Классификация текстов

+ gentle reminder on ML course

Задача классификации

- Есть заранее известные классы (2 - бинарная классификация, больше - многоклассовая классификация)
- Задача - построить алгоритм, который относит объекты в свой класс

Примеры: анализ тональности, определение интента в диалоге, выявление токсичного поведения и много всего другого.

Как будем оценивать качество?

- Как и в любой ML задаче, у нас есть train и test множества: обучаемся на train, проверяем качество на test.
- Метрики качества для бинарной классификации:
  - Доля правильных ответов (accuracy)
  - Точность (precision)
  - Полнота (recall)
  - F-мера
  - AUC-ROC

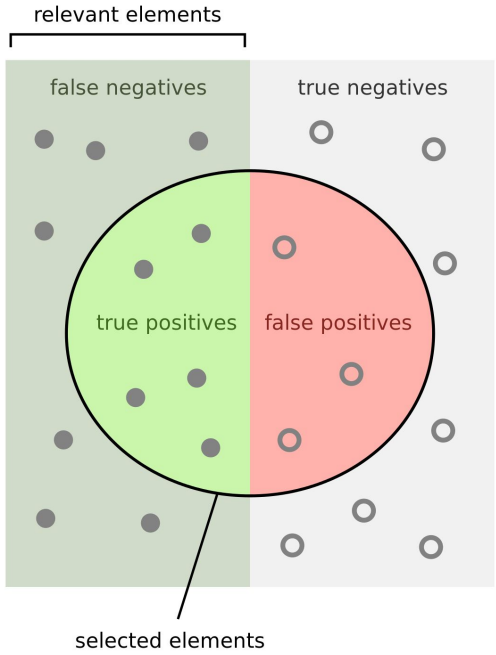
Accuracy, precision, recall

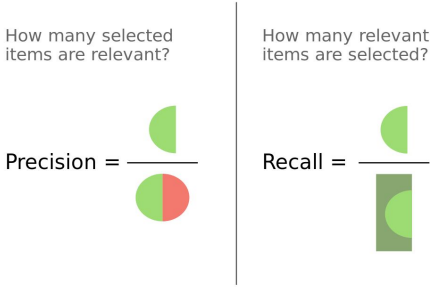
	Y = 0	Y = 1
f(x) = 0	**T**rue **N**egative	**F**alse **P**ositive
f(x) = 1	**F**alse **N**egative	**T**rue **P**ositive

$$\text{Полнота (recall)} = \frac{T_p}{P}$$

$$\text{Точность (precision)} = \frac{T_p}{T_p + F_p}$$

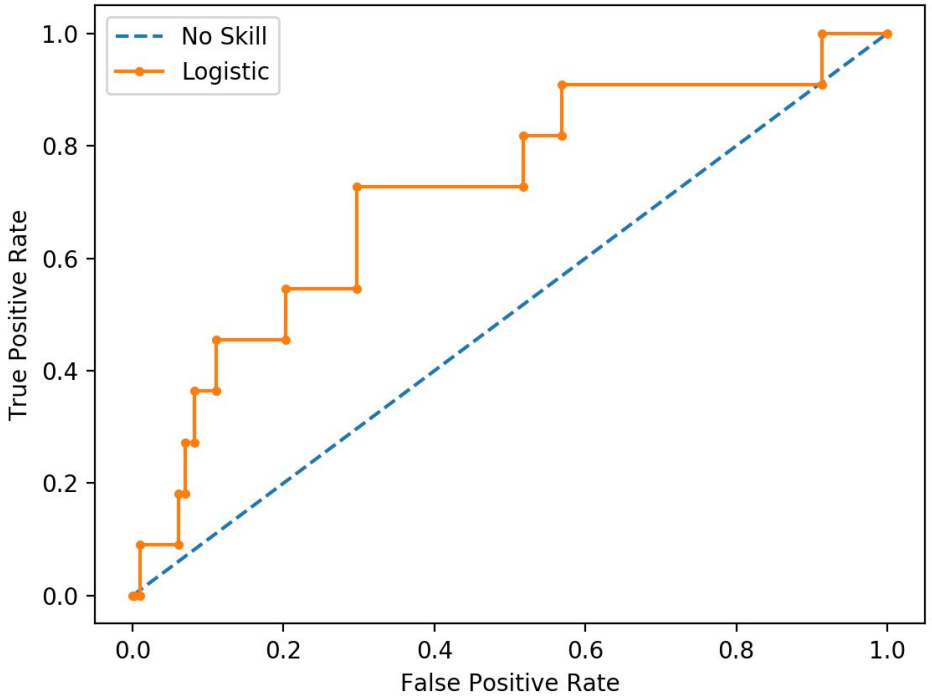
$$\text{F-мера} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$





