

# Natural Language Processing

## Lecture 1

**Konstantin Kotochigov, 2021**

# Что такое NLP

Область Computer Science: все, что связано с обработкой и анализом языка, в том числе:

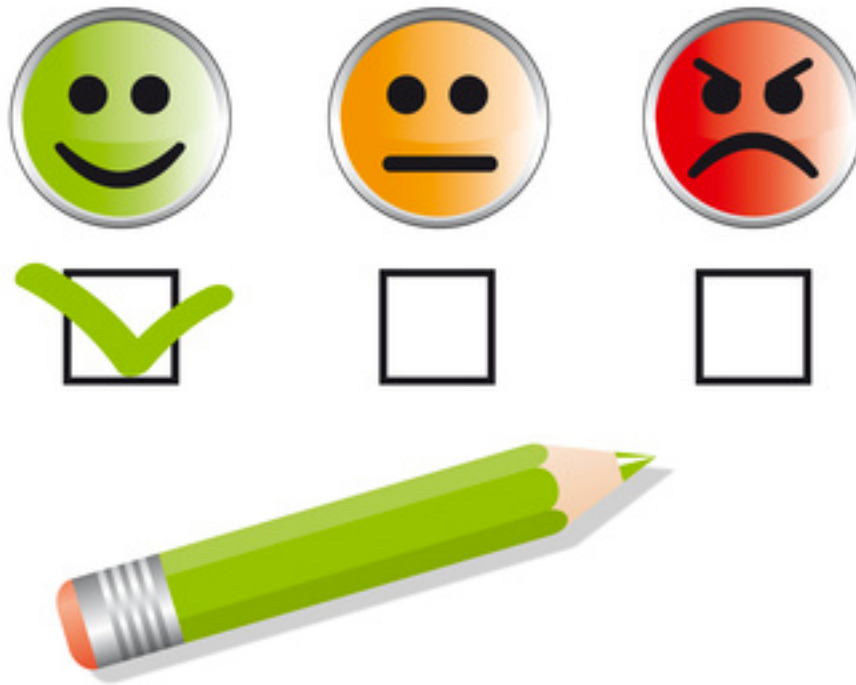
- Language Recognition - распознавание
- Language Understanding - понимание смысла сказанного
- Language Generation - генерация текста

