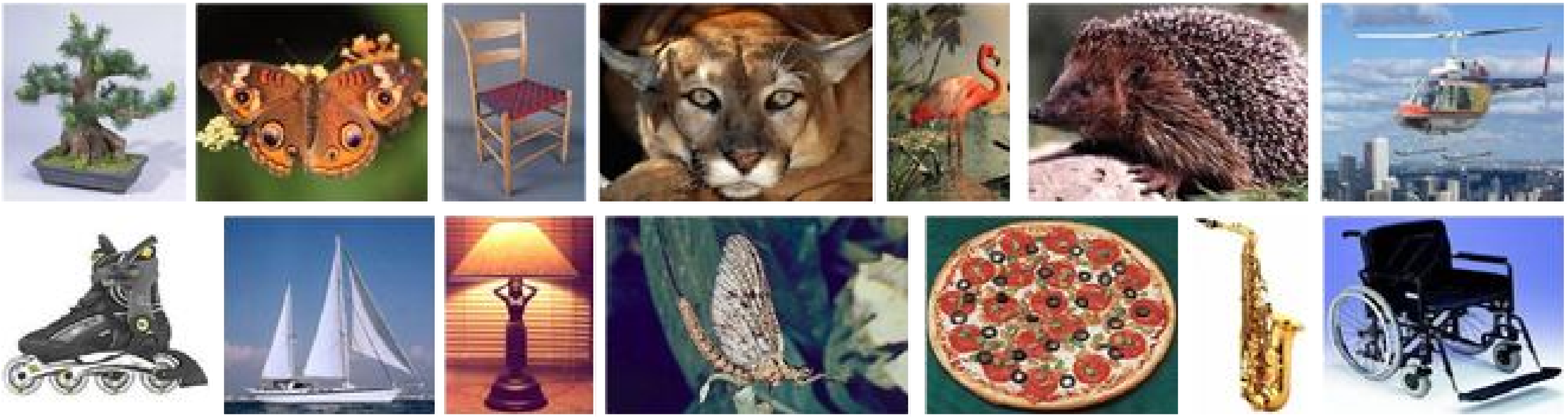
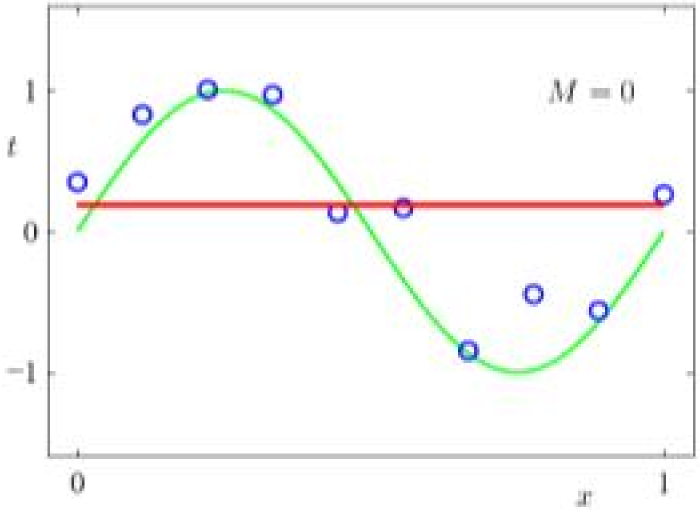
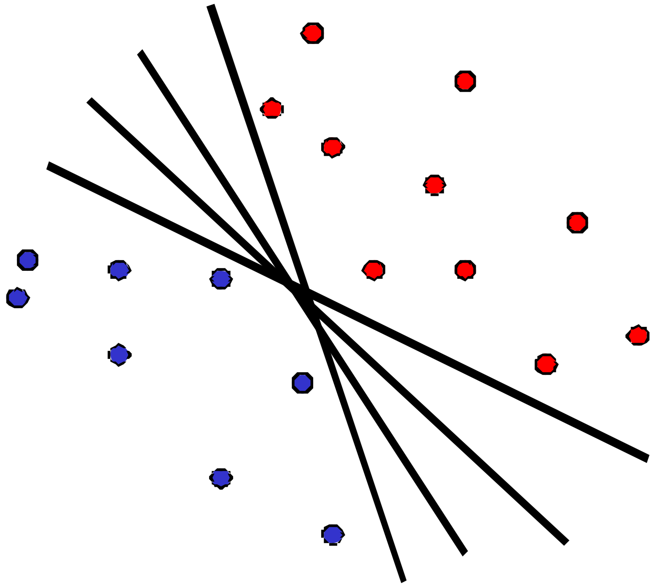
Распознавание изображений







Slides adapted from Fei-Fei Li, Rob Fergus, Antonio Torralba, Jean Ponce and Svetlana Lazebnik

План лекции

- Введение в классификацию образов с помощью машинного обучения
  - Некоторые общие понятия
  - Один из базовых методов (SVM)
  - Практические аспекты машинного обучения
- Выделение категорий объектов на изображении
  - Более сложные признаки (HOG)
  - Скользящее окно
  - Выделение пешеходов и «торса»

План лекции

- Введение в классификацию образов с помощью машинного обучения
  - Некоторые общие понятия
  - Один из базовых методов (SVM)
  - Практические аспекты машинного обучения

- Выделение категорий объектов на изображении
  - Более сложные признаки (HOG)
  - Скользящее окно
  - Выделение пешеходов и «торса»

На прошлой лекции

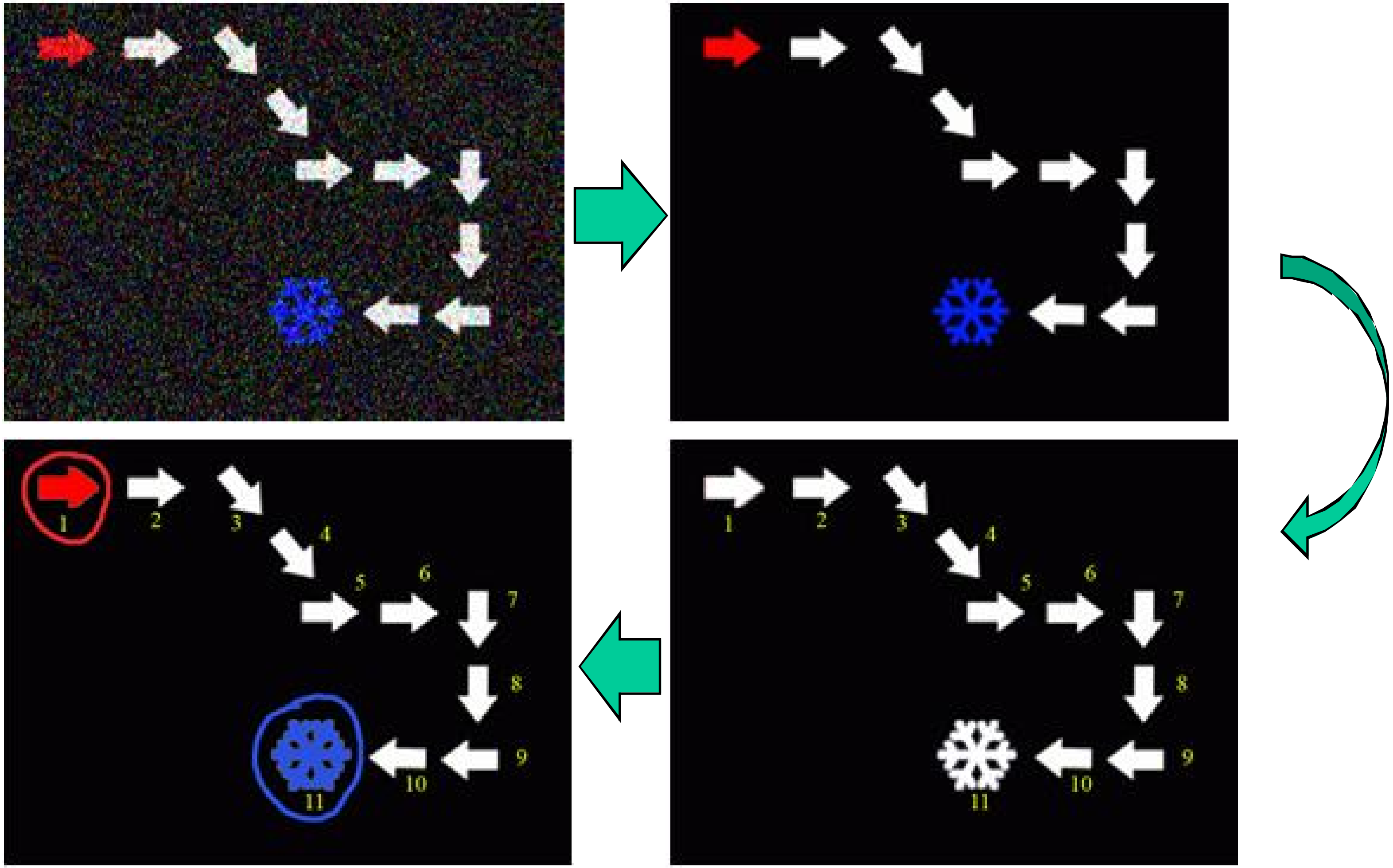
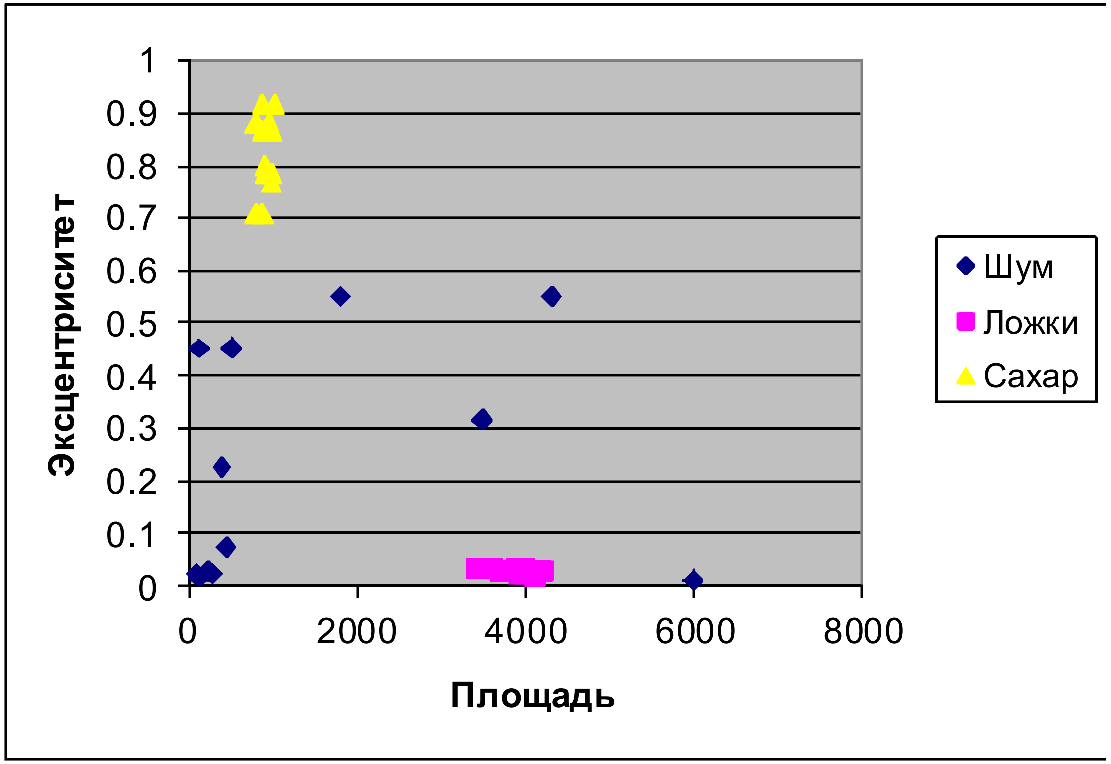


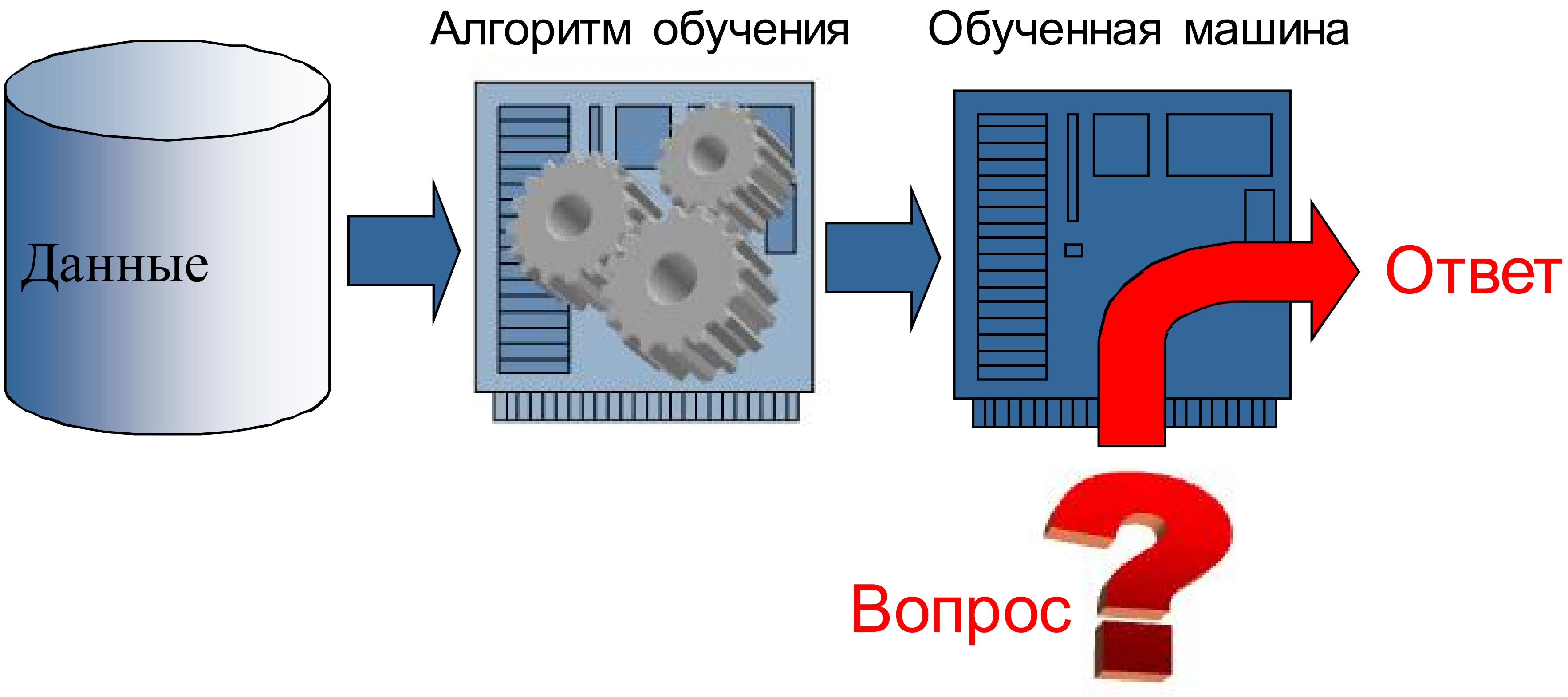
Схема простого алгоритма выделения и распознавания объектов

Что было сложно?

- Признаков может быть много (несколько моментов, площадь, положение, гистограммы яркости, каналов цвета и т.д.)
- Подбирать правила приходится вручную, много гипотез, когда признаков много и распределение сложно, это невозможно.

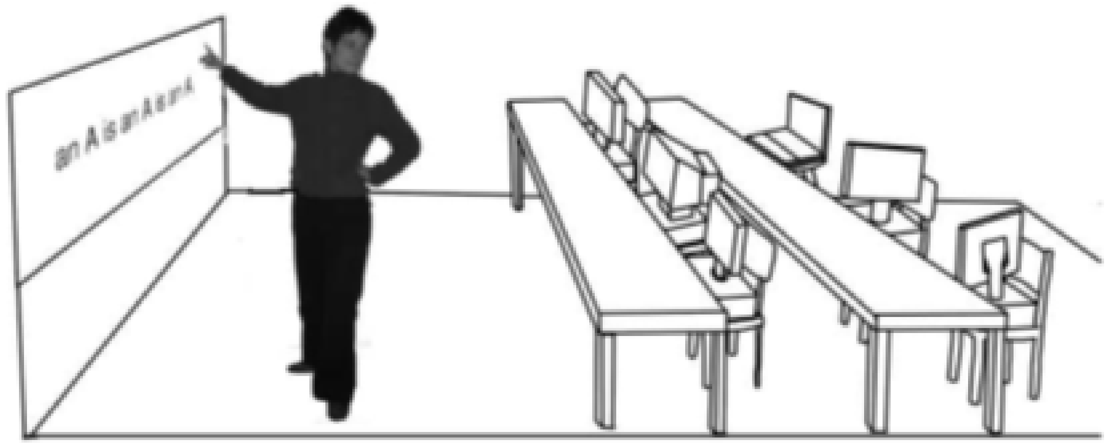


Как должно быть в идеале



Что такое машинное обучение?

- Обучение ≠ «заучивание наизусть»
  - Заучить наизусть – для машины не проблема
  - Мы хотим научить машину делать выводы!
  - Машина должна корректно работать на новых данных, которые мы ей раньше не давали
  - По конечному набору обучающих данных машина должна научиться делать выводы на новых данных



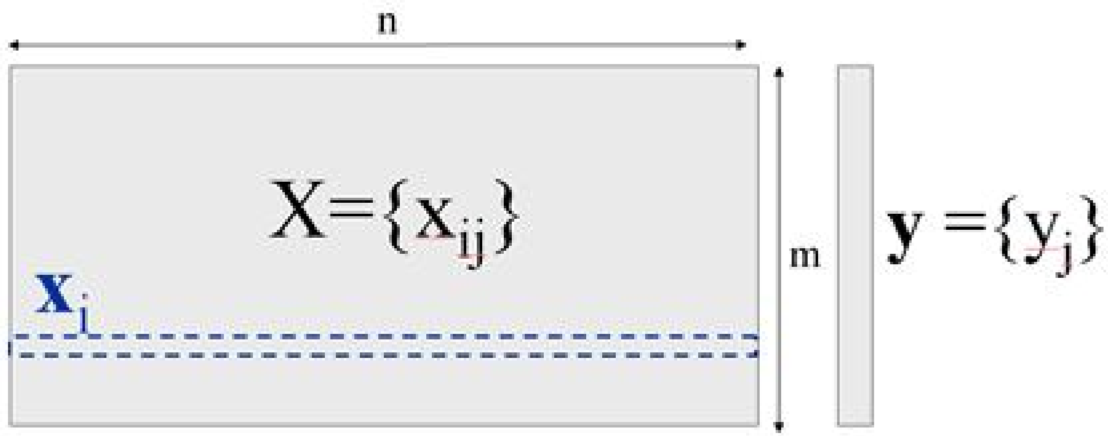
Важные этапы машинного обучения

- В 1958 году Розенблатт предложил алгоритм «персептрона»
- В 1975 году Фукушима предложил алгоритм когнитрона – многослойной нейронной сети с алгоритмом обучения
- В 1997 – 2001 года предложен ряд новых алгоритмов распознавания образов на основе машинного обучения (SVM, Boosting, Random Forest)
- Их появление привело к взрывному прогрессу в области анализа данных и компьютерного зрения в частности

- Мы рассмотрим «на пальцах» один метод – «метод опорных векторов»

Задача классификации образов

- Дано множество объектов, каждый из которых принадлежит одному из нескольких классов. Нужно определить, какому классу принадлежит данный экземпляр
- Каждый объект с номером j можно описать вектором признаков $\boldsymbol{x_j}$
- Каждому объекту можно приписать метку класса $\boldsymbol{y_j}$
- Всё множество известных наблюдений (конечное) можно записать в следующем виде:

