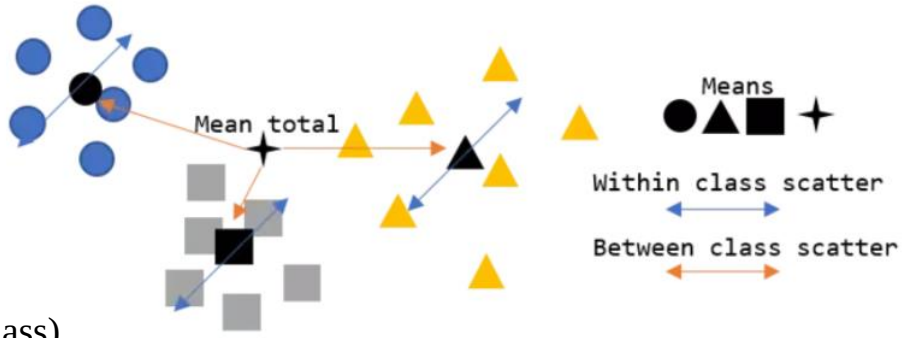
$$s_1 = \sum_{C_1} (y_n - m_1)^2, \quad s_2 = \sum_{C_2} (y_n - m_2)^2$$

- Критерий Фишера

$$J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{w^T S_B w}{w^T S_W w}$$

$$S_B = (m_2 - m_1)(m_2 - m_1)^T$$

– ковариация между классами (between-class)



$$S_W = \sum_{C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{C_2} (x_n - m_2)(x_n - m_2)^T$$

– ковариация внутри классов (within-class)

Линейный дискриминант Фишера

- Критерий Фишера максимален, когда $\left(w^T S_B w\right) S_W w = \left(w^T S_W w\right) S_B w$
- получим: $w \propto S_w^{-1}(m_2 - m_1)$
- Для случая нескольких классов $J(w) = \left\| S_W^{-1} S_B \right\|$

$S_W = \sum_{k=1}^{K} \sum_{C_k} (x_n - m_1)(x_n - m_1)^T$ – внутренняя дисперсия

$S_B = S_T - S_W$ – внешняя (межклассовая) дисперсия

$S_T = \sum_{n} (x_n - m)(m_n - m)^T$ – полная дисперсия

Дискриминантный анализ

Для выбранных переменных рассчитывается новая переменная Z (дискриминантная функция) – линейная комбинация исходных переменных, которая наилучшим образом разделит группы.

- Матрицы рассеяния внутри классов $S_W = \sum_{k=1}^{K} \sum_{x_n \in X_k} (x_n - m_k)(x_n - m_k)^T$
- Матрица рассеяния между классами $S_T = \sum_{x_n \in \bigcup X_k} (x_n - m)(x_n - m)^T$
- Матрица рассеяния смеси $S_B = S_T - S_W$
- Критерии разделимости

$$J_2 = \ln |S_w^{-1}(S_w + S_b)| = \ln\{|S_w + S_b| / |S_w|\}$$

$$J_3 = trS_b / tr(S_w + S_b)$$

$$J_1 = tr((S_w + S_b)^{-1} S_b)$$

Если группы две: получается одно уравнение $Z = \lambda_0 + \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \ldots$

Когда групп много, получают **несколько дискриминантных функций**, «перпендикулярных» друг другу. Чем больше коэффициент при переменной, тем лучше она разделяет группы, но не говорит, какие именно.

Особенности дискриминантного анализа

Требования к обучающим выборкам:

1. Внутри классов должно быть многомерное *нормальное распределение* (оценка – на основе построения гистограмм частот);

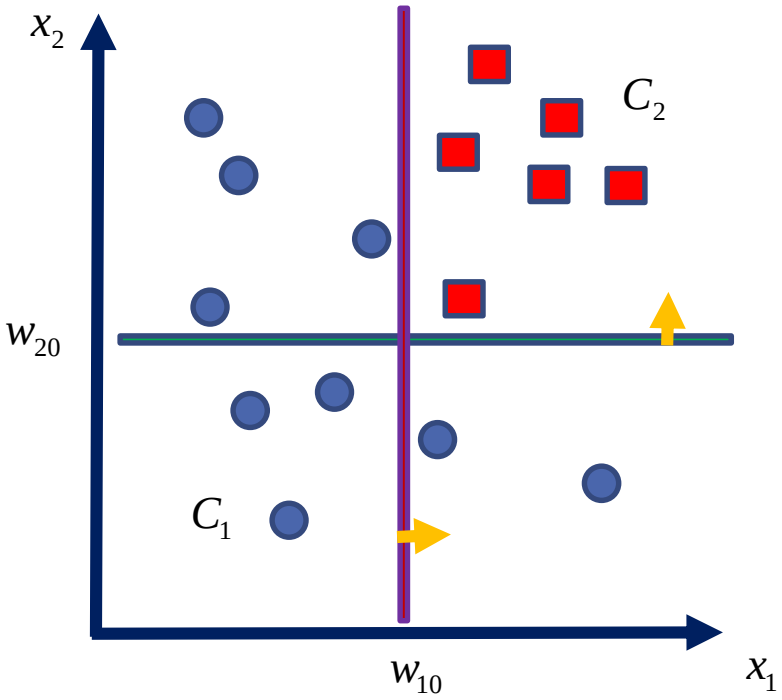
2. Гомогенность внутриклассовых *дисперсий* (не очень критичное требование);

