Лекция 13

Нерегулярности

Машинное обучение

Андрей Фильченков / Сергей Муравьёв

27.11.2020

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

- Слайды подготовлены с использованием материалов курсов
  - К.В. Воронцов «Машинное обучение»
  - В. Гулин «Data Mining»
  - А.
  - A. Ng’s “Deep Learning”
- Слайды доступны: **shorturl.at/ltVZ3**
- Видео доступны: **shorturl.at/hjyAX**

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Вспомним утиный тест

Как выглядели обучающие данные:

Много уток, **много не уток (неуток)**.

Как строился классификатор:

1. Для уток выделили **ключевые признаки**.
2. **Понятие похожести** используется для оценки близости признаков.
3. Логический сепаратор используется для классификации.

Что может пойти не так?

Мы предполагаем, что есть одинаковые по значимости классы с примерно одинаковым числом объектов, но это бывает не так:

1. Классы несбалансированы
2. Не все объекты размечены
3. В классе мало объектов
4. Классы могут добавляться
5. Классы не одинаковы по значимости
6. Метки расставлены неправильно

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Сценарий

В классах несбалансированное число объектов

При этом возможности дособрать и разметить дополнительные объекты исчерпаны

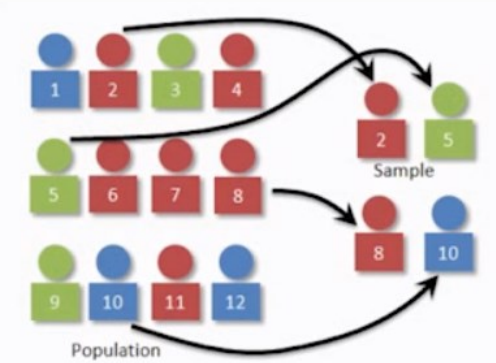
Две стратегии сэмплирования

Делать подвыборку большего класса (subsampling), или увеличить размер меньшего класса (upsampling)

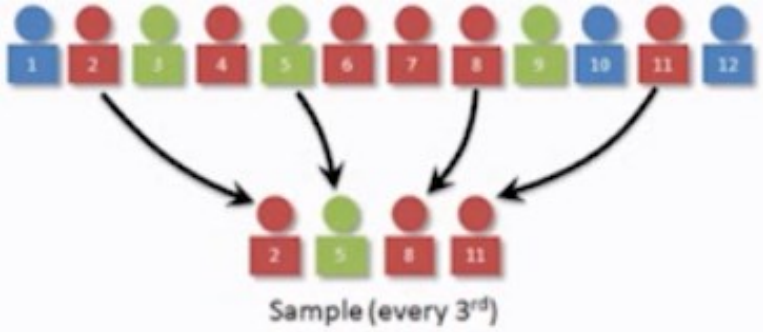
Subsampling делается в основном для исследования данных и валидации результатов

Стратегии сэмплирования

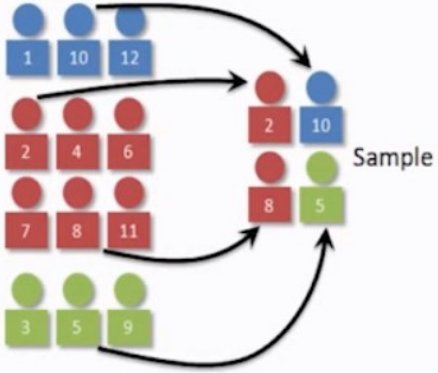
Простое случайное



Систематическое



Стратифицированное (stratified)



Кластерное

SMOTE

Synthetic Minority Over-sampling Technique (SMOTE)

1) Случайно выбрать точку a
2) Выбрать k ближайших точек из ее класса
3) Случайно выбрать из них одну, b
4) Случайно выбрать точку на отрезке (a, b)
5) Добавить ее с той же меткой класса, что и у a

Что бывает еще?

- Аугментация данных
- Синтетические данные, полученные при помощи генеративных моделей
- Использование данных с «грязными» (noisy/dirty) метками
- Дистилляция данных

Очень важное замечание

Все искусственные объекты можно использоваться только в тестовом множестве.

Что можно делать еще?

