Машинное обучение. Обзорная лекция

Алексей Колесов

Белорусский государственный университет

10 ноября 2019 г.

Содержание

1. Обзор пройденного материала
   - Теория
   - Алгоритмы
2. Избранные главы ML
   - Кластеризация
   - Понижение размерности
   - Алгоритмы
   - Другие направления
3. Избранные главы NN

Что такое машинное обучение

Том Митчелл

A computer program is said to learn from experience **E**, with respect to some task **T**, and some performance measure **P**, if its performance on **T** as measured by **P** improves with experience **E**.

Некоторые понятия

Обобщающая способность — качество программы показывать хорошее качество на примерах, которые она не видела раньше

Inductive bias — набор предположений (априорных знаний), который используется для предсказания неизвестных значений

Для успешного обучения использование априорных знаний неизбежно (No Free Lunch theorem).

Зачем нужно машинное обучение

- задачи, которые сложно запрограммировать
  - сложноформализуемые задачи (например, распознавание символов, речи, вождение автомобиля)
  - задачи неподвластные человеку (анализ астрономических данных, ранжирование веб-страниц)
- задачи, для которых нужна адаптация

Минимизация эмпирического риска

Модель: алгоритм принимает S, полученный из распределения D и размеченный функцией f. Его задача найти гипотезу $h_S : X \to Y$, который минимизирует ошибку $L_{D,f}(h_S)$ по отношению к **неизвестным** D и f.

- D и f неизвестны $\Rightarrow L_{D,f}(h_s)$
- давайте использовать ошибку на тренировочной выборке (**empirical risk, empirical error**):

$$L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$$

Минимизация эмпирического риска — парадигма обучения, заключающаяся в выборе гипотезы, минимизирующей ошибку на тренировочной выборке

ERM with inductive bias

- ERM-правило приводит к переобучению
- Вместо того, чтоб не использовать его, найдём случаи, когда это правило работает достаточно хорошо
- Хороший способ — ограничить набор гипотез
- H — семейство гипотез из X в Y;

$$\mathrm{ERM}_H(S) \in \operatorname*{argmin}_{h \in H} L_S(h)$$

- Один из важных вопросов машинного обучения: «для каких H ERM_H не переобучается»

Agnostic PAC-leanable for generalized loss functions

Класс гипотез H называют **агностически вероятно приблизительно верно изучаемым** (agnostic PAC-learnable) по отношению к множеству Z и функции потерь $l : H \times Z \to \mathbb{R}_+$, если существует такая функция $m_H : (0,1)^2 \to \mathbb{N}$ и алгоритм, такой что

- для любых $\epsilon, \delta \in (0,1)$
- для любого распределения D над Z

если мы выполним алгоритм на выборке из $m \geqslant m_H(\epsilon, \delta)$ независимых одинаково распределённых элементов из D, то алгоритм вернёт гипотезу $h \in H$ такую, что с вероятностью как минимум $1 - \delta$, выполняется $L_D(h) \leqslant \min\limits_{h' \in H} L_D(h') + \epsilon$, где $L_D(h) = \mathop{\mathbb{E}}\limits_{z \sim D}[l(h, z)]$

Неравномерная изучаемость

Гипотеза h называется (ϵ, δ)-**конкурентной с гипотезой** h' ((ϵ, δ)-competitive), если $\mathbb{P}[L_D(h) \leqslant L_D(h') + \epsilon] > 1 - \delta$

Класс гипотез H называют **неравномерно изучаемым** (nonuniform learnable) если существует такая функция $m_H^{NUL} : (0,1)^2 \times H \to \mathbb{N}$ и алгоритм, такой что

- для любых $\epsilon, \delta \in (0,1)$
- для любой $h' \in H$
- для любого распределения D над X

если мы выполним A на выборке из $m \geqslant m_H^{NUL}(\epsilon, \delta, h')$ независимых элементов из D, то с вероятностью как минимум $1 - \delta$, выполняется $L_D(A(S)) \leqslant L_D(h') + \epsilon$

Характеризация классов с неравномерной изучаемостью

Критерий неравномерной изучаемости

Класс гипотез H является неравномерно изучаемым, тогда и только тогда, когда H — объединение не более чем счётного множества PAC-изучаемых классов H_i.

Теорема о связи равномерной сходимости и неравномерной изучаемости

Пусть $H = \bigcup_{n \in \mathbb{N}} H_n$, где каждый H_n обладает свойством равномерной сходимости. Тогда H — неравномерно изучаемый

RLM

Будем искать решение вот так:

$$\text{argmin}_w(L_S(w) + R(w))$$

- $R(w)$ может отражать «сложность» гипотезы
- $H = \bigcup_i \{w : R(w) \leqslant i\}$ (см. SRM)
- $R(w) = \lambda||w||^2$ — регуляризация Тихонова
- гребневая регрессия использует регуляризацию
- выбор регуляризации — наложение prior distribution на w

Алгоритм Perceptron

Алгоритм 1 Batch perceptron

Вход: **Разделимая** тренировочная выборка $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$

Выход: w, такой что $y_i \langle w, x_i \rangle > 0 \ \forall i = 1, \ldots, m$

1: $w^{(1)} = (0, \ldots, 0)$
2: **for** $t = 1, 2, \ldots$ **do**
3: **if** $\exists i$, т.ч. $y_i \langle w^{(t)}, x_i \rangle \leqslant 0$ **then**
4: $w^{(t+1)} = w^{(t)} + y_i x_i$
5: **else**
6: **return** $w^{(t)}$
7: **end if**
8: **end for**

МНК = ММП для линейной регрессии

Теорема о ММП оценке в случае гауссовского шума

Пусть разметочная функция f имеет вид:

$$f(x_i) = h_\alpha(x_i) + \epsilon_i$$

где ϵ_i — независимые нормальные случайные величины с нулевым средним и дисперсией σ_i^2. Тогда МНК-решение и ММП-оценка для α совпадает, в случае, если веса объектов w_i обратно пропорциональны дисперсии шума σ_i^2