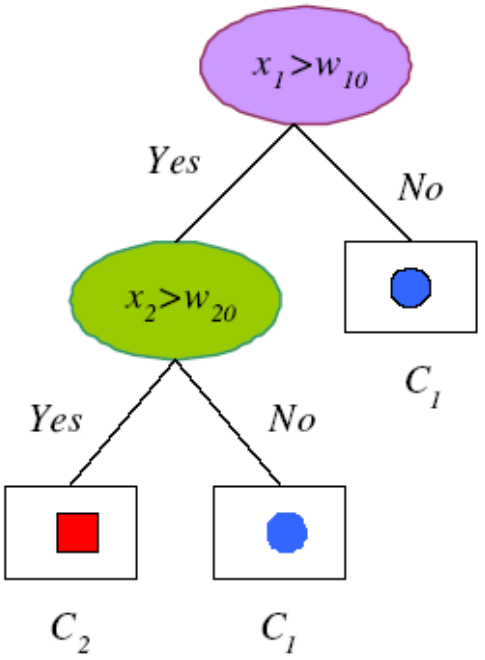
3. Не должно быть корреляции *средних* значений и *дисперсий* в классах;

4. Не должно быть чрезмерно коррелирующих друг с другом переменных

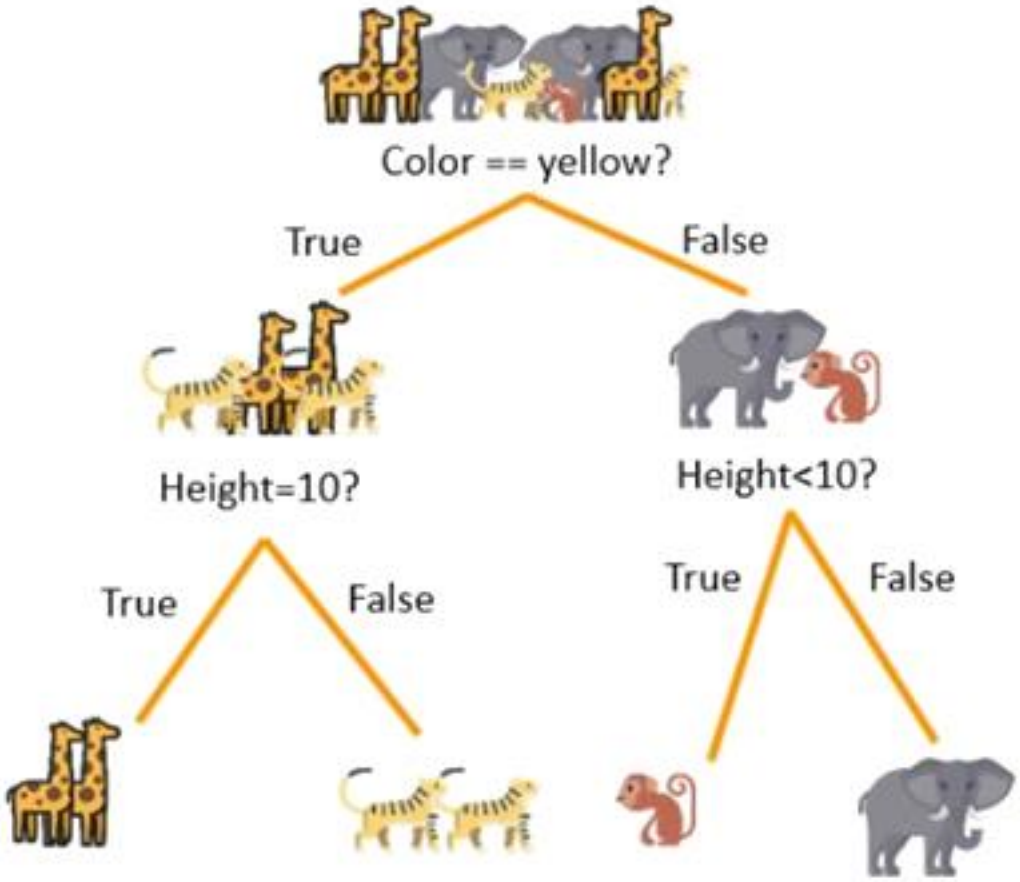
Дискриминантная функция рассчитывается только для тех измерений, для которых *известно, к какому классу они принадлежат* (т.е., только для тех образов, для которых класс известен).

Деревья решений





Деревья решений



Деревья решений

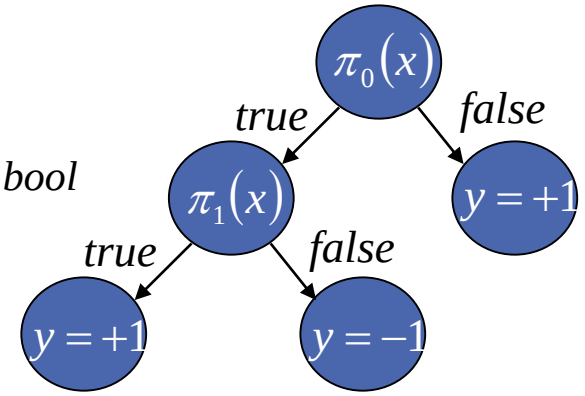
- Деревья решений – *способ представления правил в иерархической, последовательной структуре*.
- Основополагающая работа – книга Ханта (Hunt, E.B.), Мэрина (Marin J.) и Стоуна (Stone, P.J) "Experiments in Induction", увидевшая свет в 1966г.
- Метод также называют деревьями решающих правил, деревьями классификации и регрессии.
- Определение:
- **Дерево решений** – представленный в виде связного ациклического графа план, при помощи которого оценивается значение целевого атрибута объекта по набору значений независимых атрибутов.