- Задавать веса объектам (потерям на объекте)
- Использовать правильные меры:
  - F-меру
  - Каппу Коэна:

$$\kappa = \frac{\mathrm{Acc} - \mathrm{Acc}_{\mathrm{chance}}}{1 - \mathrm{Acc}_{\mathrm{chance}}},$$

где $\mathrm{Acc}_{\mathrm{chance}} = \sum_{y \in Y} \mathrm{Recall}_y \cdot \mathrm{Precision}_y$

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Сценарий

Есть доступ к большому числу объектов, но не у всех есть метки.

Объекты обычно собирать значительно дешевле, чем размечать, потому что разметка зачастую требует привлечения человеческих ресурсов.

Постановка задачи

Задано обучающие множество

$$\mathcal{D} = \{(x_1, y_1), \dots, (x_l, y_l), x_{l+1}, \dots, x_{l+u}\} = \mathcal{D}_l \cup \mathcal{D}_u,$$

где $l \ll u$. Требуется восстановить $f: X \to Y$.

Простые, но неудачные подходы:

- решать как задачу обучения с учителем (на $\mathcal{D}_l$, отбросив $\mathcal{D}_u$)
- решать как задачу обучения без учителя (на $\mathcal{D}_l \cup \mathcal{D}_u$, отбросив метки).

Предположения

Предположение плавности (smoothness assumption): две точки в области высокой плотности, лежащие близко друг от друга, с большей вероятностью имеют одинаковые метки.

Предположение кластеризованности (cluster assumption): две точки из одного кластера с большей вероятностью имеют одинаковые метки.

Предположение многообразия (manifold assumption): многомерные данные из реального мира лежат в низкоразмерном многообразии внутри соответствующего многомерного пространства.

Алгоритм самообучения

Алгоритм самообучения (self-training):

Обучить некую a на $\mathcal{D}_l = (X_l, Y_l)$

Предсказать $a(x)$ для всех $x \in X_u$

Добавить $(x, a(x))$ к $\mathcal{D}_l$ и начать заново

Варианты:

- Добавлять наиболее достоверные $(x, a(x))$
- Добавлять все $(x, a(x))$
- Добавлять все $\big(x, a(x)\big)$, взвешенные с учетом достоверности

Анализ самообучения

Достоинства:

- Простота
- Может быть обёрткой для более сложных алгоритмов классификации

Недостатки:

- Негативное влияние ошибочных прогнозов усиливается с обучением.
- Трудность в достижении сходимости алгоритма

Сообучение

Идея: использовать несколько независимых моделей, обучаемых на разных группах признаков (**разделение признаков, feature split**).

В сценарии **сообучения (co-training)** два классификатора используют метки, поставленные друг другом.

В сценарии **многовидового обучения (multi-view learning)** обучается несколько классификаторов.

Анализ сообучения

Достоинства:

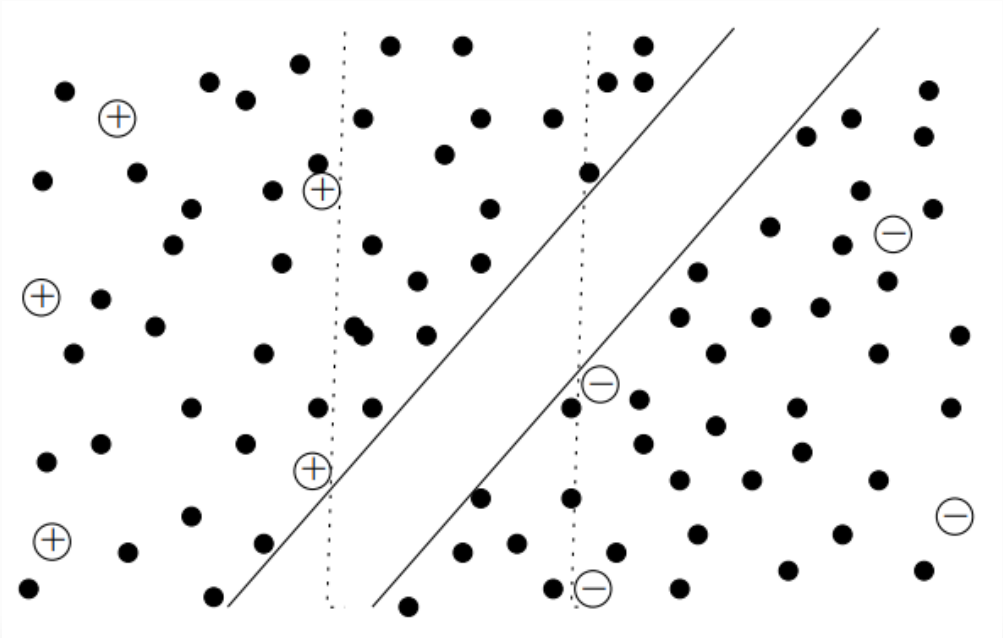
- Подходит почти ко всем известным классификаторам
- Менее чувствительно к ошибочным прогнозам