# Несколько примеров классических задач, где используется NLP

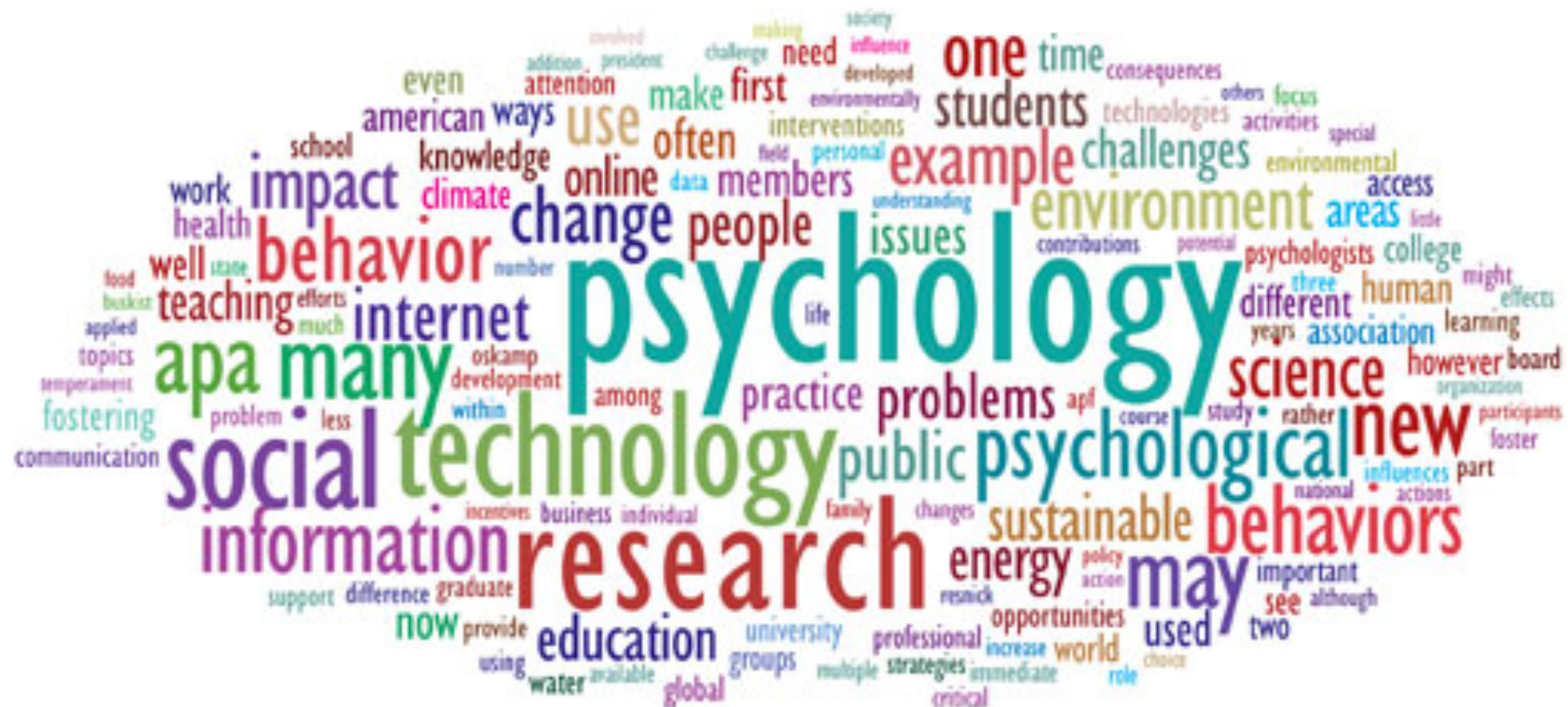
# Spam Classification



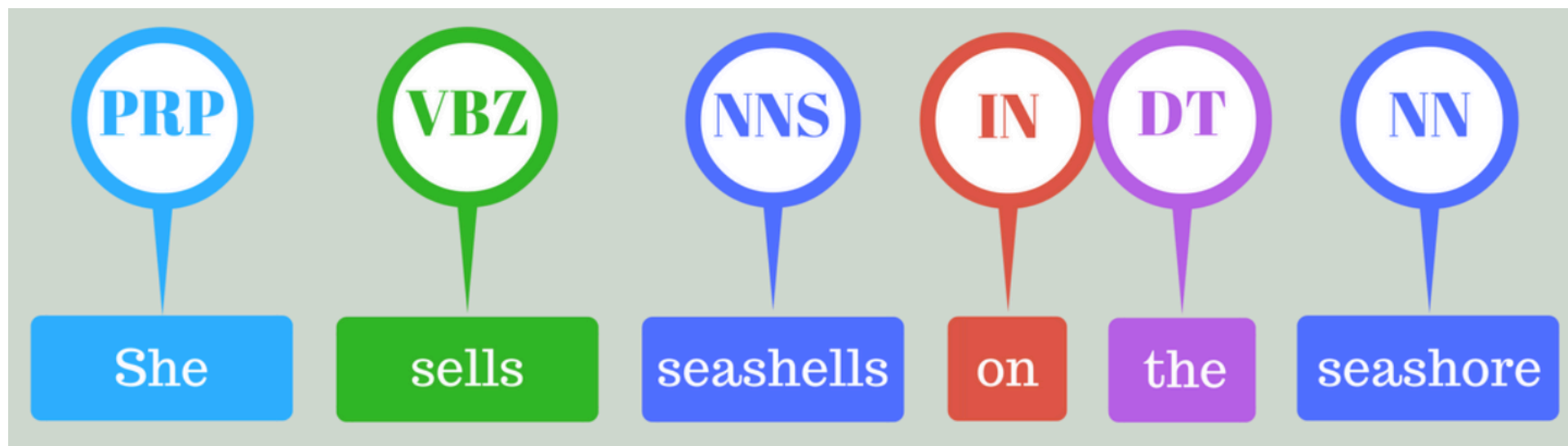
# Sentiment Analysis



# Topic Modeling



# Part-Of-Speech Tagging



# Named Entity Recognition

"There was nothing about this storm that was as expected," said **Jeff Masters**, a meteorologist and