Деревья решений

- ✓ Двоичное дерево.
- ✓ Узлы – помечены некоторым предикатом $\pi : X \to bool$
- ✓ Обычно, используются пороги по проекциям на оси координат (на элементы вектора признаков)

$$\pi\left(x = (x^0, \ldots x^m)\right) = \begin{cases} true, & x^j < \Theta \\ false, & x^j \geq \Theta \end{cases}$$

- ✓ Связи – помечены $\{true \quad false\}$.
- ✓ Листья – помечены ответами из *Y*.



- ➢ **Внутренний узел** представляет разбиение множества возможных значений той или иной независимой переменной
- ➢ Атрибуты, соответствующие внутренним узлам дерева – **атрибуты расщепления** (прогнозирующие атрибуты).
- ➢ Каждая ветвь от внутреннего узла отмечается **предикатом расщепления**.
- ➢ Информация об атрибутах и предикатах расщепления в узле – **критерий расщепления**.

Преимущества:

- ❖ **Интуитивность деревьев решений**.
- ❖ Возможность извлекать правила из базы данных на **естественном языке.**
- ❖ **Не требует от пользователя выбора входных атрибутов.**
- ❖ **Точность** моделей.
- ❖ Разработан ряд **масштабируемых алгоритмов.**
- ❖ **Быстрый процесс обучения** .
- ❖ Обработка **пропущенных значений.**
- ❖ Работа и с числовыми, и с **категориальными** типами данных .

Деревья решений

Модель алгоритма обучения

Дана обучающая выборка

1. Выбираем целевой атрибут.
2. Выбираем критерий расщепления.
3. Разделяем обучающую выборку.
4. Исключаем атрибут расщепления из выборки.
5. Для всех полученных подвыборок переходим на шаг 2.

Модель классификатора

- Выходом классификатора $a(x)$ является значение y листа, полученного при обходе:
  - Начинаем от корня.
  - Переходим в тот узел, в который ведёт связь помеченная значением предиката в текущем узле.
  - Заканчиваем, попав в лист.

Лист определяет собой область пространства *X*

Если зависимая переменная принимает дискретные значения – задача классификации.
Если непрерывные – задачу регрессии (численного прогнозирования).

Построение дерева решений

- Процесс заключается в последовательном, рекурсивном разбиении обучающего множества на подмножества с применением решающих правил в узлах.
  - Выбираем очередной атрибут ***Q***, помещаем его в корень.
  - Для всех его значений ***i***:
    - Оставляем из тестовых примеров только те, у которых значение атрибута ***Q*** равно ***i***
    - Рекурсивно строим дерево в этом потомке.
- Процесс разбиения продолжается до тех пор, пока все узлы в конце всех ветвей не будут объявлены листьями.
  - Объявление узла листом может произойти естественным образом (когда он будет содержать единственный объект, или объекты только одного класса), или по достижении некоторого условия остановки, задаваемого пользователем (например, минимально допустимое число примеров в узле или максимальная глубина дерева).

Особенности построения деревьев решений

- Алгоритмы построения деревьев решений относят к категории так называемых жадных алгоритмов.
  - Жадными называются алгоритмы, которые допускают, что локально-оптимальные решения на каждом шаге (разбиения в узлах), приводят к оптимальному итоговому решению.
  - В случае деревьев решений это означает, что если один раз был выбран атрибут, и по нему было произведено разбиение на подмножества, то алгоритм не может вернуться назад и выбрать другой атрибут, который дал бы лучшее итоговое разбиение. Поэтому на этапе построения нельзя сказать обеспечит ли выбранный атрибут, в конечном итоге, оптимальное разбиение.

Основные этапы:

- Выбор атрибута, по которому будет производиться разбиение в данном узле (атрибута разбиения).
- Выбор критерия остановки обучения.
- Выбор метода отсечения ветвей (упрощения).
- Оценка точности построенного дерева.