Недостатки:

- Естественное разделение признаков не всегда существует
- Разделение признаков ограничивает качество обучаемых моделей

Частичный метод опорных векторов

Частичный метод опорных векторов (semi-supervised support vector machine, S3VM, также Transductive SVM, T-SVM) максимизирует отступ и до неразмеченных данных



Функция потерь S3VM

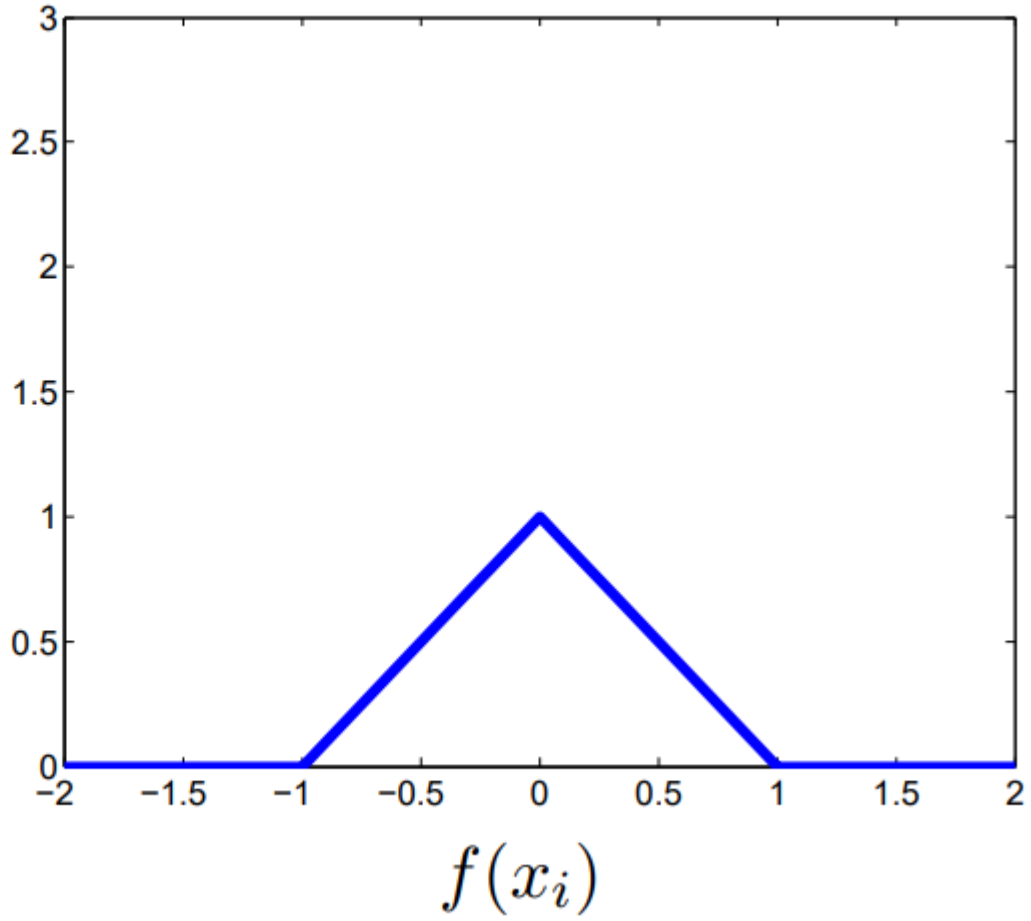
Функция потерь содержит расстояние и до неразмеченных вершин (**hat loss**):

$$\sum_{i=1}^{l}\left(1 - M_i(w, w_0)\right)_+ + \frac{1}{2C}||w||^2 +$$

$$+ \sum_{j=1}^{u}\left(1 - |M_j(w, w_0)|\right)_+ \to \min_{w,w_0}.$$

Hat loss

$$(1 - |f(x_i)|)_+$$



Проблемы Hat loss

Ее использование приводит к невыпуклой оптимизации

Подходы к решению:

- Использовать алгоритмы невыпуклой оптимизации
- Использовать сглаживание и градиентный спуск
- Использовать верхнюю оценку и разбить на несколько задач выпуклой оптимизации

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Сценарий

Есть доступ к большому числу объектов, но не у всех есть метки

Данные собираются быстро, а размечаются медленно и порционно, скорость обучения моделей происходит быстрее, чем разметка

Активное обучение

В **активном** обучении условия такие же, как в частичном обучении, но можно задавать Оракулу вопросы о значении меток.