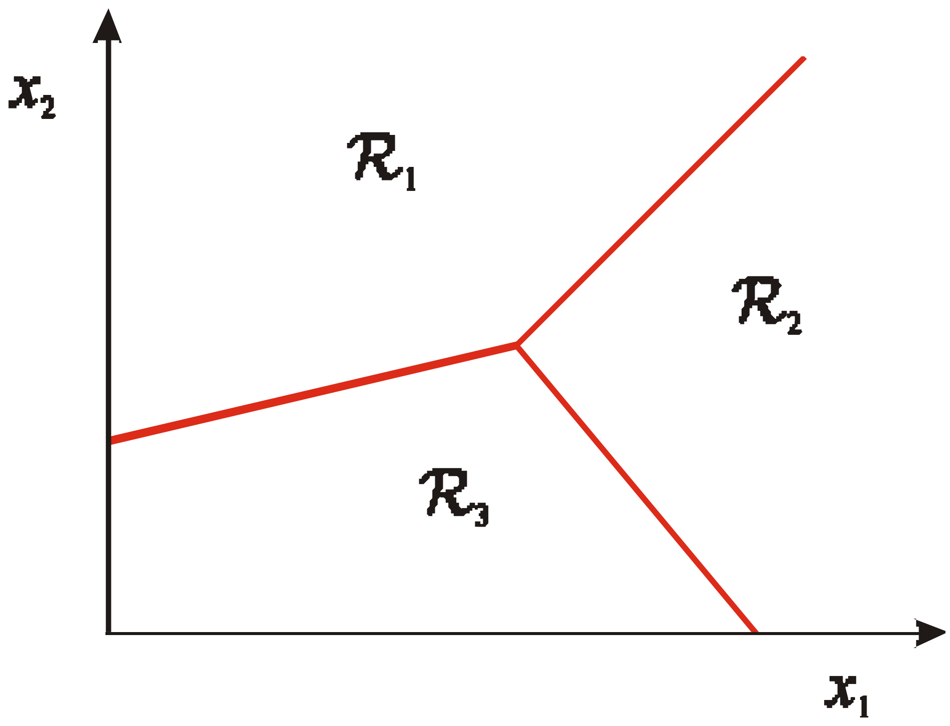


Каждое наблюдение (x,y) ещё называют прецедентом

Задача классификации образов

- Задача построить функцию **y=f(x)**, которая для каждого вектора-признаков **x** даёт ответ **y**, какому классу принадлежит объект **x**
- Функция **f()** назвается *решающее правило или классификатор*
- Любое *решающее правило* делит пространство признаков на *решающие регионы* разделенные *решающими границами*

Decision



Классификация

- Будем выбирать функции ***f*** из **параметрического семейства *F*** (т.е. будем выбирать подходящий набор параметров)
- Введем некоторую **функцию потерь *L(f(x), y)***,
  - В случае классификации часто используют $L(f(\mathbf{x}), y) = I[y \neq f(\mathbf{x})]$,где $f(\mathbf{x})$ - предсказанный класс
  - Можем использовать и другие функции потерь
- Задача обучения состоит в том, чтобы найти набор параметров классификатора f, при котором потери для новых данных будут минимальны
- «Метод классификации» = параметрическое семейство F + алгоритм оценки параметров по обучающей выборке

Общий риск

- Общий риск – математическое ожидание потерь:

$$R(f) = \mathrm{E}\Big(L\big(f(\mathbf{x}), y\big)\Big) = \int_{\mathbf{x},y} L(f(\mathbf{x}), y)dP$$

  - рассчитать невозможно, поскольку распределение ***P*** неизвестно

Эмпирический риск

- Пусть $X^m = \{\mathbf{x_1}, ..., \mathbf{x_m}\}$ - обучающая выборка

- Эмпирический риск (ошибка обучения):

$$R_{emp}\left(f, X^m\right) = \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), y_i)$$

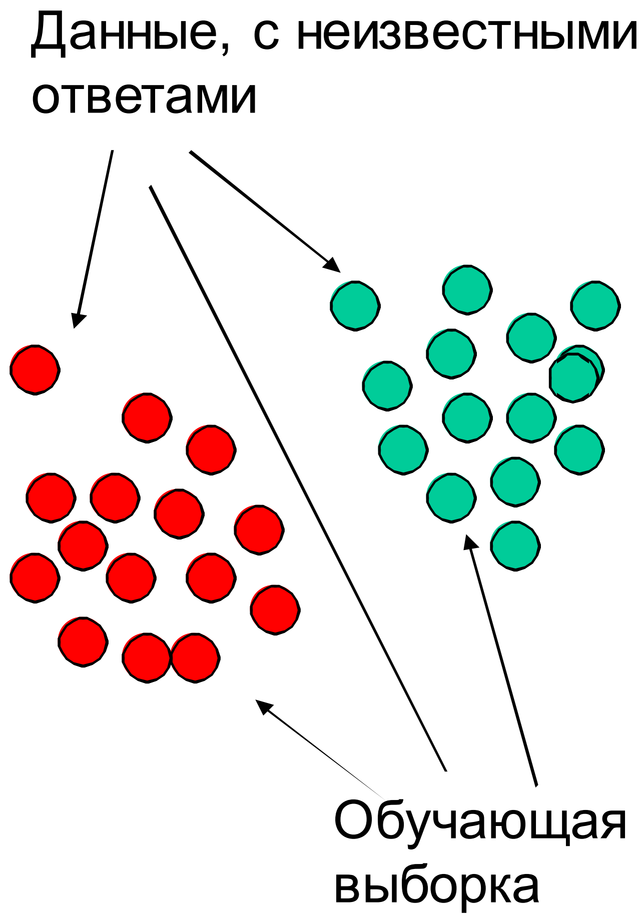
- Метод минимизации эмпирического риска:

$$f = \arg\min_{f \in F} R_{emp}\left(f, X^m\right)$$

Смысл: подбираем параметры решающего правила таким образом, чтобы эмпирический риск был минимален

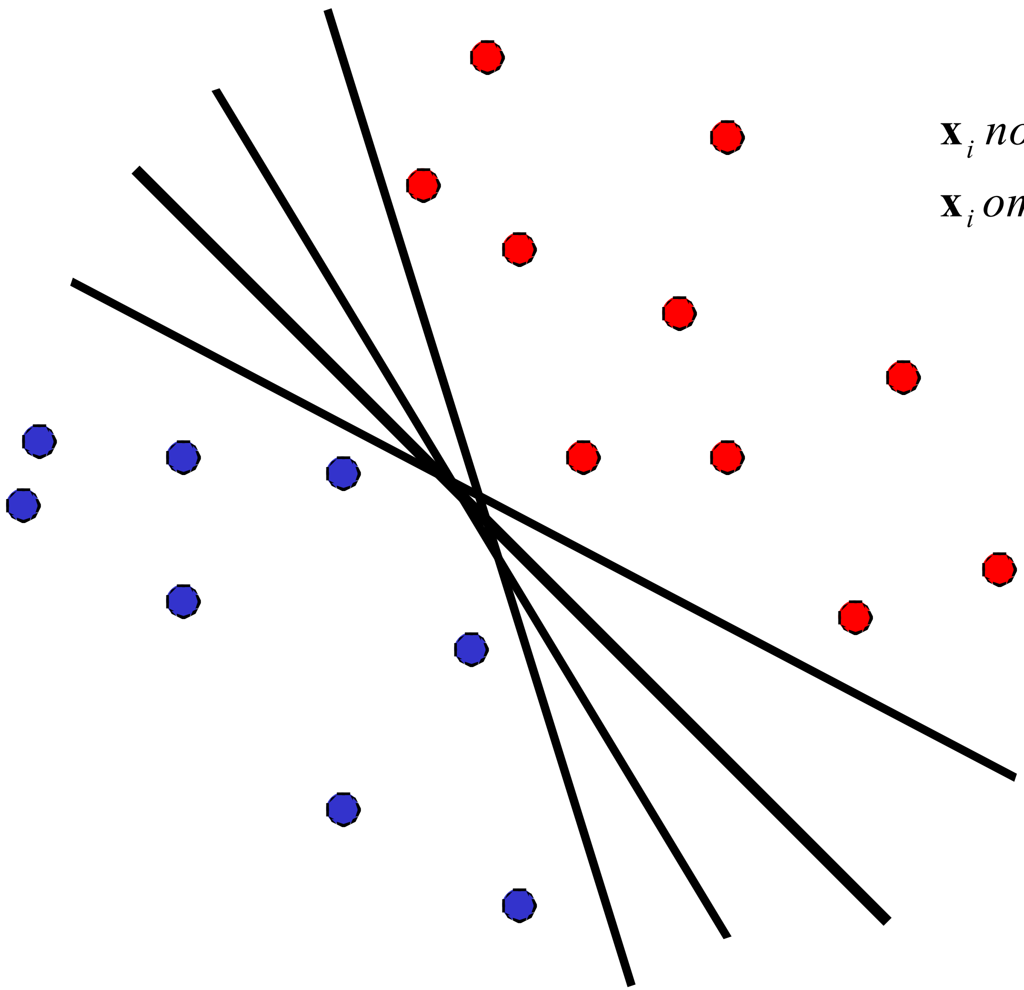
Линейная классификация

- Рассмотрим случай линейно разделимых данных
- Только для 2х классов
- Т.е. таких, что вектора в пространстве признаков можно отделить друг от друга гиперплоскостью



Линейный классификатор

- Найдем линейную функцию (гиперплоскость) и разделим положительные {y=+1} и отрицательные {y=-1} примеры



$\mathbf{x}_i$ *положительные* : $\mathbf{x}_i \cdot \mathbf{w} + b \geq 0$

$\mathbf{x}_i$ *отрицательные* : $\mathbf{x}_i \cdot \mathbf{w} + b < 0$

Для всех рассмотренных плоскостей эмпирический риск одинаковый

Какая гиперплоскость наилучшая?

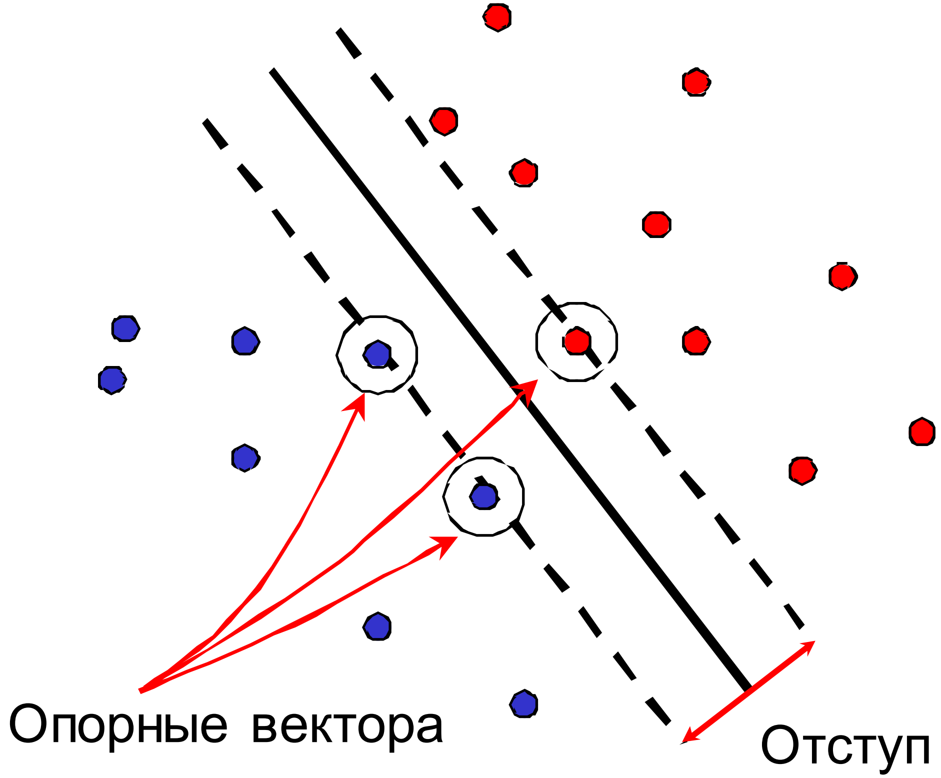
Метод опорных векторов

- Support Vector Machine (SVM)
- Найдем гиперплоскость, максимизирующую *отступ* между положительными и отрицательными примерами

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Метод опорных векторов

- Найдем гиперплоскость, максимизирующую *отступ* между положительными и отрицательными примерами



$\mathbf{x}_i$ *положительные* $(y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ *отрицательные* $(y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

Для опорных векторов, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Расстояние от точки до гиперплоскости: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Поэтому отступ равен $2 / \|\mathbf{w}\|$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Поиск гиперплоскости

1. Максимизируем $2/\|\mathbf{w}\|$
2. Правильно классифицируем все данные:

$$\mathbf{x}_i \text{ позитивные } (y_i = 1): \qquad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ негативные } (y_i = -1): \qquad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

- *Квадратичная оптимизационная задача*:

- Минимизируем $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

  При условии $y_i(\mathbf{w}\cdot\boldsymbol{x}_i+b) \geq 1$

- Решается с помощью методом множителей Лагранжа

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Поиск гиперплоскости

- Решение: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

Обученные веса

Опорные вектора

- Для большей части векторов вес = 0!
- Все вектора, для которых вес >0 называются опорными
- Определяется только <u>опорными</u> векторами

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Поиск гиперплоскости

- Решение: $$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

  $b = y_i - \mathbf{w}\cdot\mathbf{x}_i$ для любого опорного вектора

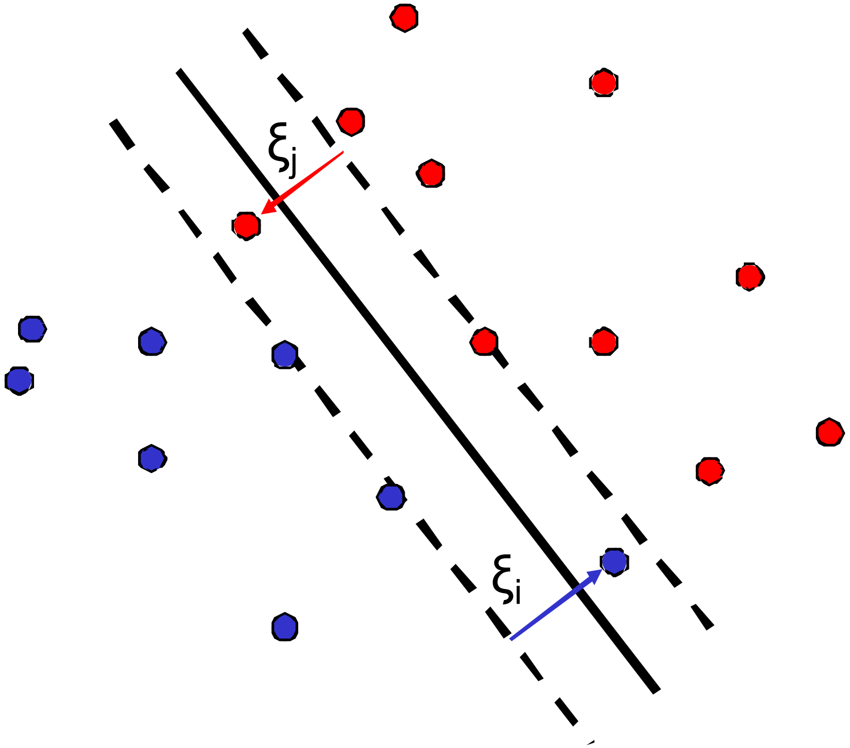
- Решающая функция:

$$\mathbf{w}\cdot\mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

- Решающая функция зависит от скалярных произведений (inner product) от тестового вектора $\boldsymbol{x}$ и опорных векторов $\boldsymbol{x}_i$
- Решение оптимизационной задачи также требует вычисления скалярных произведений $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ между всеми парами векторов из обучающей выборки

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Реальный случай



Вводим дополнительные «slack» переменные: $\xi = (\xi_1, \ldots, \xi_n)^T$

Минимизируем $\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$

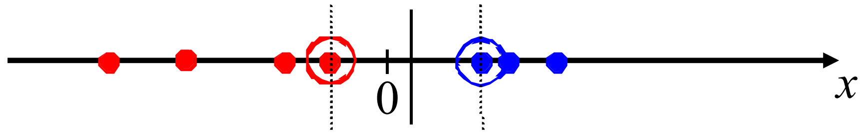
При условии $y_i(wx_i + b) \geq 1 - \xi_i$

C – параметр регуляризации

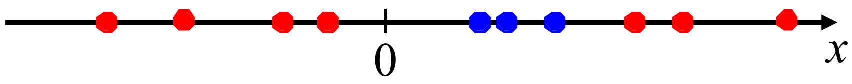
В реальном случае идеально разделить данные невозможно (эмпирический риск всегда больше 0)

Нелинейные SVM

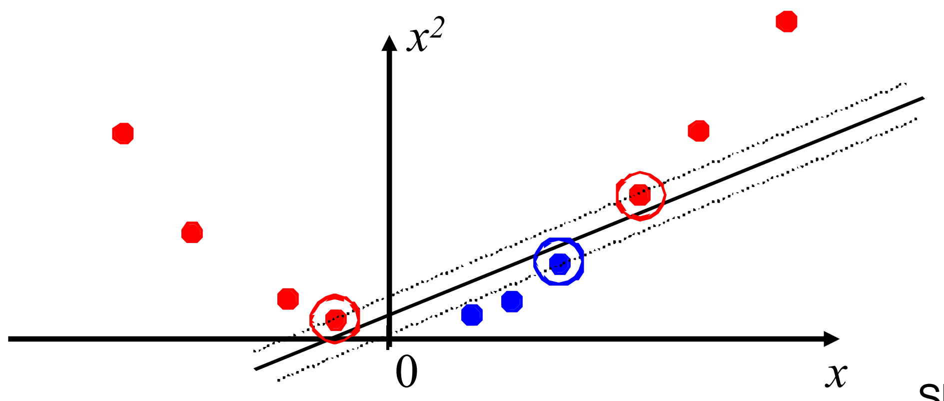
- На линейно разделимых данных SVM работает отлично:



- Но на более сложных данных не очень:



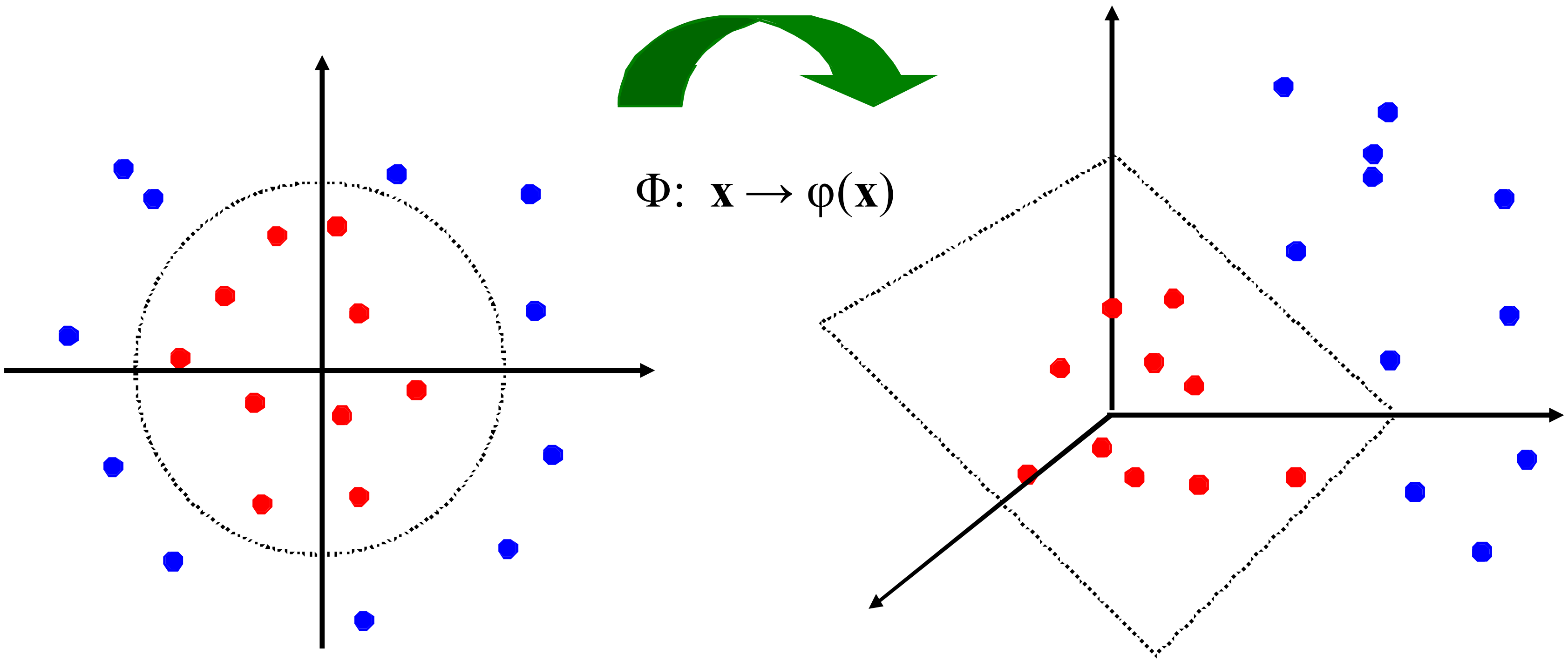
- Можно отобразить данные на пространство большей размерности и разделить их линейно там:



