Разведочный анализ данных в Python

Как получить проверяемый отчёт о сообщениях несколькими командами pandas?

Восемь сообщений достаточно, чтобы проверить весь расчёт вручную. На такой выборке освоим pandas: прочитаем таблицу, сравним классы и проследим, как фильтр меняет знаменатель доли.

1. Прочитать таблицу без скрытых шагов

Перед нами восемь сообщений из SMS Spam Collection. Для первого опыта оставим три столбца: номер строки в исходном файле, класс и длину текста. Значение `ham` обозначает обычное сообщение, `spam` обозначает спам. Длина считается по сохранённому оригиналу. Русский перевод помогает читать примеры в уроке о данных, но здесь не используется для вычисления длины.

Строка источника	Класс	Длина
371	ham	31
5384	spam	42
1353	ham	30
577	spam	69
264	ham	51
3291	spam	105
5040	ham	133
81	ham	22

Выборка специально подобрана для обсуждения разных случаев. Она не случайная, поэтому по ней нельзя оценивать долю спама во всей переписке. Зато она достаточно мала, чтобы проверить программу вручную. Сначала предскажите число строк и число спам-сообщений, затем сверяйтесь с кодом.

Библиотека **pandas** хранит таблицу в объекте `DataFrame`. Импорт `import pandas as pd` даёт библиотеке короткое имя `pd`. В упражнении таблица записана строкой в формате CSV: первая строка содержит имена столбцов, остальные строки значения через запятую. `StringIO` позволяет читать эту строку как файл. Поэтому пример запускается без загрузки вашего файла и не зависит от внешнего сервера с данными.

Пример

Источник — SMS Spam Collection, UCI, Almeida и Hidalgo, CC BY 4.0. Номера исходных строк сохранены для проверки происхождения. Восемь выбранных сообщений не описывают частоты полного корпуса.

> В учебной выборке 7 обычных сообщений и три спам-сообщения. Сколько строк получится после их объединения?

2. Посмотреть размеры и пропуски

Начните с `df.head()`: эта команда показывает первые пять строк. Она помогает заметить неверный разделитель или сместившиеся столбцы, но не доказывает, что остальные строки корректны. `df.shape` возвращает пару «число строк, число столбцов». В нашем примере это `(8, 3)`. Номер строки, который pandas печатает слева, служит индексом и не добавляет четвёртый столбец.

Команда `df.dtypes` показывает тип каждого столбца. У номеров и длин ожидаются целые числа, у класса текст. Если длины вдруг прочитались текстом, проверьте единицы, лишние символы и способ записи пропусков. Одинаково выглядящие значения могут иметь разные типы: число 42 и строка `"42"` требуют разных действий.

Выражение `df["length"]` выбирает один столбец. Проверка `df["length"].isna()` отмечает отсутствующие значения, а `.sum()` считает истинные отметки. В этой маленькой выборке пропусков нет. Это результат проверки, а не основание удалить такую проверку из следующего исследования.

Не сравнивайте пропуск с `NaN` через обычное равенство. Для поиска отсутствующих значений используйте `isna`. Номер источника также заслуживает проверки: повтор одного номера может означать, что запись случайно загрузили дважды. Совпадение длин само по себе дубликатом не считается.

Заметка

`df.shape[0]` даёт количество строк, `df.shape[1]` количество столбцов. Здесь индекс ноль выбирает первый элемент пары, как в обычном списке Python.

> Таблица pandas `df` имела 10 строк и 4 столбца. Добавили столбец с длиной текста, не меняя строк. Чему теперь равно `df.shape[1]`?

3. Отфильтровать и посчитать долю

Сравнение `df["label"] == "spam"` выполняется для каждой строки. Получается последовательность из `True` и `False`, которую называют **маской**. В нашей выборке истинны три позиции. Маска сохраняет порядок строк и позволяет выбрать только подходящие наблюдения.

Запись `df[df["label"] == "spam"]` возвращает таблицу из трёх строк. Если вместо выбора взять среднее самой логической маски, pandas посчитает долю истинных значе-

ний: `True` участвует как единица, `False` как ноль. Получится `3 / 8 = 0.375`, или 37,5 процента. У доли и процента разные масштабы: для процента нужно умножить на сто.

Посмотрите теперь на сообщения длиннее пятидесяти символов. Подходят длины 69, 51, 105 и 133. Среди них два спама из четырёх, поэтому доля спама в этой группе равна половине. Она отличается от доли среди всех восьми строк. Обе величины рассчитаны правильно, но отвечают на разные вопросы.

Записывайте условие фильтра рядом с выводом. Фраза «доля спама равна половине» без упоминания длины создала бы неверное впечатление о выборке. Когда фильтров несколько, полезно после каждого печатать оставшееся число строк: неожиданное уменьшение часто выдаёт ошибку в условии.

Ссылки

pandas: выбор подмножества показывает выбор столбцов и фильтрацию. Проверяйте сначала одну маску, затем соединяйте условия.