Для $\mathcal{D} = \{(x_1, y_1), \dots, (x_l, y_l), x_{l+1}, \dots, x_{l+u}\} = \mathcal{D}_l \cup \mathcal{D}_u$, где $l \ll u$, восстановить $f: X \to Y$ за наименьшее число обращений к Оракулу (найти стратегию обращений к Оракулу, оптимизирующую качество аппроксимации f).

Выбор по степени неуверенности

Выбор по степени неуверенности (uncertainty sampling)

Идея: спрашивать про объекты, про которые меньше всего уверенности.

Пусть $\hat{y} = \arg\max_y P_\theta(y|x)$ — это наиболее вероятный класс для текущей модели с параметрами θ.

Варианты

Минимальная уверенность (last confident):

$$x_{LC}^* = \arg\max_x 1 - P_\theta(\hat{y}|x)\,,$$

По отступу (margin sampling):

$$x_{MS}^* = \arg\max_x P_\theta(\hat{y}_1|x) - P_\theta(\hat{y}_2|x)\,,$$

Максимальная энтропия (maximum entropy):

$$x_{ME}^* = \arg\max_x -\sum_i P_\theta(y_i|x)\log P_\theta(y_i|x)\,.$$

Отбор по несогласию в комитете

Отбор по несогласию в комитете (query-by-committee)

Пусть есть комитет натренированных моделей $C = \{\theta_{(1)}, \dots, \theta_{(|C|)}\}$. Каждая голосует за объекты из H.

Идея: спрашивать про объекты, на которых наименьшее согласие.

Энтропия голосования:

$$x^*_{VE} = \arg\max_{x \in H} -\sum_i \frac{V(y_i)}{|C|} \log \frac{V(y_i)}{|C|},$$

где $V(y_i)$ число голосов на метку y_i для объекта x.

Ожидаемое изменение модели

Ожидаемое изменение модели (expected model change)

Идея: спрашивать про объекты, которые приведут к наибольшему изменению ошибки:

$$x^*_{EGL} = \arg\max_x -\sum_i P_\theta(y_i|x)\|\nabla\mathcal{L}(x, y_i)\|\,.$$

Ожидаемое сокращение ошибки

Ожидаемое сокращение ошибки (expected error reduction)

Идея: спрашивать про объекты, которые позволят уменьшить ошибку.

Минимизация ожидания точности:

$$x^*_{0/1} = \arg\max_x \sum_i P_\theta(y_i|x) \sum_{x' \in H} 1 - P_{\theta^+(x', y_i)}(\hat{y}|x'),$$

Минимизация ожидания перекрестной энтропии:

$$x^*_{0/1} = \arg\max_x \sum_i P_\theta(y_i|x) \left(-\sum_j \sum_{x' \in H} P_+(j) \log P_+(j) \right),$$

где $P_+(j) = P_{\theta^+(x', y_j)}(y_j|x')$

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Сценарий

- Число классов может увеличиться
- В некоторых классах сравнительно мало объектов

Обучение на одном примере

Обучение на одном примере (one short learning) это постановка, в которой алгоритм должен дообучиться классификации на новый класс, содержащий всего один объект.

Обучение на нескольких примерах (few short learning) предполагает все то же, но с несколькими объектами.