AUC-ROC

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}};$$
$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}.$$



Многоклассовая классификация: One-VS-All

Обучим K (K = число классов) линейных классификаторов: $b_1(x), \ldots, b_K(x)$

Каждый классификатор (бинарный!) будет отличать один класс от всех остальных. $b_k(x) = \langle w_k, x \rangle + w_{0k}.$

Итоговый класс будем вычислять как наиболее вероятный, исходя из прогнозов всех алгоритмов:

$$a(x) = \underset{k \in \{1,\ldots,K\}}{\arg\max}\, b_k(x).$$

Многоклассовая классификация: All-VS-All

Обучим C_K^2 классификаторов (для всех возможных пар классов):

$$b_k(x) = \text{sign}(\langle w_k, x \rangle + w_{0k}).$$

Каждый классификатор (бинарный!) обучаем на подвыборке, содержащей только 2 класса.

Для классификации нового объекта подадим его на вход всем построенным классификаторам; в качестве ответа выберем наиболее “частый” среди ответов класс.

$$a(x) = \underset{k \in \{1,\dots,K\}}{\arg\max} \sum_{i=1}^{K} \sum_{j \neq i} [a_{ij}(x) = k]$$

Микро- и макро-усреднение

Рассмотрим K двухклассовых задач (one-VS-all), для каждой вычислим матрицу ошибок: $\mathrm{TP}_k, \mathrm{FP}_k, \mathrm{FN}_k, \mathrm{TN}_k$

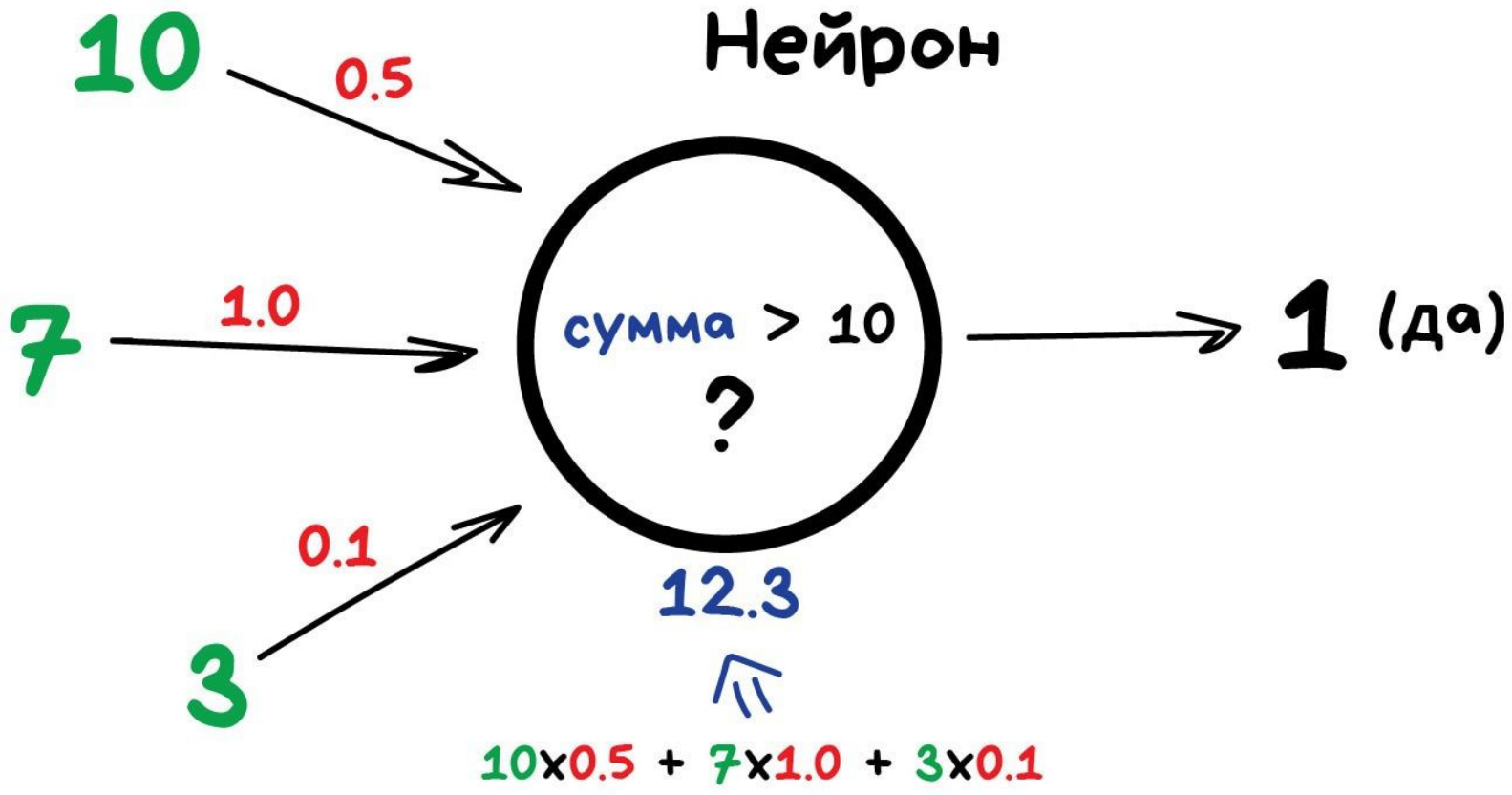
Микро-усреднение: $$\mathrm{precision}(a, X) = \frac{\overline{\mathrm{TP}}}{\overline{\mathrm{TP}} + \overline{\mathrm{FP}}}, \qquad \overline{\mathrm{TP}} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{TP}_k$$