Алгоритм

Алгоритм 2 AdaBoost

Вход: $S = ((x_1, y_1), \ldots, (x_m, y_m))$; $y_i \in \{-1; +1\}$

for $i = 1, \ldots, m$ **do**
    $D_1 = \frac{1}{m}$
end for
for $t = 1, \ldots, T$ **do**
    $h_t =$ базовая гипотеза с ошибкой $\epsilon_t = \mathbb{P}_{D_t}[h_t(x_i) \neq y_i]$
    $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
    $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$
    for $i = 1, \ldots, m$ **do**
        $D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
    end for
end for
return $A = \text{sign}(\sum_{i=1}^{t} \alpha_t h_t)$

Алгоритм

Алгоритм 3 Stochastic gradient descent для минимизации $f(w)$

Вход: $\eta > 0,\ T > 0$

$w^{(1)} = 0$
for $t = 1, \dots T$ **do**
    v_t — случайный вектор, т.ч. $\mathbb{E}[v_t | w^{(t)}] \in \partial f(w^{(t)})$
    $w^{(t+1)} = w^{(t)} - \eta v_t$
end for
return $\overline{w} = \sum_{t=1}^{T} w^{(t)}$

SGD для Soft-SVM

Алгоритм 4 SGD для Soft-SVM

Вход: $T > 0$

$\theta^{(1)} = 0$
for $t = 1, \dots T$ **do**
    $w^{(t)} = \frac{1}{\lambda t}\theta^{(t)}$
    выбрать i равновероятно из $[m]$
    if $y_i \langle w^{(t)}, x_i \rangle < 1$ **then**
        $\theta^{(t+1)} = \theta^{(t)} + y_i x_i$
    else
        $\theta^{(t+1)} = \theta^{(t)}$
    end if
end for
return $\overline{w} = \sum_{t=1}^{T} w^{(t)}$

Содержание

1. Обзор пройденного материала
   - Теория
   - Алгоритмы
2. Избранные главы ML
   - Кластеризация
   - Понижение размерности
   - Алгоритмы
   - Другие направления
3. Избранные главы NN

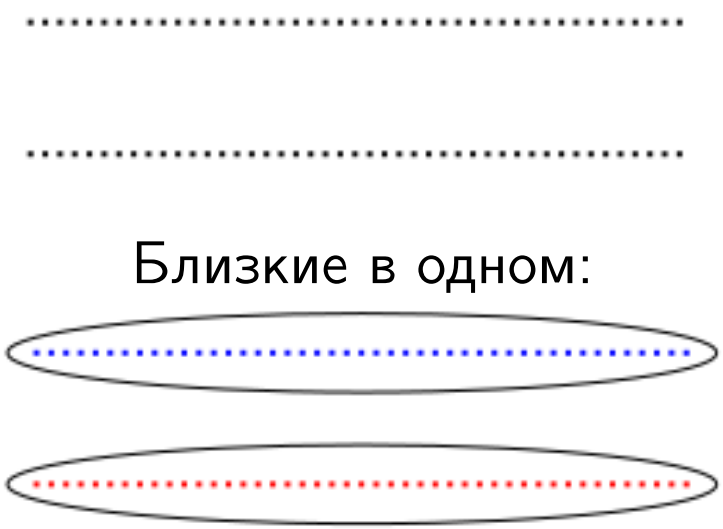
Кластеризация

- кластеризация — группировка множества объектов таким образом, чтоб похожие объекты были в одной группе (кластере), а непохожие — в разных
- применяется в анализе данных, как один из первых этапов
- магазины кластеризуют покупателей по покупкам; астрономы — звёзды по близости друг к другу; биологи — гены по их показателям в экспериментах

Сложность кластеризации

- кластеризация преследует две цели:
  - близкие объекты — в одном классе
  - далёкие — в разных
- близость — не транзитивное понятие
- разбиение на кластеры — отношение эквивалентности
- можно предложить последовательность $x_1, \ldots, x_m$, что x_i близка к соседям, но x_1 далёк от x_m

Пример

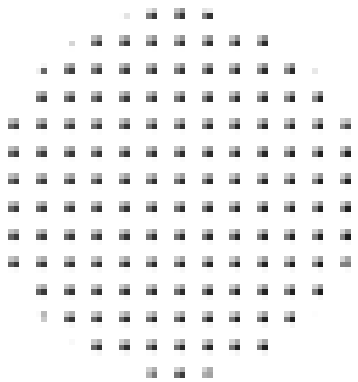
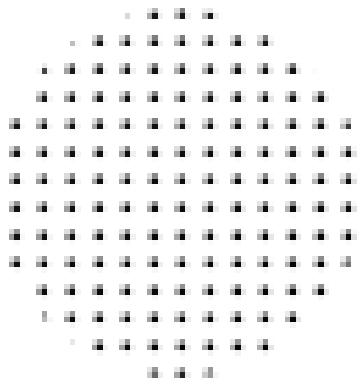
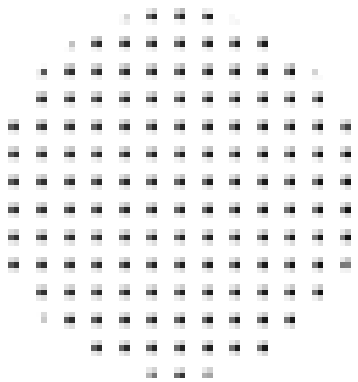
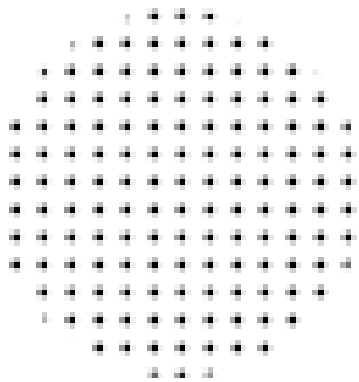


Отсутствие ground truth

- вторая проблема — отсутствие ground truth
- в supervised learning мы можем оценить качество решения по тренировочной выборке
- в кластеризации нет чёткого критерия успеха (что такое «правильная» кластеризация?)