founder of **Weather Underground**. "**Irma** could have been so much worse. If it had traveled 20 miles north of the coast of **Cuba**, you'd have been looking at a (Category) 5 instead of a (Category) 3."

Person

Organization

Location

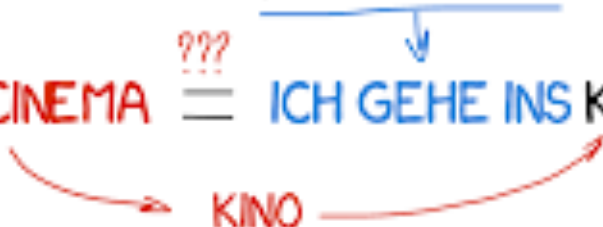


# Machine Translation

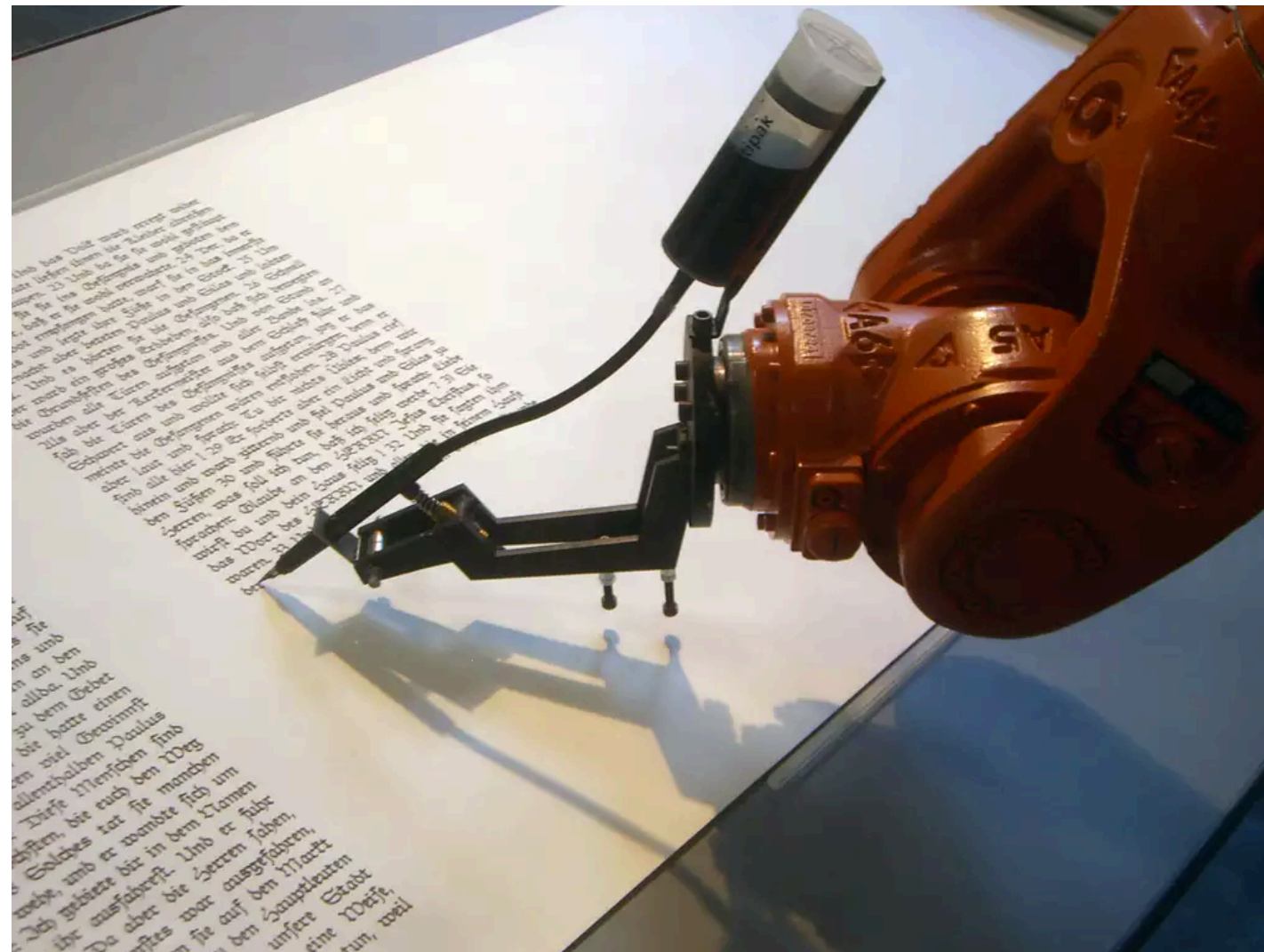
(ALREADY FAMILIAR EXAMPLE)

