Очистка данных (data cleaning) – это процесс обнаружения и удаления (или исправления) повреждённых, ложных или неинформативных записей таблицы или целой базы данных. Процесс состоит из двух этапов: поиск и ликвидация (или редактирование).

Основные проблемы «грязных данных»:

- Пропущенные значения
- Выбросы
- Дубликаты
- Неинформативные признаки
- Опечатки

Основные причины появления пропусков:

- Ошибка ввода данных
- Ошибка передачи данных
- Намеренное сокрытие информации
- Прямое отсутствие информации
- Мошенничество

Методы отображения пропусков: список столбцов

Метод	Значение
`data.isnull().sum()`	количество пропусков в каждом столбце
`data.isnull().mean()`	доля пропусков в каждом столбце

Методы отображения пропусков: столбчатая диаграмма

```
null_data = data.isnull()
cols_with_null = null_data[null_data > 0].index
cols_with_null.plot(kind='bar')
```

Методы отображения пропусков: тепловая карта

```
null_data = data.isnull()
cols_with_null = null_data[null_data > 0].index
sns.heatmap(data[cols_with_null])
```

Методы обработки пропущенных значений

Метод	Реализация

Удаление строк/ столбцов, содержащих пропуски	`thresh = data.shape[0]*0.7`
	`data= data.dropna(`
	`  how='any',`
	`  thresh=thresh,`
	`  axis=1`
	`)`
	`data= data.dropna(`
	`  how='any',`
	`  axis=0`
	`)`

Замещение пропусков константой	`data= data.fillna( {'col':data['col'].median()})`

Замещение пропусков константой с добавлением индикатора	`for col in cols_with_null:`
	`  data[col+'was_null'] = data[col].isnull()`
	`data= data.fillna( {'col':data['col'].median()})`

Основные параметры метода dropna()

Параметр	Значение
`axis`	Ось, по которой происходит удаление пропусков (axis=0 — строки, axis-1 — столбцы)
`how`	Определяет тип удаления (any — удаляются все записи, где есть хотя бы один пропуск, all — удаляются только те записи, в которых отсутствуют все значения)
`thresh`	Порог удаления, который определяет минимальное число непустых значений в строке/столбце, при котором она/он сохраняется. Например, если мы установим thresh в значение 2, то мы удалим строки, где число пропусков n-2 и более, где n — число признаков (если axis=0)

Основные методы выявления выбросов: ручной поиск

Поиск аномальных значений с помощью определения границ признака и несостыковок в данных

```
data['col'].describe()
outliers = data[data['col']> 50].index
data.drop(outliers, axis=0)
```

Основные методы выявления выбросов: межквартильный размах

Поиск выбросов по правилу Тьюки: выбросами помечаются наблюдения, не попавшие в интервал $Q_{25} - 1.5 \times IQR$, $Q_{75} + 1.5 \times IQR$. Разработан для признаков, распределённых приблизительно нормально.

```
def outliers_iqr(data, feature):
    x = data[feature]
    quartile_1, quartile_3 = x.quantile(0.25), x.quantile(0.75),
    iqr = quartile_3 - quartile_1
    lower_bound = quartile_1 - (iqr * 1.5)
    upper_bound = quartile_3 + (iqr * 1.5)
    outliers = data[(x<lower_bound) | (x > upper_bound)]
    cleaned = data[(x>lower_bound) & (x < upper_bound)]
    return outliers, cleaned
```

Основные методы выявления выбросов: z-отклонение

Поиск выбросов по правилу правилу трёх сигм: выбросами помечаются наблюдения, не попавшие в интервал $\mu - 3\sigma, \mu + 3\sigma$. Разработан для признаков, распределённых приблизительно нормально.

```
def outliers_z_score(data, feature):
    mu = x.mean()
    sigma = x.std()
    lower_bound = mu - 3 * sigma
    upper_bound = mu + 3 * sigma
    outliers = data[(x < lower_bound) | (x > upper_bound)]
    cleaned = data[(x > lower_bound) & (x < upper_bound)]
    return outliers, cleaned
```

Поиск дубликатов: метод duplicated()

```
mask = data.duplicated(subset=cols)
data = data[mask]
```

Удаление дубликатов: метод drop_duplicates()

```
data= data.drop_duplicates(subset=cols)
```

Поиск неинформативных признаков

```
low_information_cols = []
for col in data.columns:
    top_freq = data[col].value_counts(normalize=True).max()
    nunique_ratio = data[col].nunique() / data[col].count()
    if (top_freq > 0.95) | (nunique_ratio > 0.95):
        low_information_cols.append(col)
```