Slide credit: Andrew Moore

Нелинейные SVM

- Идея: отображение исходного пространства параметров на какое-то многомерное пространство признаков (feature space) где обучающая выборка линейно разделима:



Slide credit: Andrew Moore

Нелинейные SVM

- Вычисление скалярных произведений в многомерном пространстве вычислительно сложно
- *The kernel trick:* вместо прямого вычисления преобразования $\boldsymbol{\varphi}(\mathbf{x})$, мы определим ядровую функцию K:

$$K(\mathbf{x}_i\,,\,\mathbf{x}_{jj}) = \boldsymbol{\varphi}(\mathbf{x}_i\,)\cdot\boldsymbol{\varphi}(\mathbf{x}_j)$$

- Чтобы все было корретно, ядро должно удовлетворять условию Мерсера (*Mercer's condition*)
  - Матрица $K(x_i,x_j)$ должна быть неотрицательно определенной
- С помощью ядра мы сможем построить нелинейную решающую функцию в исходном пространстве:

$$\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

C. Burges, [A Tutorial on Support Vector Machines for Pattern Recognition], Data Mining and Knowledge Discovery, 1998

Пример ядра

- Полиномиальное: $K(x,y) = ((\mathbf{x}, y) + c)^d$

- Пусть d=2, x=(x_1,x_2):

$$K(x,y) = ((\mathbf{x}, y) + c)^2 = (x_1 y_1 + x_2 y_2 + c)^2 = \varphi(x) \cdot \varphi(y)$$

$$\varphi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2, \sqrt{2c} x_1, \sqrt{2c} x_2, c)$$

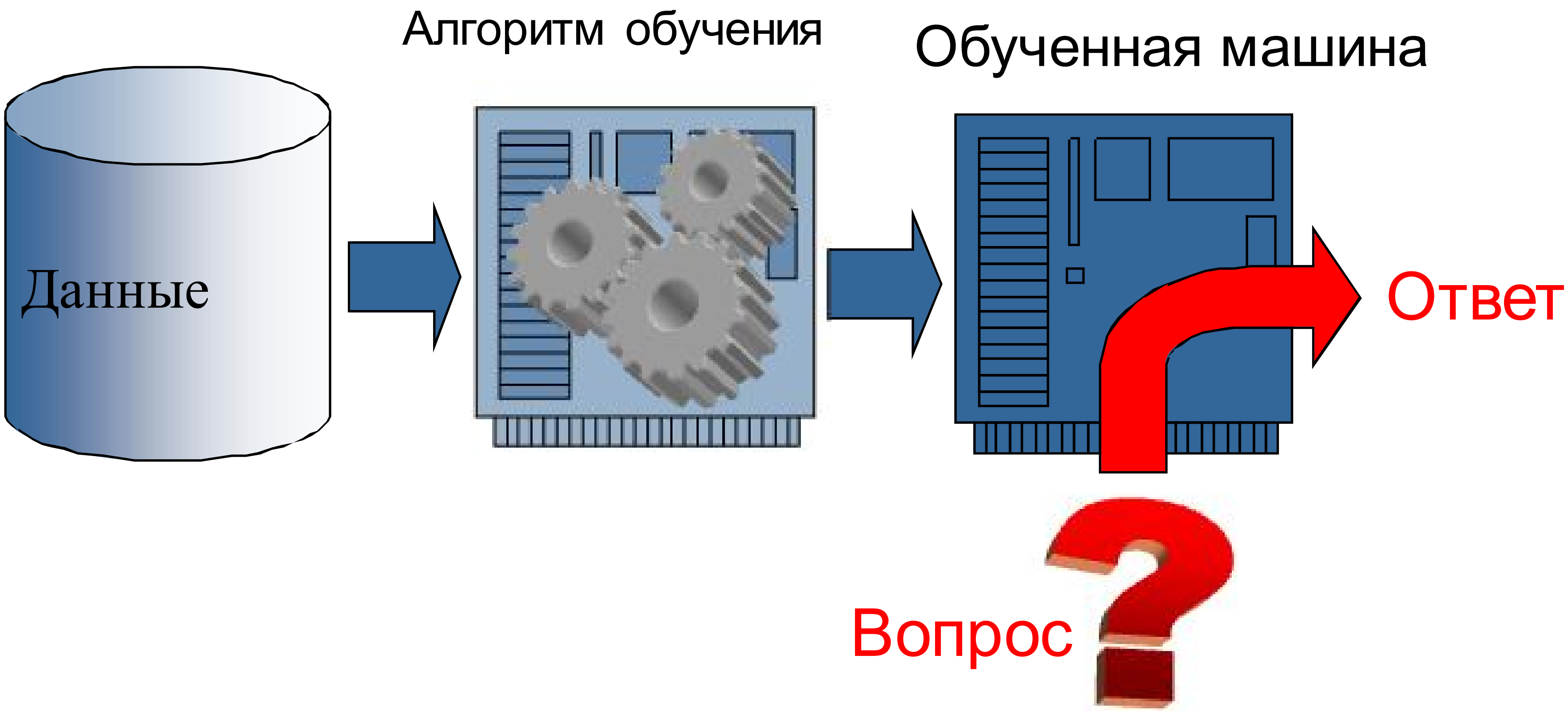
- Т.е. из 2х мерного в 6и мерное пространство

SVM - резюме

- Мощный и гибкий подход на основе ядер
  - Придумано много разных ядер
- На практике работает очень хорошо, даже для маленьких обучающих выборок
- Много доступных библиотек: http://www.kernel-machines.org/software

Оценка классификаторов

- Обучить классификатор SVM умеем
- Как же оценить, насколько хорошо он обучился?
- Нужны количественные оценки качества обучения – предсказательной способности классификатора



Предсказательная способность

- Нам нужно, чтобы классификатор хорошо работал на всех данных
- Нужно минимзировать «общий риск»:

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

- Однако, напрямую его посчитать невозможно (требует вычислений на неограниченном множестве)
- У нас же есть только ограниченный объём данных (конечная обучающая выборка)
- Как быть?
  - «Удерживание»
  - «Скользящий контроль»
  - 5-2 контроль

Общая идея

- Будем разбивать имеющуюся обучающую выборку на части
  - На одной части будем обучать классификатор
  - На другой будет измерять ошибку (оценивать предстказательную способность)

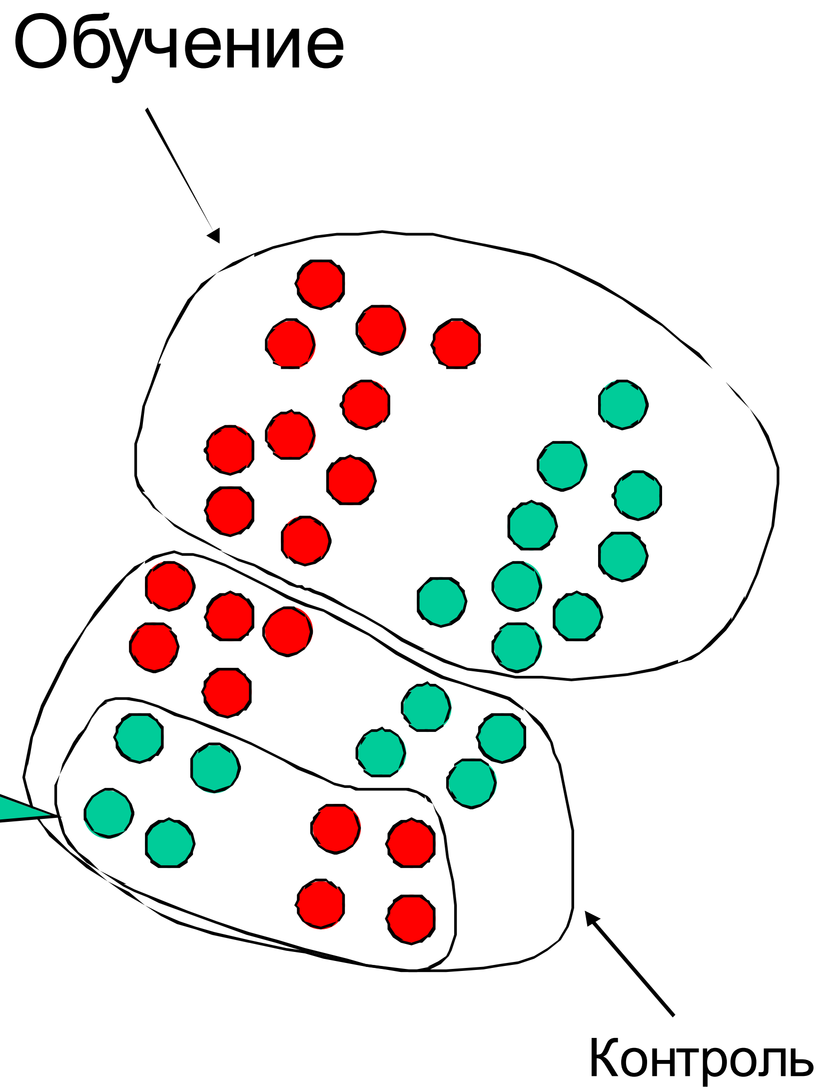
$$R(f, X) \sim P\left(f(x) \neq y \mid X^c\right) = \frac{1}{c}\sum_{j=1}^{c}\left[f(x_j) \neq y_j\right]$$

Удерживание

- Пусть, имеется набор данных $X^k = \{x_1, ..., x_k\}$ с известными ответами
- Разобьем $X^l \bigcup X^c = X^k : X^l \bigcap X^c = 0$
- Будем использовать для обучения X^l , а для контроля X^c
- То есть: $P(f(x) \neq y) \approx P\left(f(x) \neq y \mid X^c\right)$

Свойства «удерживания»

- Быстро и просто рассчитывается
- Некоторые «сложные» прецеденты могут полностью попасть в только одну из выборок и тогда оценка ошибки будет смещенной



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

Скользящий контроль

- Разделим выборку на d непересекающихся частей и будем поочередно использовать одно из них для контроля а остальные для тренировки

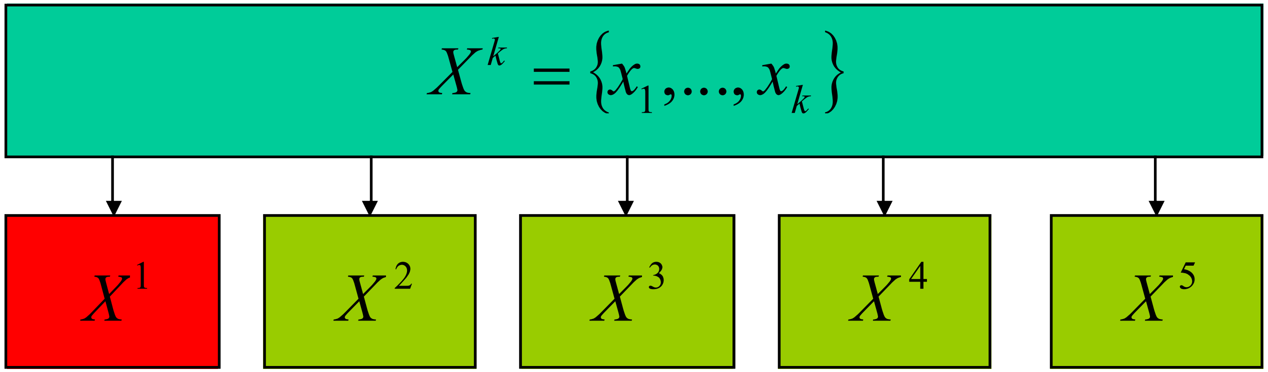
  - Разбиваем: $\{X^i\}_1^d : X^i \cap X^j = 0, i \neq j$

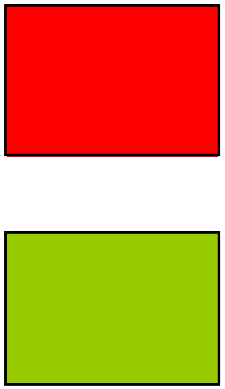
$$\bigcup_{i=1}^{d} X^i = X^k$$

  - Приближаем риск:

$$P(f(X^k) = y^*) \approx \frac{1}{d} \sum_{i=1}^{d} P(f(X^i) \neq y^* \mid \bigcup_{i \neq j} X^i)$$

Иллюстрация





Контроль

Обучение

Результат считается как средняя ошибка по всем итерациям

Свойства

- 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 одна группа «сложных прецедентов» попала полностью в один сегмент, то оценка будет смещенной

5-2 контроль

- 5-2 cross validation
- Некоторый компромисс:
  - Разделим выборку случайным образом пополам
  - Обучим на одной половине, протестируем на другой, и наоборот
  - Повторим этот эксперимент пять раз и усредним результат

- Свойства:
  - Каждый из прецедентов будет учувствовать в контрольных выборках на каждом из 5 этапов

Виды ошибок

- Измерения ошибки как «вероятности выдать неверный ответ» может быть не всегда достаточно
  - 15% ошибки при постановке диагноза может означать как и то что, 15 % больных будут признаны здоровыми (и возможно умрут от отсутствия лечения), так и то, что 15% здоровых больными (и деньги на лечение будут потрачены зря)

- При неравнозначности ошибок для разных классов вводят понятие ошибки первого и второго рода и замеряют их по отдельности

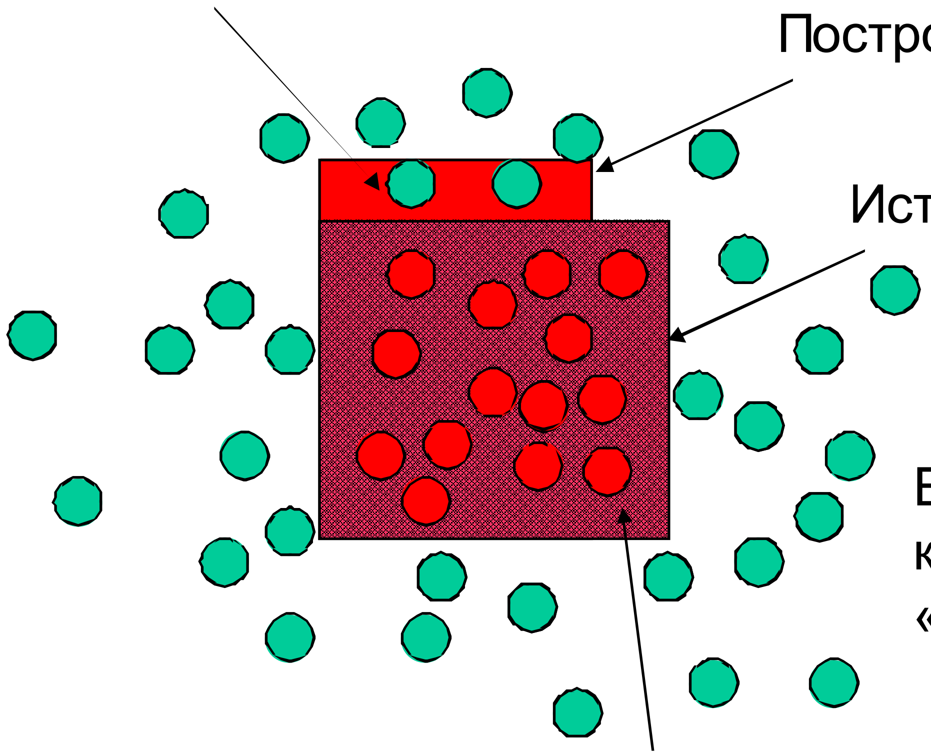
Ошибки I и II рода

- Пусть, существует «основной класс»
  - Обычно, это класс, при обнаружении которого, предпринимается какое-либо действие;
  - Например, при постановке диагноза основным классом будет «болен», а вторичным классом «здоров».

- Ошибка первого рода равна вероятности принять основной класс за вторичный
  - Вероятность «промаха», когда искомый объект будет пропущен

- Ошибка второго рода равна вероятности принять вторичный класс за основной
  - Вероятность «ложной тревоги», когда за искомый объект будет принят «фон»

Ошибки I и II рода



Будем считать
красные точки
«основным классом»

Ошибки I и II рода

- Что считать основным классом зависит полностью от прикладной специфики
  - В компьютерном зрении – почти всегда сильно несбалансированы
- Особенно важно оценивать ошибки I и II рода раздельно при несбалансированности классов:
  - Пусть $P(y=+1)=0.01; P(y=-1)=0.99$
  - Тогда при ошибке II рода 0 и ошибке I рода 0.5

    $$P(f(x)=-1 \mid y=+1)=0.5$$

  - Общая ошибка всего лишь

    $$P(a(x)\neq y)=0.005$$

Чувствительность vs Избирательность

- **Чувствительность** – вероятность дать правильный ответ на пример основного класса

$$sensitivity = P(f(x) = y \mid y = +1)$$

  - Также *уровень обнаружения (detection rate)*

- **Избирательность** – вероятность дать правильный ответ на пример вторичного класса

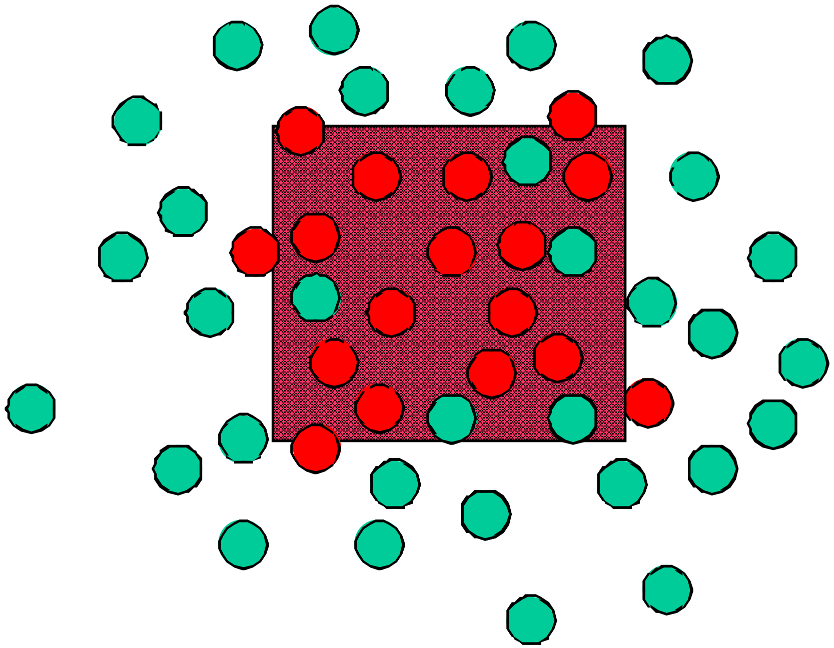
$$specificity = P(f(x) = y \mid y = -1)$$