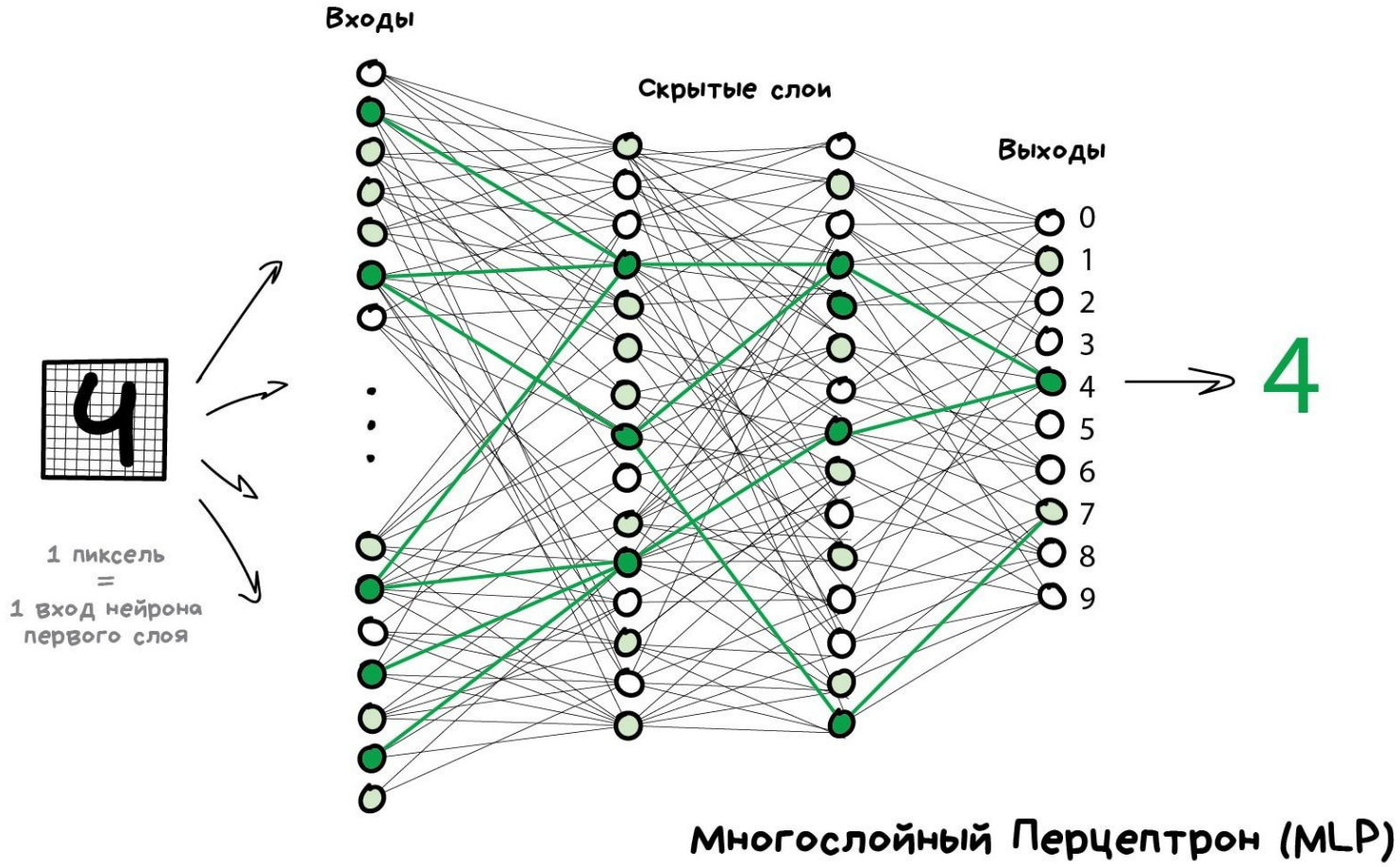
Макро-усреднение: $$\mathrm{precision}(a, X) = \frac{1}{K}\sum_{k=1}^{K} \mathrm{precision}_k(a, X); \qquad \mathrm{precision}_k(a, X) = \frac{\mathrm{TP}_k}{\mathrm{TP}_k + \mathrm{FP}_k}.$$

