СӘТБАЕВ УНИВЕРСИТЕТІ



SATBAYEV UNIVERSITY

Основы искусственного интеллекта. Лекция 9. Обработка текста и NLP. Введение в обработку естественного языка, основные методы и модели

Преподаватель: **Қасенхан Арай Мейрамбайқызы, доктор PhD, ассоциированный профессор кафедры «Программная инженерия»**

a.kassenkhan@satbayev.university

Содержание

1. Что такое NLP
2. Основные задачи NLP
3. Токенизация
4. Нормализация текста
5. Bag-of-Words и TF-IDF
6. Word Embeddings
7. Классификация текста
8. NLP pipeline
9. TF-IDF в Scikit-learn
10. Обучение модели
11. Предсказание новых текстов

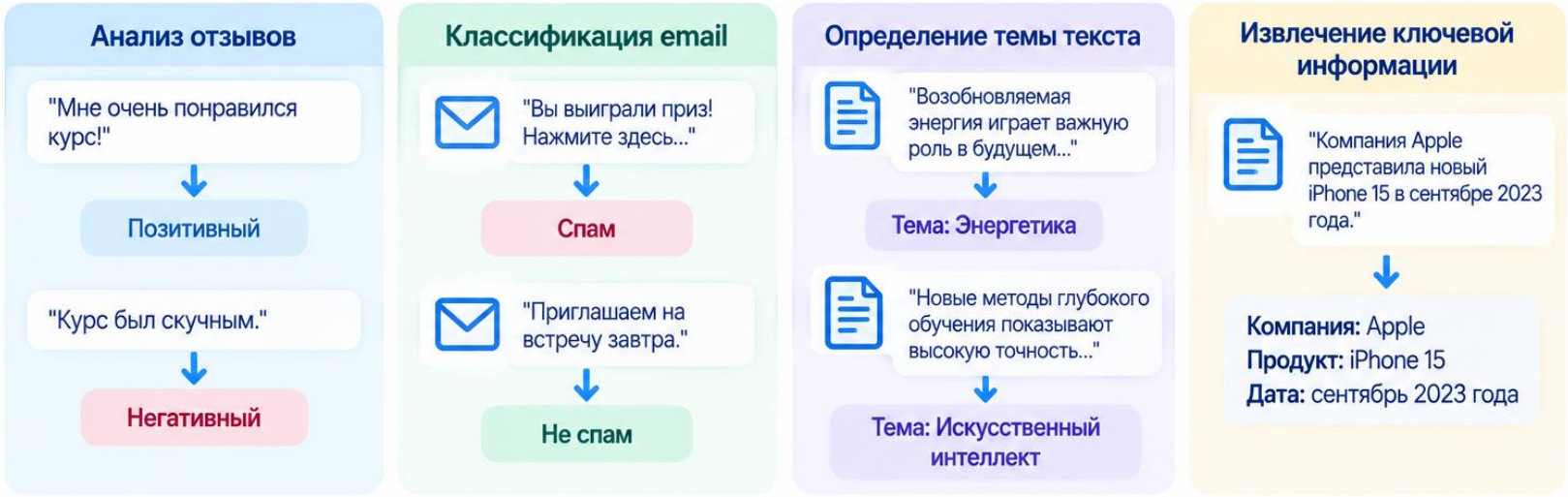
По завершению урока Вы будете знать:

1. **Что такое NLP и какие задачи оно решает**
2. **Как подготовить текст к машинному обучению**
3. **Что такое токенизация, лемматизация и стоп-слова**
4. **Как преобразовать текст в числовые признаки**
5. **Чем отличаются BoW, TF-IDF и embeddings**
6. **Как построить простую модель классификации текста**
7. **Как использовать Scikit-learn для NLP.**