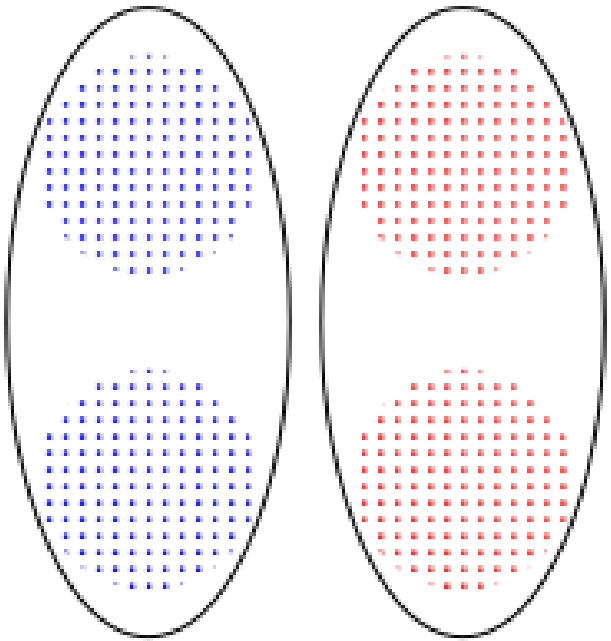
Отсутствие ground truth

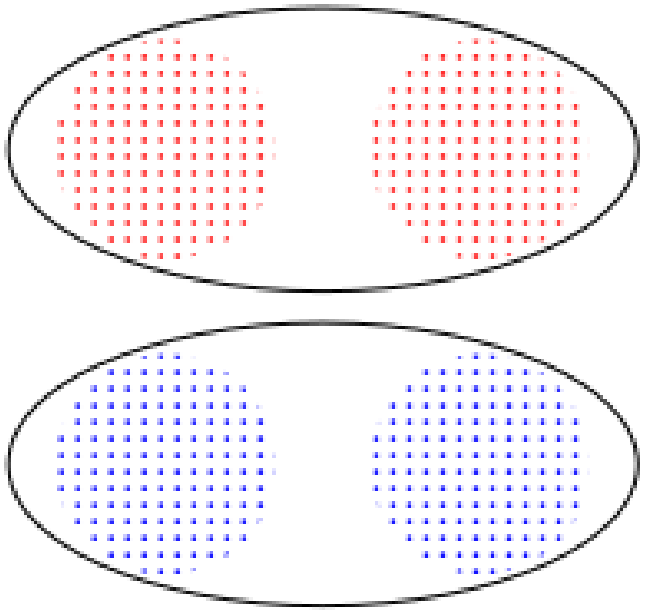
Пусть хотим кластеризовать:

Отсутствие ground truth

Какой вариант выбрать?





Отсутствие ground truth

Проблема встречается и в приложениях:

- как кластеризовать речь: по акценту или по содержанию?
- как кластеризовать фильмы: по жанру или по рейтингу?

Алгоритмы кластеризации

- linkage-based
- минимизация стоимости кластеризации
- спектральные методы

k-means

- в k-means каждый кластер C_i представляется своим центроидом: μ_i
- предполагается, что $\mu_i \in X'$, $X \subseteq X'$, d расширяется на X'
- $\mu_i(C_i) = \underset{\mu \in X'}{\operatorname{argmin}} \sum_{x \in C_i} d(x, \mu)^2$
- $G_{\text{k-means}}((X, d), (C_1, \ldots, C_k)) = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i(C_i))^2$
- можно переписать:

$$G_{\text{k-means}}((X, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in X'} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2$$

Часто применяемые стоимости

- $G_{\text{k-means}}(\ldots) = \min\limits_{\mu_1,\ldots,\mu_k \in X'} \sum\limits_{i=1}^{k} \sum\limits_{x \in C_i} d(x, \mu_i)^2$
- $G_{\text{k-medoids}}(\ldots) = \min\limits_{\mu_1,\ldots,\mu_k \in X} \sum\limits_{i=1}^{k} \sum\limits_{x \in C_i} d(x, \mu_i)^2$
- $G_{\text{k-medians}}(\ldots) = \min\limits_{\mu_1,\ldots,\mu_k \in X} \sum\limits_{i=1}^{k} \sum\limits_{x \in C_i} d(x, \mu_i)$
- $G_{\text{SOD}}(\ldots) = \min\limits_{\mu_1,\ldots,\mu_k \in X} \sum\limits_{i=1}^{k} \sum\limits_{x,y \in C_i} d(x, y)$

Алгоритм k-means

Алгоритм 5 k-means

Вход: $X \subset \mathbb{R}^n$,

Вход: k — количество кластеров

1: Случайно выбрать начальные центроиды: $\mu_1, \ldots, \mu_k$
2: **while** не сошлось **do**
3: $C_i = \{x \in X : i = \text{argmin}_j \|x - \mu_j\|\}, \forall i \in [k]$
4: $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x, \forall i \in [k]$
5: **end while**
6: **return** центроиды: $\mu_1, \ldots, \mu_k$

Методы понижения размерности

Понижение размерности — отображение данных высокой размерности в низкоразмерное пространство

- уменьшение вычислительной сложности
- улучшения обобщения (например, k-NN)
- повышение интерпретируемости данных

Рассматриваемые методы

- будем отображать данные из $\mathbb{R}^d$ в $\mathbb{R}^n$ $(n < d)$
- наиболее распространены линейные методы: $x \mapsto Wx$, где $W \in \mathbb{R}^{n \times d}$
- выбирать W стоит так, чтоб можно было «восстановить» x из Wx
- точное восстановление не всегда возможно

Задача PCA

Давайте решим:

$$\underset{W \in \mathbb{R}^{n \times d}, U \in \mathbb{R}^{d \times n}}{\operatorname{argmin}} \sum_{i=1}^{m} ||x_i - UWx_i||_2^2$$

Полученный метод носит название **Principal Component Analysis** (**метод главных компонент**). Изобретён Карлом Пирсоном в 1901-м году.