I'M GOING TO THE THEATER = ICH GEHE INS THEATER

I'M GOING TO THE CINEMA <sup>???</sup> = ICH GEHE INS KINO



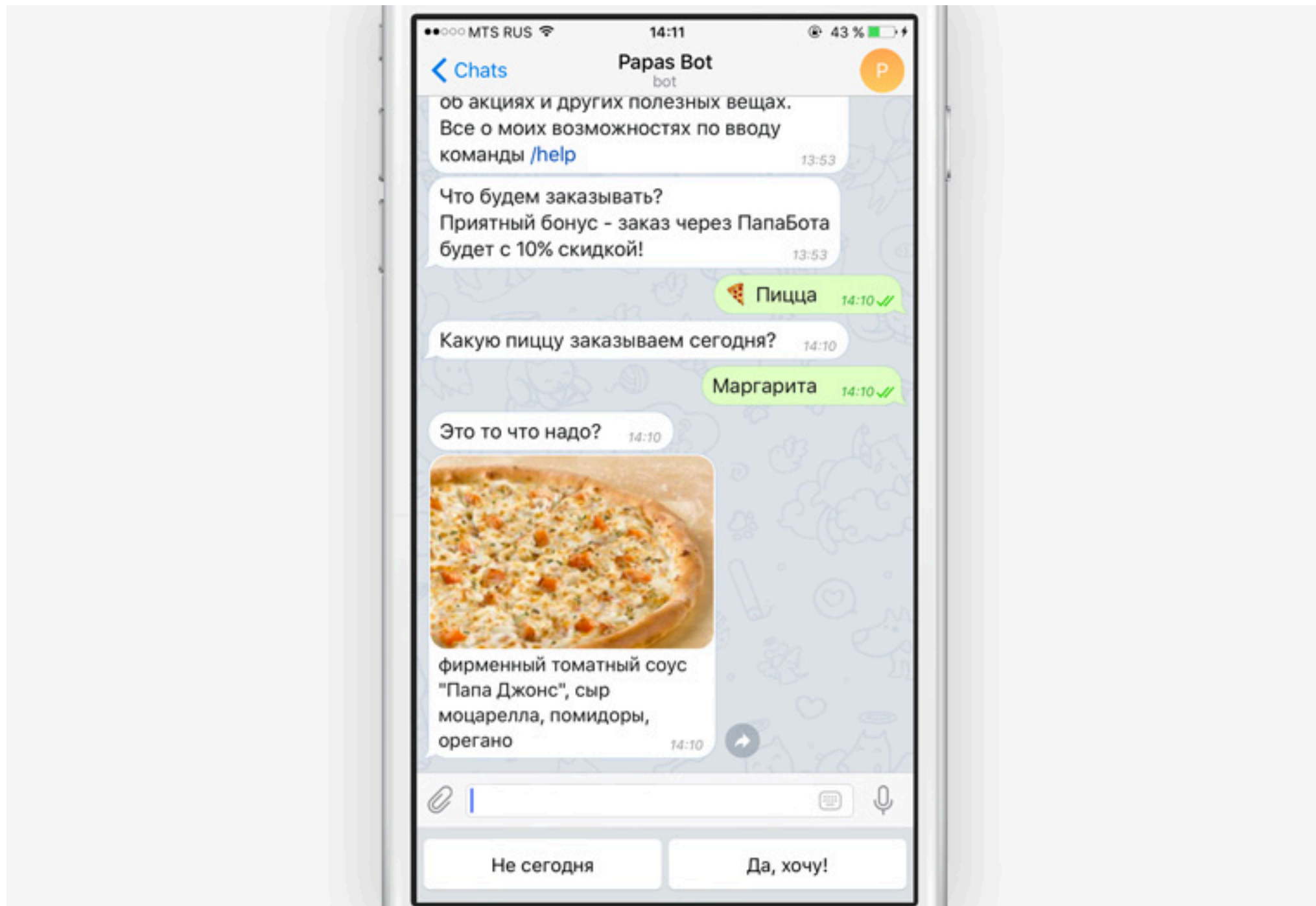
# Natural Language Generation



# Question Answering Systems



# Chat Bots



# Chat Bots

Два вида чат-ботов:

- Специализированные (goal oriented)