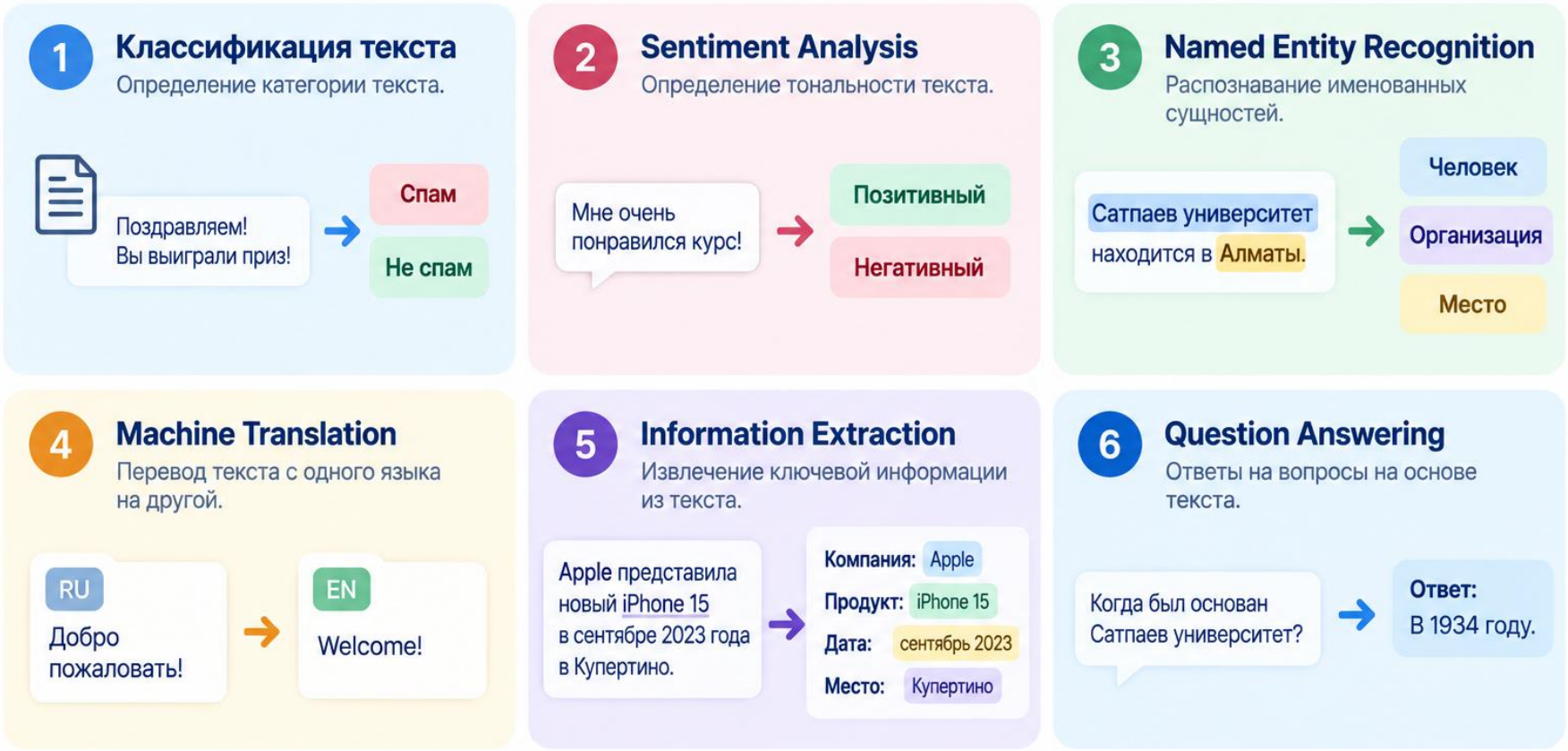
Что такое NLP?

Natural Language Processing — область ИИ, которая позволяет компьютерам обрабатывать и анализировать человеческий язык.