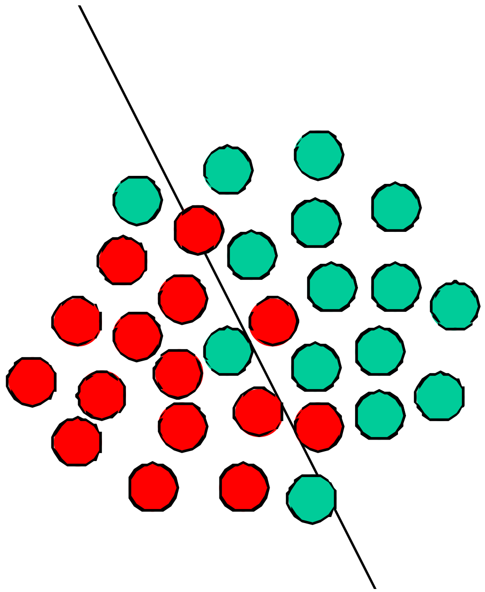
Другие метрики

- Точность (Precision)
  - Доля истинных объектов основного класса среди всех классифицированных, как первый класс
- Полнота (Recal)
  - Доля правильно распознанных объектов основного класса среди всех объектов основного класса из тестовой выборки
- Интегральный показатель F (F-measure)
  - $\mathrm{F} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$

Регулировка баланса

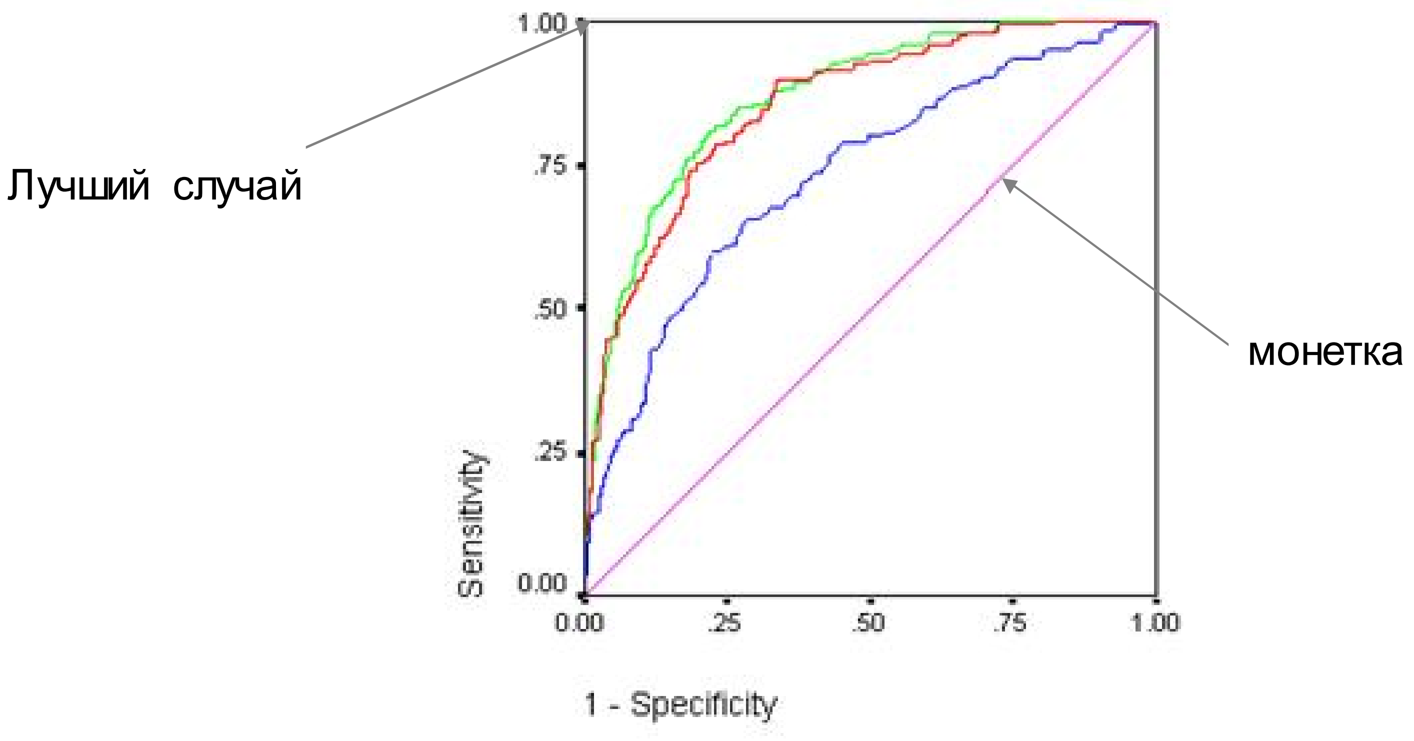
- Почти все алгоритмы классификации допускают регулировку соотношения ошибки I и II рода за счет варьирования некоторого параметра





ROC кривая

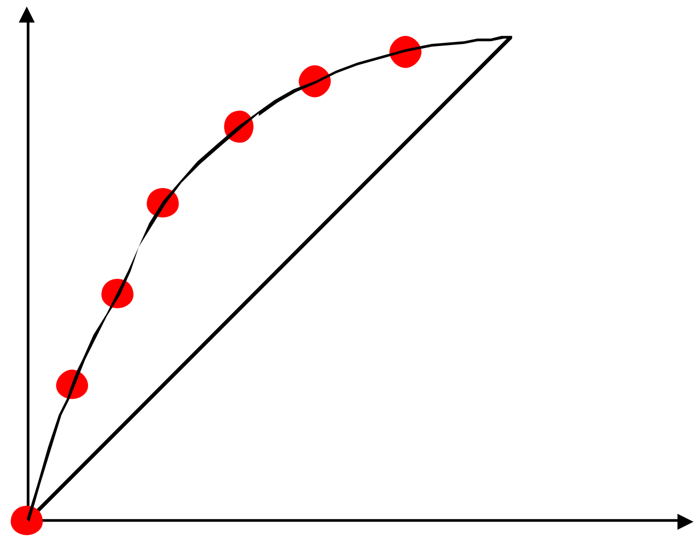
- ROC – Receiver Operating Characteristic curve
  - Кривая, отражающая зависимость чувствительности и ошибки второго рода



ROC кривая - Построение

- Для различных значений параметра строится таблица ошибок
  - Сам параметр в таблице не участвует!
  - Классификатор строится и оценивается на разных выборках!
- По таблице строиться набор точек в плоскости sensitivity/FP
  - Каждая строка таблицы - точка
- По точкам строиться кривая

Sensitivity	False Positive
0.0	0.0
0.25	0.5
0.5	0.8
…	…
1.0	1.0



Анализ ROC кривой

- Площадь под графиком – AUC
  - Дает некоторый объективный показатель качества классификатора
  - Позволяет сравнивать разные кривые

- Соблюдение требуемого значения ошибок I и II рода
  - Зачастую, для конкретной задачи существуют рамки на ошибку определенного рода. С помощью ROC можно анализировать возможность текущего решения соответствовать требованию

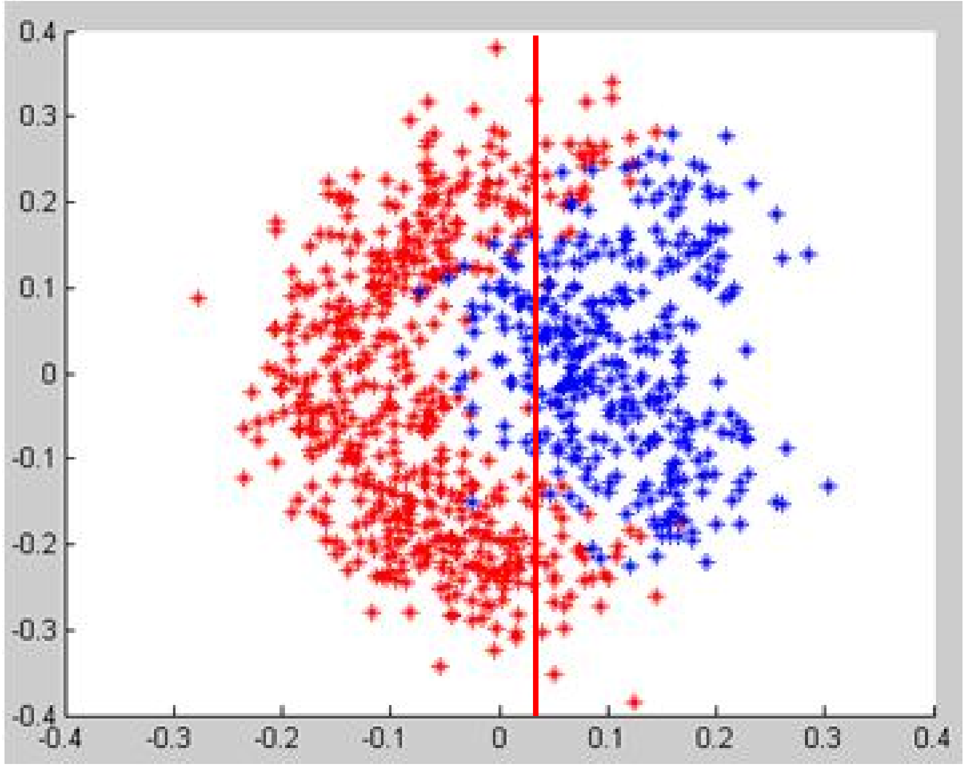
Пример

- Данные – точки на плоскости

- «Классификатор» – порог по оси X

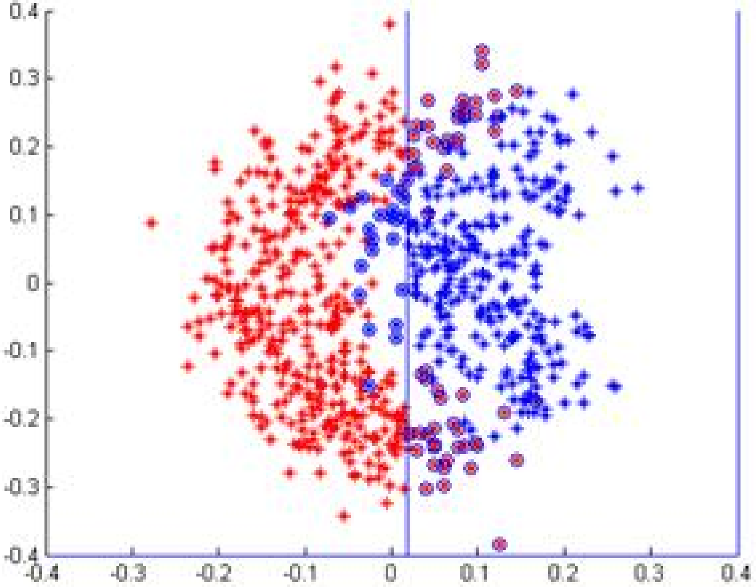
$$a(x^1, x^2) = \begin{cases} +1, x_1 > \Theta \\ -1, x_1 \leq \Theta \end{cases}$$

- Будем обучать его, пользуясь разными подходами и измерять ошибки
  - Удерживание
  - Скользящий контроль



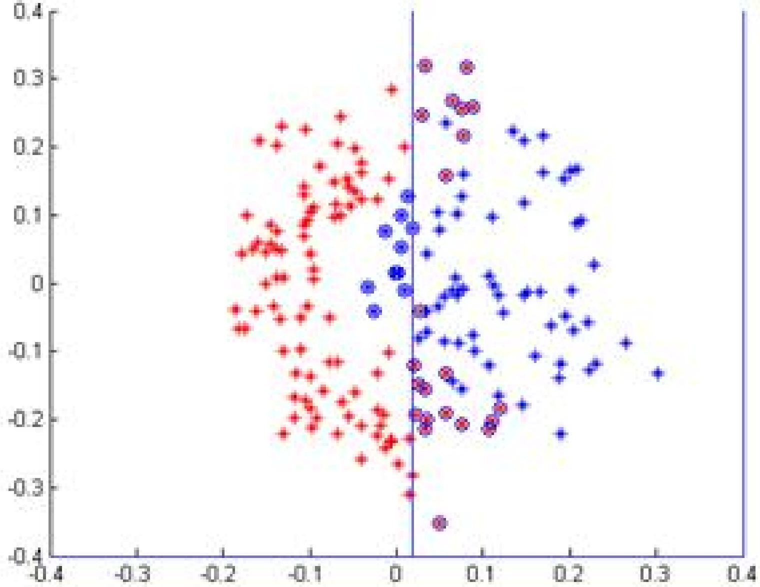
Удерживание

Ошибка: 0.1133



Тренировочная выборка

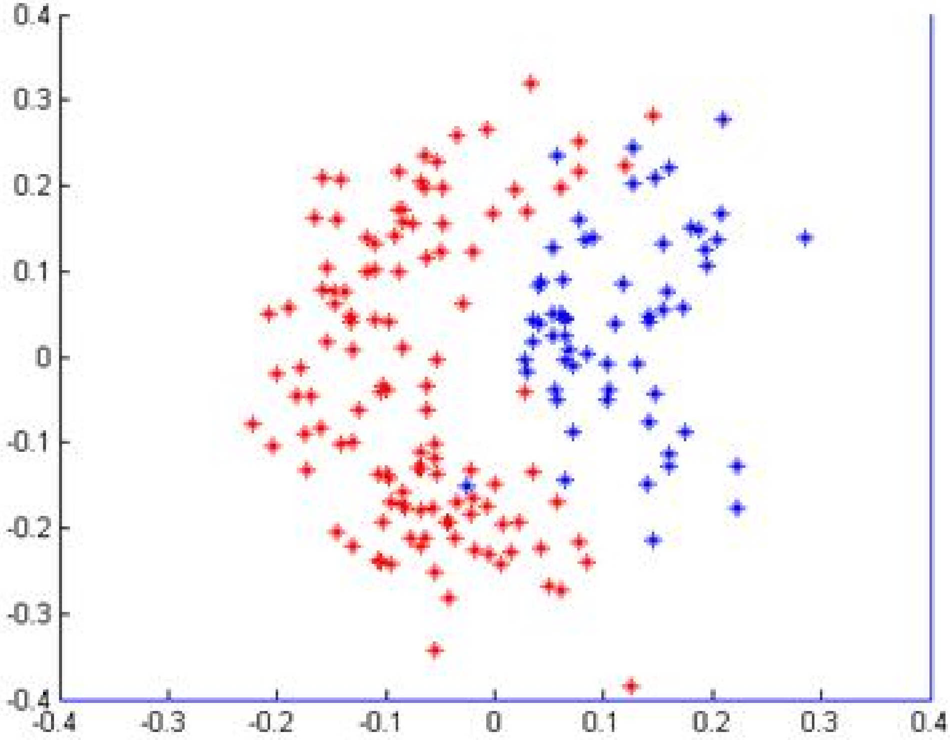
Ошибка: 0.1433



Контрольная выборка

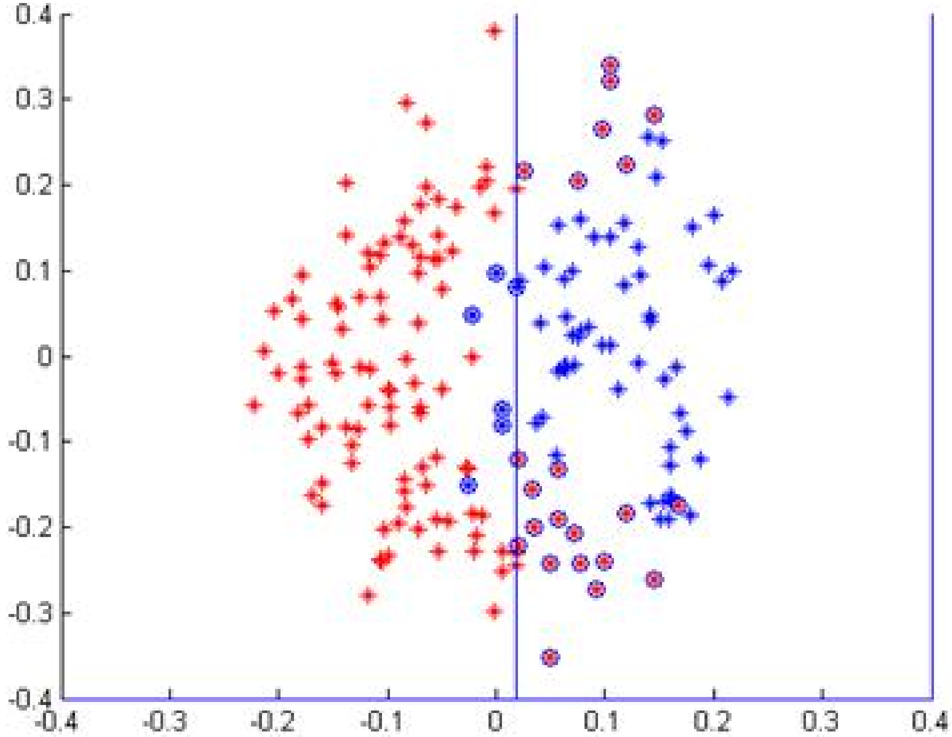
Разобъём данные на 2 части, обучим на одной, проверим на другой

Скользящий контроль: разбиение



Разобъём данные на 5 частей

Скользящий контроль: измерение



Обучим и измерим ошибку 5 раз

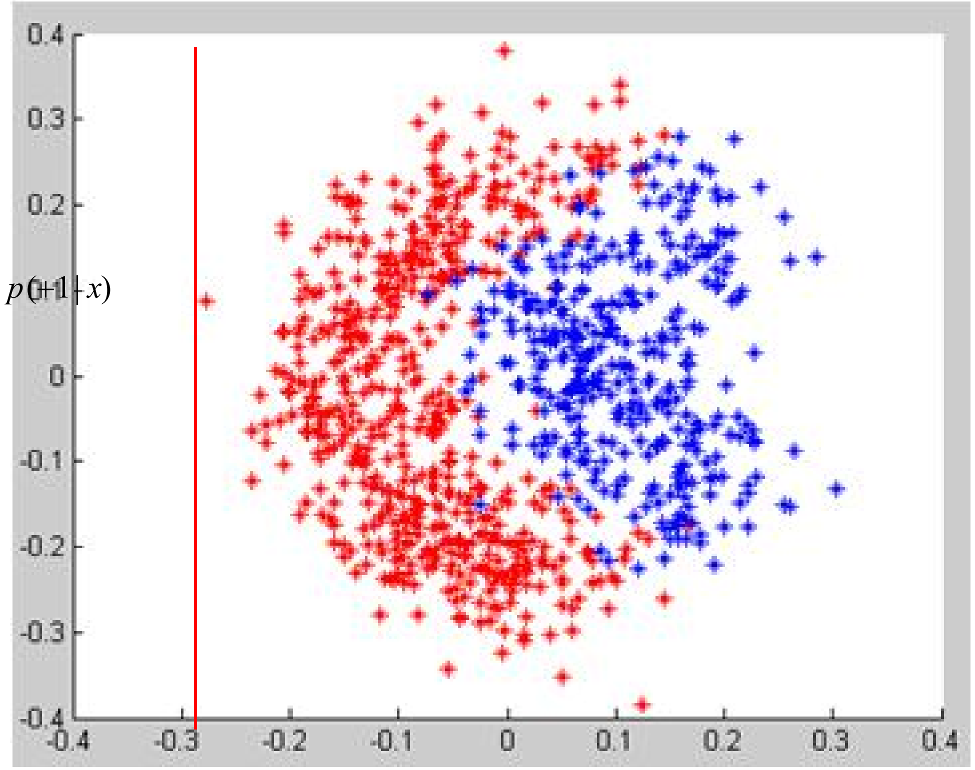
Скользящий контроль

- Тренировочная ошибка:
  - *{0.1236 0.1208 0.1250 0.1097 0.1306}*
  - Среднее = *0.1219*
- Ошибка на контроле
  - *{0.1500 0.1333 0.1222 0.1778 0.1000}*
  - Среднее = *0.1367*