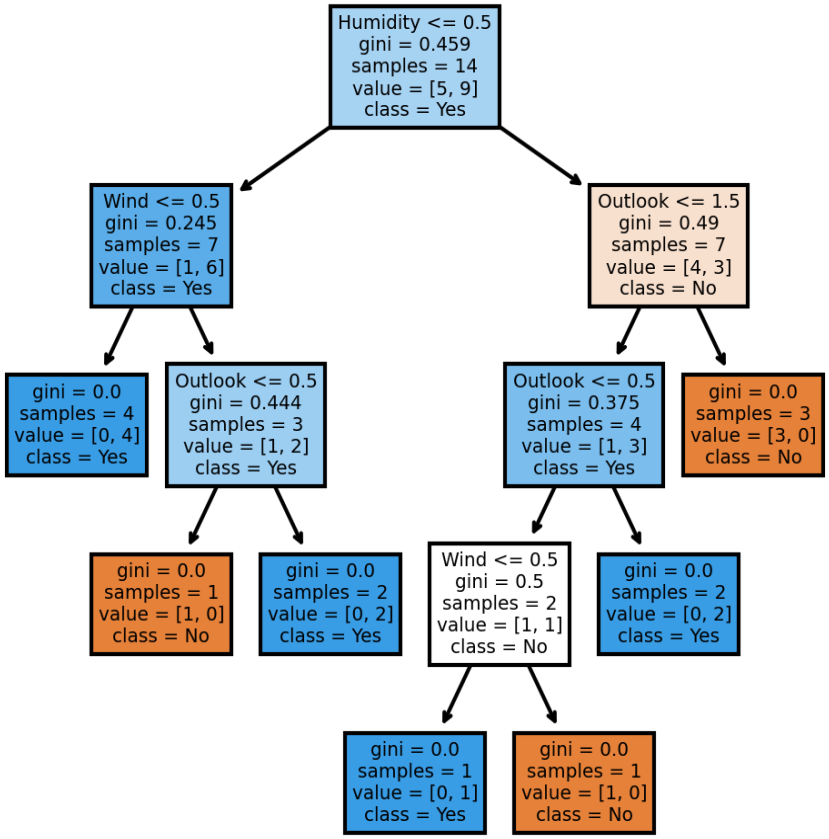
Алгоритмы обучения

- **ID3 (Iterative Dichotomizer 3)** — алгоритм позволяет работать только с дискретной целевой переменной, поэтому деревья решений, построенные с помощью данного алгоритма, являются классифицирующими. Число потомков в узле дерева не ограничено. Не может работать с пропущенными данными.
- **C4.5** — усовершенствованная версия алгоритма ID3, в которую добавлена возможность работы с пропущенными значениями атрибутов *по версии издания Springer Science в 2008 году алгоритм занял 1-е место в топ-10 наиболее популярных алгоритмов Data Mining*.
- **CART (Classification and Regression Tree)** — алгоритм обучения деревьев решений, позволяющий использовать как дискретную, так и непрерывную целевую переменную, то есть решать как задачи классификации, так и регрессии. Алгоритм строит деревья, которые в каждом узле имеют только два потомка.

Деревья решений. Пример.

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



x=(Sunny, Cool, High, Strong)

Выбор атрибута: мера энтропии

- Предположим, что имеется множество ***A*** из ***n*** элементов, ***m*** из которых обладают некоторым свойством ***S***. Свойство ***S*** не бинарное и может принимать ***s*** различных значений.
- Тогда «неоднородность» ***A*** по отношению к множеству ***S*** выражается как:

$$H(A,S) = -\sum_{i=1}^{s} \frac{m_i}{n} \log \frac{m_i}{n}$$

$H(Y \mid x) = -\sum_{y_i \in Y} p(y_i) \log p(y_i \mid x)$ — условная энтропия:

- Энтропия — это среднее количество битов, которые требуются, чтобы закодировать атрибут ***S*** у элемента множества ***A***.
- Если вероятность появления ***S*** равна ***1/2***, то энтропия равна ***1***, и нужен полноценный бит; а если ***S*** появляется не равновероятно, то можно закодировать последовательность элементов ***A*** более эффективно.

Выбор атрибута: прирост информации

- Предположим, что множество ***A*** элементов, характеризующихся свойством ***S***, классифицировано посредством атрибута ***Q***, имеющего ***q*** возможных значений.
- A_i – множество элементов из ***A*** , на которых атрибут ***Q*** имеет значение ***i.***
- Тогда «прирост информации» выражается как:

$$Gain(A) = H(A,S) - \sum_{i=1}^{q} \frac{|A_i|}{|A|} H(A_i,S)$$

- **Атрибут для классификации нужно выбирать так чтобы, после классификации энтропия (относительно целевой функции) стала как можно меньше.**
- Таким образом, задача выбора атрибута разбиения в узле заключается в максимизации величины *Gain(A)*, (от англ. gain — прирост, увеличение). Этот теоретико-информационный подход известен как критерий прироста информации.

Выбор атрибута. Статистический подход.

Индекс Джини *(назван в честь итальянского статистика и экономиста Коррадо Джини)*

- Статистический смысл данного показателя в том, что он показывает — насколько часто случайно выбранный пример обучающего множества будет распознан неправильно, при условии, что целевые значения в этом множестве были взяты из определённого статистического распределения.
- Индекс Джини фактически показывает расстояние между двумя распределениями — распределением целевых значений, и распределением предсказаний модели.
- Индекс Джини может быть рассчитан по формуле:

$$Gini(Q) = 1 - \sum_{i=1}^{n} p_i$$

где Q — результирующее множество, n — число классов в нём, p_i — вероятность i-го класса (выраженная как относительная частота примеров соответствующего класса).

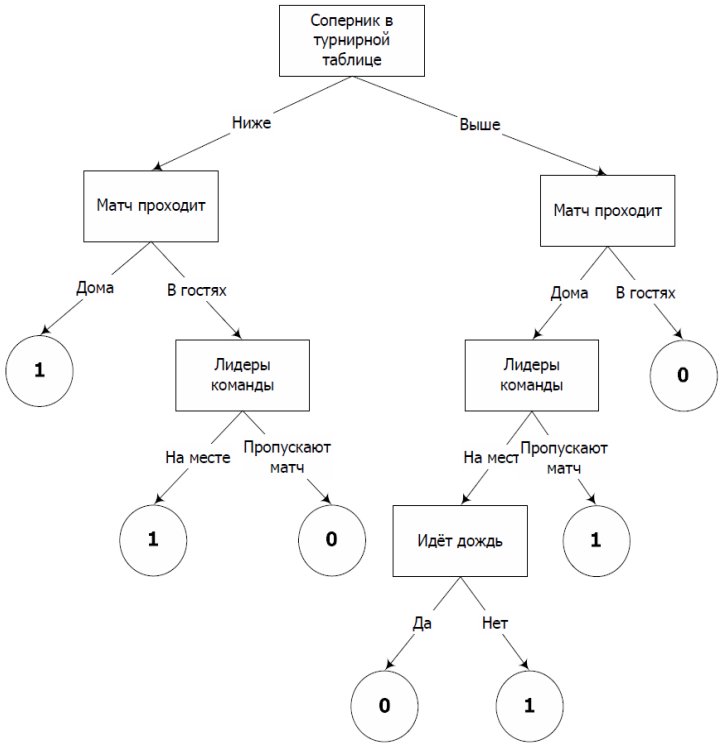
Очевидно, что данный показатель меняется от *0* до *1*. При этом он равен *0*, если все примеры *Q* относятся к одному классу, и равен *1*, когда классы представлены в равных пропорциях и равновероятны. Тогда лучшим будет то разбиение, для которого значение индекса Джини будут минимальным.