*заказать пиццу*

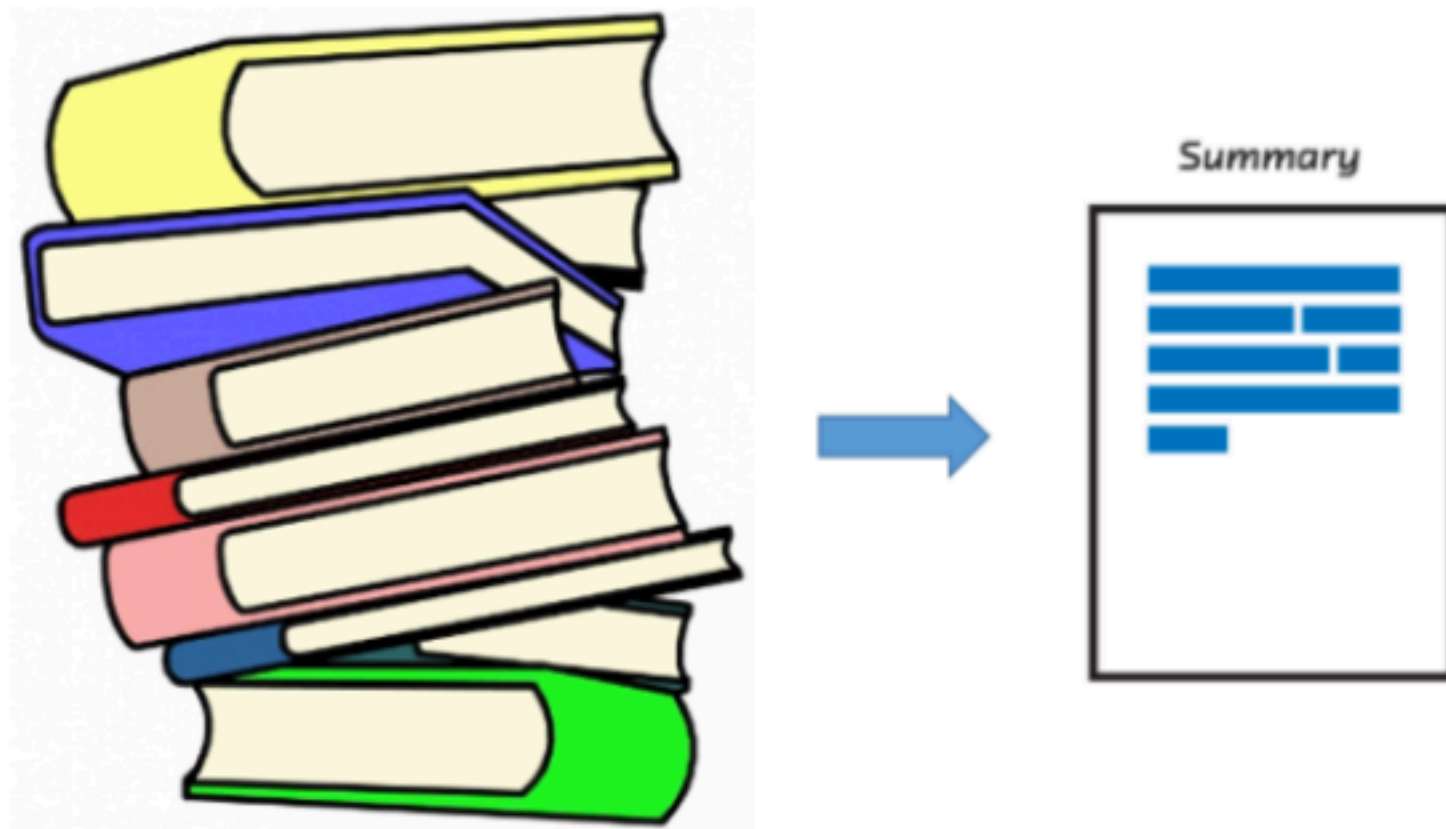
*узнать погоду*

*вызвать такси*

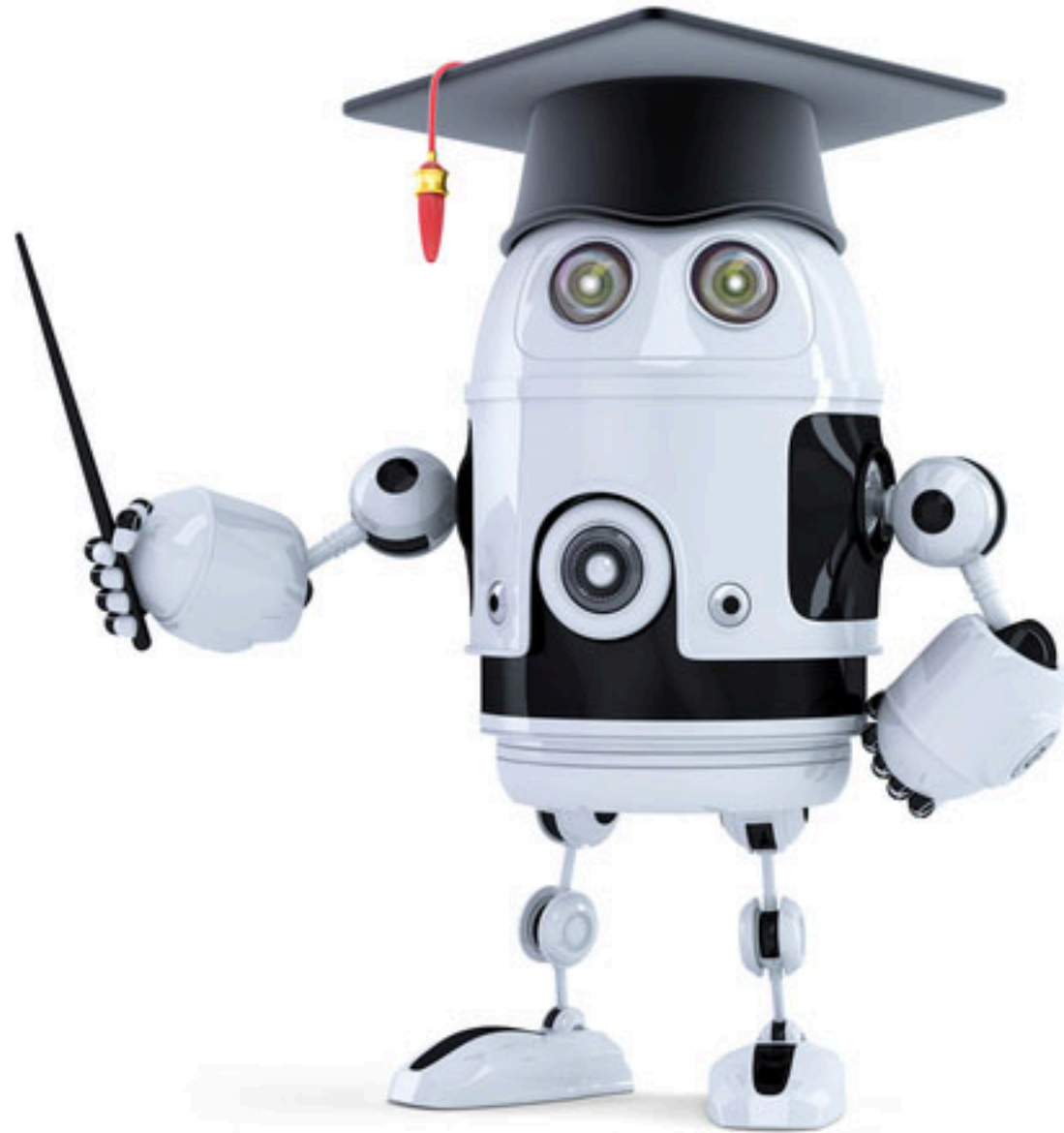
- Общего назначения (chit-chat боты)