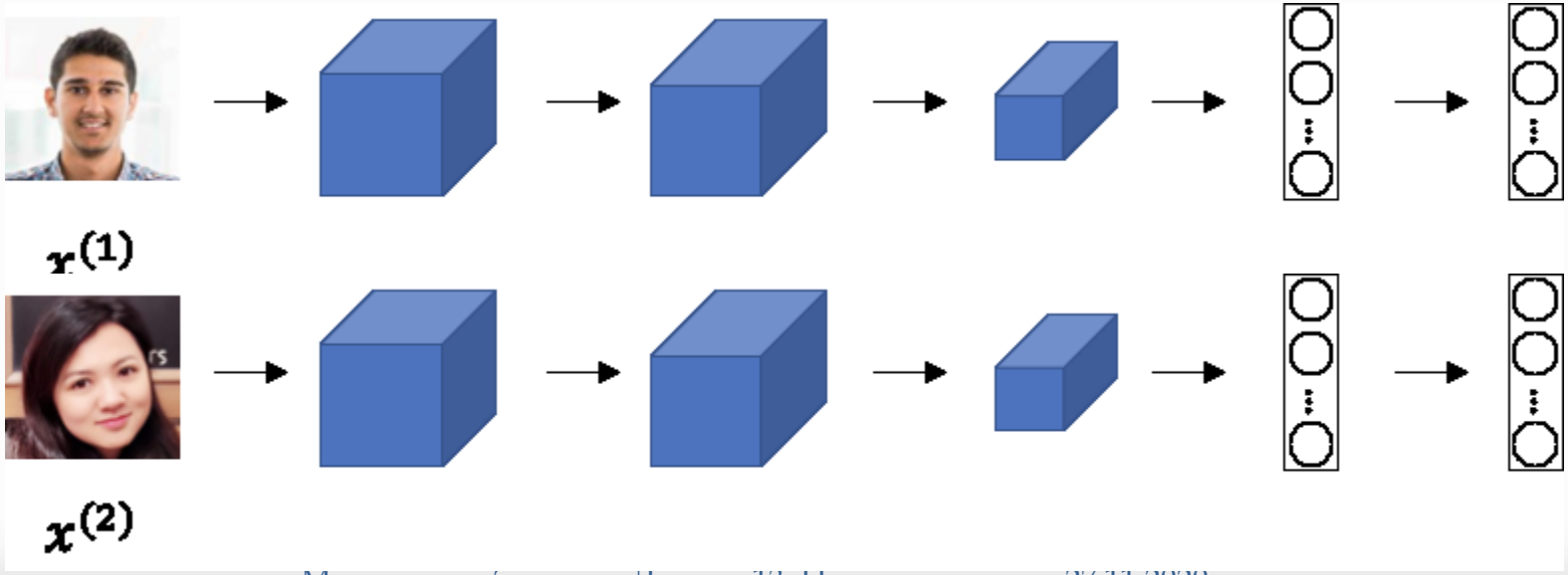
Добавление центроидов

Обучение на одном примере основано на метрической классификации.

Добавление центроида не заставляет изменять другие центроиды. Однако необходимо, чтобы выученная метрика позволяла хорошо по ним классифицировать.

Сиамская сеть

Сиамская сеть (Siamese network) состоит из двух идентичных сетей, возвращающих векторные представления входов



Triplet loss

Для обучения используется **triplet loss**:

$$\mathcal{L}(a, p, n) = \max(\text{dist}(a, p) - \text{dist}(a, n) + \varepsilon, 0)$$

Обучение и вывод

Обучение. Сеть обучается по батчам троек градиентным спуском

Вывод. Для каждого класса хранится объект (центроид), с которыми сравнивается вход по косинусному расстоянию.
Чтобы добавить класс, добавляется новый объект в качестве центроида.

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Сценарий

Много объектов, но мало меток, и мы хотели бы эффективно предобучиться на объектах без меток

Самообучение

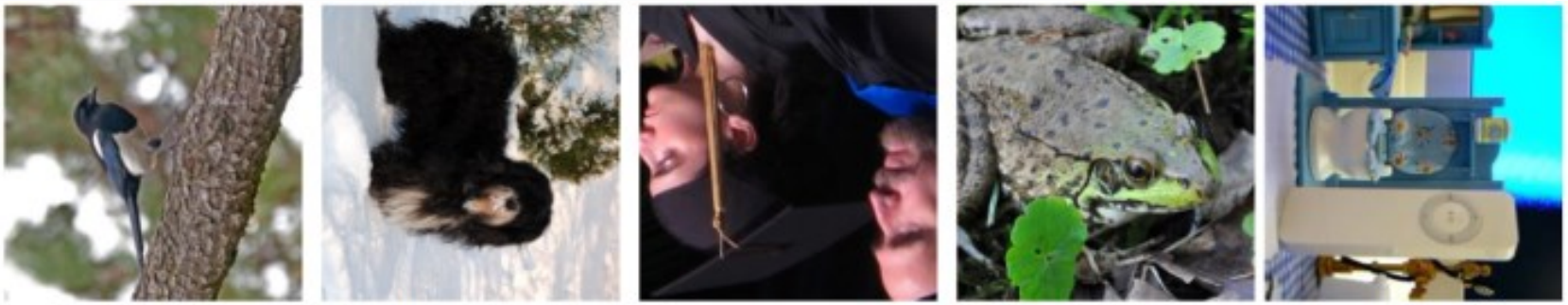
Самообучение (self-supervised learning) — подход, в котором для обучения представлений используются задачи, в которых разметку можно получить автоматически

Предварительная задача (pretext task) — задача с искусственно созданными метками (псевдо-метками), на которой обучается модель, чтобы выучить хорошие представления (representations) объектов.

