ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ

Лекция 6

Очистка данных

Оспан Ә.Ғ., PhD, доцент-исследователь

КазНУ, кафедра ИИ и Big Data

2026-2027 учебный год

Цель и содержание лекции

Цель и результаты обучения

- Различать дефекты схемы, типов и значений.
- Выбирать обоснованную обработку пропусков и повторов.
- Проверять результат очистки через счетчики и постусловия.

Содержание лекции

1. **Паспорт набора**
2. **Пропуски и типы**
3. **Строки и фильтры**
4. **Дубликаты**
5. **Контроль результата**

Очистка: от правила к проверке

1

Определить правило

Назвать допустимые типы, значения и связи между полями.

2

Найти нарушения

Построить маски и посчитать затронутые записи.

3

Выбрать действие

Исправить, оставить неизвестным или отправить в карантин.

4

Проверить результат

Сопоставить счетчики, распределения и сохранность исходника.

- Правила зависят от смысла признака и задачи.
- Карантин сохраняет сомнительные строки с причиной.

Воспроизводимая очистка = исходник + правила + код + журнал.

Три уровня проверки качества

Структура

- Число и названия колонок.
- Разделитель, кодировка, заголовок файла.

Представление

- Число записано текстом или с пробелами.
- Условный маркер скрывает неизвестное значение.

Смысл

- Число нарушает предметный диапазон.
- Поля одной записи противоречат друг другу.

Один общий счетчик «плохих строк» скрывает разные причины; каждому уровню нужна своя проверка.

Паспорт football.csv и ограничения

Поле или свойство	Что известно	Следствие
Объем	12897 строк, 42 колонки	Проверяем форму после чтения
Пропуски	В исходном файле их нет	Не создаем «найденные» NaN
Unnamed: 0	Старый индекс строк	Не аналитический признак
Name	Уникальность не гарантирована	Не удаляем совпадения имен
Penalties	Рейтинг навыка пенальти	Не число голов
Wage / Value	Единицы и период неизвестны	Не называем годовыми евро

Исторический снимок не описывает текущую реальность.

Загрузка и первичный аудит

Пример Python

```
from pathlib import Path
import pandas as pd

raw = pd.read_csv(Path('data') / 'football.csv')
assert raw.shape == (12897, 42)
print(raw.isna().sum().sum())
raw.info()
```

Результат: Ожидаемая форма: (12897, 42); число пропусков: 0.

Проверяем

- Форму таблицы и число пропусков.
- Типы после автоматического распознавания.

Интерпретируем

- Ноль NaN относится к этому файлу.
- Тип данных не подтверждает смысл значения.

Ошибка распознавания и ошибка значения

Распознавание

- «ошибка» не превращается в число.
- errors="coerce" создает служебный пропуск.
- Исходный пропуск учитываем отдельно.

Предметный диапазон

- «-4» превращается в число -4.
- Правило возраста выявляет нарушение смысла.
- Границы должны быть обоснованы задачей.

После преобразования типа нужна отдельная валидация значений и прозрачный счетчик для каждой причины нарушения.

Считаем две причины ошибок отдельно

Пример Python

```
demo = pd.DataFrame({
    'age_text': ['21', ' 25 ', 'ошибка', '-4']
})
demo['age'] = pd.to_numeric(
    demo['age_text'], errors='coerce'
)
parse = demo['age_text'].notna() & demo['age'].isna()
domain = demo['age'].notna() & ~demo['age'].between(0, 120)
print(parse.sum(), domain.sum())
```

Результат: 1 1

Маска parse

- Нечисловой текст при известном исходнике.
- Не смешиваем с первоначальным пропуском.

Маска domain

- Число распознано, но нарушает границы.
- 0-120 здесь только учебное правило.

Что делать с неизвестным значением

Решение	Польза	Ограничение
Оставить NA	Сохраняется неизвестность	Не все расчеты определены
Карантин	Сохраняется причина проблемы	Основной набор уменьшается
Исключить из расчета	Простой локальный анализ	Может измениться состав
Заполнить медианой	Число строк сохраняется	Разброс искусственно сужается
Запросить источник	Можно восстановить факт	Не всегда доступно сразу

Неизвестное значение не равно нулю.

