

Математический анализ данных и машинное обучение

Лекция 3

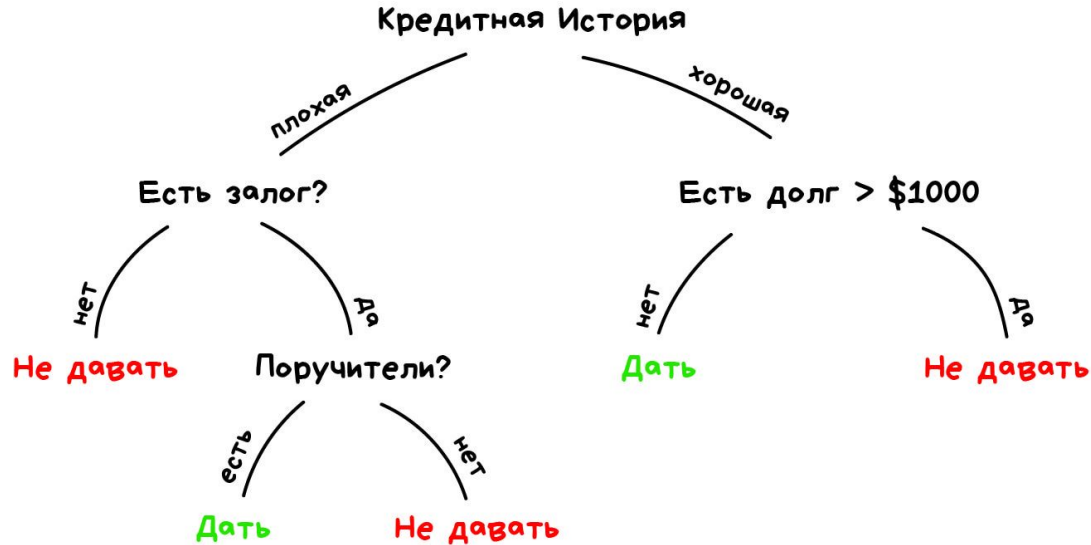
Саркисян Вероника

План на сегодня

9:30 - 11:00	Лекция: Решающие деревья, беггинг и бустинг.
11:15 - 12:30	Обзорная лекция: нейросети.
12:30 - 13:30	Обед
13:30 - 15:30	Семинар 1: Деревья, случайные леса
15:45 - 17:30	Семинар 2: XGBoost, CatBoost

Решающие деревья

Давать ли кредит?



Дерево Решений

Решающие деревья

Достоинства:

- Высокая интерпретируемость
- Нелинейность: можно обучиться под любую зависимость в данных

Недостатки:

- Очень быстро переобучаются
- Дискретная структура
=> нельзя продифференцировать
=> нельзя найти максимум (даже локальный)
- Неустойчивость

Рассмотрим бинарное дерево, в котором:

- каждой внутренней вершине v приписана функция (или предикат) $\beta_v : \mathbb{X} \rightarrow \{0, 1\}$;
- каждой листовой вершине v приписан прогноз $c_v \in Y$ (в случае с классификацией листу также может быть приписан вектор вероятностей).

Рассмотрим теперь алгоритм $a(x)$, который стартует из корневой вершины v_0 и вычисляет значение функции β_{v_0} . Если оно равно нулю, то алгоритм переходит в левую дочернюю вершину, иначе в правую, вычисляет значение предиката в новой вершине и делает переход или влево, или вправо. Процесс продолжается, пока не будет достигнута листовая вершина; алгоритм возвращает тот класс, который приписан этой вершине. Такой алгоритм называется *бинарным решающим деревом*.

На практике в большинстве случаев используются одномерные предикаты β_v , которые сравнивают значение одного из признаков с порогом:

$$\beta_v(x; j, t) = [x_j < t].$$

Существуют и многомерные предикаты, например:

- линейные $\beta_v(x) = [\langle w, x \rangle < t]$;
- метрические $\beta_v(x) = [\rho(x, x_v) < t]$, где точка x_v является одним из объектов выборки любой точкой признакового пространства.

Функционал ошибки

R_m - множество объектов, попавших в вершину для разбиения

R_l, R_r - множества объектов, определенных в левой и правой поддереве соответственно

$H(R_m)$ - критерий информативности (минимизируем: меньшее значение H соответствует меньшему разнообразию целевой переменной в R)

$Q(R_m, j, t)$ - функционал качества (максимизируем)

$$Q(R_m, j, s) = H(R_m) - \frac{|R_\ell|}{|R_m|} H(R_\ell) - \frac{|R_r|}{|R_m|} H(R_r).$$

Критерии информативности

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} L(y_i, c)$$

Общее определение; L - это функция потерь, зависящая от задачи

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2$$

В случае регрессии - это (например) MSE.