Последующая задача (downstream task) — целевая задача, для которой используются полученные представления с простым дискриминатором.

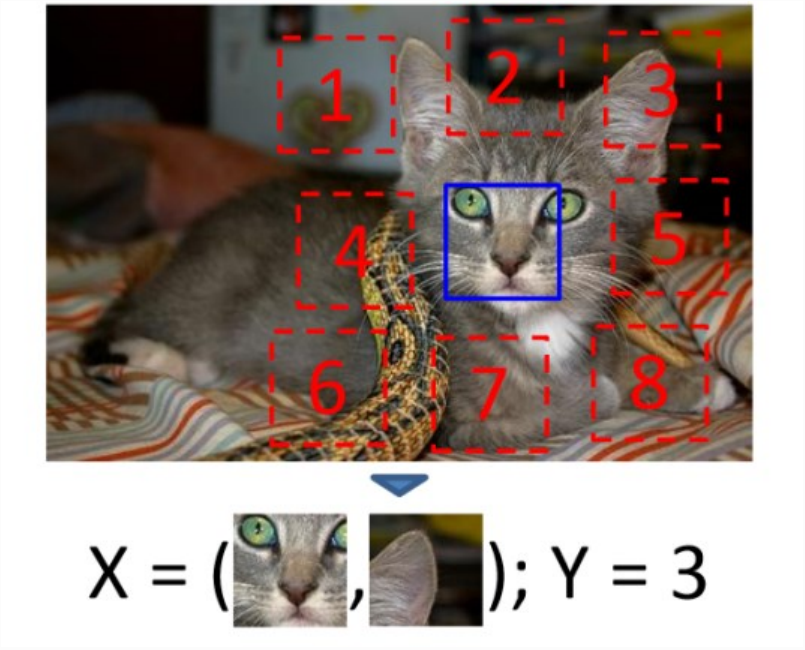
Определение поворота

Нужно предсказать, как повернут кадр



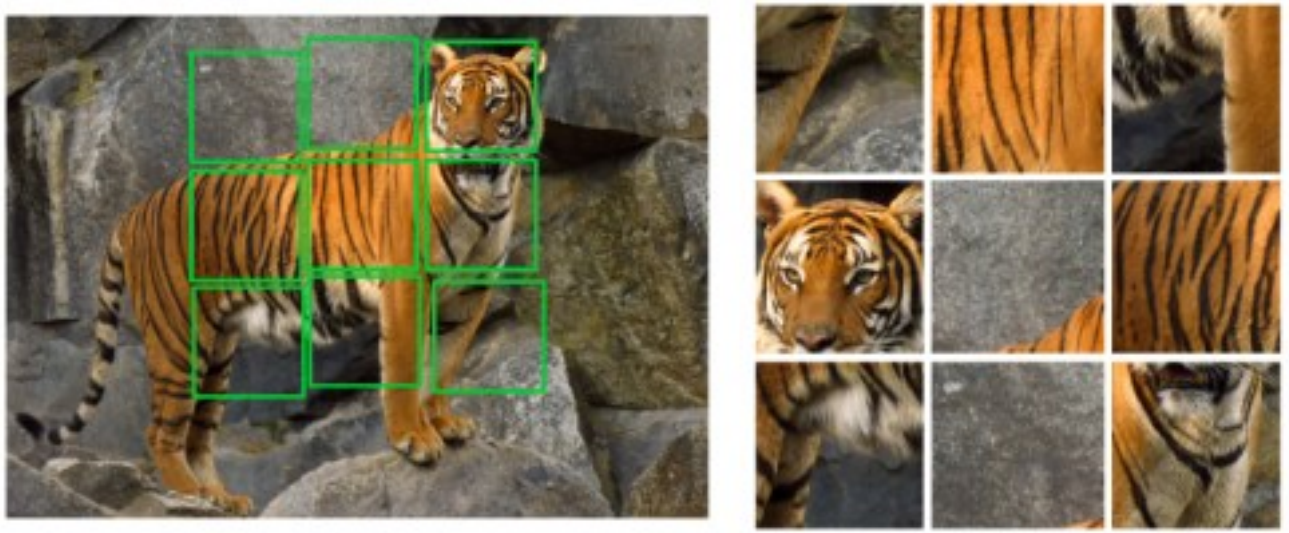
Предсказание контекста

Нужно предсказать, как расположен патч относительно центрального



Решение головоломки

Нужно определить перестановку патчей, чтобы восстановить правильный порядок



Что еще?

Восстанавливать часть изображения

Восстанавливать цвет изображения

Восстанавливать детали изображения

…

Предсказывать слово по контексту

Предсказывать контекст по слову

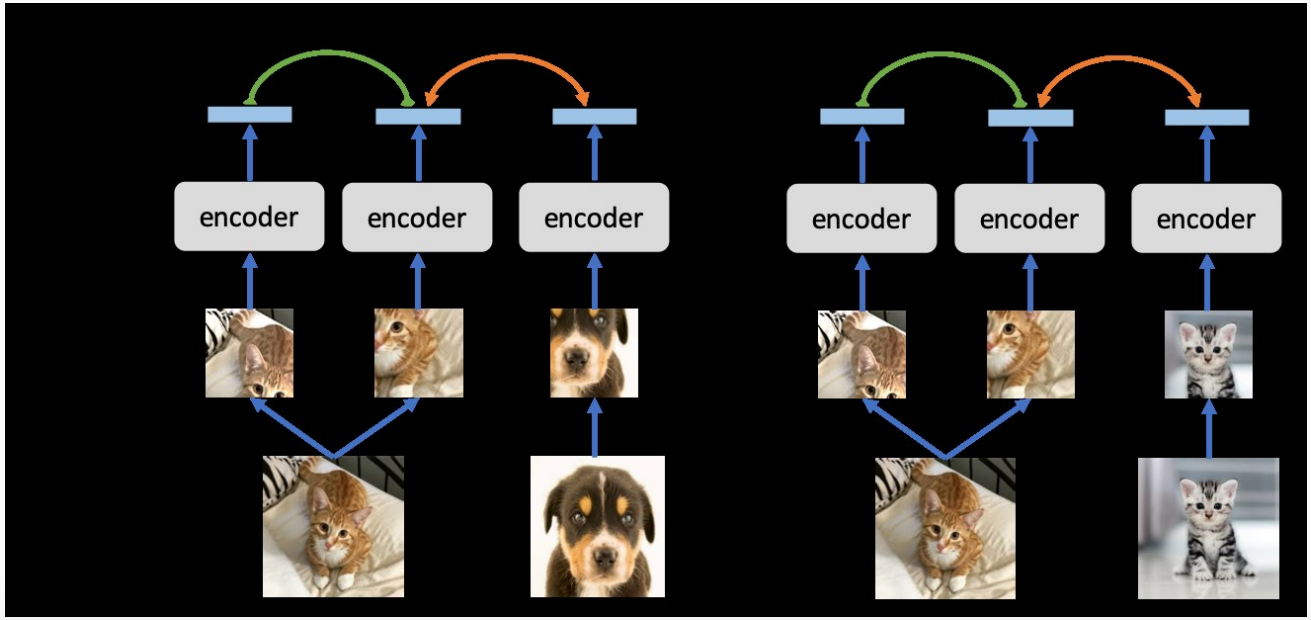
…

Сравнительное обучение

Сравнительное обучение (contrastive learning) — обучение за счет объединения аугментированных или иным образом полученных трансформаций в один класс с исходным объектом в парадигме метрической классификации.

Пример: использование кропов

Мы должны выучить векторное представление объектов, удовлетворяющее тому, что части объектов более похожи на объект, чем любой другой объект



План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Одноклассовая классификация

На самом деле утиный тест про всего лишь один класс. Второй класс определяется через непринадлежность к первому. Это — **задача одноклассовой классификации (one-class classification)**.

Поиск аномалий (anomaly / exception / surprise **detection)** — это задача одноклассовой классификации.