Основные задачи NLP



NLP Pipeline



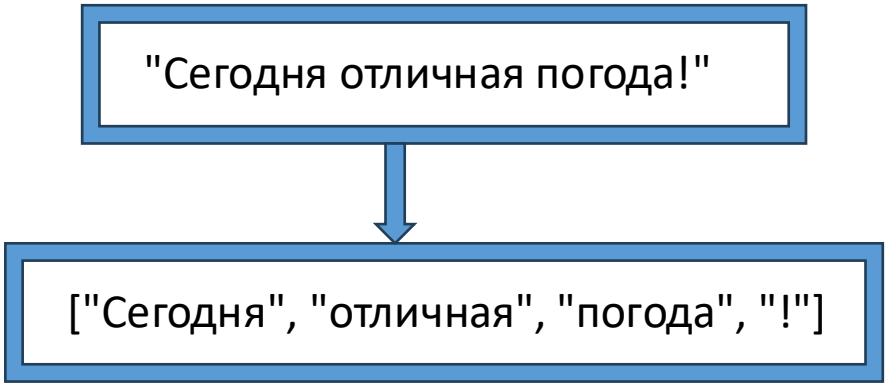
Например:

Отзыв → обработка → TF-IDF → классификатор → Positive

Компьютер не работает с текстом напрямую — текст необходимо преобразовать в числовое представление.

Токенизация

Токенизация — разбиение текста
на отдельные элементы — токены.



Токеном может быть слово, часть слова, символ или другой фрагмент текста.

Нормализация текста

1 Лемматизация

машины → **машина**

ходил → **ходить**

лучше → **хороший**

Получаем нормальную словарную форму.

2 Стемминг

машины → **машин**

машинами → **машин**

машинам → **машин**

Получаем основу слова.

3 Stop words

"Мы идем в парк на прогулку"

↓

["идем", "парк", "прогулку"]

Удаляем стоп-слова (часто встречающиеся слова).

Цель предобработки — уменьшить шум и привести текст к форме, удобной для анализа.

Как представить текст числами?



Текст → числовые признаки → ML-модель

Bag-of-Words

Есть два текста:
D1: "ИИ анализирует текст"
D2: "ИИ генерирует текст"

Словарь:
ИИ | анализирует | генерирует | текст

Получаем:

	ИИ	анализирует	генерирует	текст
D1	1	1	0	1
D2	1	0	1	1

Плюс: просто.
Минус: не учитывает смысл и порядок слов.

TF-IDF: какие слова важнее?

Документ 1

"машинное обучение использует данные"

Документ 2

"анализ данных использует данные"

Часто встречается в одном документе

Значение ↑

Слово	Частота в документе 1
машинное	1
обучение	1
использует	1
данные	1

Встречается почти во всех документах

Значение ↓

Слово	В скольких документах
машинное	1
обучение	1
анализ	1
использует	2
данные	2

TF-IDF даёт больший вес словам, которые важны для конкретного документа, но встречаются реже в других документах.

$$TF\text{-}IDF = TF \times IDF$$

Word Embeddings

BoW / TF-IDF

слова → отдельные признаки

Текст: "кошка и кот — домашние животные"

↓

кошка | и | кот | домашние | животные

↓

Слово	Документ 1 (BoW / TF-IDF)	Документ 2 (BoW / TF-IDF)
кошка	1	0
кот	1	0
домашние	1	1
животные	1	1
и	0	1
...	...	...

Каждое слово — отдельный признак (обычно число: количество или TF-IDF).

Embeddings

слово → [0.21, -0.37, 0.84, ...]

кошка → [0.21, -0.37, 0.84, ...]

Слова с похожим смыслом находятся ближе в пространстве векторов.



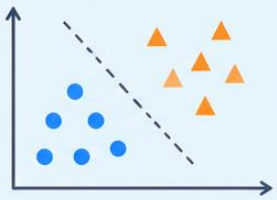
Embeddings представляют слова плотными векторами, где близкие по смыслу слова имеют похожие векторы.

Классификация текста



Классификация текста = преобразование текста в признаки + ML-модель.

Какие модели используются для текста?