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i \neq c].$$

В случае классификации - это доля неверных ответов.

Чуть более хитрые критерии для классификации

Критерий Бриера:

$$H(R) = \min_{\sum_k c_k = 1} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^K (c_k - [y_i = k])^2.$$

Путем несложных математических преобразований находим оптимум

$$c_* = (p_1, \dots, p_K)$$

Подставляем его в исходный критерий и получаем **критерий Джини**:

$$H(R) = \sum_{k=1}^K p_k (1 - p_k).$$

Энтропийный критерий:

$$H(R) = \min_{\sum_k c_k = 1} \left(-\frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^K [y_i = k] \log c_k \right)$$

Путем несложных математических преобразований находим оптимум, он такой же

$$c_* = (p_1, \dots, p_K)$$

$$H(R) = - \sum_{k=1}^K p_k \log p_k$$

Борьба с переобучением

1. Критерии останова

- a. Ограничение максимальной глубины дерева
- b. Ограничение максимального количества листьев
- c. Ограничение минимального числа объектов в листе
- d. Останавливаемся, если все объекты в листе лежат в одном классе
- e. На очередном шаге функционал качества улучшился менее чем на n процентов

2. Стрижка дерева

Обработка пропущенных значений

- Усреднение
$$a_{mk}(x) = \begin{cases} a_{\ell k}(x), & \beta_m(x) = 0; \\ a_{rk}(x), & \beta_m(x) = 1; \\ \frac{|R_\ell|}{|R_m|} a_{\ell k}(x) + \frac{|R_r|}{|R_m|} a_{rk}(x), & \beta_m(x) \text{ нельзя вычислить.} \end{cases}$$
- Суррогатные предикаты: используем другой признак, дающий разбиение, максимально близкое к данному
- Замена пропуска на минимальное (максимальное) значение

Обработка категориальных признаков

В случае с бинарной классификацией упорядочим все значения категориального признака на основе того, какая доля объектов с таким значением имеет класс +1:

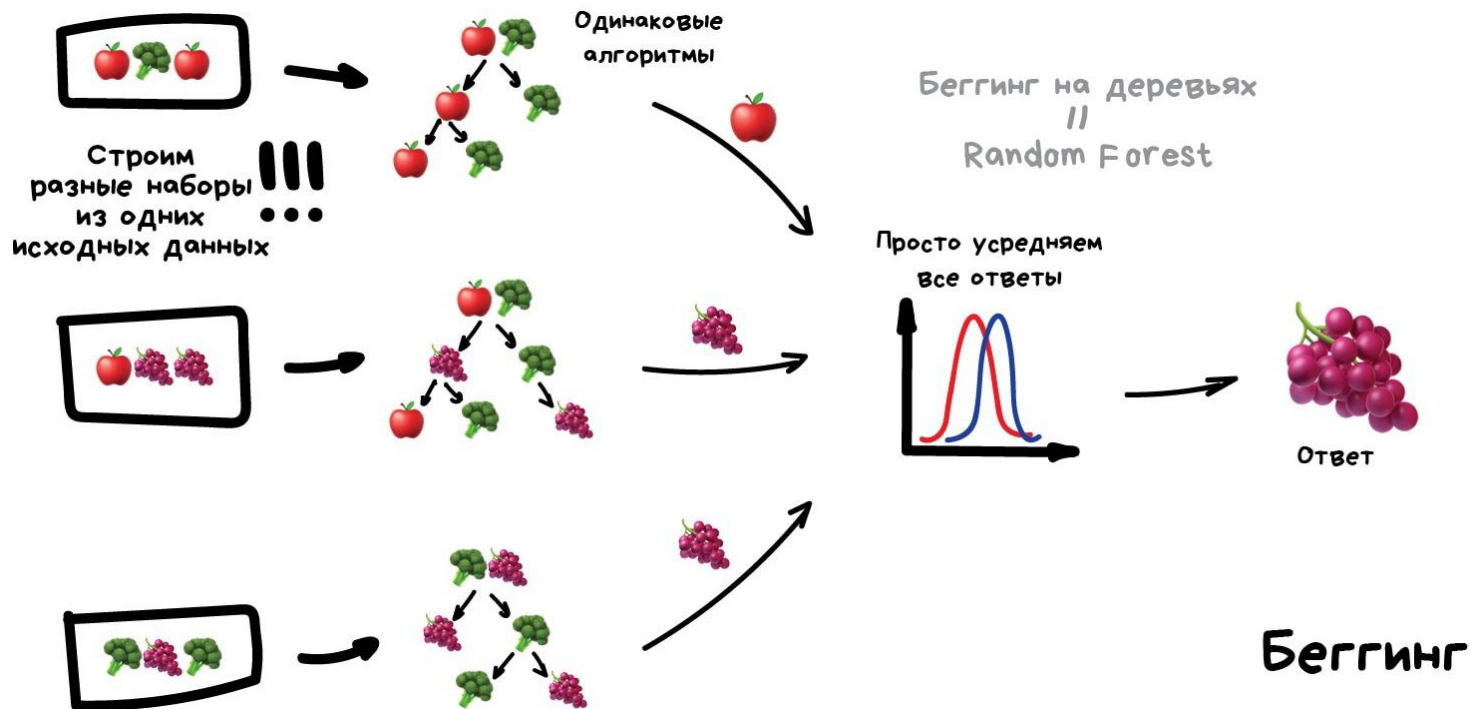
$$\frac{1}{N_m(u_{(1)})} \sum_{x_i \in R_m(u_{(1)})} [y_i = +1] \leq \dots \leq \frac{1}{N_m(u_{(q)})} \sum_{x_i \in R_m(u_{(q)})} [y_i = +1],$$

