СӘТБАЕВ УНИВЕРСИТЕТІ SATBAYEV UNIVERSITY

Основы искусственного интеллекта. Лекция 2. Основы Python для ИИ. Основы языка Python, структуры данных, библиотеки для научных вычислений (NumPy, Pandas)

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

a.kassenkhan@satbayev.university

Содержание

1. Python как инструмент искусственного интеллекта
2. NumPy: массивы и векторные вычисления
3. Pandas: работа с табличными данными
4. Подготовка данных для задач ИИ

По завершению урока Вы будете знать:

1. **Почему Python широко используется в ИИ**
2. **Как создавать и обрабатывать массивы NumPy**
3. **Как применять векторные операции и broadcasting**
4. **Как загружать, исследовать и преобразовывать данные с помощью Pandas**
5. **Как подготовить простой набор данных для дальнейшего использования в машинном обучении**

Почему Python используется в ИИ?

Простой синтаксис

→ меньше кода, легче экспериментировать

Экосистема библиотек

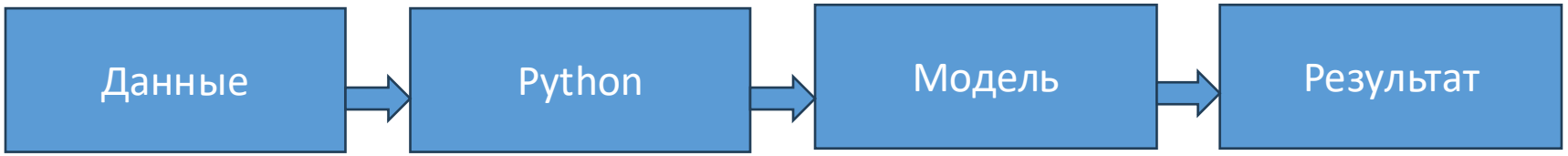
→ NumPy · Pandas · Scikit-learn · TensorFlow · PyTorch

Работа с данными

→ таблицы · массивы · изображения · текст

Интеграция

→ C/C++ · GPU · облачные платформы



От Python-списка к NumPy-массиву

```
numbers = [10, 20, 30, 40]

print(numbers)
print(type(numbers))
```

```
[10, 20, 30, 40]
<class 'list'>
```

```
import numpy as np

arr = np.array([10, 20, 30, 40])

print(arr)
print(arr.shape)
print(arr.dtype)
```

```
[10 20 30 40]
(4,)
int64
```

NumPy array = основная структура для числовых вычислений

shape — размерность массива
dtype — тип данных

NumPy: векторизация

Без NumPy-векторизации:

```
result = []

for x in arr:
    result.append(x * 2)

print(result)
```

[20, 40, 60, 80]

С NumPy:

```
result = arr * 2

print(result)
```

[20, 40, 60, 80]

Векторизация = меньше циклов + короче код + быстрее вычисления

NumPy: индексация и фильтрация

Индексация:

```
scores = np.array([45, 72, 88, 56, 91])

print(scores[0])
print(scores[1:4])
```

```
45
[72 88 56]
```

Фильтрация:

```
scores >= 70
```

```
[False True True False True]
```

```
high_scores = scores[scores >= 70]

print(high_scores)
```

```
[72 88 91]
```

NumPy позволяет выбирать данные по условию без написания циклов.

NumPy: матрицы и Broadcasting

Матрица:

```
X = np.array([
    [10, 20, 30],
    [40, 50, 60]
])

print(X.shape)
```

(2, 3)

2 строки × 3 столбца

Broadcasting:

```
X = X / 10

print(X)
```

```
[[1. 2. 3.]
 [4. 5. 6.]]
```

```
bias = np.array([1, 2, 3])

print(X + bias)
```

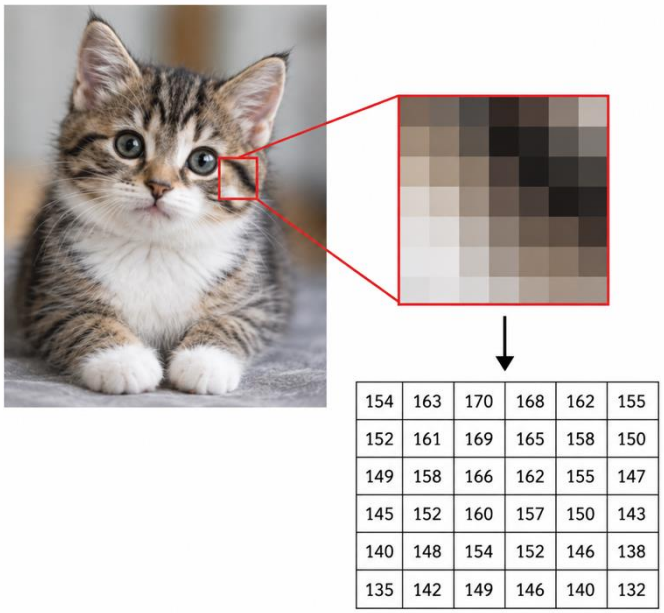
```
[[2. 4. 6.]
 [5. 7. 9.]]
```

В ИИ: строки — объекты, столбцы — признаки

Broadcasting → операции без ручного повторения данных

Как ИИ «видит» данные

ИЗОБРАЖЕНИЕ:



154	163	170	168	162	155
152	161	169	165	158	150
149	158	166	162	155	147
145	152	160	157	150	143
140	148	154	152	146	138
135	142	149	146	140	132

ТЕКСТ:

"AI is useful"

↓

["AI","is",...]

↓

[101,45,892]

ТАБЛИЦА:

Age	Score	Hours
20	85	4.5
21	72	3.0

↓

NumPy-массив

Реальные данные → Числа → Модель ИИ → Результат

Строки — объекты • Столбцы — признаки

Pandas: создаём DataFrame

Pandas = работа с табличными данными

```
import pandas as pd

data = {
   "hours": [2, 5, 3, 7],
   "score": [60, 85, 70, 95]
}

df = pd.DataFrame(data)

print(df)
```

	hours	score
0	2	60
1	5	85
2	3	70
3	7	95

DataFrame — таблица из строк и столбцов.

Pandas: загрузка и первичный анализ данных

Перед обучением модели сначала изучаем данные

ЗАГРУЖАЕМ:

```
import pandas as pd

df = pd.read_csv("students.csv")

print(df.head())
```

	age	hours	score
0	20	2.5	60
1	21	5.0	85
2	19	3.5	70
3	22	7.0	95

```
print(df.shape)
df.info()
print(df.describe())
print(df.isnull().sum())
```

ИССЛЕДУЕМ:

df.head()	первые строки
df.shape	размер таблицы
df.info()	типы данных
df.describe()	статистика
isnull()	пропуски

Pandas: фильтрация и создание признаков

После изучения данных начинаем их преобразовывать

1. ФИЛЬТРАЦИЯ:

Студенты с score ≥ 70

```
good_students = df[df["score"] >= 70]
```

name	score
Ali	85
Dana	70
Max	95

2. СОЗДАНИЕ ПРИЗНАКА:

```
df["passed"] = df["score"] >= 70
```

Score	passed
60	False
85	True
70	True
95	True

passed → новый feature

CSV → Загрузить → Исследовать → Фильтровать → Преобразовать → Создать признаки → Данные для ML

От данных к машинному обучению

```
X = df[["hours"]]
y = df["score"]
```

X → признаки (features)
y → целевая переменная (target)

В большинстве задач машинного обучения модель не получает «сырые» данные — сначала их необходимо подготовить.