Критерии остановки алгоритма

- Если «рост» дерева не ограничить, то в результате будет построено сложное дерево с большим числом узлов и листьев.
  - Проблема переобучения — точное распознавание примеров, участвующих в обучении и полная несостоятельность на новых данных. Переобученные деревья имеют очень сложную структуру, и поэтому их сложно интерпретировать.
- Ранняя остановка — алгоритм будет остановлен, как только будет достигнуто заданное значение некоторого критерия, например процентной доли правильно распознанных примеров.
  - Единственным преимуществом подхода является снижение времени обучения. Главным недостатком является то, что ранняя остановка всегда делается в ущерб точности дерева, поэтому многие авторы рекомендуют отдавать предпочтение отсечению ветвей.
- Ограничение глубины дерева — задание максимального числа разбиений в ветвях, по достижении которого обучение останавливается.
  - Данный метод также ведёт к снижению точности дерева.
- Задание минимально допустимого число примеров в узле — запретить алгоритму создавать узлы с числом примеров меньше заданного (например, 5).
  - Это позволит избежать создания тривиальных разбиений и, соответственно, малозначимых правил.

Пример построения дерева

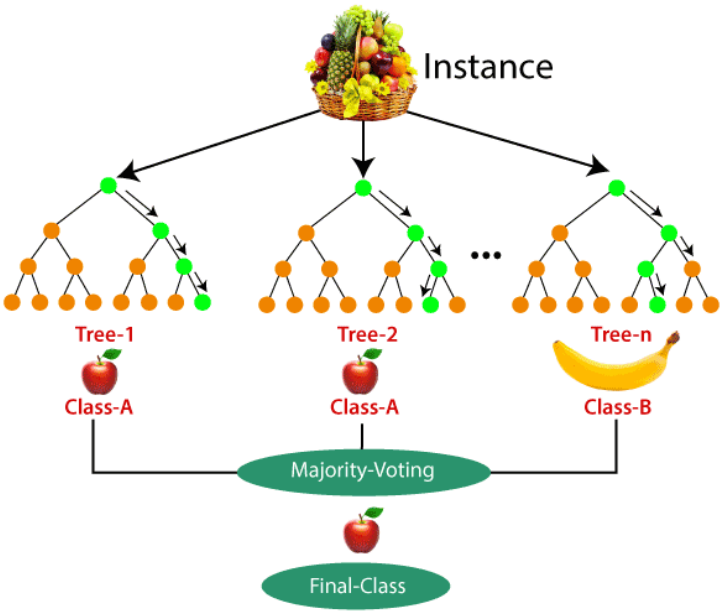
- Как сыграет команда?



Соперник	Играем	Лидеры	Дождь	Победа
Выше	Дома	На месте	Да	Нет
Выше	Дома	На месте	Нет	Да
Выше	Дома	Пропускают	Нет	Да
Ниже	Дома	Пропускают	Нет	Да
Ниже	В гостях	Пропускают	Нет	Нет
Ниже	Дома	Пропускают	Да	Да
Выше	В гостях	На месте	Да	Нет
Ниже	В гостях	На месте	Нет	*???*

Случайный лес

- Метод случайного леса (англ. **random forest**) — алгоритм машинного обучения (предложен в 2001 г.), сочетает в себе две основные идеи: метод бэггинга и метод случайных подпространств
- Метод основан на использовании большого **(комитета) ансамбля решающих деревьев**, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат улучшается.



Случайный лес

Алгоритм обучения:

- Обучающая выборка состоит из N образцов, размерность пространства признаков равна M ,
- Построение ансамбля деревьев:
  - Генерируется случайная подвыборка размером N из обучающей выборки. Некоторые образцы попадут в неё два или более раза, тогда как в среднем $N(1-\frac{1}{N})^N$ образцов не войдут в подвыборку.
  - Формируется решающее дерево, классифицирующее образцы данной подвыборки, причём в ходе создания очередного узла дерева случайным образом выбирается m признаков из всех M признаков. Выбор наилучшего из этих m признаков может осуществляться различными способами.
  - Дерево строится до полного исчерпания подвыборки и не подвергается процедуре прунинга (англ. Pruning) — отсечение ветвей),

Классификация объектов проводится путём голосования: каждое дерево комитета относит классифицируемый объект к одному из классов, а побеждает класс, за который проголосовало наибольшее число деревьев.

Оптимальное число деревьев подбирается таким образом, чтобы минимизировать ошибку классификатора на тестовой выборке. В случае её отсутствия, минимизируется оценка ошибки на не вошедших в набор образцах.

Характеристика деревьев решений

+ Просты в понимании и интерпретации.
+ Не требуют подготовки данных.
+ Используют модель «белого ящика».
+ Позволяют оценить модель при помощи статистических тестов.
+ Дают возможность извлекать из базы данных правила на естественном языке.
+ Позволяют создавать классификационные модели в тех областях, где аналитику достаточно сложно формализовать знания.
+ Алгоритм конструирования дерева решений не требует от пользователя выбора входных атрибутов.
+ Быстро обучаются.

– Проблема получения оптимального дерева решений бывает *NP*-полной.
– Могут появиться слишком сложные конструкции, которые при этом недостаточно полно представляют данные.
– Существуют концепты, которые сложно понять из модели, так как модель описывает их сложным путём.
– Для данных, которые включают категориальные переменные с большим набором уровней, больший информационный вес присваивается тем атрибутам, которые имеют большее количество уровней.