после чего заменим категорию $u_{(i)}$ на число i , и будем искать разбиение как для вещественного признака. Можно показать, что если искать оптимальное разбиение по критерию Джини или энтропийному критерию, то мы получим такое же разбиение, как и при переборе по всем возможным $2^{q-1} - 1$ вариантам.

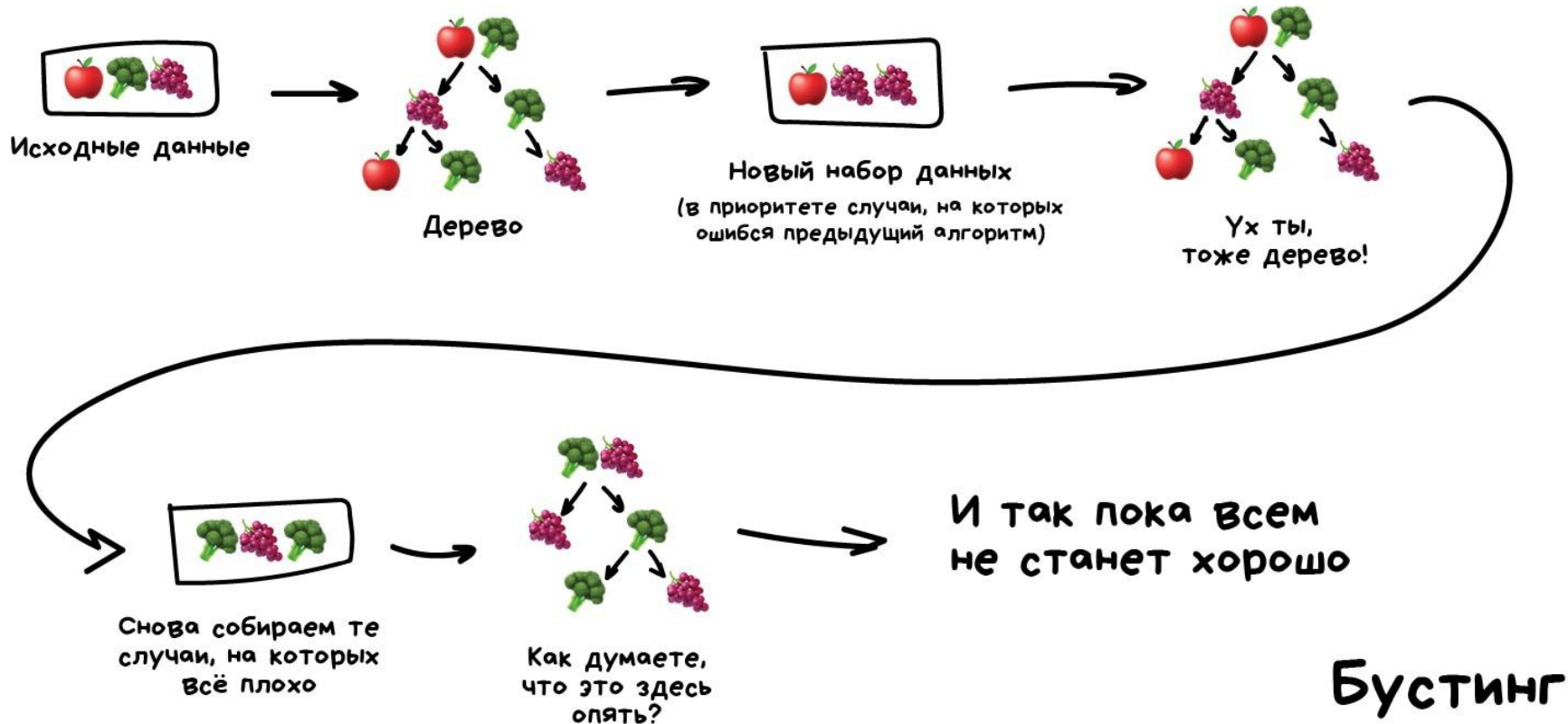
Для задачи регрессии с MSE-функционалом это тоже будет верно, если упорядочивать значения признака по среднему ответу объектов с таким значением:

$$\frac{1}{N_m(u_{(1)})} \sum_{x_i \in R_m(u_{(1)})} y_i \leq \dots \leq \frac{1}{N_m(u_{(q)})} \sum_{x_i \in R_m(u_{(q)})} y_i.$$

Как еще можно бороться с переобучением?










Как еще можно бороться с переобучением?

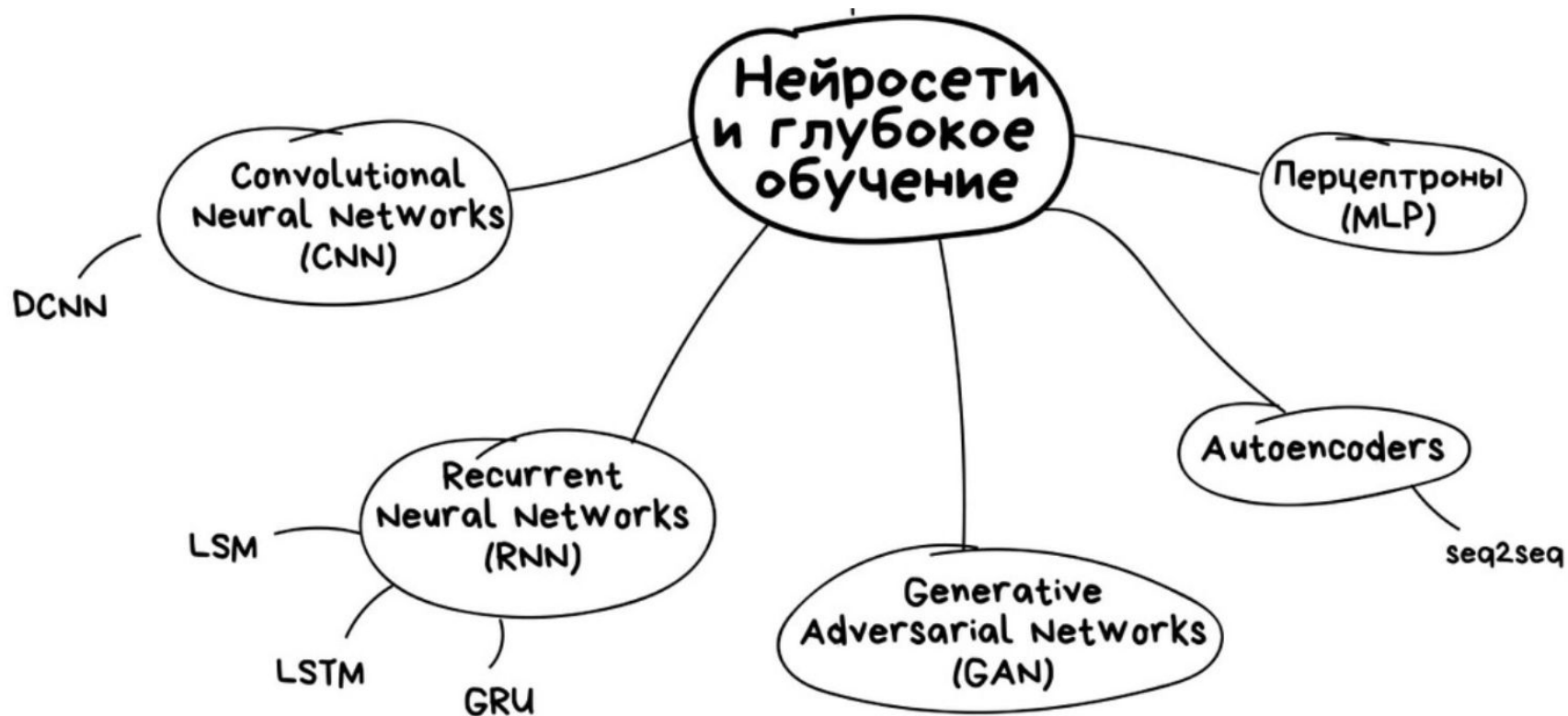


Градиентный бустинг на деревьях решений

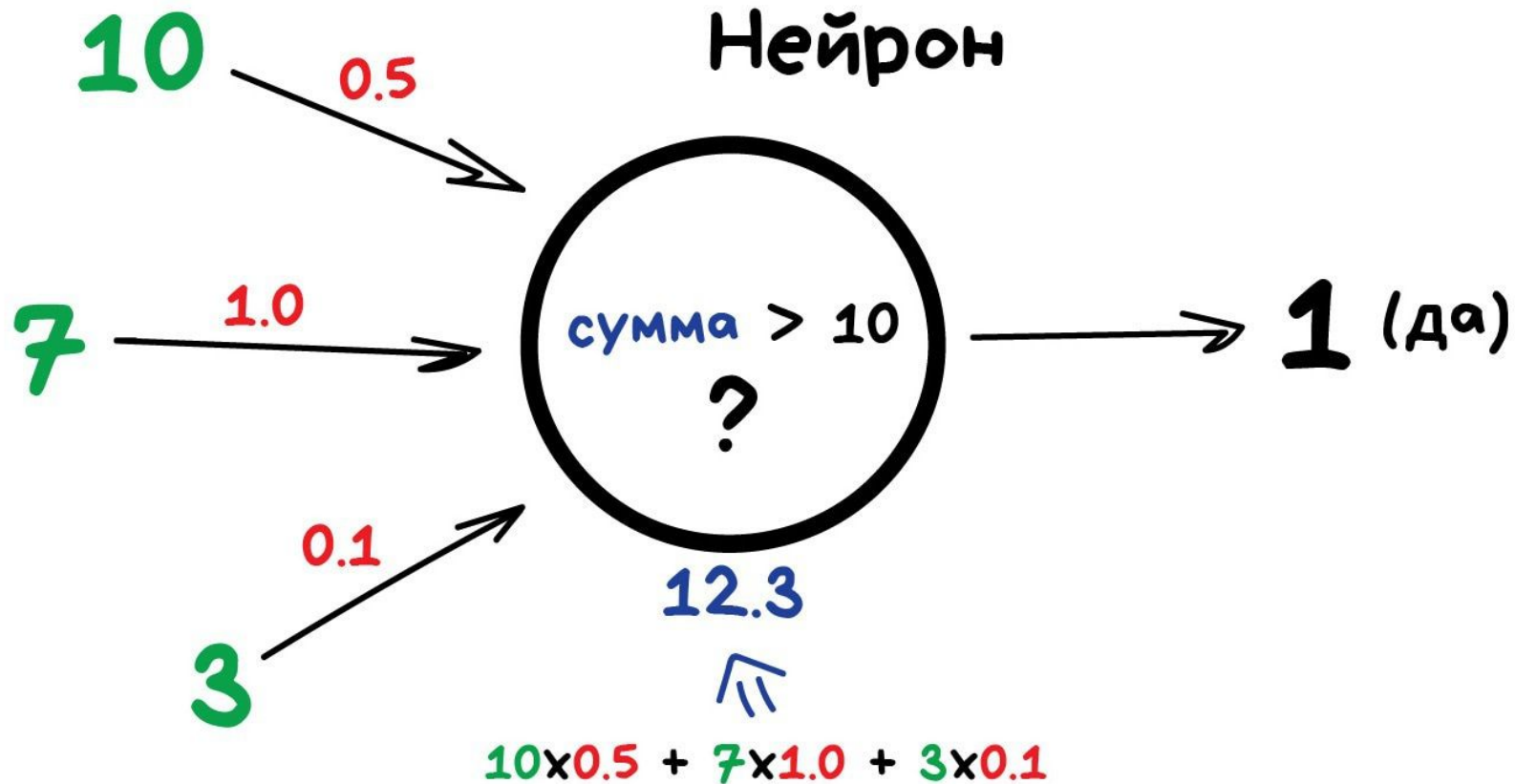
- XGBoost (2014)
- LightGBM (январь 2017)
- CatBoost (июль 2017)