Строковая нормализация без потери смысла

Обычно прозрачные действия

- strip удаляет внешние пробелы.
- casefold создает поисковый ключ.
- Исходное написание сохраняется отдельно.

Нужна предметная проверка

- Удаление внутренних пробелов.
- Замены по коротким подстрокам.
- Объединение неизвестных категорий.

Храните соответствия «исходное значение - нормализованный ключ» и проверяйте, какие категории объединены вашим правилом.

Поисковый ключ и буквальный поиск

Пример Python

```
clubs = pd.Series([' Alpha ', 'alpha', None],
                  dtype='string')
key = clubs.str.strip().str.casefold()
has_al = key.str.contains('al', regex=False, na=False)
known_without = key.notna() & ~has_al
print(has_al.tolist())
print(known_without.tolist())
```

Результат: has_al: [True, True, False]; known_without: [False, False, False]

Буквальный поиск

- regex=False не трактует текст как шаблон.
- na=False делает маску определенной.

Три группы

- Совпадение, известное несовпадение, NA.
- Отрицание маски само по себе включает NA.

Фильтрация: .loc и query

Пример Python

```
work = raw.drop(columns=['Unnamed: 0']).copy()
limit = 30
mask = (work['Age'] < limit) & (work['Stamina'] >= 70)
selected = work.loc[mask].copy()
same = work.query('Age < @limit and Stamina >= 70')
assert selected.index.equals(same.index)
print(len(selected))
```

Явная маска

- Скобки вокруг каждого сравнения.
- & вместо скалярного and.

Проверка

- Сравниваем выбранные индексы.
- copy обозначает независимую рабочую таблицу.

Одинаковое число строк не гарантирует одинаковый отбор.

Создание категорий: сначала валидация

Правило age_group

- Меньше 23: «до 23».
- 23-35 включительно: «23-35».
- Больше 35: «старше 35»; NA отдельно.

Порядок операций

- Распознать и проверить age.
- Решить судьбу ошибочных значений.
- Проверить функцию на 22, 23, 35, 36, NA.

Категоризация не должна маскировать ошибочные исходные числа возраста.

Повторы строк и повторы сущностей

Совпадение записей

- duplicated сравнивает выбранные поля.
- Служебный индекс может скрыть повтор.
- Удаление требует понятного происхождения.

Совпадение реального объекта

- Name не является надежным ID.
- Версии записи могут различаться.
- Нужны ключ и правило выбора версии.

Вход = сохраненные строки + карантин + подтвержденные повторы.

Ошибки очистки и способы контроля

Ошибка	Последствие	Контроль
Перезаписать исходник	Нельзя пересмотреть решение	Отдельная рабочая копия
Заменить все NA нулем	Искажение смысла и средних	Правило для каждой колонки
Удалить совпавшие имена	Потеря разных сущностей	Надежный ключ
Цепочка присваиваний	Изменение временного объекта	Один явный .loc
Удалить все редкости	Смещение состава	Проверка групп до и после
Скрыть предупреждения	Незамеченная проблема API	Прочитать и устранить причину

Основные результаты лекции

1. Очистка опирается на явное предметное правило.
2. Распознавание числа и проверка диапазона - разные этапы.
3. Пропуск не равен нулю; решение зависит от задачи.
4. Имя не заменяет надежный идентификатор сущности.
5. Журнал и баланс строк делают изменения проверяемыми.

Контрольные вопросы и самостоятельная работа

1. Чем ошибка распознавания отличается от недопустимого числа?
2. Почему ноль пропусков не доказывает качество всего набора?
3. Что произойдет при отрицании contains с na=False?
4. Когда совпадение Name можно считать повтором сущности?
5. Какие счетчики должны объяснять изменение числа строк?

Самостоятельная работа

На шести учебных строках сравните удаление NA и заполнение медианой. Покажите объем, среднее, разброс и ограничения каждого решения.

Источники и документация

Документация

1. pandas: пропущенные значения

https://pandas.pydata.org/docs/user_guide/missing_data.html

2. pandas: строковые данные

https://pandas.pydata.org/docs/user_guide/text.html

3. pandas: выбор и присваивание

https://pandas.pydata.org/docs/user_guide/indexing.html

Основа лекции

- Лекция 6 Очистка данных.pdf
- Лабораторная работа 6 .docx
- Лабораторная работа 6 с ответами.docx