Характеристика случайного леса

+ Высокая точность предсказания, которая сравнима с результатами градиентного бустинга.
+ Эффективная обработка данных с большим числом признаков и классов.
+ Нечувствительность к масштабированию (и вообще к любым монотонным преобразованиям) значений признаков.
+ Использует методы оценивания значимости отдельных признаков в модели.
+ Внутренняя оценка способности модели к обобщению (тест по неотобранным образцам).
+ Высокая параллелизуемость и масштабируемость.

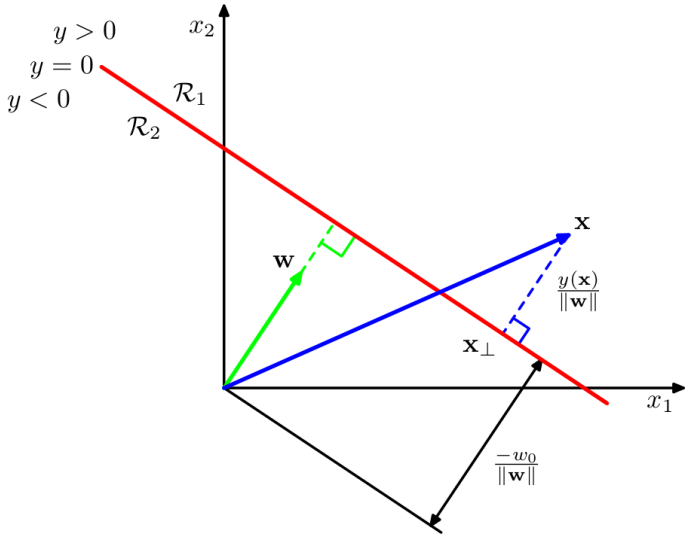
– Большой размер получающихся моделей.
– Очень трудоемкий процесс прогнозирования по сравнению с другими алгоритмами.
– Алгоритм склонен к переобучению на зашумленных данных.
– В отличие от более простых алгоритмов, результаты случайного леса сложнее интерпретировать.

Разбиение пространства и классификация.

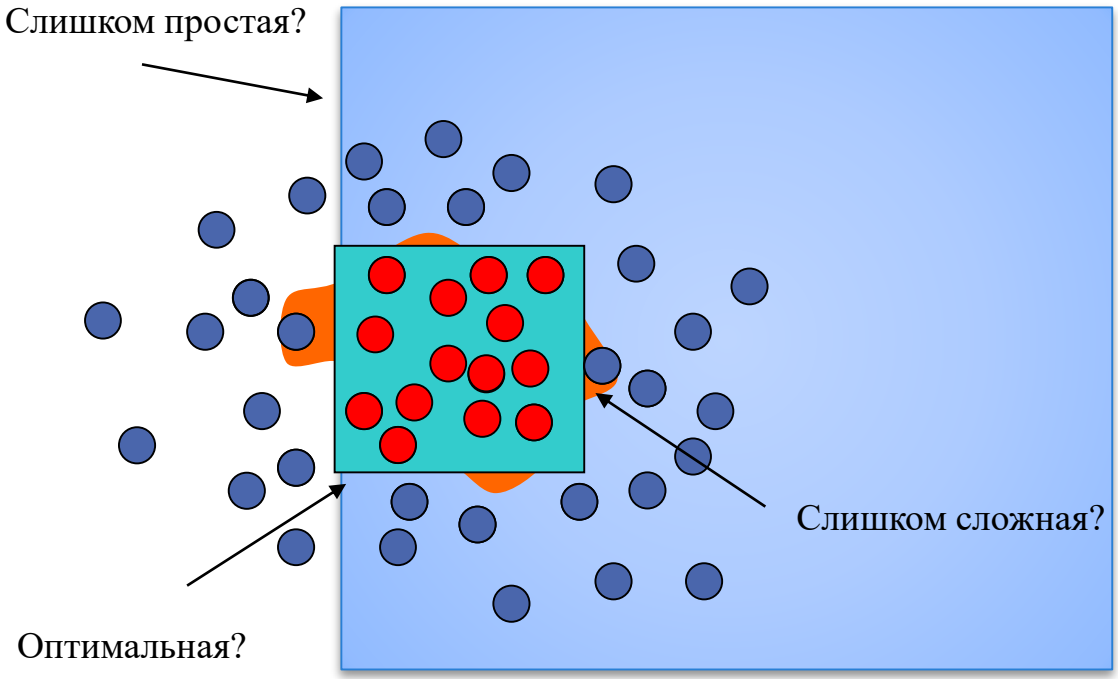
- Задача классификации: определить вектор ***x*** в один из классов ***Y*** .
- Рассматриваем линейную дискриминантную функцию:

$$y(x) = w^T x + w_0$$

- В линейном случае мы спроецировали все точки в размерность 1 (на нормаль разделяющей гиперплоскости) так, чтобы в этой размерности 1 они «хорошо» разделялись.



Иллюстрация



Состоятельность метода

- Метод состоятелен, если он с большой вероятностью делает маленькую ошибку на данных не присутствовавших в обучающей выборке;

  Мы говорим *«с большой вероятностью»* потому что точно мы не сможем сказать никогда.

- Метод обучения μ называется состоятельным, если для достаточно малых значений точности и больших значений надёжности:

Точность Надежность

$$P_{X^l,X}\left\{\underbrace{R(\mu(X^l),X)}_{\text{Общий риск}} < \varepsilon\right\} > \eta$$

Теория Вапника-Червоненкиса

- Как оценить состоятельность метода?
- Как выбрать оптимальный, с точки зрения общего риска, алгоритм?
- Теория, предложенная русским математиком Владимиром Вапником (конец 60-х), для оценки способности к обучению различных моделей алгоритмов
- Основные результаты теории:
  - Оценка сложности (ёмкости) модели алгоритма
  - Оценка состоятельности через эмпирический риск и сложность модели

Владимир Наумович Вапник (род. 6 декабря 1936),

Алексей Яковлевич Червоненкис (7 сентября 1938 — 22 сентября 2014)

Теория распознавания образов (статистические проблемы обучения),
В. Н. Вапник, А. Я. Червоненкис. Издательство «Наука»,
Главная редакция физико-математической литературы, М., 1974, 416 стр.