Решение PCA

Решение PCA

Пусть $x_1, \ldots, x_m$ вектора из $\mathbb{R}^d$, $A = \sum_{i=1}^{m} x_i x_i^T$, пусть $u_1, \ldots, u_n$ — n собственных векторов A, соответствующие n наибольшим собственным значениям A. Тогда решение задачи PCA — взять U матрицу, колонки которой — вектора $u_1, \ldots, u_n$, а $W = U^T$

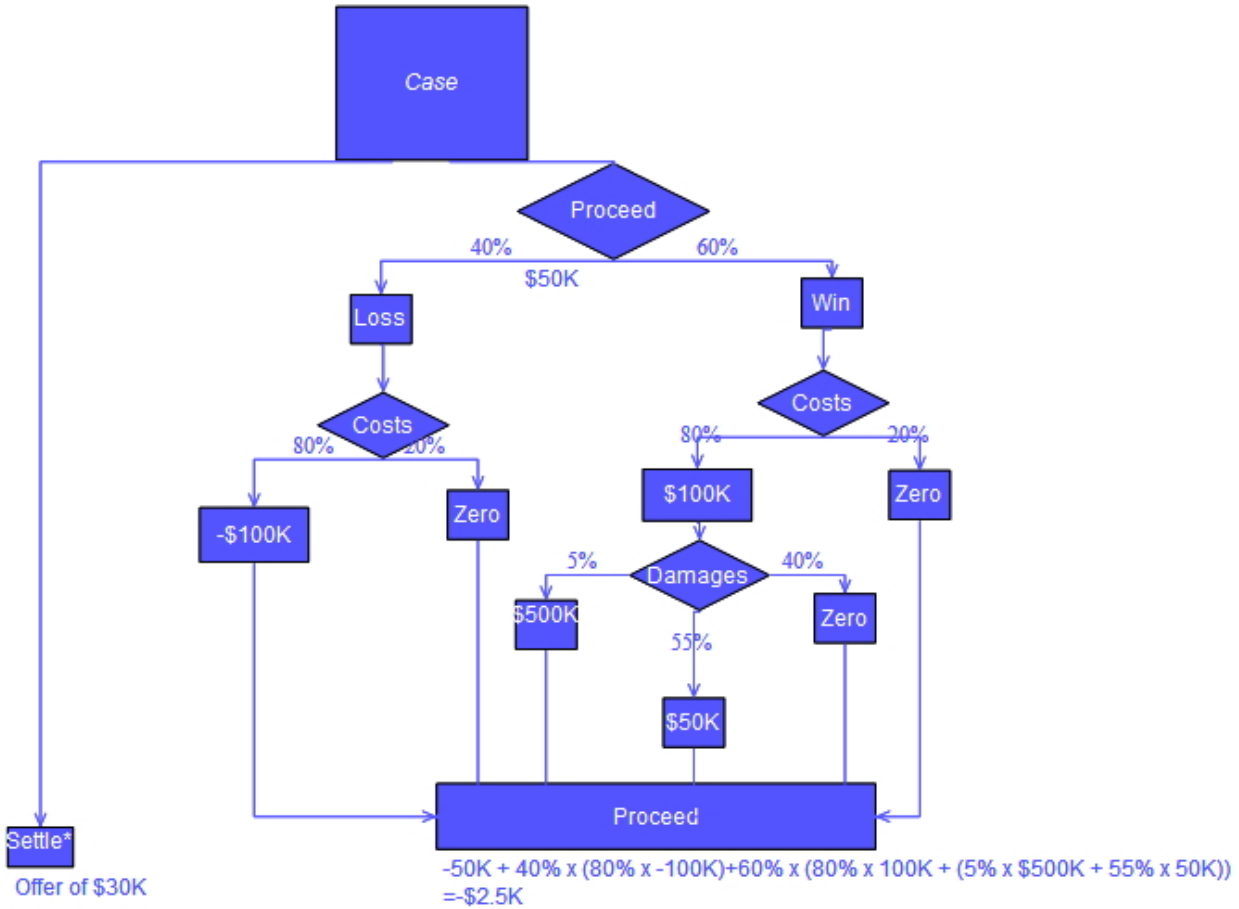
Другие методы

- случайные проекции
- Linear Discriminative Analysis
- Autoencoders
- t-SNE

Метод ближайших соседей

- обучение: запомнить выборку
- классификация: найти ближайший объект из выборки ответить тем же классом
- можно находить k ближайших (k-NN) и выбирать мажоритарный класс
- можно применять для задач регрессии

Decision trees



Boostrap aggregating (Bagging)

- пусть есть выборка размера n
- получим из неё сэмплингом k выборок размера n' (с повторениями)
- на каждом научим классификатор
- объединим выходы классификаторов

Random forest

- зададимся числом k
- получим k выборок из исходной выборки путём сэмплинга объектов и признаков
- обучим decision tree на каждом подмножестве
- объединим деревья