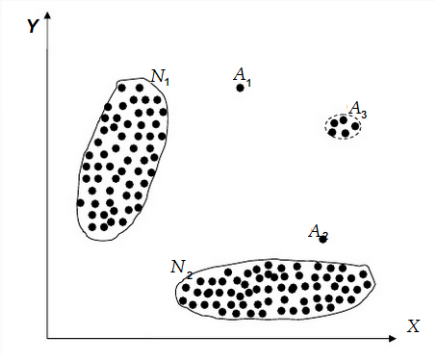
Новизна и выбросы

Детектирование выбросов (outlier detection) — определение того, что объекты в выборке аномальны

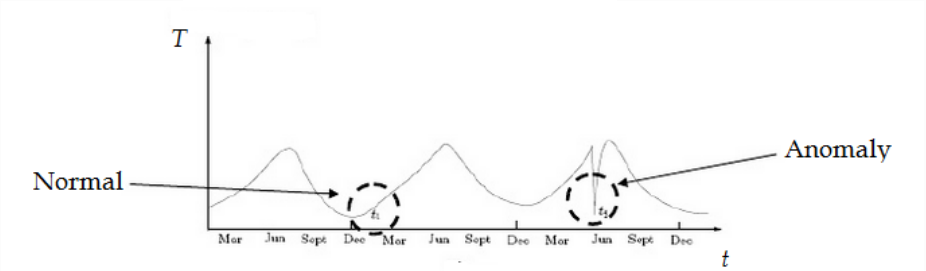
Детектирование новизны (novelty detection) — определение того, что новые поступающие объекты аномальны

Типы аномалий

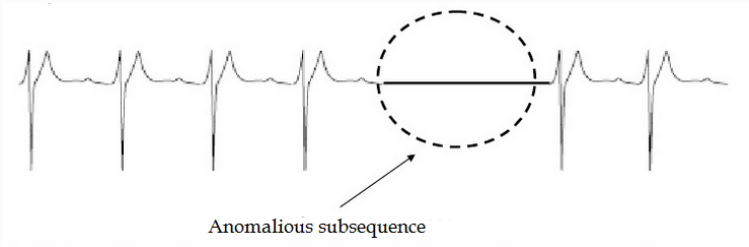
- Точечная аномалия



- Контекстуальная аномалия



- Коллективная аномалия



Подходы к поиску аномалий

- **методы на основе плотности**: аномалии находятся в регионах низкой плотности
- **методы на основе расстояний**: аномалии удалены от других точек
- **специфические алгоритмы**

Одноклассовый SVM

Будем отделять выборку от начала координат

Работает с ядром радиально-базисных функций

Заточен именно под поиск аномалий, потому что воспринимает весь набор данных

План лекции

- Нерегулярность в данных
- Сэмплирование данных
- Частичное обучение
- Активное обучение
- Обучение на одном примере
- Самообучение
- Поиск аномалий
- Обработка выбросов

Что делать с выбросами?

Простой вариант — выкинуть из выборки и обучить модель на чистых данных.

Более сложный вариант — обучать модель с учетом того, что выбросы все-таки есть.

Ошибка как индикатор выброса

Идея: чем больше ошибка на объекте

$$\varepsilon_i = \text{LOO}(x_i) = \mathcal{L}(a(x_i; \mathcal{D}\backslash\{(x_i, y_i)\})),$$

тем с большей вероятностью он является выбросом и тем меньше мы хотим его учитывать.

Это можно добавить в качестве веса к непараметрической регрессии:

$$\mathcal{L}(a, \mathcal{D}) = \sum_{i=1}^{|\mathcal{D}|} K(\varepsilon_i)(f(x_i, \theta) - y_i)^2 \to \min_{\theta}.$$

Локально взвешенное сглаживание

Локально взвешенное сглаживание (LOcally WEighted Scatter plot Smoothing, LOWESS) для непараметрической регрессии:

$\varepsilon_i = |a - y_i|$ будет функцией потерь;

возьмем квартичное ядро $K(\varepsilon_i) = K_Q\left(\frac{\varepsilon_i}{6\text{med}\{\varepsilon_i\}}\right)$.

Будем итеративно повторять обучение модели и обновление ε, пока они не стабилизируются.

Робастная регрессия

Идея: поменяем асимптотику функции потерь, чтобы ограничить влияние выбросов

Функция Мешалкина:

$$\mathcal{L}(a,x) = 1 - \exp\left(-\frac{1}{2\sigma}\big(a(x) - y(x)\big)^2\right).$$