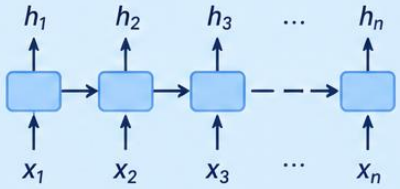


Классические ML

Ручное представление признаков

Naive Bayes · Logistic Regression · SVM

Простые, эффективные, хорошо работают на небольших данных.



Deep Learning

Работа с последовательностями

RNN · LSTM

Учитывают контекст и порядок слов, хорошо работают с последовательными данными.



Transformers

Масштабируемые языковые модели

BERT · GPT · современные языковые модели

Используют механизм внимания, могут обучаться на больших данных и решать широкий круг задач.

Выбор модели зависит от задачи, объёма данных и требуемого качества.

Практика: наши данные

```
import pandas as pd

data = {
   "review": [
      "Мне очень понравился продукт",
      "Качество ужасное",
      "Отличный сервис",
      "Очень разочарован товаром"
   ],
   "label": [1, 0, 1, 0]
}

df = pd.DataFrame(data)
```

1 → Positive

0 → Negative

Преобразуем текст с TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()

X = vectorizer.fit_transform(df["review"])
y = df["label"]
```



fit → создаёт словарь и вычисляет веса
transform → преобразует текст в числа
fit_transform → делает оба шага

Обучаем классификатор

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.25,
    random_state=42
)

model = MultinomialNB()

model.fit(X_train, y_train)
```



Проверяем модель

```
from sklearn.metrics import accuracy_score

y_pred = model.predict(X_test)

print(
    accuracy_score(y_test, y_pred)
)
```

y_test → реальные ответы
y_pred → ответы модели
accuracy → доля правильных ответов

Проверяем новый отзыв

```
new_review = [
    "Мне очень понравилось обслуживание"
]

X_new = vectorizer.transform(new_review)

prediction = model.predict(X_new)

print(prediction)
```

Для новых данных используется transform(), а не fit_transform().



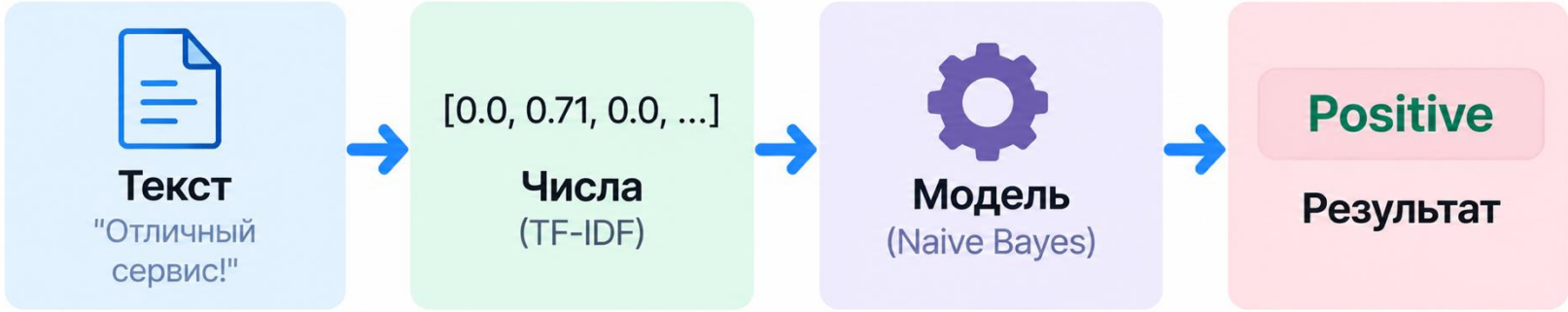
Весь NLP Pipeline





Заключение

NLP → работа компьютера с человеческим языком
Preprocessing → подготовка текста
TF-IDF / Embeddings → преобразование текста в числа
ML-модель → анализ и предсказание



Чтобы алгоритм машинного обучения мог работать с языком, текст необходимо подготовить и представить в числовой форме.