Построение ROC: таблица

- Меняем порог и оцениваем ошибку

Sensitivity	False Positive
0.0	0.0
0.25	0.5
0.5	0.8
…	…
1.0	1.0

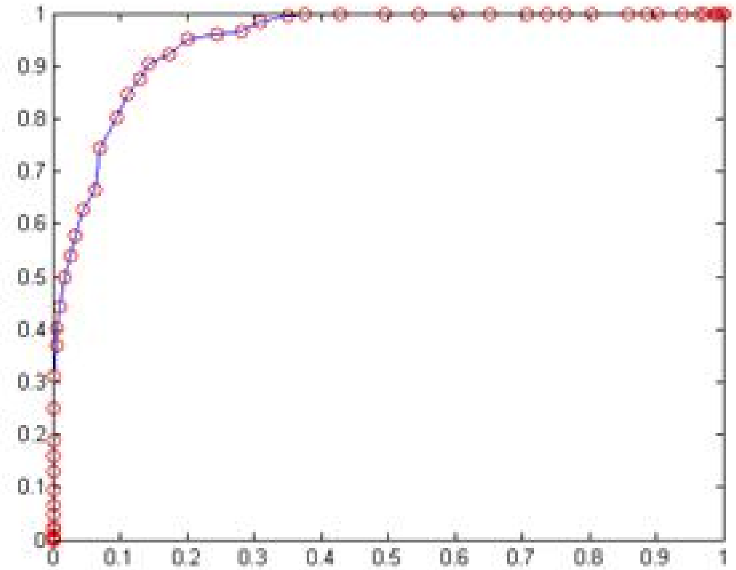


- Теперь построим ROC-кривую
- Поскольку у нас только 1 параметр, будем его менять напрямую
  - Это не обучение, должно быть обучение, но для примера сойдёт

Построение ROC-кривой

- По таблице строим точки

- Точки интерполируем кривой



Резюме бинарной классификации

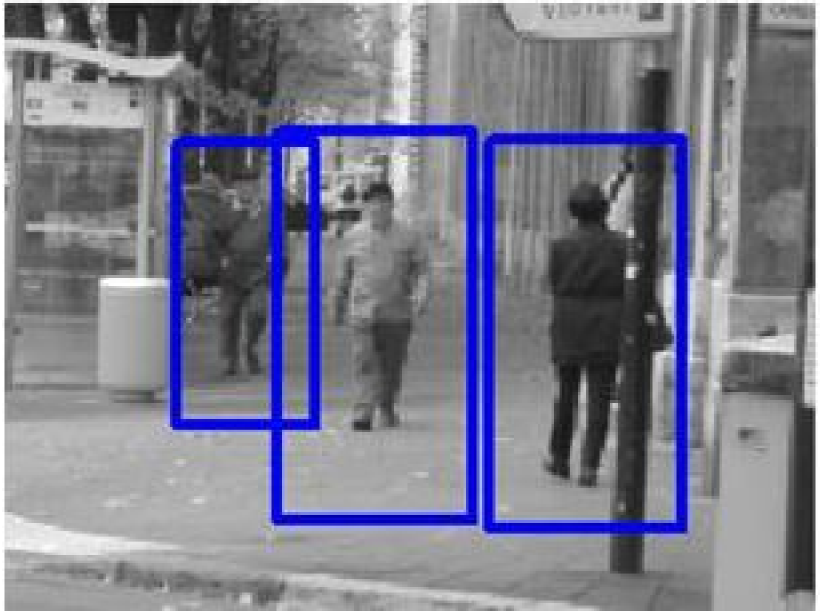
- Соберём обучающую выборку
- Подберём оптимальные параметры SVM (ядро, параметры ядра, количество опорных векторов)
  - Зафиксируем параметры
  - Воспользуемся кросс-валидацией, например, с разбиением выборки на 5 блоков
  - 5 раз проведем операцию
    - Выберем блок для контроля
    - Обучим на оставшихся 4х
  - Усредним характеристики (ошибку на контроле, ошибки 1 и 2 рода)
  - Построим ROC-кривую
  - Выберем точку с оптимальными для нашей задачи ошибками

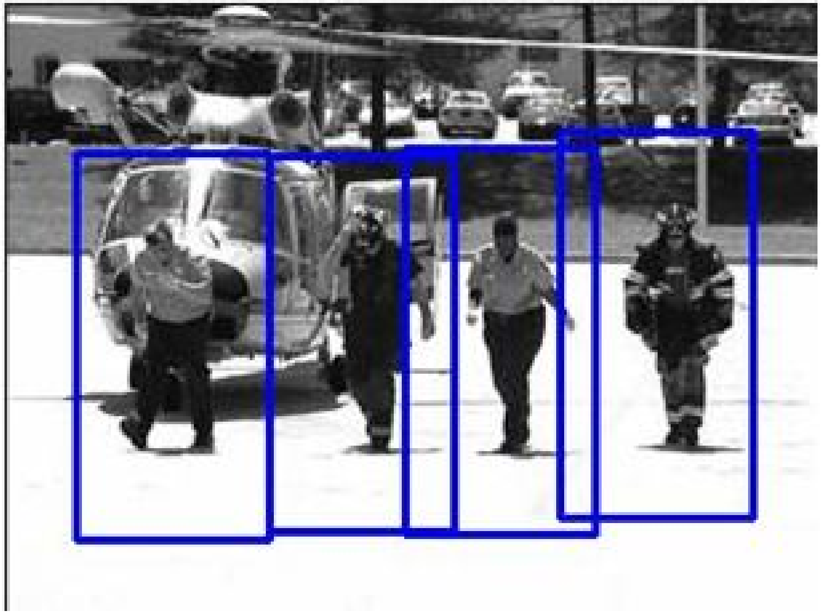
План лекции

- Введение в классификацию образов с помощью машинного обучения
  - Некоторые общие понятия
  - Один из базовых методов (SVM)
  - Практические аспекты машинного обучения

- Выделение категорий объектов на изображении
  - Более сложные признаки (HOG)
  - Скользящее окно
  - Выделение пешеходов и «торса»

Поиск пешеходов

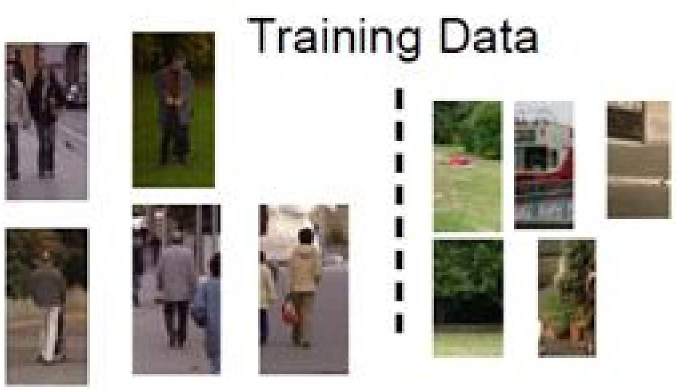




- Нужно найти «пешеходов» (если есть) и выделить их прямоугольной рамкой
- Пешеход (pedestrian) – это «категория», а не «объект»
  - Люди все похожи, но все разные
  - По простым признакам (цвету, площади) выделить пешехода нельзя

Классификация изображения

- Простая бинарная классификация может ответить на вопрос «да» / «нет»
- Например: «есть» пешеход на изображении или «нет» пешехода на изображении



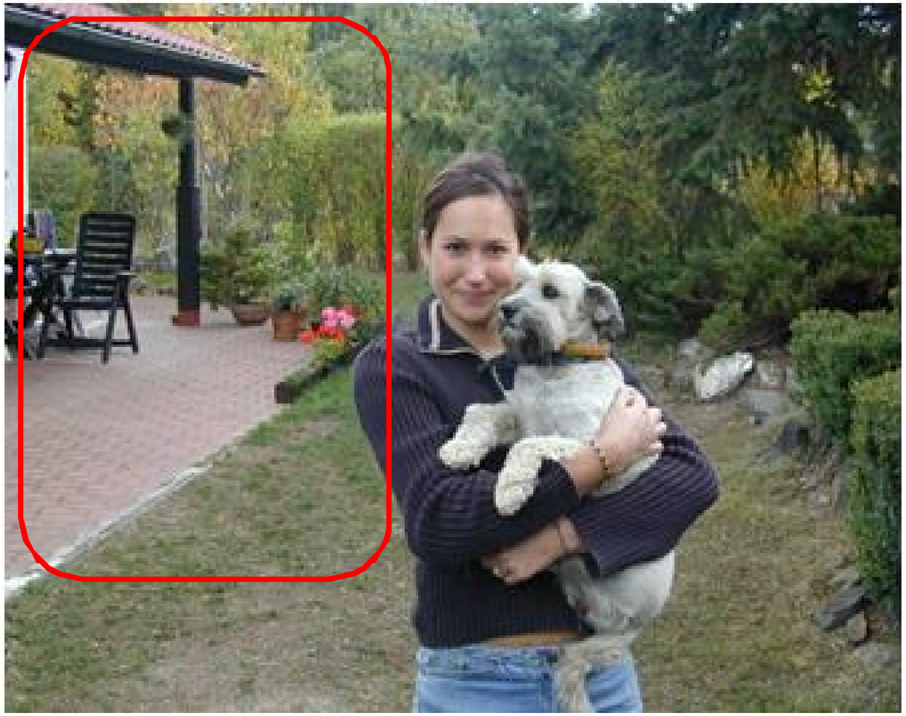
Обучающая выборка

«Картинка ли это пешехода»?

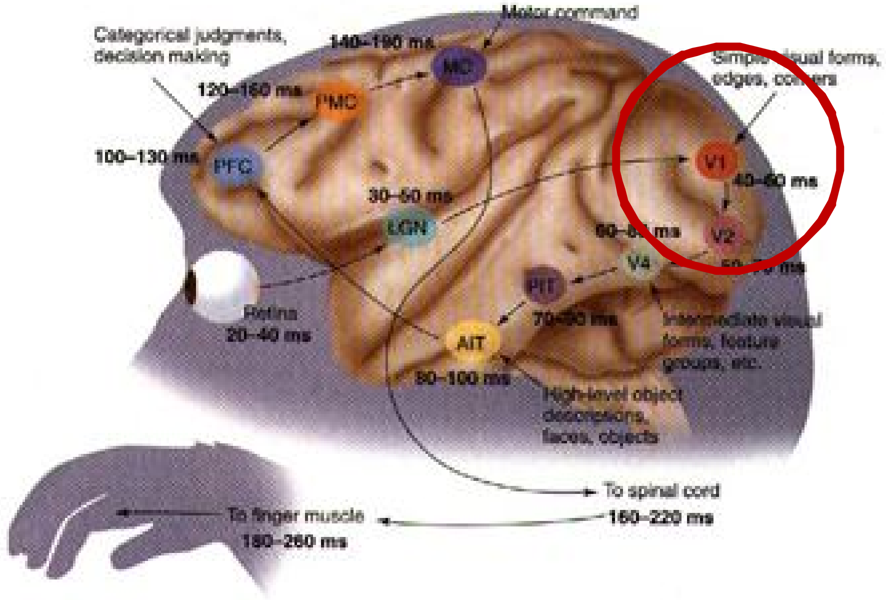
- А нам нужно знать, где ещё на картинке пешеход!

Скользящее окно (*sliding window*)

- Возьмём «окно», пусть размером 64*128 пикселов
- Сканируем изображение «окном»
- Применяем классификатор «пешеход?» к каждому окну
- Скользящее окно – форма сегментации изображения!



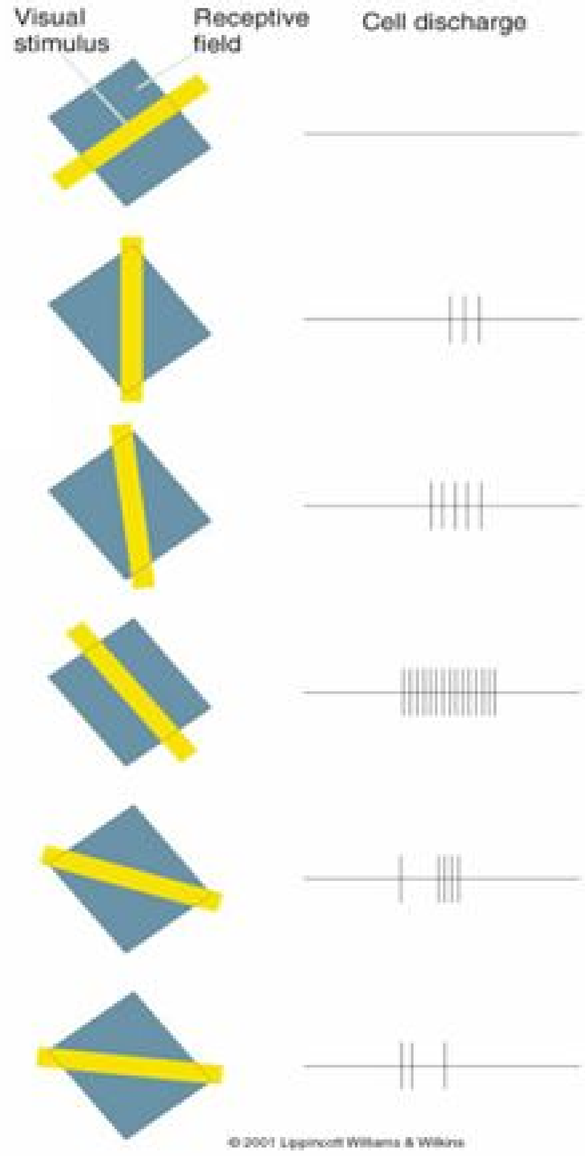
Края и обработка изображений мозгом



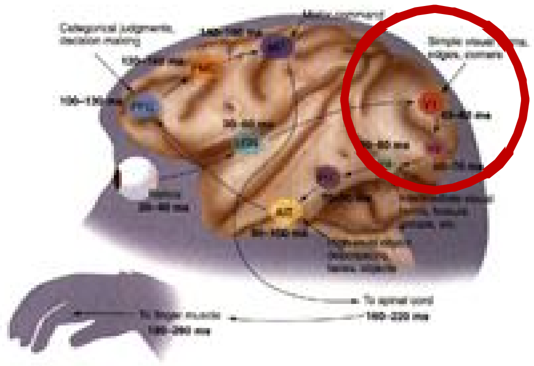
- В краях содержится много визуальной информации
- Как оказалось, первичная зрительная кора в основном занимается анализом краёв и градиентов в изображении
- «Ретинотопичкая организация» - распределение обработки в коре соответствует распределению на сетчатке

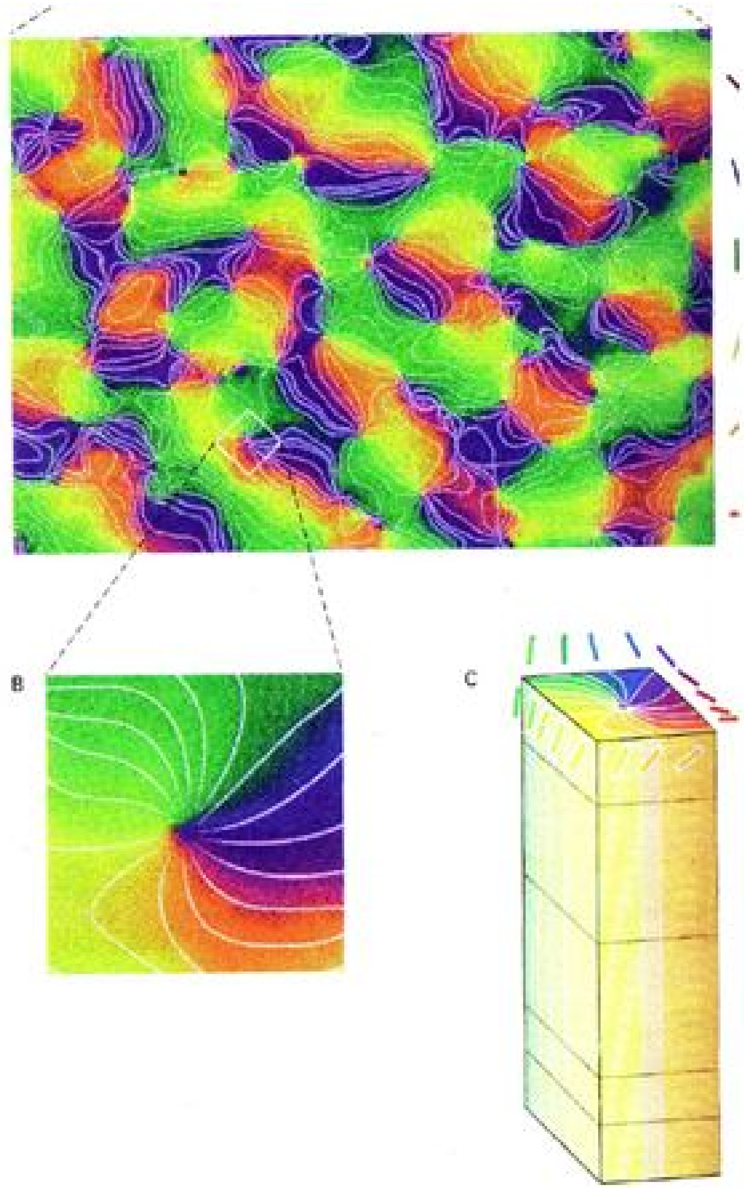
«Простые клетки»

Каждая «простая» клетка из первичной зрительной коры чувствительна к краю строго определенной ориентации



«Ретинотопическая» организация

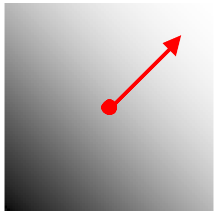




Для каждой области визуального поля есть клетки, чувствительные к краям разной ориентации (отображены на рисунке цветом)

Оценка градиентов

- Градиент изображения – направление максимального изменения яркости изображения



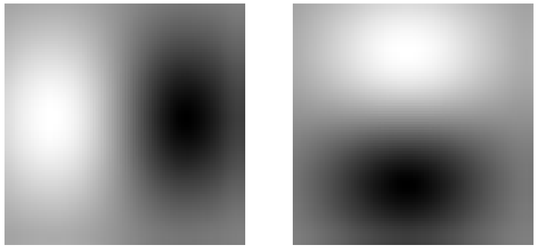
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Сила края: $\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$

Направление градиента: $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

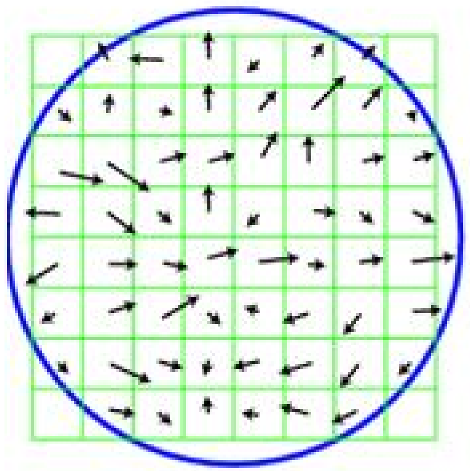
$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$ Собеля

Вычисление разностной производной через свёртку



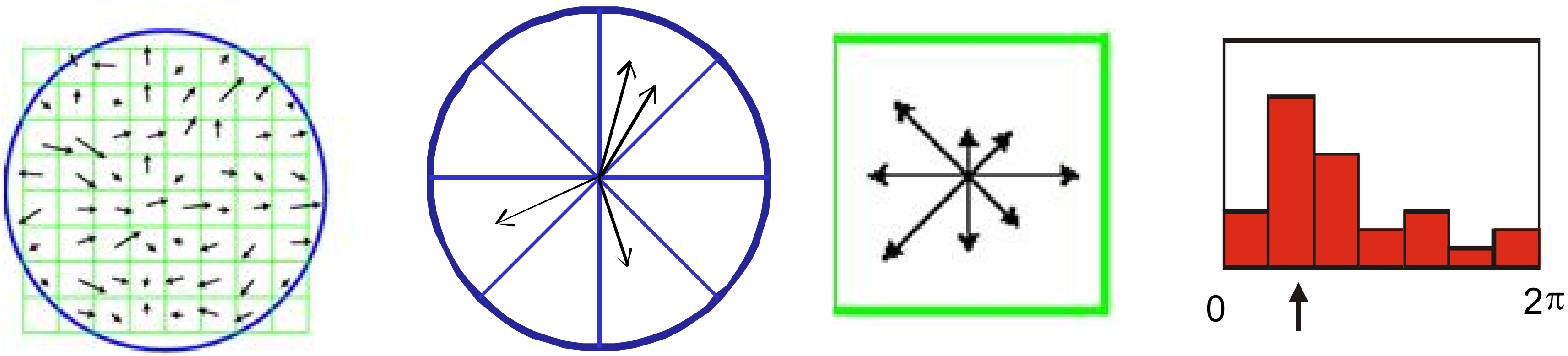
Вычисление градиента со сглаживанием

Признаки через градиенты



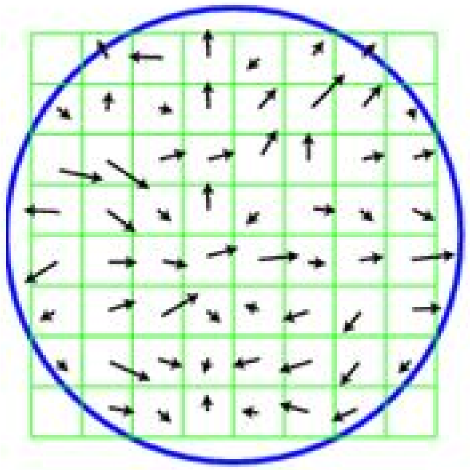
- Вычислим градиенты в каждом пикселе
  - Направление градиента
  - Силу градиента
- Какие признаки мы можем получить из градиентов?
  - Использовать напрямую (слишком много, неудобно)
  - Посчитать какие-нибудь статистики