Классические алгоритмы классификации

- Метод опорных векторов (Support Vector Machine, SVM)
- Логистическая регрессия
- Метод K ближайших соседей (K Nearest Neighbors, KNN)
- Нейросетевые алгоритмы

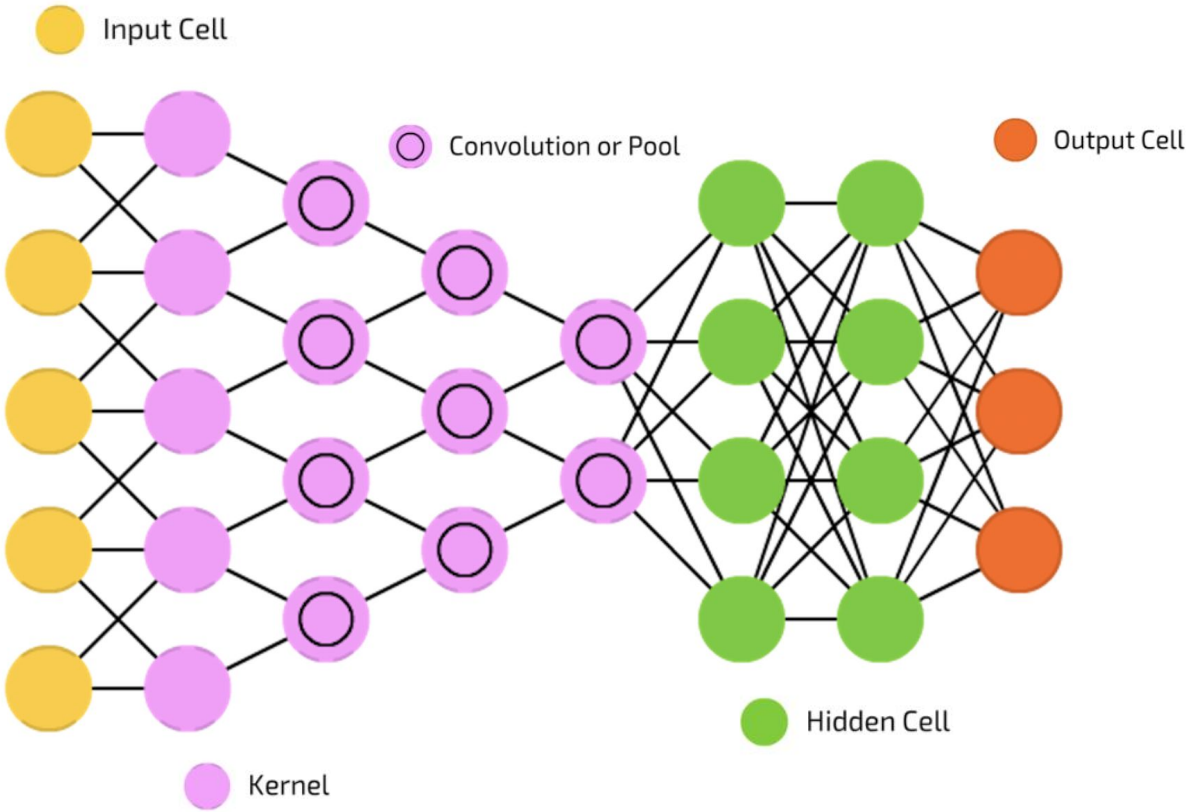

Нейросети и глубокое обучение
Convolutional Neural Networks (CNN)
DCNN
Перцептроны (MLP)
Recurrent Neural Networks (RNN)
LSM
LSTM
GRU
Generative Adversarial Networks (GAN)
Autoencoders
seq2seq