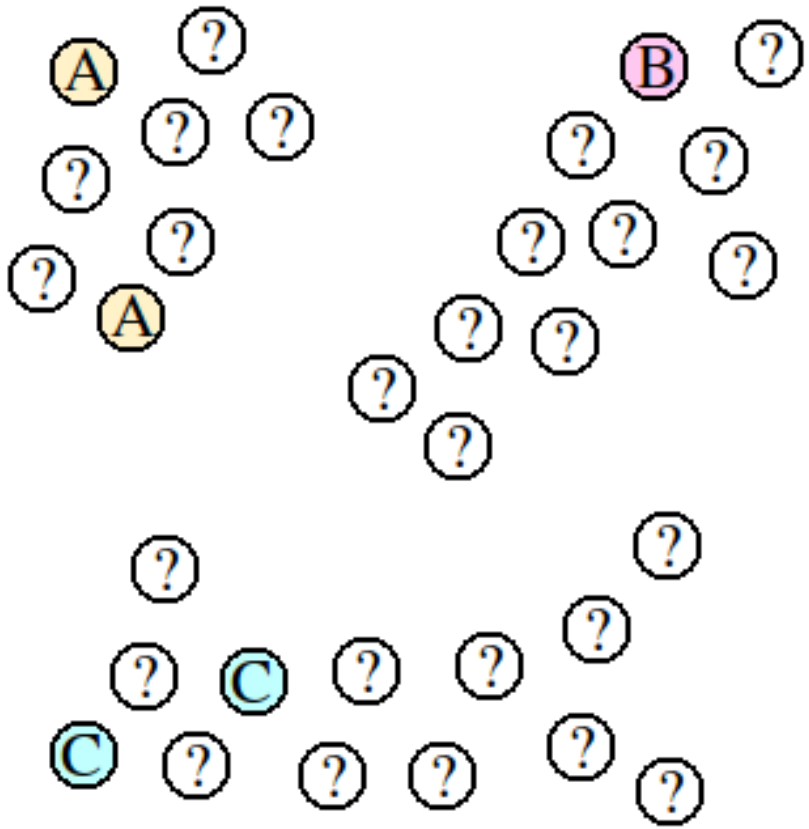
Ансамблирование

- методы голосования
- стекинг моделей (обучаем одну модель на выходах другой)

Онлайн-обучение

- что делать, если данные приходят по одному и сразу нужен ответ?
- SGD!
- Online Convex Optimization

Transduction



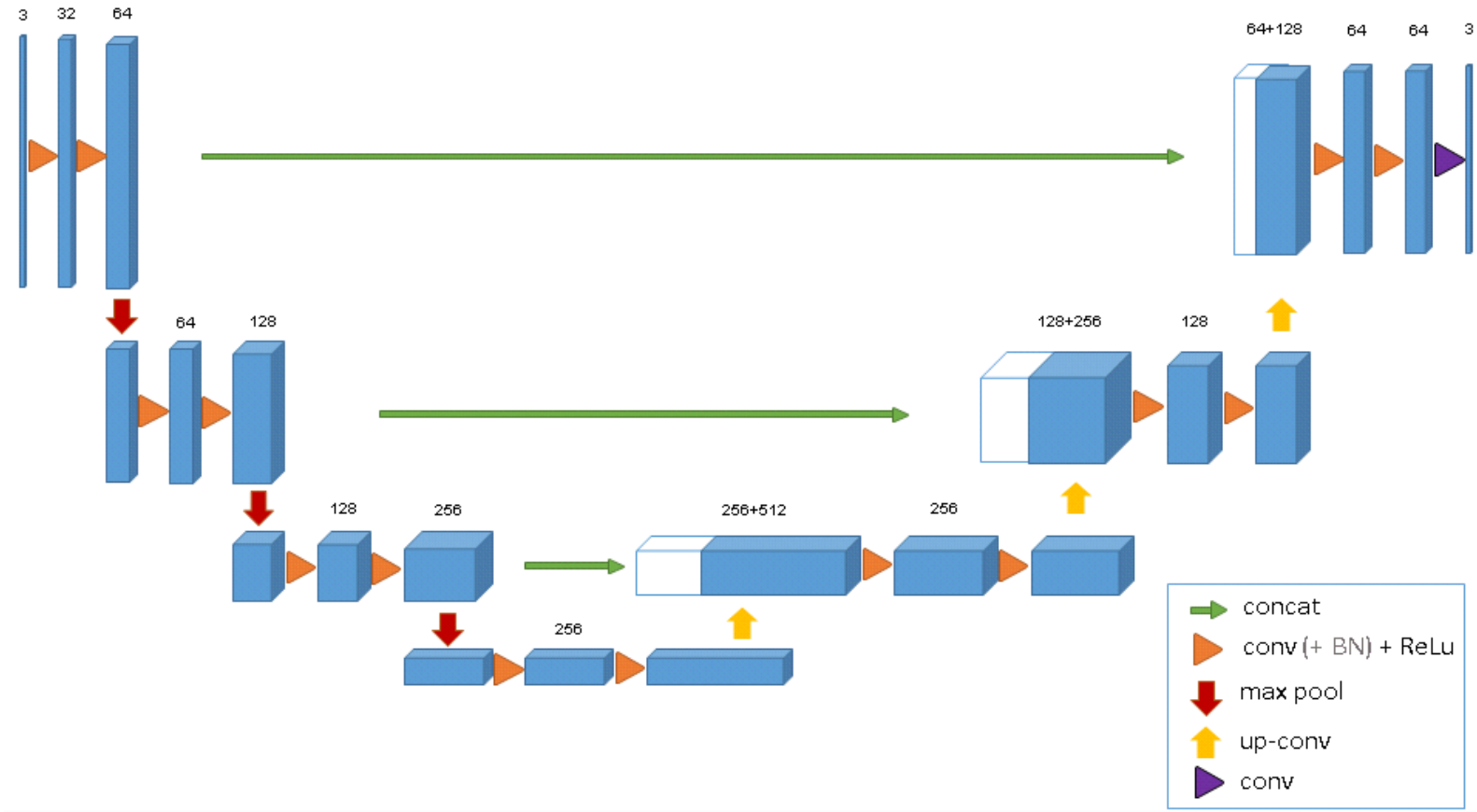
Ещё теория

- Radamacher complexity
- Feature selection
- Multiclass learning
- Learning to rank!
- Transfer learning
- Federative learning

Содержание

1. Обзор пройденного материала
   - Теория
   - Алгоритмы
2. Избранные главы ML
   - Кластеризация
   - Понижение размерности
   - Алгоритмы
   - Другие направления
3. Избранные главы NN