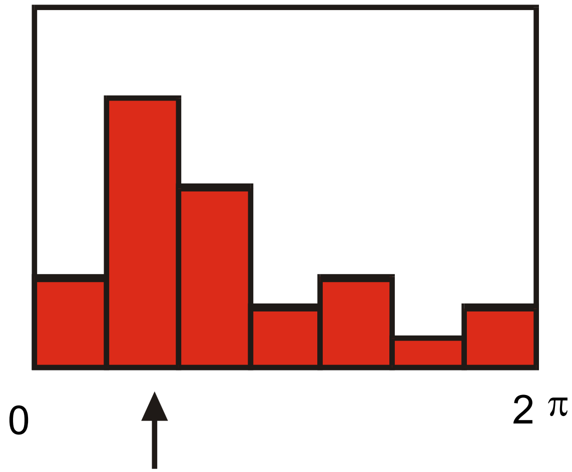
Гистограмма ориентаций градиентов



- Вычислим направление градиента в каждом пикселе
- Квантуем ориентации градиентов на 8 ячеек (направлений)
  - Пометим каждый пиксель номером ячейки
- Посчитаем гистограмму направлений градиентов
  - Для каждой ячейки посчитаем количество пикселов с номером этой ячейки

Гистограмма ориентаций градиентов





- Histogram of oriented gradients (HOG)
- Мощный класс признаков
- Очень широко используется при анализе изображений
- Устойчив к изменениям освещенности изображения
  - Т.к. считаем только направления градиентов

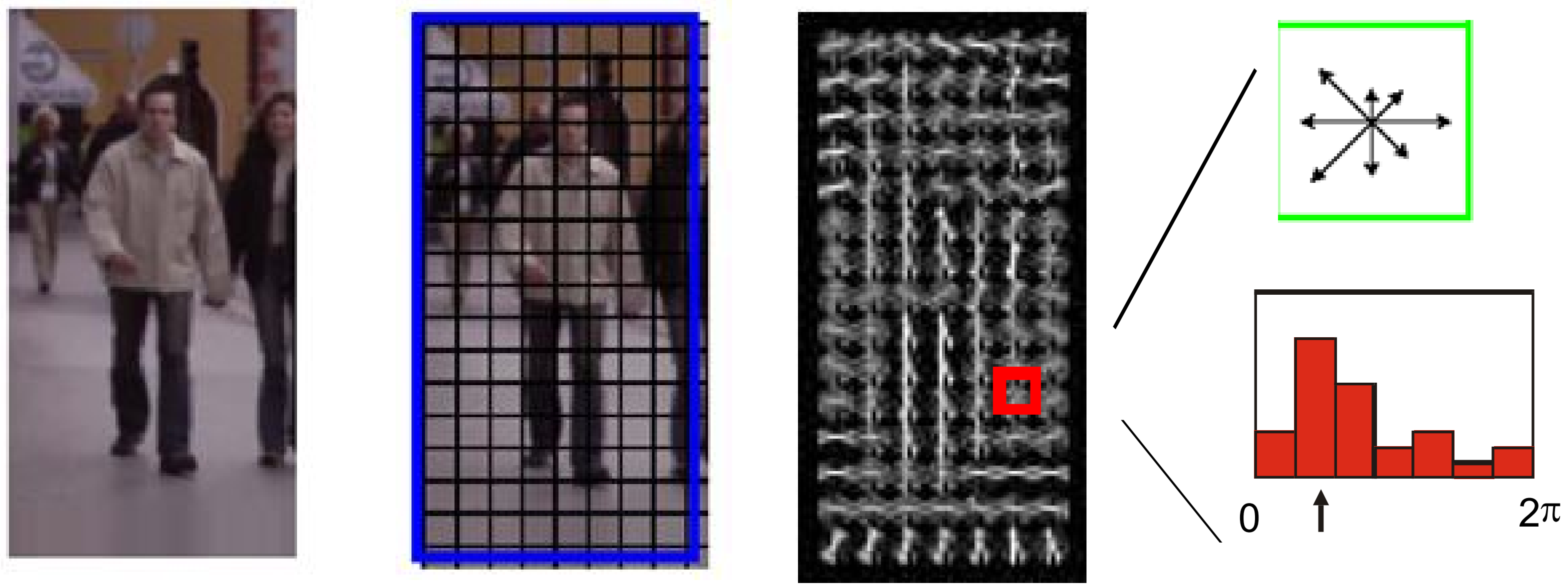
Алгоритм поиска пешеходов

- Алгоритм поиска:
  - Скользящее окно поиска
  - Вычисляем признаки («гистограммы ориентаций градиентов»
  - Бинарный классификатор «пешеход?» SVM

- Хотя схема HOG + SVM изначально была предложена для пешеходов, она успешно применялась в дальнейшем к разным категориями объектов

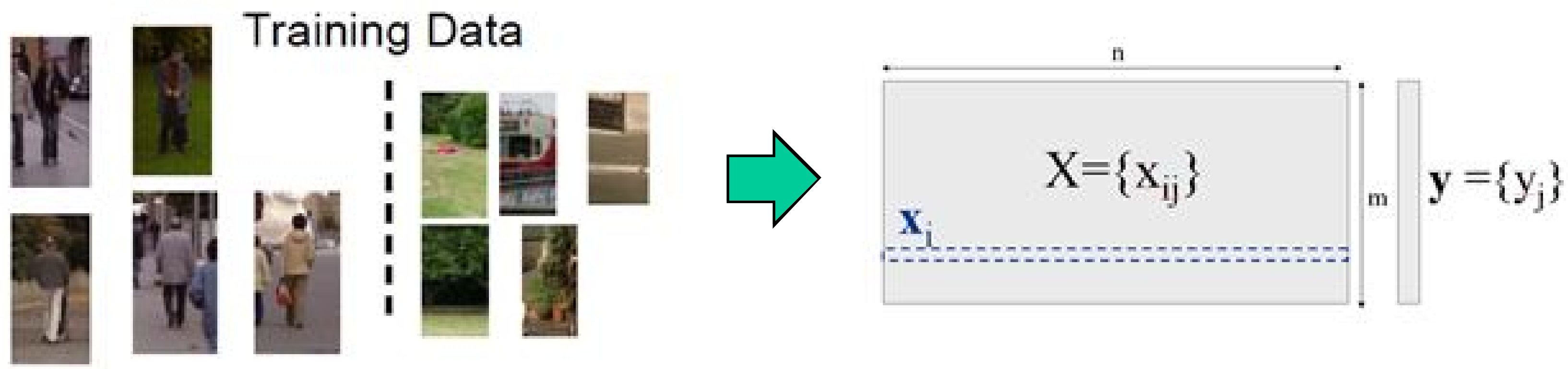
Navneet Dalal, Bill Triggs,Histograms of Oriented Gradients for Human Detection, CVPR 2005

Вычисление признаков



- Возьмём окно 64 x 128 пикселей
- Разобъём его на блоки 8 x 8 пикселей
- Всего будет 8 * 16 = 128 блоков
- В каждом блоке посчитаем гистограмму ориентаций градиентов с 8 ячейками (8 параметров)
- Всего у нас получится 128 x 8 = 1024 признака

Обучение классификатора



- Соберём обучающую выборку фрагментом изображения с пешеходами и без
- Для каждого фрагмента посчитаем вектор признак x и метку y
- На полученной обучающей выборке обучим линейный классификатор SVM

SVM

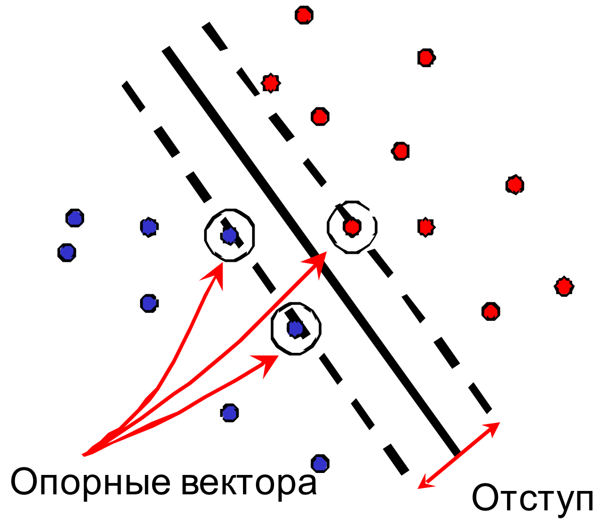
- Обучаем линейную SVM:

$$f(x) = w^T x + b$$

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

- Опорные вектора с положительными и отрицательными весами

- Чем фактически в нашем случае являются ***x*** ?



SVM

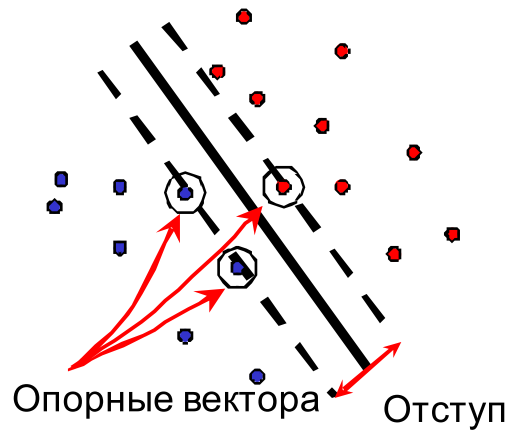
• Обучаем линейную МОВ:

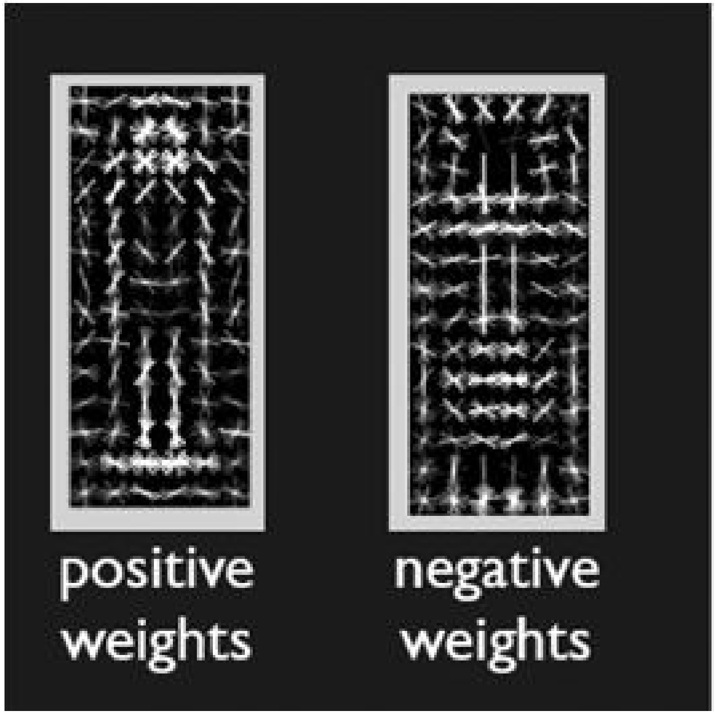
$$f(x) = w^T x + b$$

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

• Опорные вектора с положительными и отрицательными весами

• Линейную МОВ можно рассматривать как линейный фильтр изображения



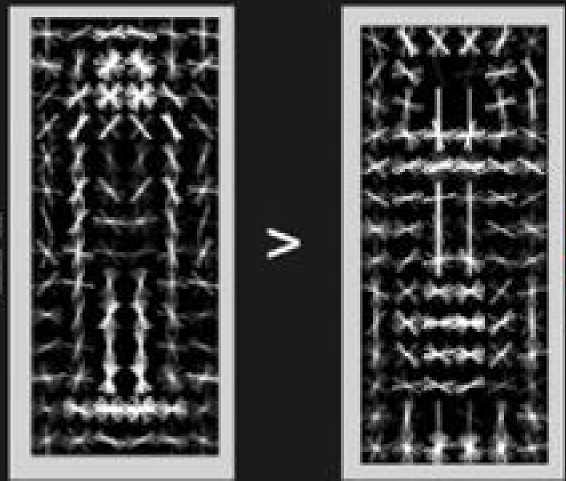


What do negative weights mean?

$$wx > 0$$

$$(w_+ - w_-)x > 0$$

$$w_+ > w_- x$$



Complete system should compete pedestrian/pillar/doorway models

Discriminative models come equipped with own bg

(avoid firing on doorways by penalizing vertical edges)

Slide from Deva Ramanan

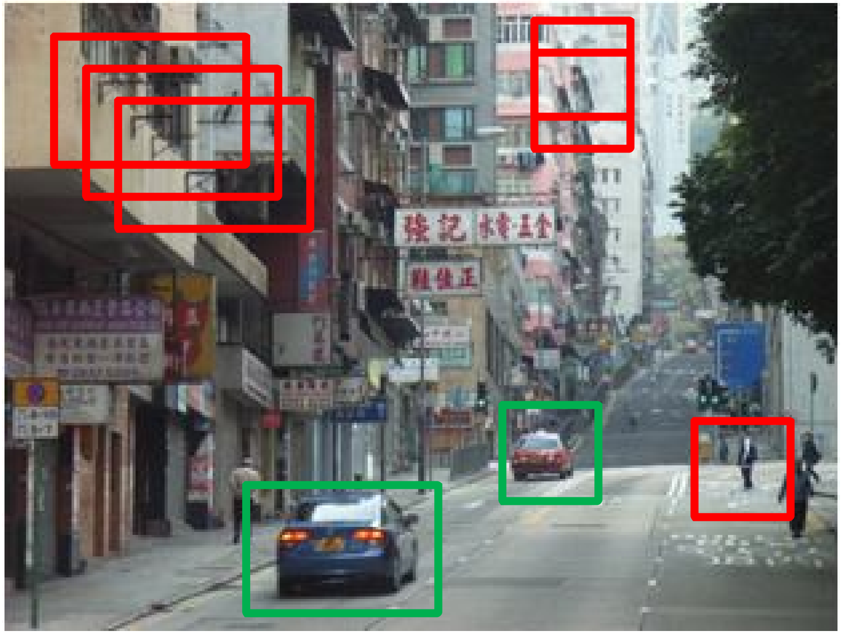
«Детектор пешеходов»

- Получили работающий «детектор пешеходов»
- Алгоритм:
  - Сканирующее окно
  - Вычисление признаков (гистограмм ориентаций градиентов)
  - Классификация с помощью SVM

- Метод работает достаточно неплохо, но неидеально. Надо улучшить.
- В чём проблемы?

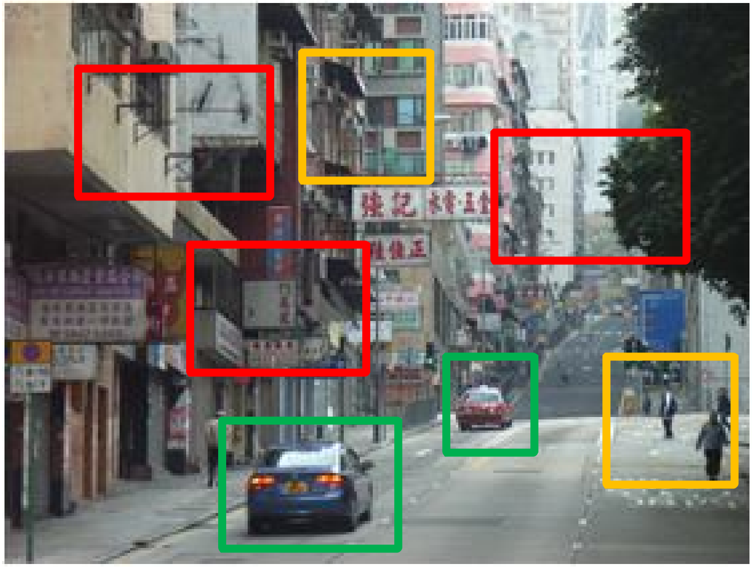
Свойства детекторов

- Поиск объектов ассиметричная задача: объектов гораздо меньше, чем «не-объектов»



- Классификатору нужна очень низкая ошибка 2 рода (мало ложных обнаружений)
- Класс «не объект» очень сложный – нужно много данных для обучения
- Объектов правильных тоже выделяется мало

Бутстраппинг (Bootstrapping)



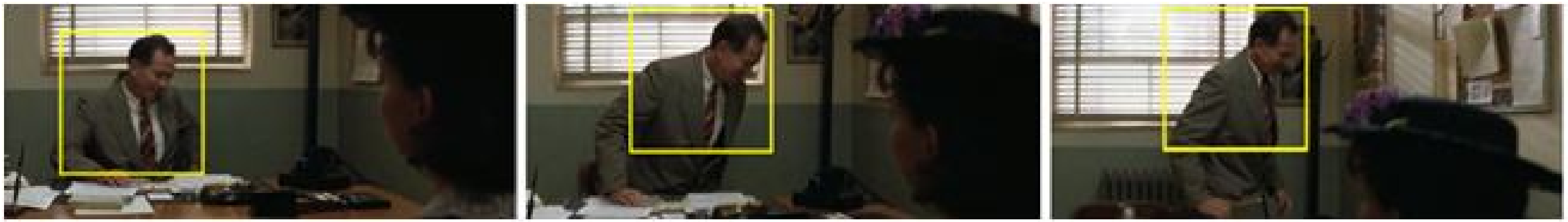
- Выбираем отрицательные примеры случайным образом
- Обучаем классификатор
- Применяем к данным
- Добавляем ложные обнаружение к выборке
- Повторяем

- Смысл:
  - Соберем ограниченную, но представительную выборку «не-обьектов»
  - Заставим классификатор сконцентрироваться на сложных отрицательных (**hard negative**) примерах