Нейрон
10
0.5
7
1.0
3
0.1
сумма > 10
?
12.3
1 (да)
10x0.5 + 7x1.0 + 3x0.1



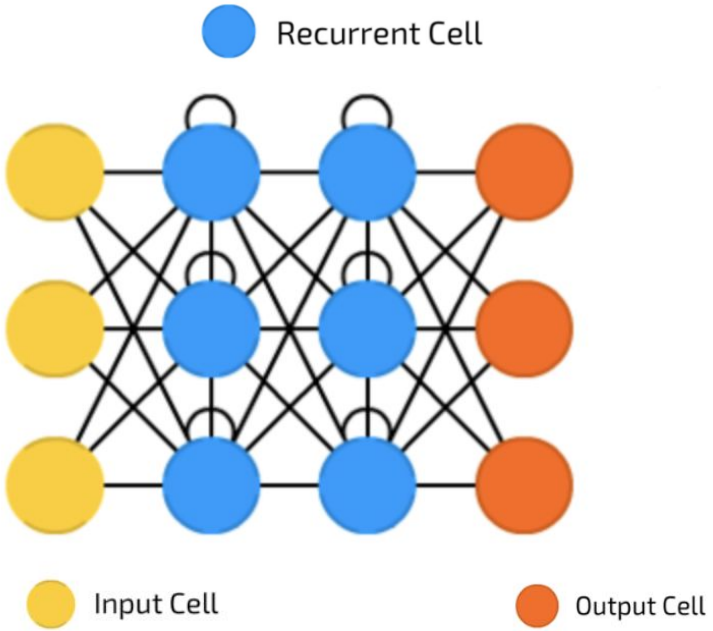
[Видео с илюстрацией]

Сверточные нейросети (convolutional neural networks, CNN)



Рекуррентные нейросети (recurrent neural network, RNN)

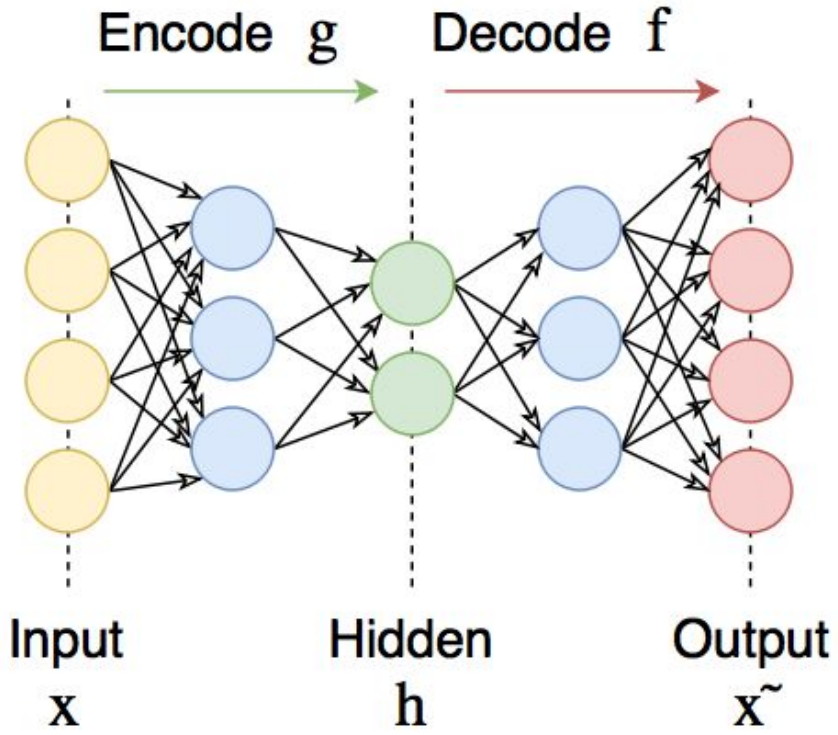
Все, что связано с последовательностями данных: тексты, речь, музыка, временные ряды и т.д.



Генеративные состязательные нейросети (generative adversarial network, GAN)

Две нейросети: одна учится генерировать данные, вторая - отличать настоящие от сгенерированных.

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))
```

Sum up

- Хороший бейзлайн для классификации - fasttext
- Если есть модель эмбеддингов (или можно ее обучить) - модели классификации поверх эмбеддингов
- Нейросетевые классификаторы (о них во второй лекции)