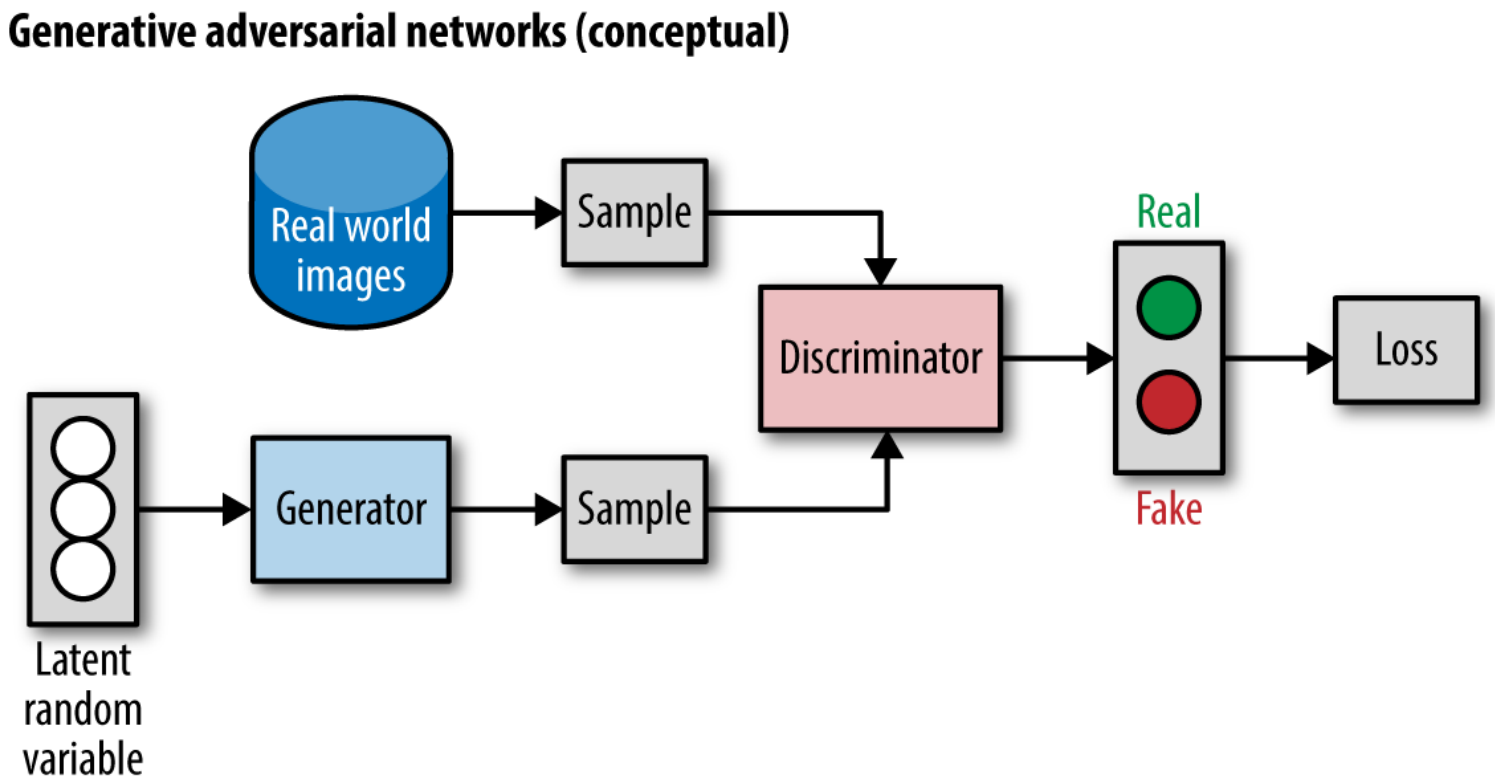
Segmentation



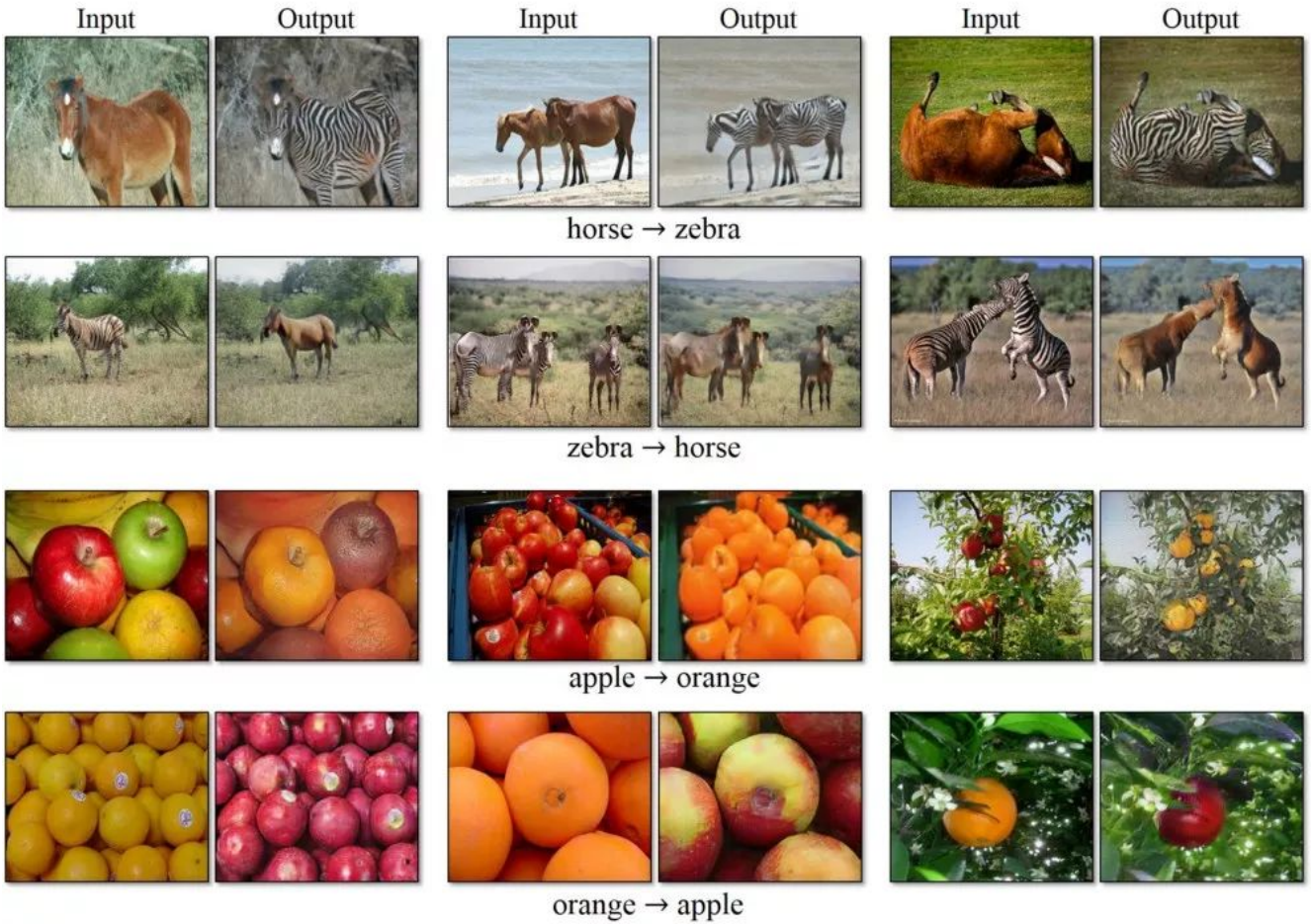
NLP

- задачи: машинный перевод, языковое моделирование, саммаризация
- word embeddings
- beam search

GAN

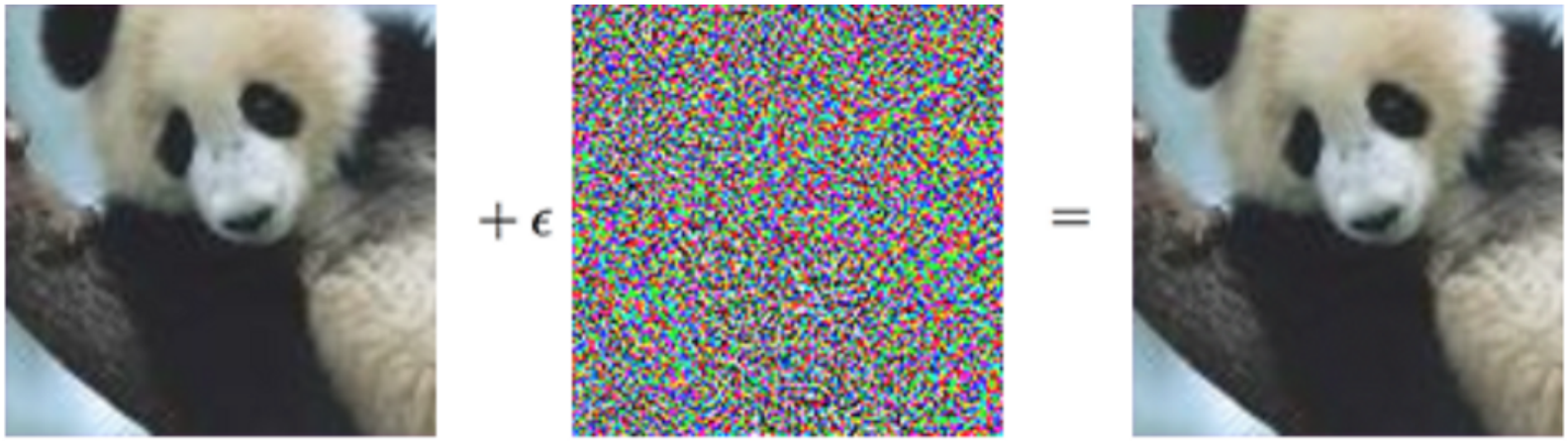

Generative adversarial networks (conceptual)
Real world images
Sample
Real
Discriminator
Loss
Generator
Sample
Fake
Latent random variable

CycleGan


Input
Output
Input
Output
Input
Output
horse → zebra
zebra → horse
apple → orange
orange → apple