*общение на произвольные темы*

# Text Summarization



# Automatic Essay Scoring



# И другие задачи...

- Классификация статей/документов
- Определение дубликатов документов
- Выявления плагиата в научных работах
- Детектирование языка
- Forensic Linguistics (определение авторства текста)



# Modern NLP approaches

Этапы развития методов анализа естественного языка:

- 1960-2010, pre Deep Learning era
- 2010-now, Deep Learning era

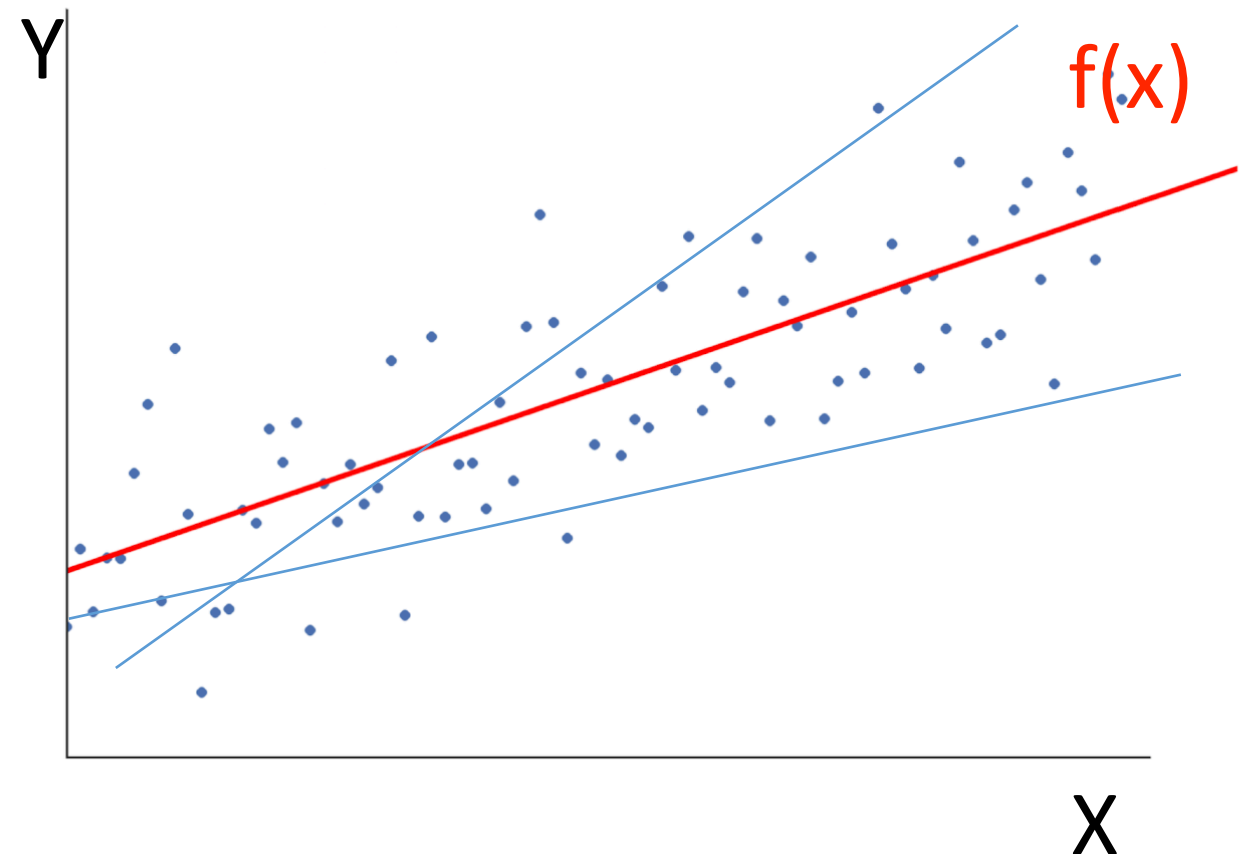
# Recap: Machine Learning

Целевая переменная  
*Target variable*

Исследуемый  
класс моделей

$$y = f(x), \text{ где } f \in F$$

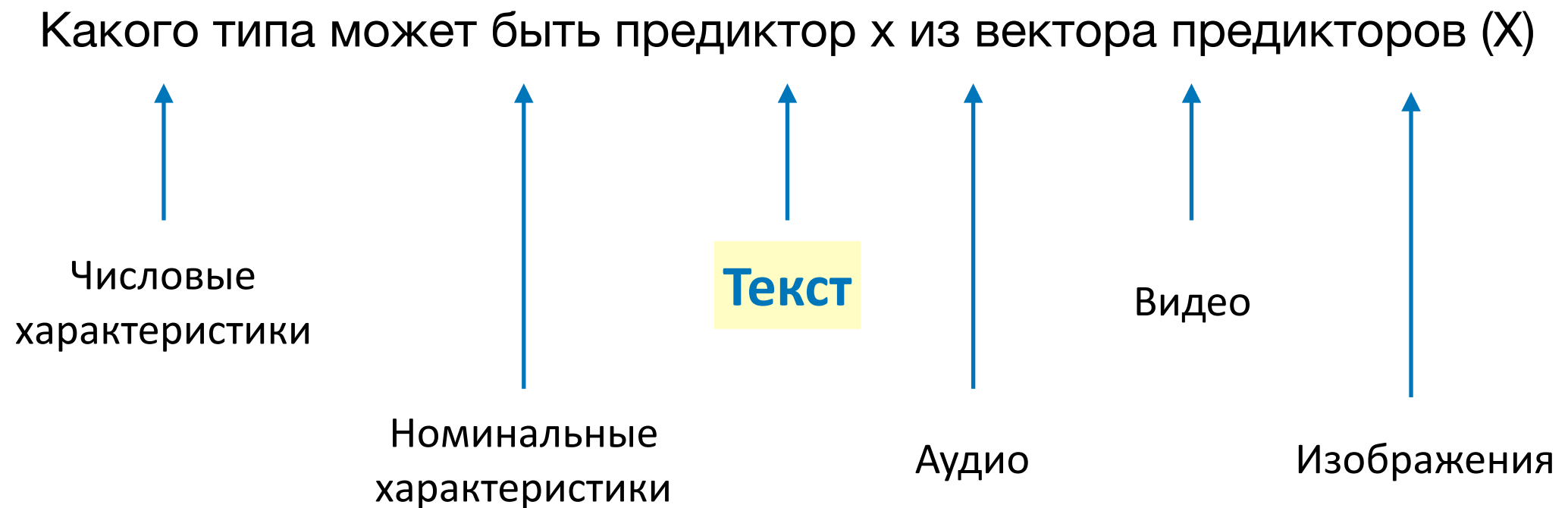
Вектор предикторов  
*Features*



# Recap: Machine Learning

Три главные задачи Machine Learning	
Классификация	$y \in \mathbb{X}$ Категориальная переменная
Регрессия	$y \in \mathbb{R}$ Числовая переменная
Кластеризация	$y$ отсутствует

# Recap: Machine Learning



# Text Representations

Главный вопрос: Как представить текст в виде вектора?

## Основная терминология

**Document** - исследуемая единица текста (например, предложение, абзац, книга)

**Corpus** - весь набор документов, который нам доступен

**Term** - составная часть документа (слово, символ или n-грамм)

**Vector-Space Model** - представление текстовых документов в виде векторов

**Bag-Of-Words** - представление текста в виде неупорядоченного множества слов

**Vocabulary** - проиндексированный список всех термов в корпусе

# Text Representations

1. На входе:

1. John likes to watch movies. Mary likes movies too.
2. John also likes to watch football games.

2. Нумеруем (в произвольном порядке) все слова корпуса. Например, так

- 'John' : 0, 'likes' : 1, 'to' : 2, 'watch' : 3, 'movies' : 4 ... 'games': 10

3. Записываем оба документа в 10—мерном пространстве слов. Каждому слову ставим в соответствие его частоту (Term Frequency) в документе.

- $\text{text}_1 = (1, 2, 1, 1, 2, 1, 0, 0, 0, 0)$
- $\text{text}_2 = (1, 1, 1, 1, 0, 0, 1, 0, 1, 1)$

Пример обучающей выборки с текстом

$$X_1 = (\text{'Male'}, 29, 1, 2, 1, 1, 2, 1, 0, 0, 0, 0), \quad Y_1 = 0$$

$$X_2 = (\text{'Female'}, 24, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1), \quad Y_2 = 1$$

# Text Representations

Как быть, если размерность такого представления (число различных слов) 15 миллионов?

**Sparse** Vector Representation

$X = (1, 2, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \dots)$

$X = (0:1, 1:2, 4:1, 12:1, 18:2, 23:1, 25:1 \dots)$

Храним только ненулевые элементы!

# Text Representations

Как быть, если словарь составили, а тексты приходят с новыми словами?

- Можно каждый раз расширять словарь новыми термами, но лучше:
- Hashing Trick

Словарь не используется. Порядковый номер каждого слова назначается (по некоторому правилу) хэш-функцией

## Пример

1. Приходит новый текст: “Пионервожатый Всеволод воплотил мечты в реальность”
2. Для новых слов документа назначаются новые индексы  
Пионервожатый => индекс 205308  
Всеволод => индекс 12928
3. Документ кодируется (205308:1, 12928:1, 1056:1, 64:1, 521:1)