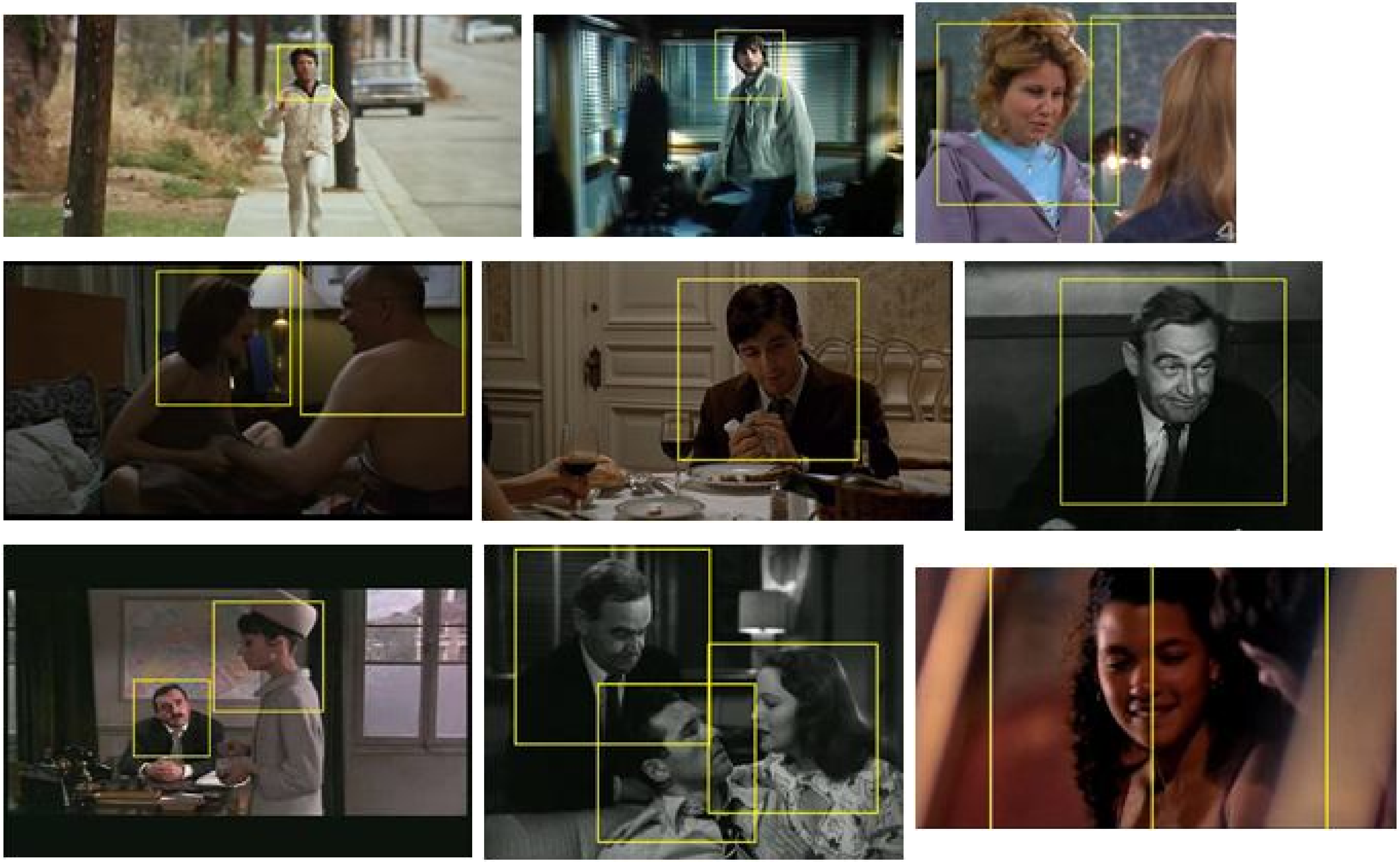
I. Laptev "Improvements of Object Detection Using Boosted Histograms« BMVC 2006

Пример

- Детектор «верхней части тела и головы»
- Обучающая выборка
  - 33 Hollywood2 фильмов
  - 1122 кадров с размеченным объектами
- Первая стадия (бутстреппинг)
  - 1607 размеченных объектов искажается (jittered) для получения 32k примеров
  - 55k отрицательных примеров выбирается из видео
- Вторая стадия (перетренировка)
  - 50k сильных отрицательных примеров выбираются



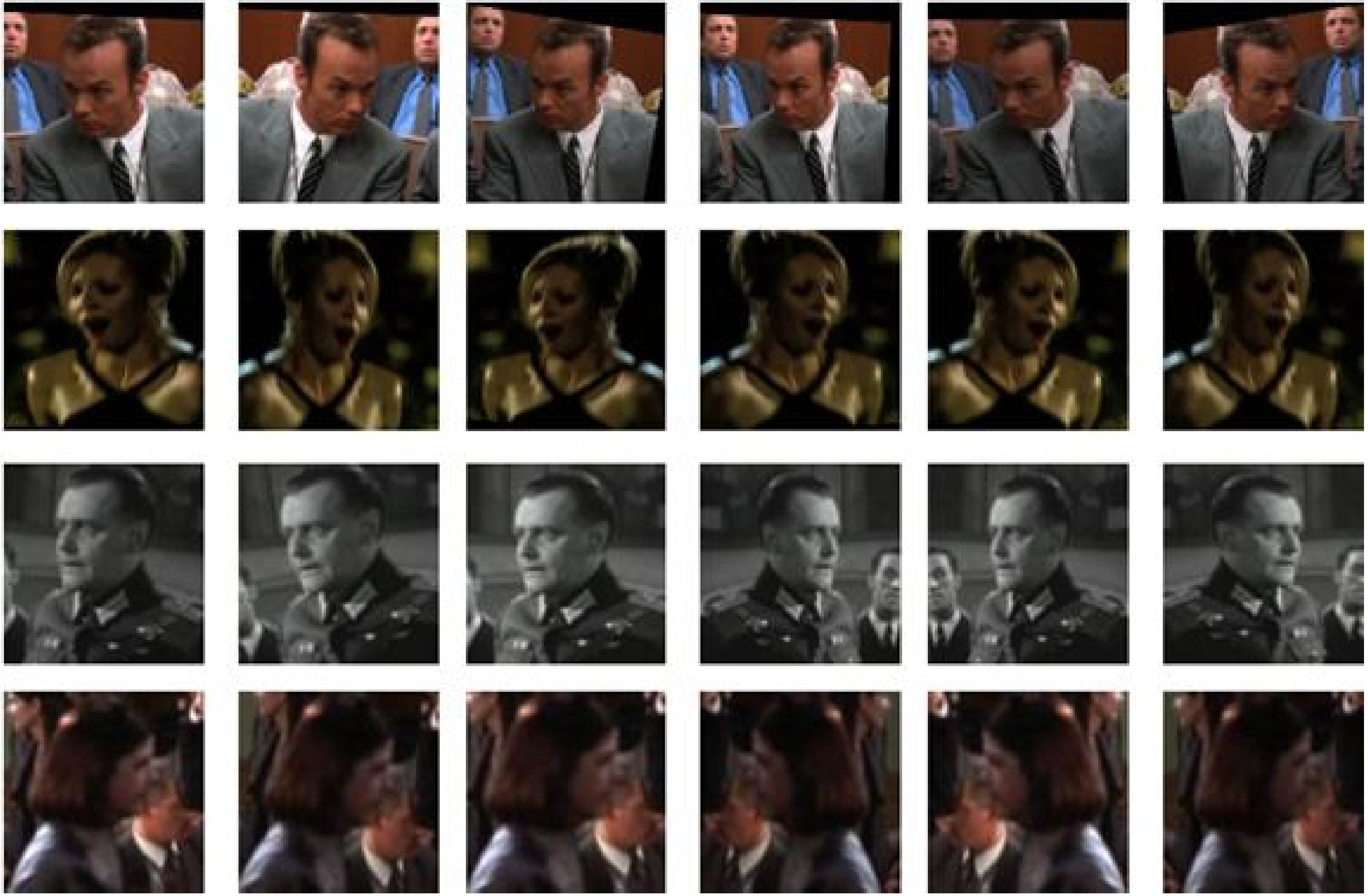
Положительные примеры



Положительные окна

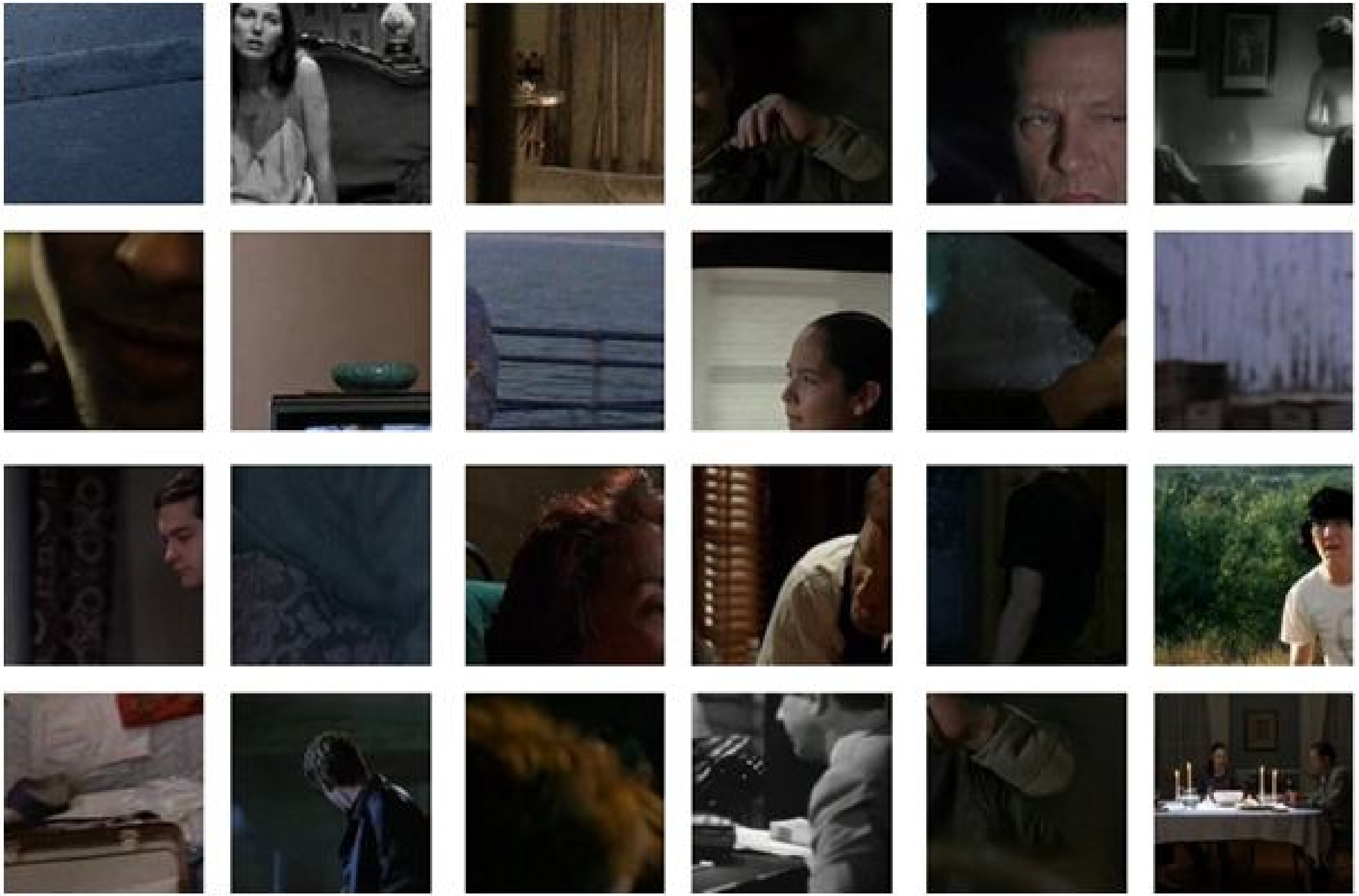
- Внимание: похожие положение и ориентация

Искаженные примеры



- Небольшие сдвиги, отображения, повороты, изменения масштаба

Random negatives

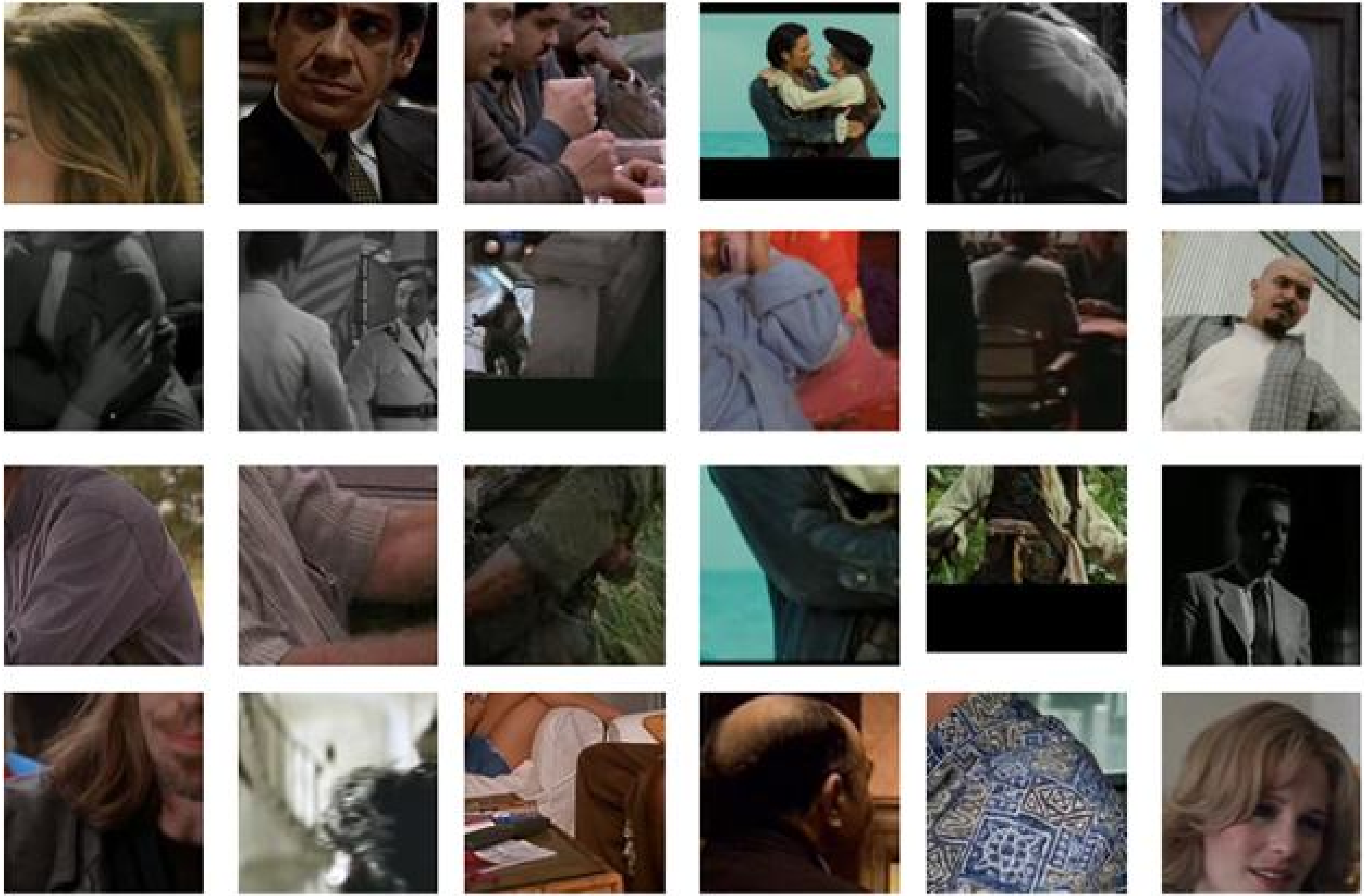


Первая стадия

Трудный отрицательный пример

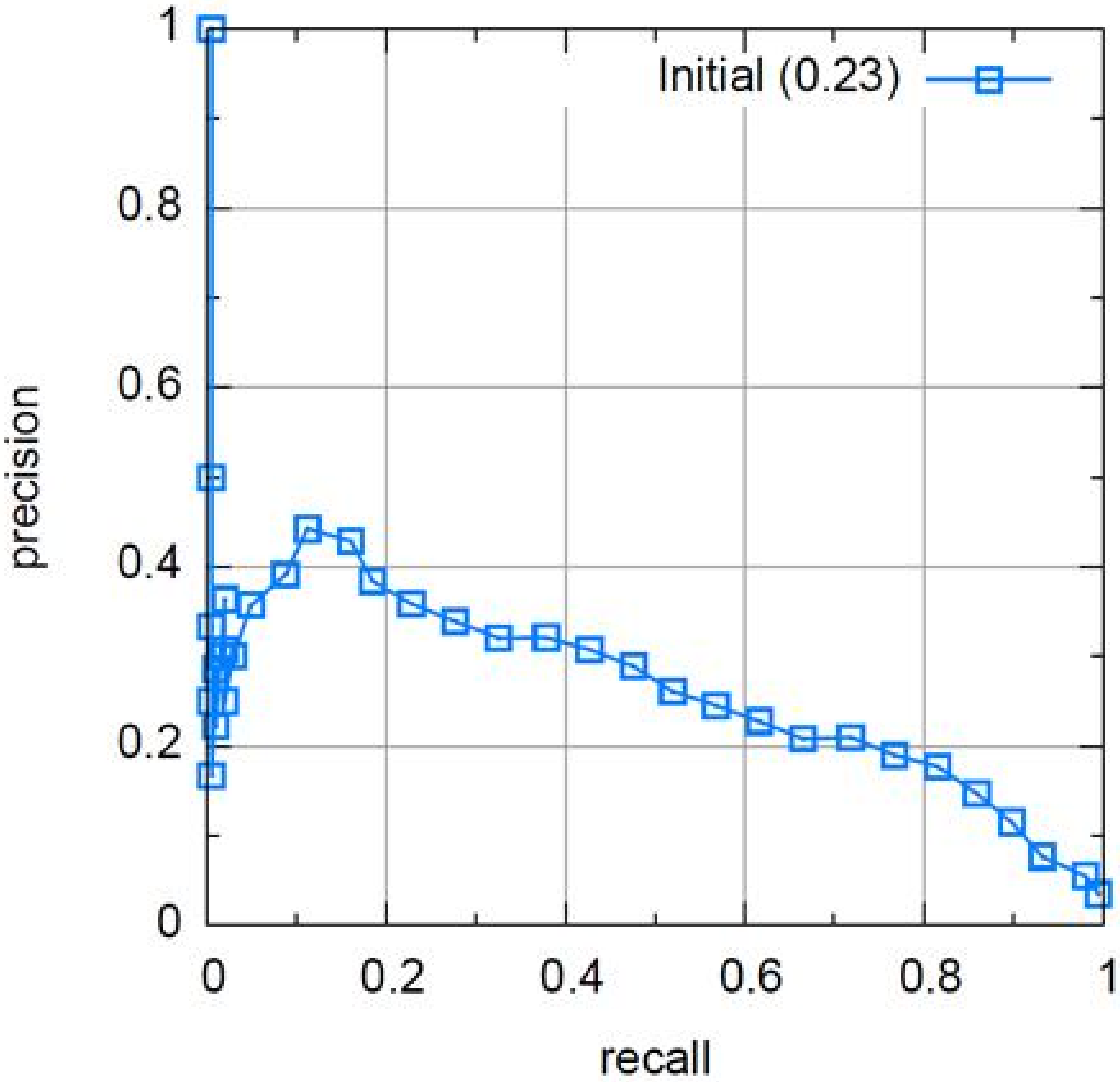
- Ищем ложные обнаружения с высоким рейтингом
- Используем их как трудные отрицательные примеры
- Затраты: # количество изображений x поиск в каждом