CycleGan

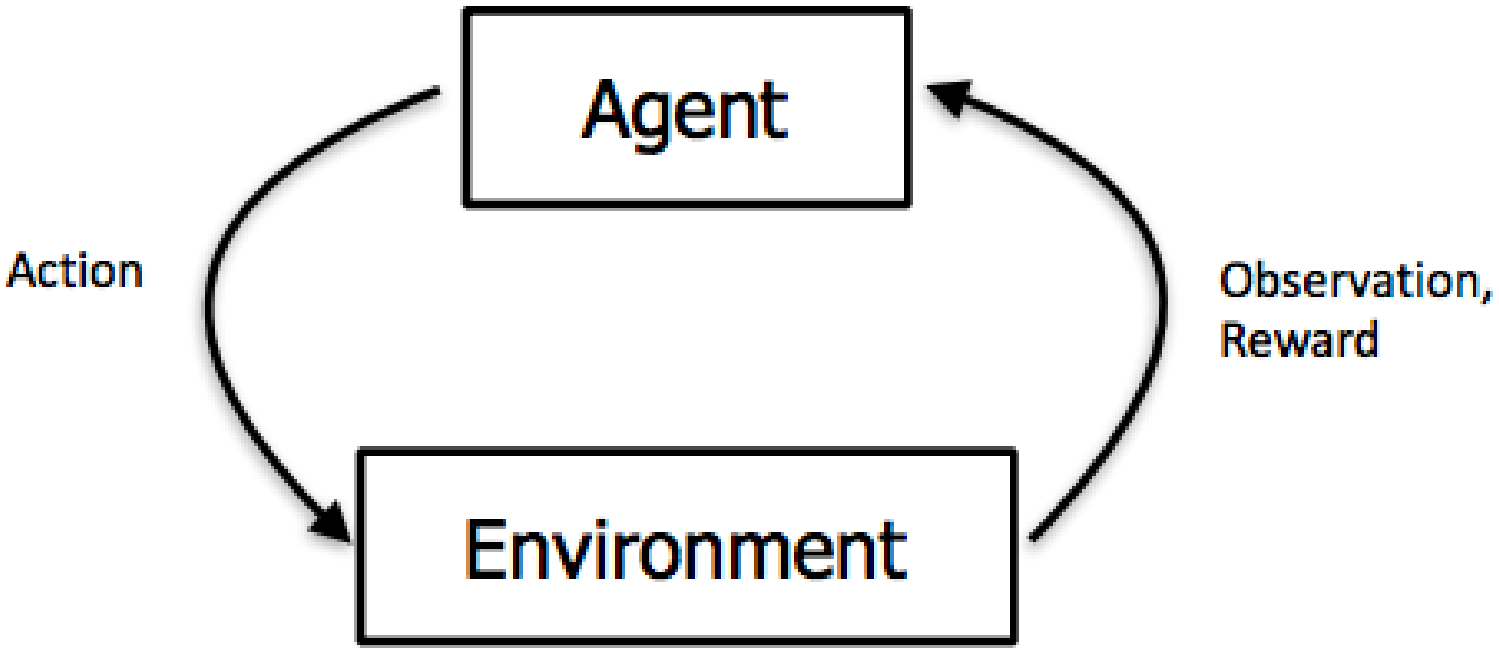
Adversarial attacks



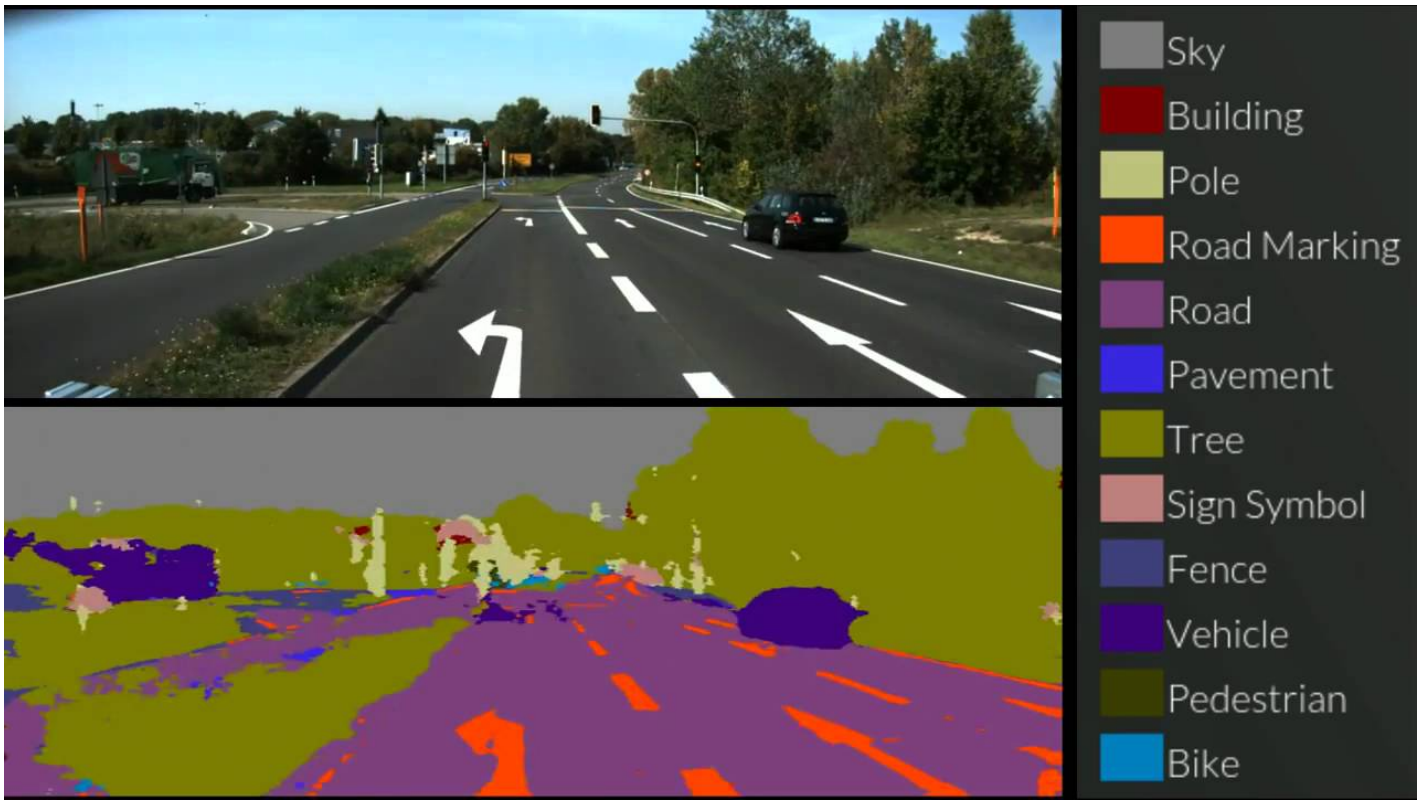
"panda"
57.7% confidence

"gibbon"
99.3% confidence

Reinforcement learning



Reinforcement learning: selfdriving car



Reinforcement learning: starcraft

Содержание

1. Обзор пройденного материала
   - Теория
   - Алгоритмы
2. Избранные главы ML
   - Кластеризация
   - Понижение размерности
   - Алгоритмы
   - Другие направления
3. Избранные главы NN

Что делать дальше?

- join ODS — `ods.ai`
- применяйте — `kaggle.com`
- читайте книги — `http://www.deeplearningbook.org`, «Pattern Recognition and Machine Learning» (Bishop)
- слушайте подкасты — `https://lexfridman.com/ai/`