Принцип структурной минимизации риска

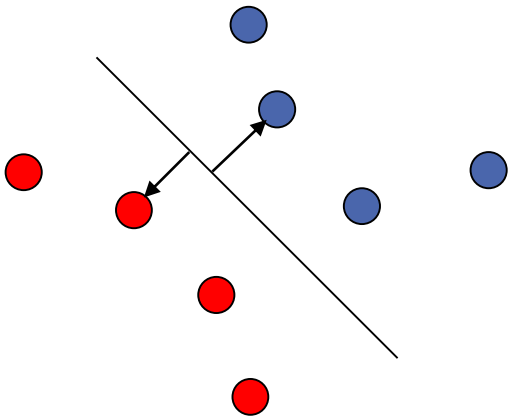
- Основная идея - «Выбрать модель наиболее простую из достаточно точных».
- Пусть есть семейство моделей A с выделенной последовательностью подсемейств возрастающей сложности:

$$A_1 \subset A_2 \subset \ldots \subset A_h = A$$

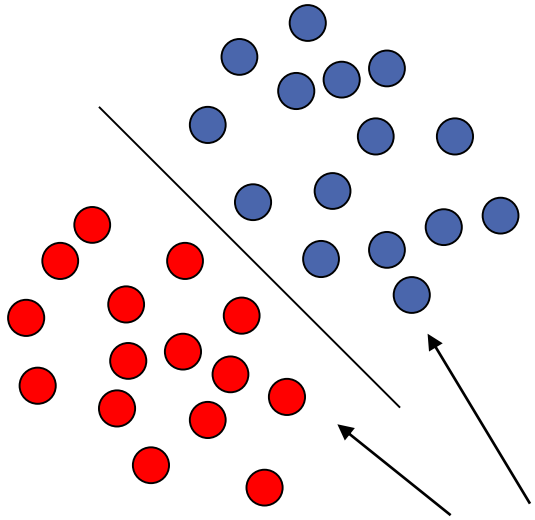
- Выберем семейство с минимальной сложностью, но обеспечивающее нужную точность.
- Требуется найти баланс между сложностью модели, обеспечивающей низкий эмпирический риск и простотой, обеспечивающей способность к обобщению.
- В случае бинарной классификации – каждый объект данных представляется как вектор (точка) в N-мерном пространстве, каждая из этих точек принадлежит только одному из двух классов.
- Типичный случай линейной разделимости – разделение точек гиперплоскостью размерности (N-1).

Максимизация отступа

1. Построим гиперплоскость, которая равно и максимально удалена от обоих классов?



2. Проверим изменится ли разделяющая поверхность если убрать какие-то объекты?



Прецеденты, которые нельзя убрать без изменения поверхности

Метод опорных векторов (SVM): оптимальная гиперплоскость

- Линейный дискриминант Фишера и МНК ищут разделяющую гиперплоскость, но не оптимально.
- Метод опорных векторов (SVM support vector machine) находит оптимальное решение.
  - Максимизирует расстояние между гиперплоскостью и трудными точками, близкими к границе раздела.
  - Интуитивно: если нет точек около границы раздела, то нет и сложных (неопределённых) примеров.
- Методы SVM принадлежит семейству линейных классификаторов и может также рассматриваться как специальный случай *регуляризации по Тихонову*.
  - Особым свойством метода опорных векторов является непрерывное уменьшение эмпирической ошибки классификации и увеличение зазора, поэтому метод также известен как метод ***классификатора с максимальным зазором***.

SVM: случай линейной разделимости

- Задача формулируется как поиск гиперплоскости максимально отдалённой от выпуклой оболочки классов.
- Такую гиперплоскость всегда можно записать в виде линейной комбинации «опорных векторов» - прецедентов, принадлежащих выпуклой оболочке.
- Задачу поиска такой гиперплоскости можно записать как задачу оптимизации:

$$w = \arg\max\left\{\frac{1}{\|w\|^2}\right\} = \arg\min\|w\|^2$$

$$y_i(w \circ x_i - b) \geq 1$$

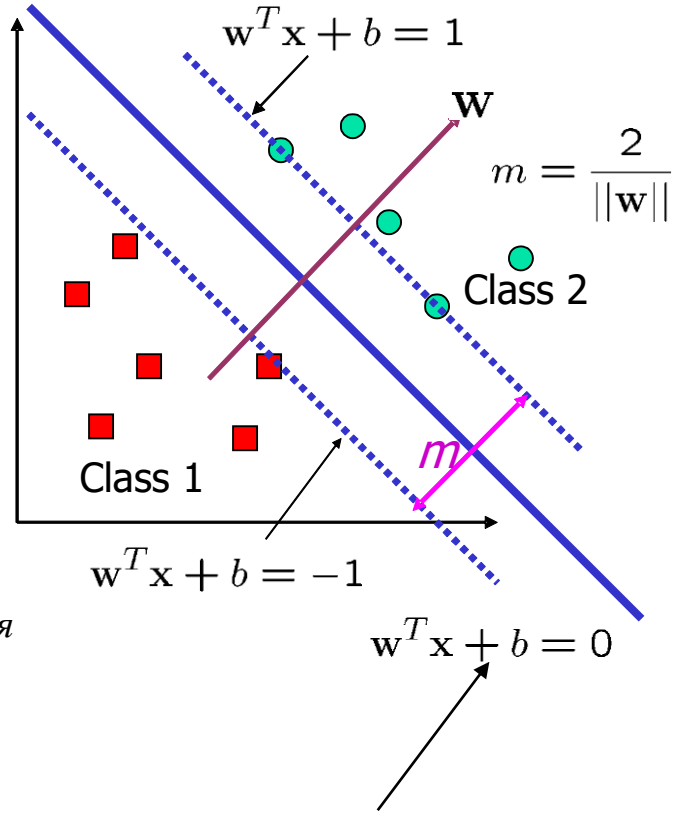
Глобальный минимум находится методом квадратичного программирования