Трудные примеры

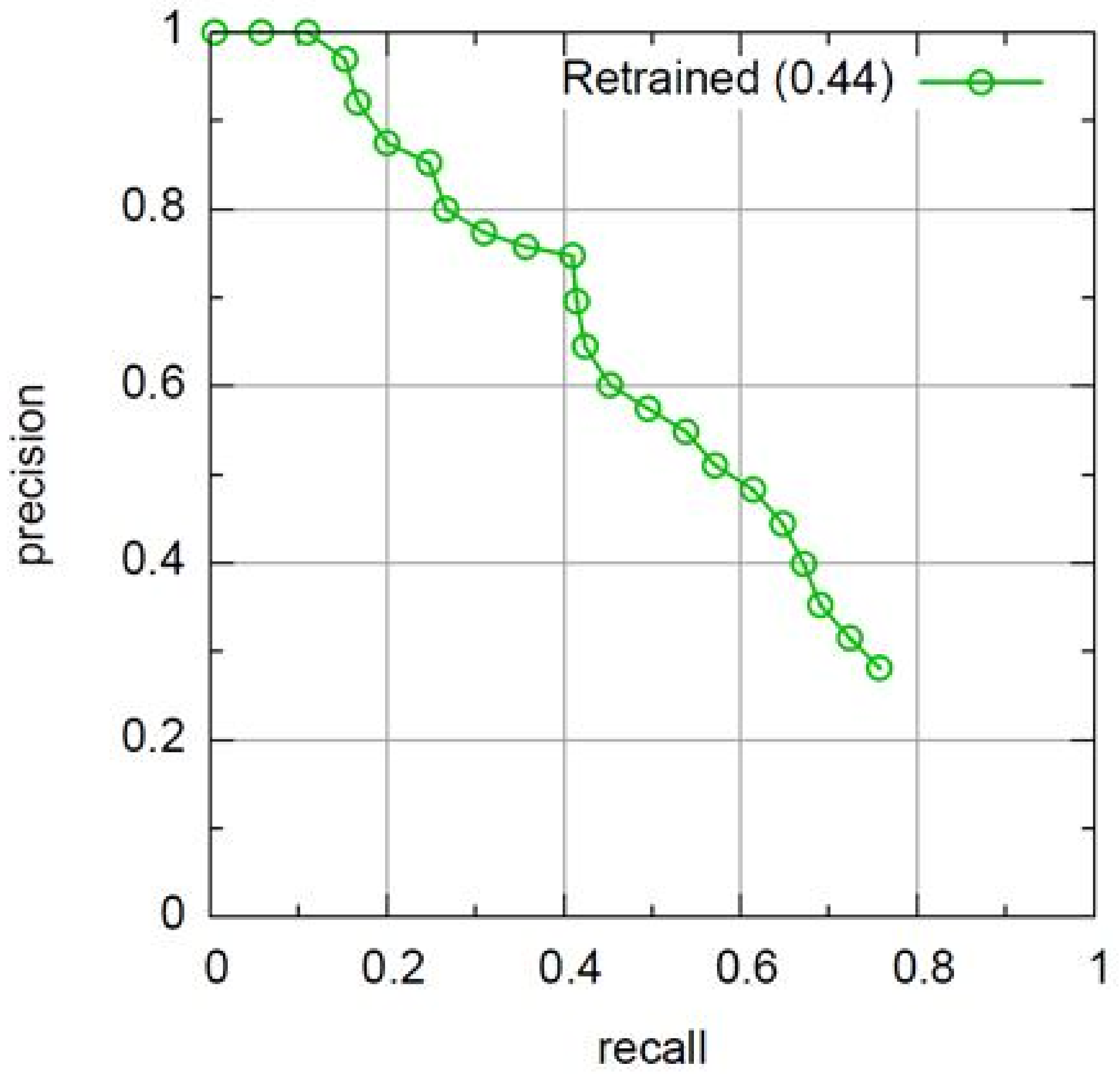


Трудные примеры

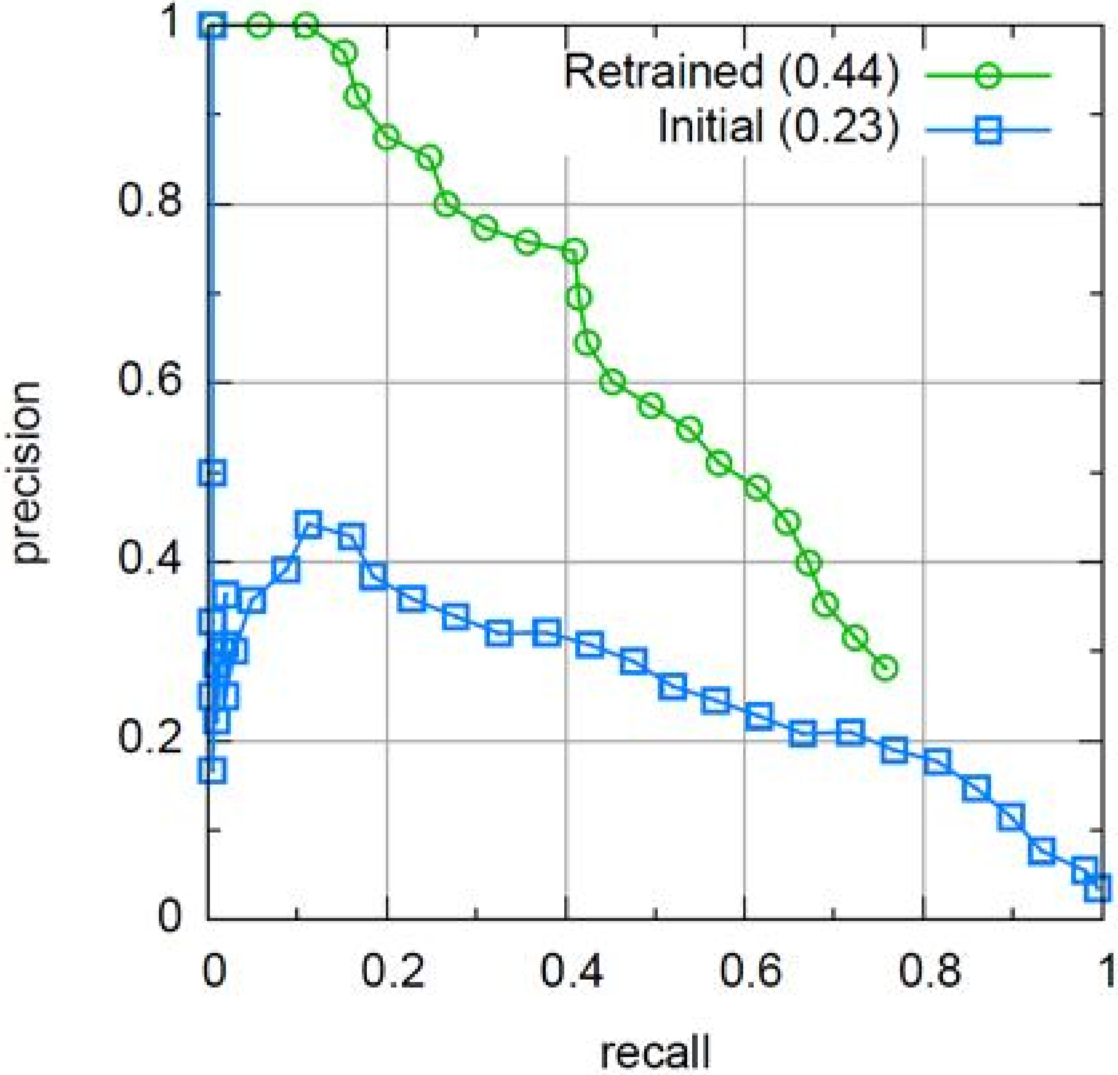
Измерение качества



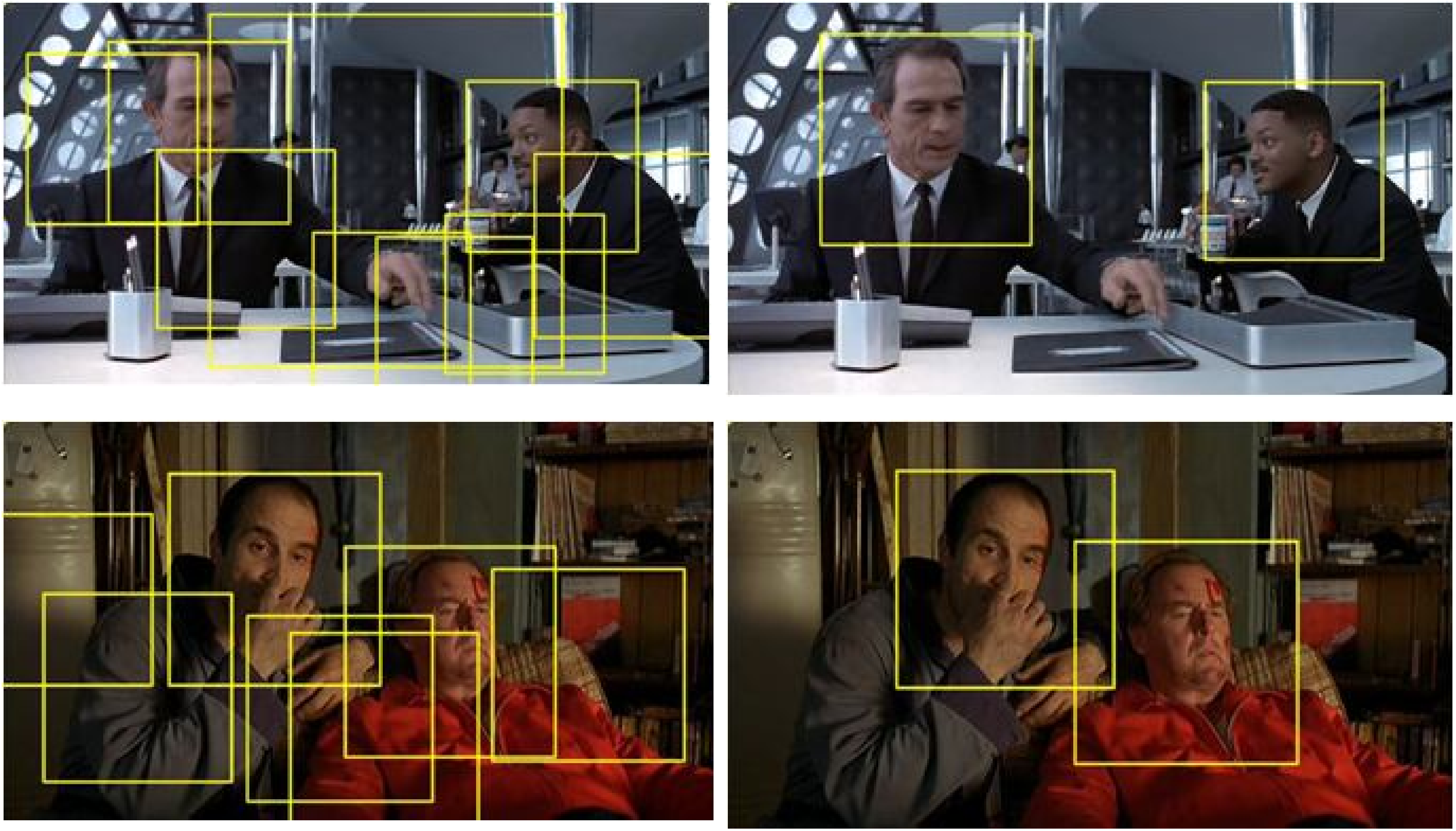
После перетренировки



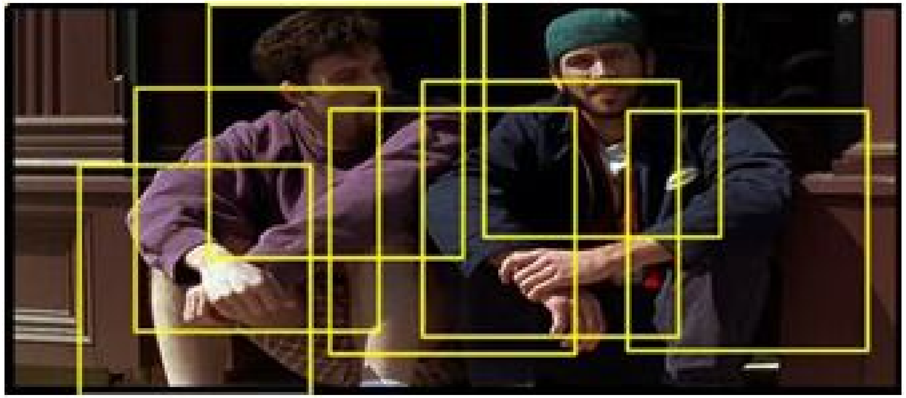
Сравнение

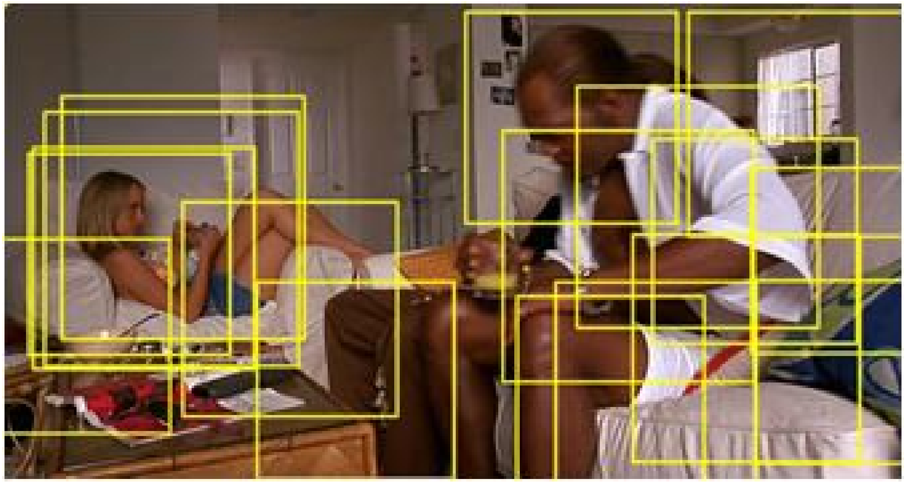


Сравнение

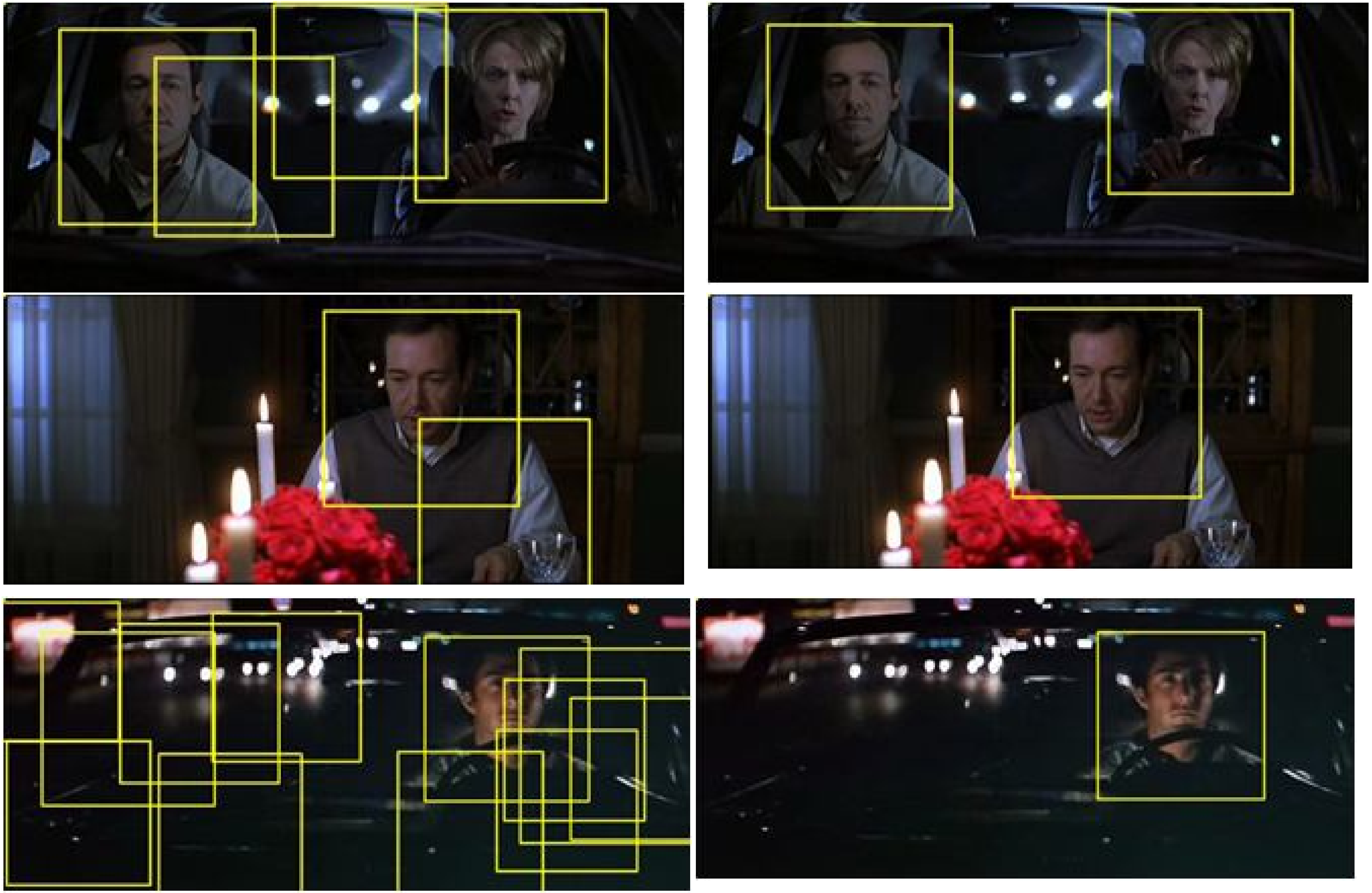


Сравнение

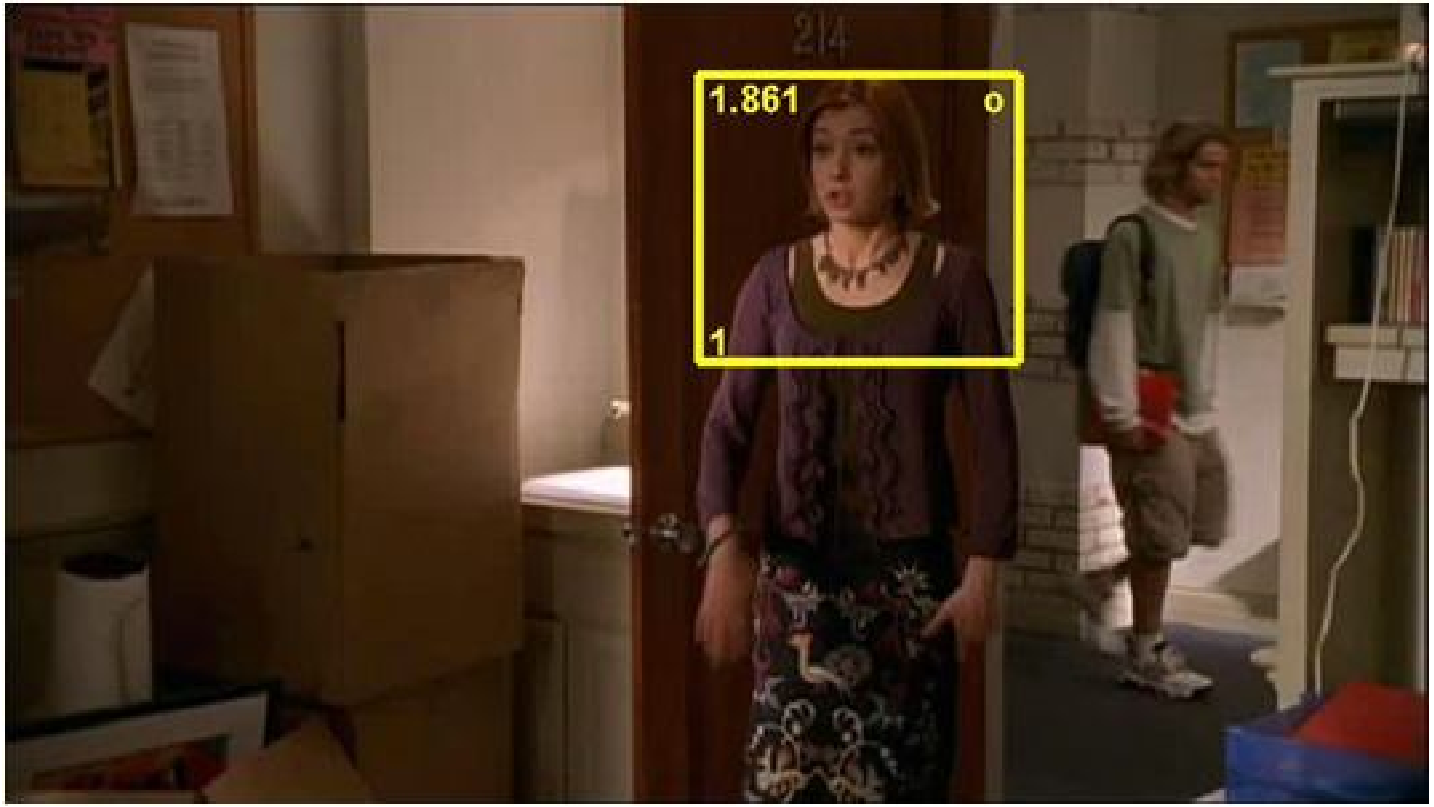





Сравнение



Пример



Резюме лекции

- Машинное обучение позволяет использовать большое количество неинтуитивных простых признаков

- Методы и понятия
  - SVM (Метод опорных векторов)
  - HOG (Гистограммы градиентов)
  - Скользящее окно
  - Бутстраппинг и размножение выборки