

Программная инженерия. Разработка ПО (Python для продвинутых специалистов. Машинное обучение)






Модуль: Предобработка данных и машинное обучение

Лекция 3: Обработка пропущенных данных

Дата: 19.05.2025

- Почему в данных бывают пропуски?
- Почему пропущенные значения являются проблемой?
- Методы обработки пропущенных значений

Почему в данных бывают пропуски?

-  Ошибки сбора данных (датчики, сбои систем)
-  Неполные анкеты и опросы
-  Проблемы при интеграции из разных источников
-  Пользователь не указал информацию (возраст, пол и др.)
-  Специально не заполняют (например, не применимо)

Почему пропуск - это проблема?

На примере алгоритма линейная регрессия:

- Цель — предсказать доход (income) на основе возраста (age) и стажа (experience):

ID	Age	Experience	Income
1	25	2	40 000
2	30	NaN	50 000
3	NaN	5	55 000
4	35	10	70 000

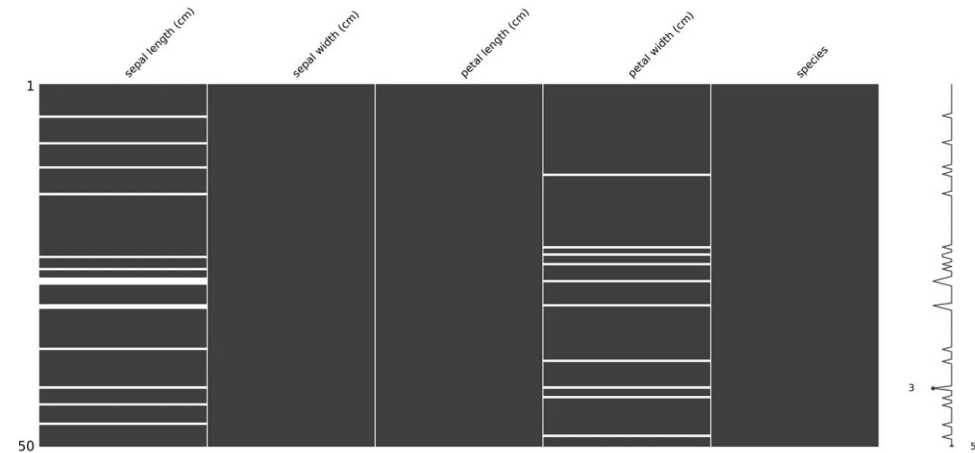
если подать эти данные в **LinearRegression** из **scikit-learn**?

- Модель **выдаст ошибку**: **ValueError: Input contains NaN**
- Она не сможет построить уравнение вида:
$$\text{Income} = a \cdot \text{Age} + b \cdot \text{Experience} + c \cdot \text{Income} =$$
- ...потому что для некоторых наблюдений **нет одного из слагаемых** — и это ломает всю формулу, то есть с NaN невозможно провести ни одну арифметическую операцию, например,
$$\text{NaN} * \text{число} = \text{NaN} - \text{невозможно}$$

Обнаружение пропущенных значений

- Методы describe, либо isna().sum()
- Визуализация пропусков (Seaborn heatmap, missingno)

```
○ import missingno as msno  
○ msno.matrix(df_miss, figsize=(10, 6))
```



- Простые EDA-подходы: группировка, сравнение количества

Стратегии обработки пропущенных значений

Удаление строк

применимо только в том случае, когда

- количество удаленных строк составляет небольшой процент
- удаление не искажает выборку, то есть до удаления и после удаления не должно сильно измениться, например распределение целевого признака

Синтаксис на python, пример

- `df.dropna()`

Стратегии обработки пропущенных значений

Удаление столбцов (если много пропусков)

применимо только в том случае, когда

- столбец почти весь состоит из NaN (пропуск) (>70-80%)
- столбец не несет никакой важной информации, перед удалением необходимо проверить связь целевого столбца с потенциально удаляемым столбцом

Синтаксис на python, пример

- `df.drop(columns=['feature_with_many_nans'])`

Стратегии обработки пропущенных значений

Простая импутация (заполнение) значениями

- Заполнение константами: UNK, 0, -1, 99999
- Может ввести шум (например, 0 в доходе — не значит "нет дохода")
- Требуется добавления дополнительного бинарного признака: `was_missing`

Синтаксис на python, пример

- `df['grade'].fillna('unknown', inplace=True)`

Стратегии обработки пропущенных значений

Заполнение средним/медианой/модой

- Среднее (mean) — при симметричном распределении
- Медиана — при скошенном (например, доход)
- Мода — для категориальных (например, чаще всего встречающееся значение)