Решение:

$$w = \sum_{i=1}^{n} \lambda_i y_i x_i,$$

$$b = w \circ x_i - y_i,$$

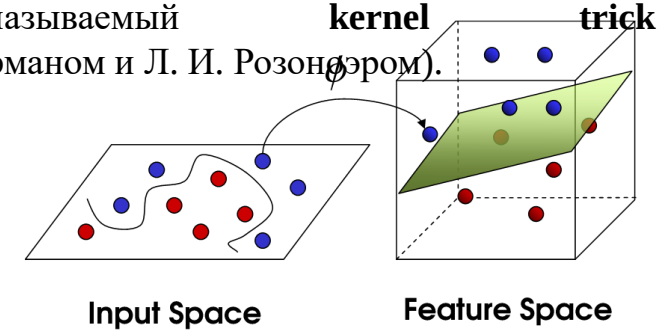
$\lambda_i > 0$ – *двойственные переменные*



Чисто геометрическая задача

Особенности SVM при отсутствии линейной разделимости

- Алгоритм построения оптимальной разделяющей гиперплоскости, предложенный в 1963 году Владимиром Вапником и Алексеем Червоненкисом — **алгоритм линейной классификации**.
- Однако в 1992 году Бернхард Босер, Изабель Гийон и Вапник предложили способ создания нелинейного классификатора, в основе которого лежит переход от скалярных произведений к произвольным ядрам, так называемый **kernel trick** (предложен впервые М. А. Айзерманом, Э. М. Браверманом и Л. И. Розоноэром).



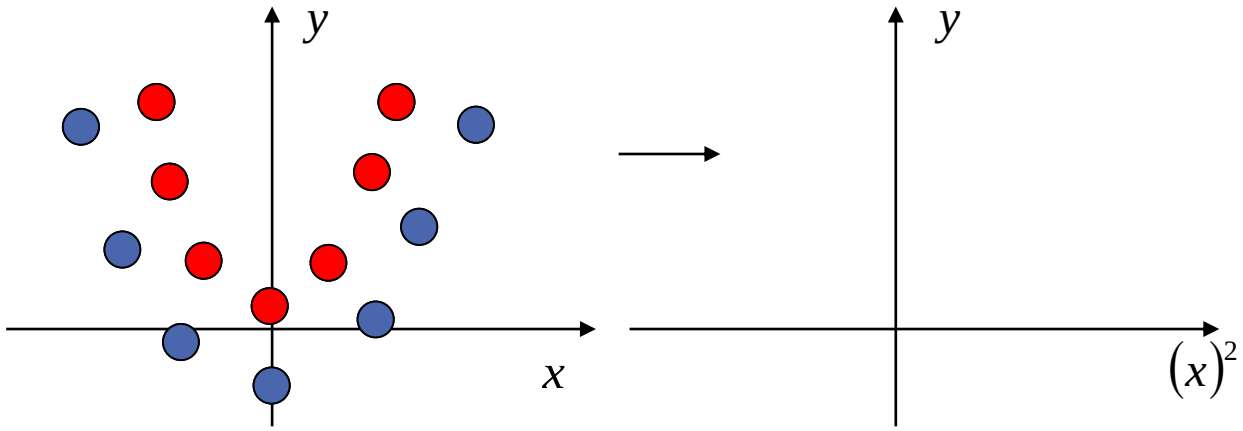
- ✓ Результирующий алгоритм крайне похож на алгоритм линейной классификации, с той лишь разницей, что каждое скалярное произведение в приведённых выше формулах заменяется **нелинейной функцией ядра** (скалярным произведением в пространстве с большей размерностью).
- ✓ Ядровые преобразования можно быстро вычислять для скалярных произведений, даже при бесконечной размерности целевого пространства

$$u \circ v = (u \cdot v')^d$$

$$u \circ v = \exp\left(-\gamma \left\| u \cdot v \right\|^2\right), \gamma > 0$$

$$u \circ v = \exp\left(-\frac{\left\| u - v \right\|^2}{2\sigma^2}\right)$$

SVM: случай нелинейной разделимости



- ✓ Допустим, примеры живут на плоскости *(x, y)*, а граница между положительными и отрицательными областями — это парабола $y=x^2$.
- ✓ Добавим третью координату *z* и установим координату *z* каждого примера равной квадрату его координаты *x*,
- ✓ Данные окажутся в пространстве *(x, y, z)* граница в котором будет просто диагональной плоскостью $y=z$.

Характеристика SVM

+ Однозначность решения (глобальный минимум).
+ Теоретически доказана минимальность оценки общего риска (Вапник).
+ При помощи ядровых преобразований обобщается на случай сложных поверхностей.

– Требует подбора множества параметров (ядро, параметры ядра, штраф за ошибки).
– Теоретические оценки верны только для случая разделимости (схема со штрафами - эвристика).
– Очень чувствителен к нормализации данных!