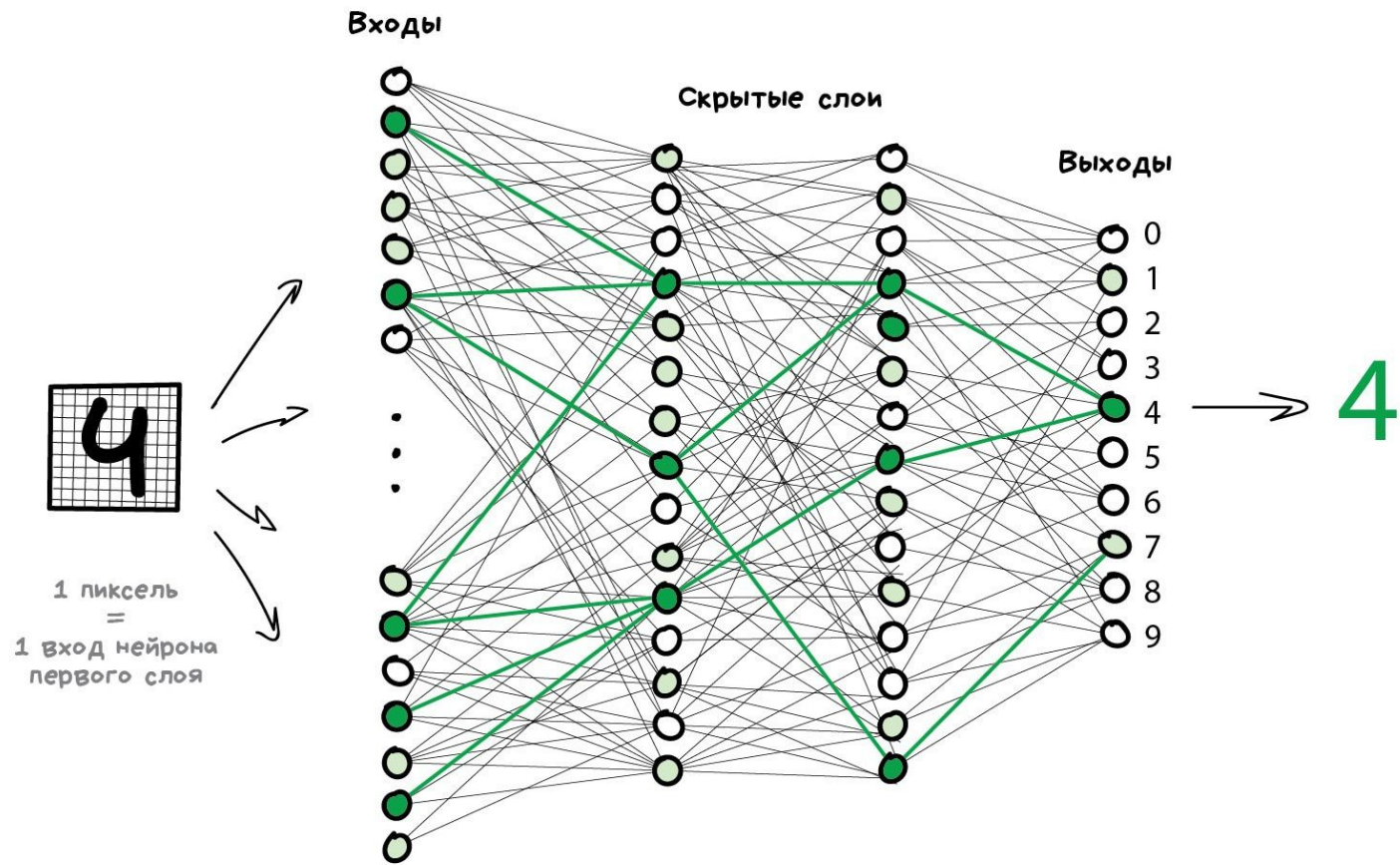
	CatBoost		LightGBM		XGBoost	
	Tuned	Default	Tuned	Default	Tuned	Default
 Adult	0.26974	0.27298 +1.21%	0.27602 +2.33%	0.28716 +6.46%	0.27542 +2.11%	0.28009 +3.84%
 Amazon	0.13772	0.13811 +0.29%	0.16360 +18.80%	0.16716 +21.38%	0.16327 +18.56%	0.16536 +20.07%
 Click prediction	0.39090	0.39112 +0.06%	0.39633 +1.39%	0.39749 +1.69%	0.39624 +1.37%	0.39764 +1.73%
 KDD appetency	0.07151	0.07138 -0.19%	0.07179 +0.40%	0.07482 +4.63%	0.07176 +0.35%	0.07466 +4.41%
 KDD churn	0.23129	0.23193 +0.28%	0.23205 +0.33%	0.23565 +1.89%	0.23312 +0.80%	0.23369 +1.04%
 KDD internet	0.20875	0.22021 +5.49%	0.22315 +6.90%	0.23627 +13.19%	0.22532 +7.94%	0.23468 +12.43%
 KDD upselling	0.16613	0.16674 +0.37%	0.16682 +0.42%	0.17107 +2.98%	0.16632 +0.12%	0.16873 +1.57%