# Text Representations

N-граммы - наборы из n последовательных слов

*Пример би-граммов*

to be or not to be

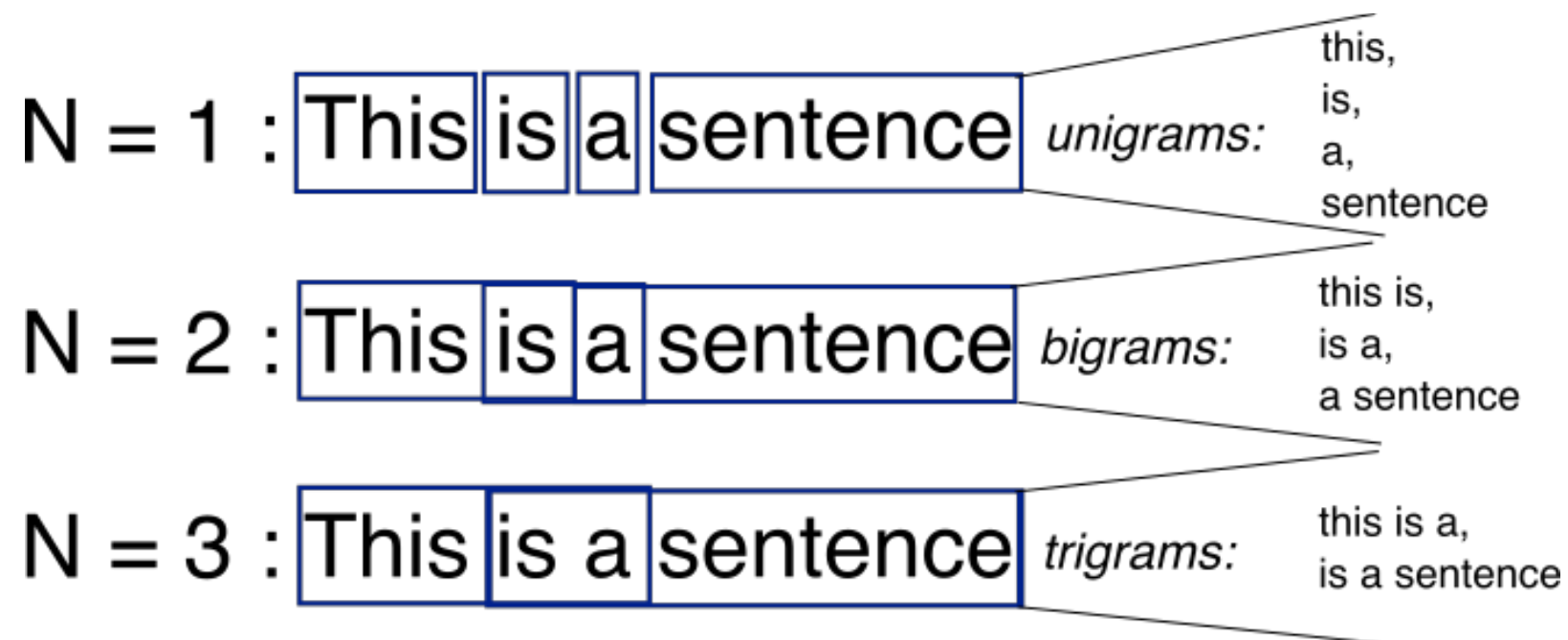
- to be (2)
- be or (1)
- or not (1)
- not to (1)

Текст выше может быть закодирован так: (2, 1, 1, 1)

# Text Representations

Типичные виды n-граммов

- униграммы
- биграммы
- триграммы
- skip-граммы (исключается слово посередине)



# Text Representations

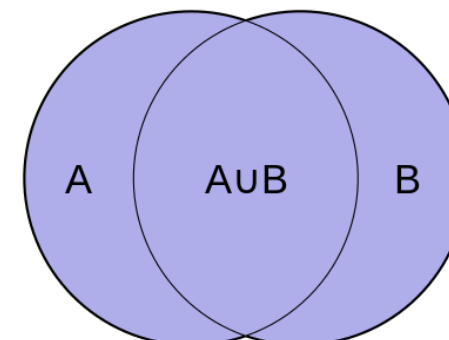
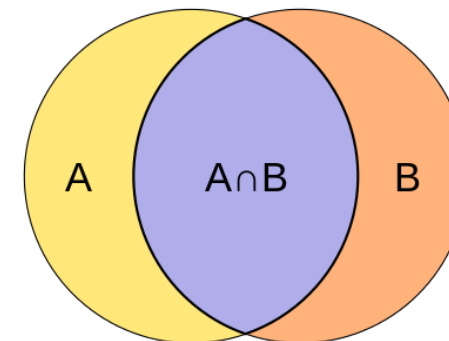
Как определить близость двух документов?

## Cosine Distance

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

## Jaccard Distance

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$



# Text Representations

Какой характеристикой кроме частоты можно описать слово?

TF-IDF transformation - большой вес у более редких слов

$$w_{i,j} = tf_{i,j} \times \log \left( \frac{N}{df_i} \right)$$

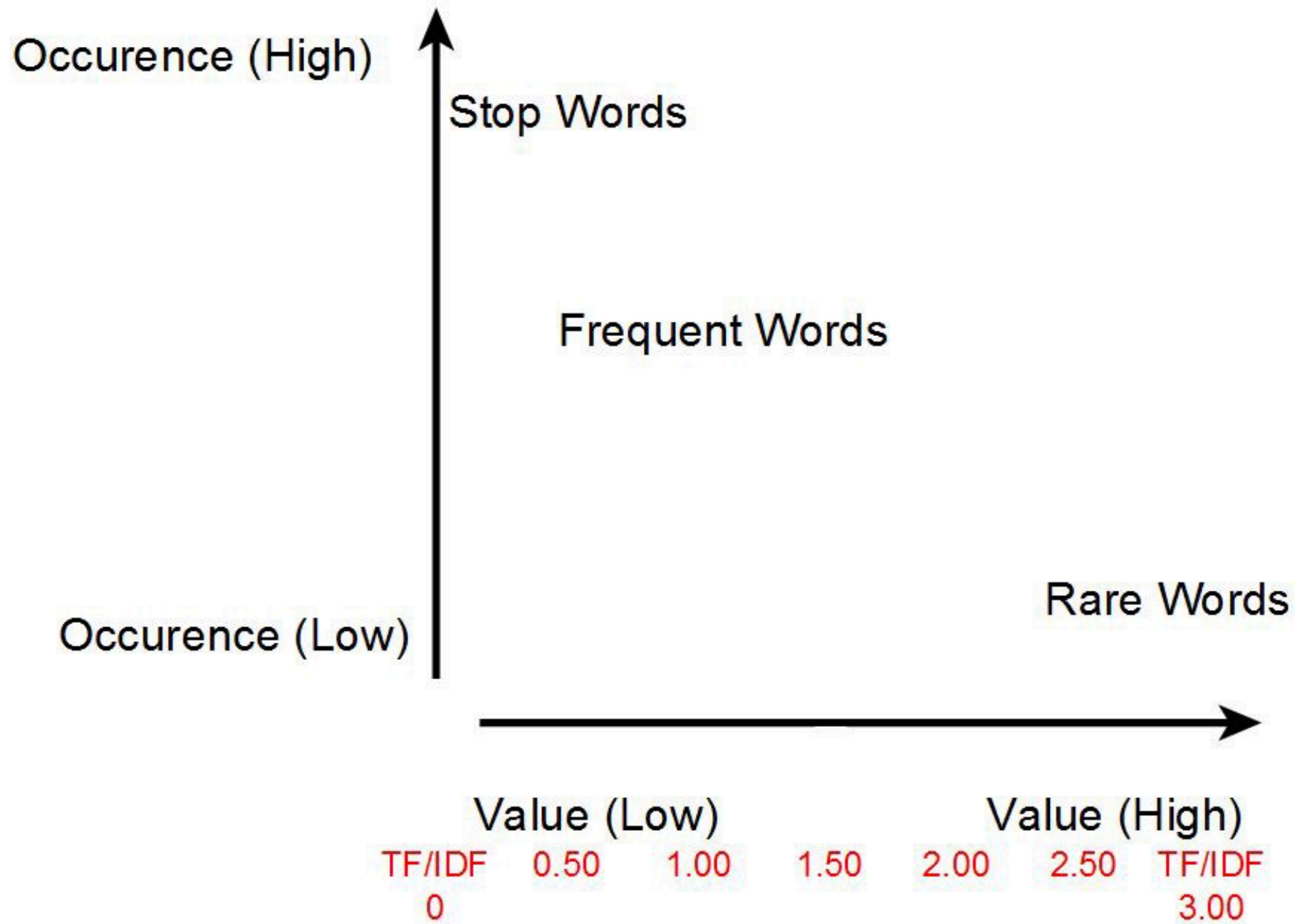
$tf_{ij}$  = number of occurrences of  $i$  in  $j$   
 $df_i$  = number of documents containing  $i$   
 $N$  = total number of documents