Синтаксис на python, пример

- `df['age'].fillna(df['age'].median(), inplace=True)`

Стратегии обработки пропущенных значений

Заполнение средним/медианой/модой

Минусы:

- Не учитывает взаимосвязи между признаками
- Может "размывать" распределения

например, мы можем рассчитать среднее = 1198 и пропуски заменить на 1198

ID магазина	площадь	количество этажей	в ТЦ?	доход от магазина
1	1000	1	1	1000000
2	1569	2	0	200000
3	870	1	0	300000
4	2000	2	0	500000
5	900	1	1	600000
6	850	1	1	1000000
7	1700	2	1	200000
8		2	1	300000
9		2	0	500000
10	700	1	0	600000

ID магазина	площадь	количество этажей	в ТЦ?	доход от магазина
1	1000	1	1	1000000
2	1569	2	0	200000
3	870	1	0	300000
4	2000	2	0	500000
5	900	1	1	600000
6	850	1	1	1000000
7	1700	2	1	200000
8	1198	2	1	300000
9	1198	2	0	500000
10	700	1	0	600000

Стратегии обработки пропущенных значений

Импутация на основе других признаков

- **Импутация с учетом группировки**

Пример: заполнение пропущенного возраста средним по полу:

```
df['age'] = df.groupby('gender')['age'].transform(lambda x: x.fillna(x.median()))
```

- **Использование модели (предсказание пропуска)**

Идея: обучаем модель предсказывать пропущенное значение на основе других признаков

Стратегии обработки пропущенных значений

То есть строим регрессию на наблюдениях с 1 по 7 и 10. Целевая переменная y = площадь магазина, независимые признаки: x_1 = “количество этажей” и x_2 = “в ТЦ”

$$y = 700 + 300 \cdot x_1 + 100 \cdot x_2$$

ID магазина	площадь	количество этажей	в ТЦ?	доход от магазина
1	1000	1	1	1000000
2	1569	2	0	200000
3	870	1	0	300000
4	2000	2	0	500000
5	900	1	1	600000
6	850	1	1	1000000
7	1700	2	1	200000
8		2	1	300000
9		2	0	500000
10	700	1	0	600000

ID магазина	площадь	количество этажей	в ТЦ?	доход от магазина
1	1000	1	1	1000000
2	1569	2	0	200000
3	870	1	0	300000
4	2000	2	0	500000
5	900	1	1	600000
6	850	1	1	1000000
7	1700	2	1	200000
8	1400	2	1	300000
9	1300	2	0	500000
10	700	1	0	600000

Стратегии обработки пропущенных значений

Рекомендуется добавление флагов пропуска

```
df['income_missing'] = df['income'].isna().astype(int)  
df['income'].fillna(df['income'].median(), inplace=True)
```

Иногда сам факт пропуска **информативен**

– Например, отсутствие информации о доходе может указывать на его высокий уровень

М

- MCAR (Missing Completely at Random) = Пропущенные значения отсутствуют полностью случайно
- MAR (Missing At Random) = Пропущенные значения зависят от других известных признаков
- MNAR (Missing Not At Random) = Пропуски зависят от самого значения, которое пропущено



MCAR (Missing Completely at Random) – Пропущенные значения отсутствуют полностью случайно

Что это значит:

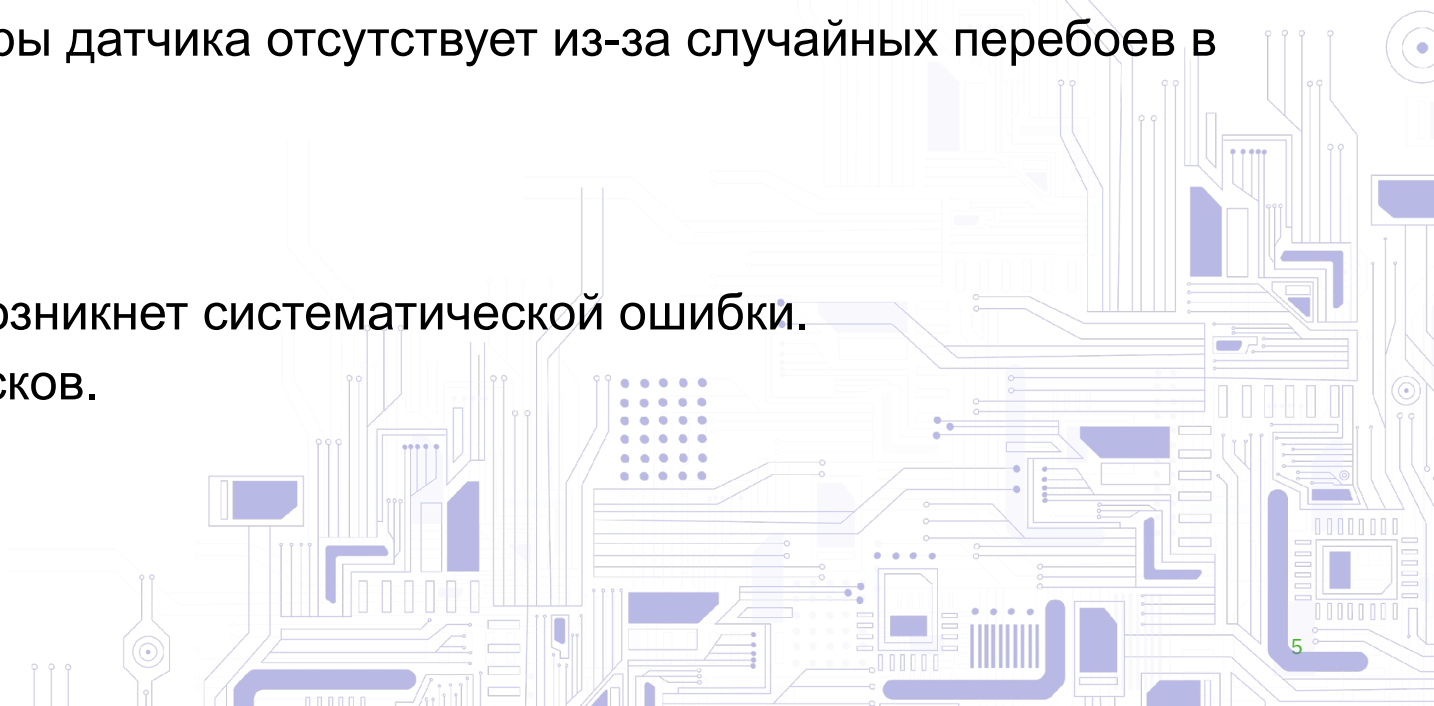
Пропуски не зависят ни от наблюдаемых, ни от ненаблюдаемых данных.

Пример:

- Опросник, где случайно "заглючил" веб-интерфейс и не записал ответы 5% пользователей.
- В датасете часть значений температуры датчика отсутствует из-за случайных перебоев в соединении.

Последствия:

- Если вы удалите такие строки — не возникнет систематической ошибки.
- Это наименее проблемный тип пропусков.



MAR (Missing At Random) = Пропуски зависят от других известных признаков

Что это значит:

Вероятность пропуска **зависит от других столбцов, но не от самого признака с пропусками.**

Пример:

- Женщины чаще не указывают возраст в анкете → пропуски в возрасте **зависят от пола.**
- Доход чаще не заполняют люди младше 25 лет → пропуски в доходе зависят от возраста.

Последствия:

- Просто удалять строки — уже **может ввести в систематическую ошибку** (например, "у нас остались только мужчины").
- Лучше использовать **импутацию с учётом других признаков** — среднее по полу, возрасту и т.п.
- Модели, умеющие учитывать контекст (например, **IterativeImputer**), будут работать лучше

Типы пропущенных данных

MNAR (Missing Not At Random) = Пропуски зависят от самого значения, которое пропущено

Что это значит:

Пропуски **связаны с самим значением**, которого нет — это **самый коварный тип**.

Пример:

- Люди с высоким доходом **скрывают** его → пропуски зависят от дохода.
- Пациенты с тяжёлыми симптомами **не приходят на обследование** → пропуски зависят от скрытого состояния здоровья.

Последствия:

- Здесь **невозможно просто так "восстановить"** пропущенные значения, так как они несут в себе информацию.
- Импутация может **исказить реальность** — например, недооценить доход.
- Иногда лучше: **добавить флаг пропуска** как отдельный признак (например, `income_missing = 1`), и не трогать сами пропуски.
- В идеале — разобраться в бизнес-контексте и **понять причину пропусков**.

Ситуация	Что делать
Мало пропусков и MCAR	Удалить строки или заполнить средним/медианой/квантиль/мода/константа
Пропуски зависят от других признаков (MAR)	Заполнять по группам или использовать модель
Пропуски не случайны (MNAR)	Добавить флаг
Много пропусков в одном признаке	Удалить признак или заменить константой



Передовые
инженерные
школы



МИНОБРНАУКИ
РОССИИ



УНИВЕРСИТЕТ
ИННОПОЛИС



онлайн
университет

Спасибо за внимание