Перерыв!



Нейрон

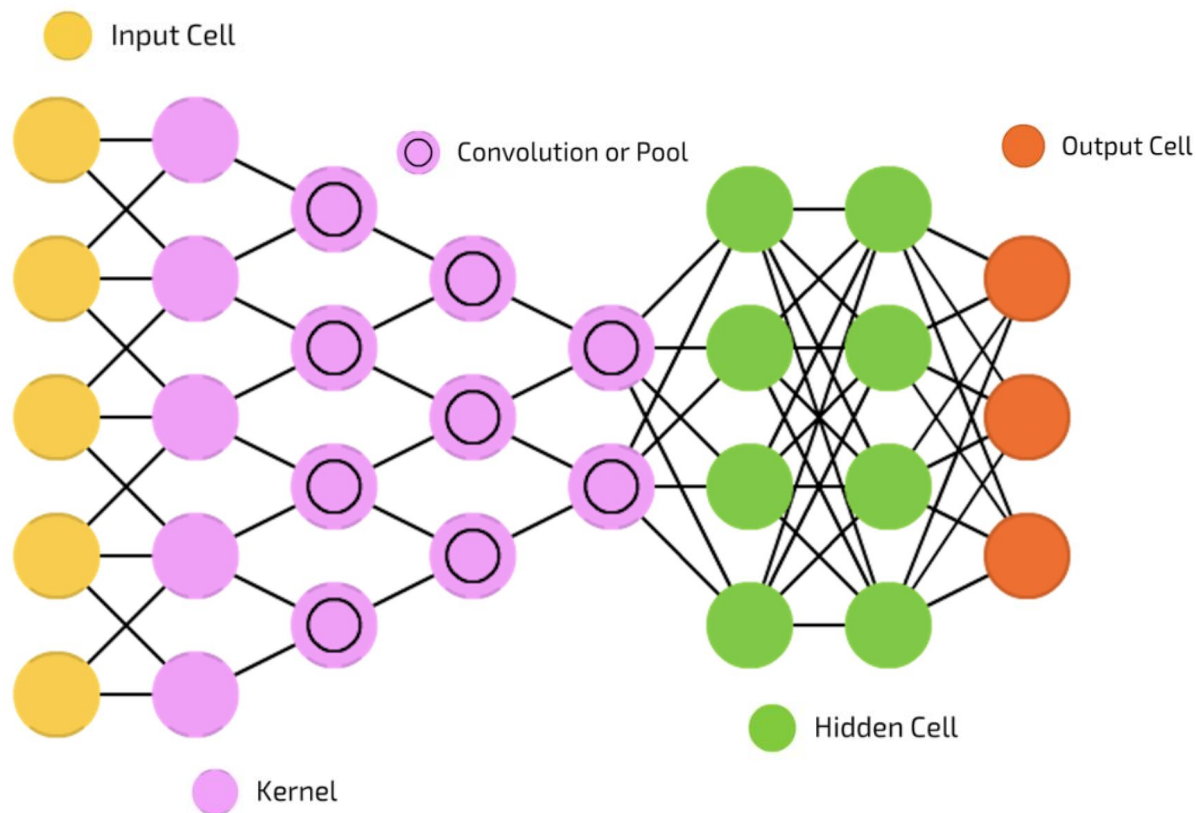




Многослойный Перцептрон (MLP)