Зачем? Совпадение по редким словам гораздо больше говорит о схожести текстов, чем совпадение по союзам и предлогам.

## Пример

(‘и’, ‘высококогерентный’, ‘картошка’) => (0.5, 23.0, 2.3)

# Text Representations



# Text Representations

**Word Embedding** - представление слова в пространстве меньшей размерности. При этом “близкие” по смыслу слова оказываются рядом.

**Word2Vec** - классический алгоритм, решающий данную задачу.

Контекстом называют соседние слова (слева и справа) в предложении.

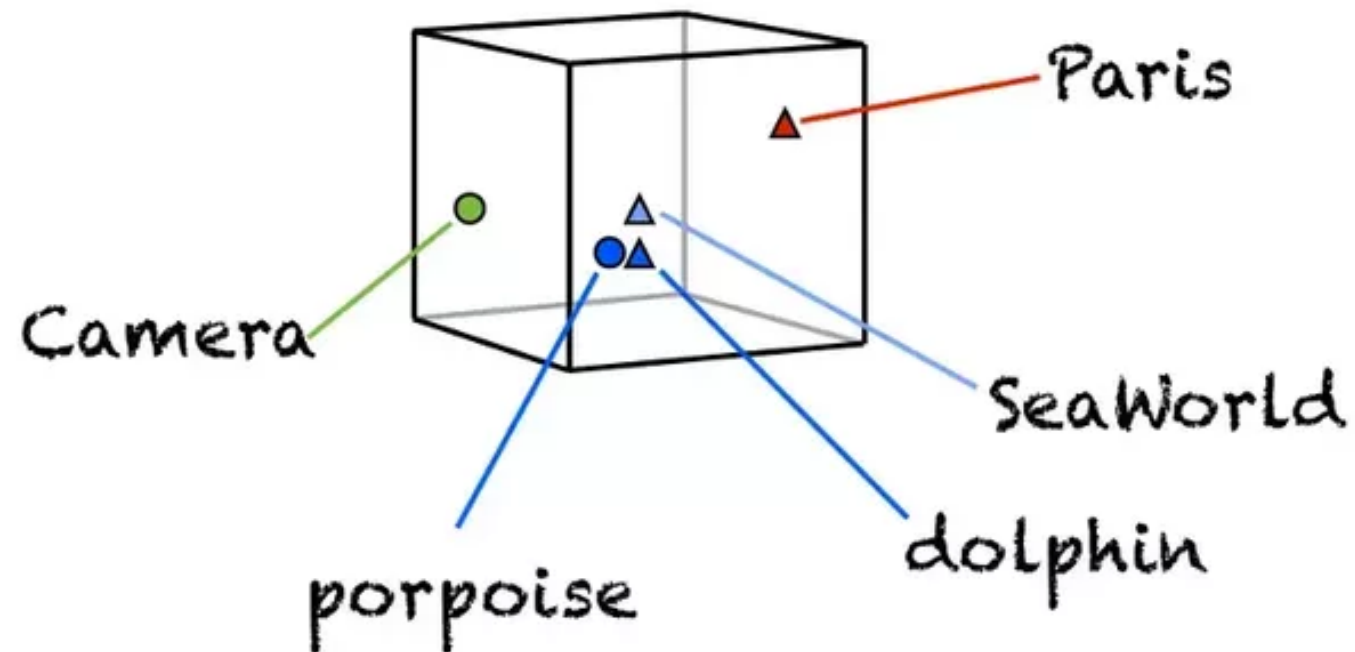
Два слова “близки” по смыслу, если часто встречаются в одном контексте.

The	quick	brown	fox jumps over the lazy dog.	→	(the, quick) (the, brown)
The	quick	brown	fox jumps over the lazy dog.	→	(quick, the) (quick, brown) (quick, fox)
The	quick	brown	fox jumps over the lazy dog.	→	(brown, the) (brown, quick) (brown, fox) (brown, jumps)

# Text Representations

Зачем нужен Word2Vec?

- Векторное представление слов в пр-ве произвольной размерности
- Учет семантики (смысла) слова
- Удобный инструмент для определения синонимов



# Text Preprocessing

## Основные преобразования

- Normalization

*приведение к нижнему регистру, удаление знаков препинания*

- Tokenization

*разбиение текста на слова*

- Stemming

*выделение основы слова*

- Lemmatization

*приведение к нормальной форме слова*

- Stop-Word Removing

*удаление общих слов*



# NLP Libraries

Классические библиотеки, используемые для NLP

- **Sklearn** (модуль `feature_extraction.text`)
- **NLTK**
- Gensim (фокус на topic modeling)
- Word2Vec (векторное представление а алгебры над словами)
- core NLP (профессиональная java-библиотека)
- Keras (есть встроенный preprocessing)
- bigARTM (инструмент для Topic Modeling)