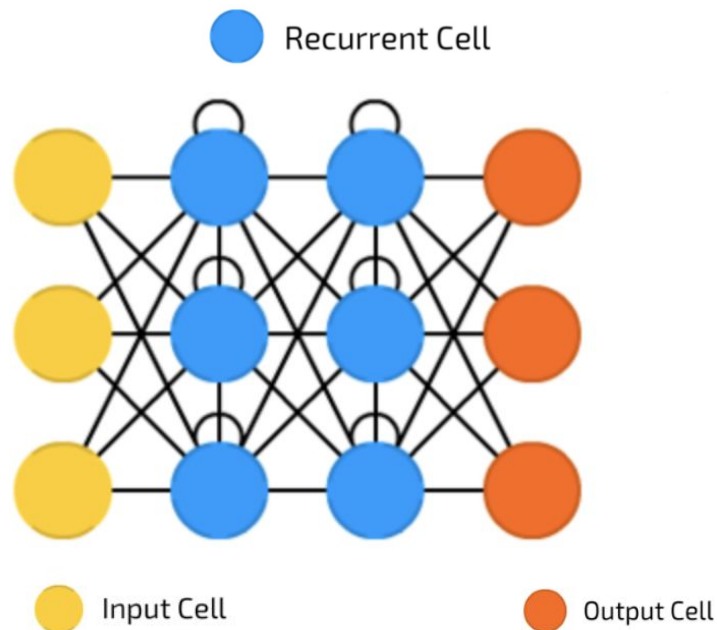
Сверточные нейросети (convolutional neural networks, CNN)

- Картинки
- Аудио (реже)



Рекуррентные нейросети (recurrent neural network, RNN)

Все, что связано с последовательностями данных: тексты, речь, музыка, временные ряды и т.д.

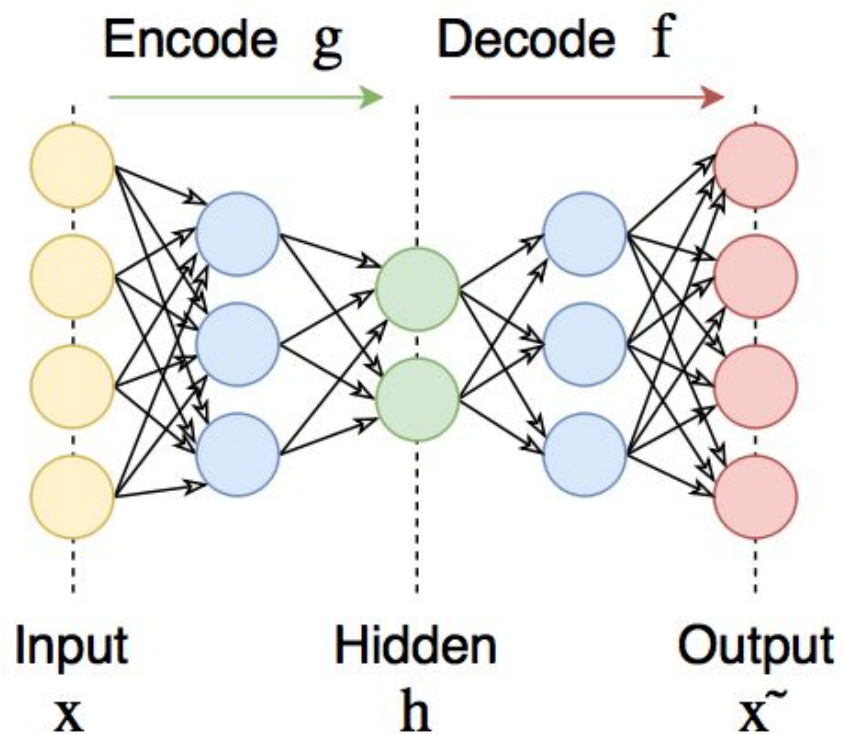


Генеративные состязательные нейросети (generative adversarial network, GAN)

Модель учится
генерировать данные,
аналогичные тем, на
которых обучается.



Автоэнкодеры (Autoencoders)



Библиотеки для написания моделей

Criteria	Winner	Runner up
Community and Support	Tensorflow	Pytorch
Ease of Use	Pytorch	Tensorflow
Academia(Prototyping)	Pytorch	Torch/Caffe
Industry	Tensorflow	Caffe2
Embedded Computer vision	Caffe	Tensorflow

Пример нейросети на Keras

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
```