> Из 12 сообщений после фильтра осталось 4, среди них 3 спама. Какова доля спама именно среди оставшихся? Введите число от 0 до 1.

Профи. **Почему среднее маски — это доля, и где прячется знаменатель**

Маска — столбец из `True` и `False`. В арифметике pandas считает `True` единицей, а `False` нулём, поэтому `mask.sum()` — число подходящих строк, а `mask.mean()` — та же сумма, делённая на длину маски. Знаменатель здесь — длина маски, а не число строк, которые вас интересуют. Отсюда типичная ошибка.

Пусть нужна доля спама среди сообщений длиннее 50 символов. Два выражения выглядят похоже:

```
long_only = df[df["length"] > 50]
share_a = (long_only["label"] == "spam").mean()
share_b = ((df["label"] == "spam") & (df["length"] > 50
```

В `share_a` сначала отобраны четыре длинных сообщения, затем среди них посчитана доля спама: `$2/4 = 0{,}5$`. В `share_b` маска построена на всех восьми строках: истинны две позиции, длинный спам, а длина маски по-прежнему восемь, поэтому $2/8 = 0{,}25$. Это не доля спама среди длинных, а доля длинного спама среди всех сообщений. Оба числа верны, но отвечают на разные вопросы, и pandas не предупредит о подмене.

Как поймать ошибку: перед `mean()` напечатайте `len()` того объекта, по которому усредняете. Если ожидали четыре, а увидели восемь, знаменатель не тот. Ещё надёжнее записать расчёт в два шага, числитель `sum()` и знаменатель `len()` отдельно, и сверить оба числа вручную на маленькой выборке.

> В таблице pandas `df` 13 сообщений; столбец `label` хранит класс (`spam` или `ham`), `length` — длину в символах. Длиннее 50 символов ровно 6 сообщений, из них 5 — спам. Какое число вернёт выражение `((df["label"] == "spam") & (df["length"] > 50)).mean()`? Ответ с точностью до сотых.

Эпик. **Когда среднее маски лжёт: пропуск в столбце класса**

Правило «доля равна среднему маски» опирается на то, что каждая строка либо `spam`, либо `ham`. Нарушим его. Пусть в восьми сообщениях три спама, три обычных, а у двух класс не записан: в столбце `label` стоит `NaN`. Сравнение `NaN == "spam"` даёт `False`, поэтому маска `df["label"] == "spam"` содержит три `True` и пять `False`, а её среднее равно 3/8 = 0,375. Строки без класса молча попали в знаменатель как «не спам».

Теперь другой способ: `df["label"].value_counts(normalize=True)`. По умолчанию эта команда пропуски отбрасывает, и доля спама получится 3/6 = 0,5. Два правильных вызова, два разных числа, и ни один не сообщил, что сделал с пропусками. Верного ответа нет, пока не решено, что означает отсутствие класса: «не размечено» или «точно не спам».

Что делать: сначала `df["label"].isna().sum()`, затем явно выбрать знаменатель и записать его в отчёт: «среди 6 размеченных сообщений» или «среди всех 8, где неразмеченные считаются обычными». Проверка пропусков нужна именно здесь, а не только для числовых столбцов.

> В таблице pandas 20 сообщений: 4 помечены `spam`, 12 помечены `ham`, у 4 класс отсутствует (`NaN`). Что вернёт `(df["label"] == "spam").mean()`?
>
> - `NaN`: сравнение с пропуском делает результат неопределённым
> - Ошибка: pandas не сравнивает пропуск со строкой
> - 0,25: четыре `True` из шестнадцати размеченных
> - 0,2: четыре `True` из двадцати, строки с `NaN` считаются как `False`

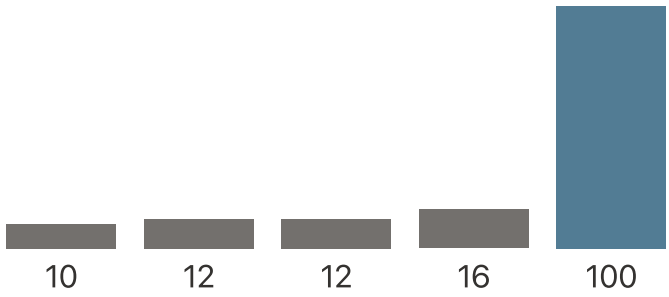
4. Сравнить группы и распределение

Команда `df.groupby("label")["length"].median()` разбивает строки на группы по классу, берёт в каждой длины и вычисляет медиану. У обычных сообщений отсортированные длины равны 22, 30, 31, 51, 133: медиана 31. У спама длины 42, 69, 105: медиана 69. Групповая сводка не требует вручную писать два отдельных фильтра.

При этом обычное сообщение длиной 133 длиннее любого спам-сообщения в выборке. Более высокая медиана у спама не означает, что всякое длинное сообщение — спам. Сводка описывает группу и допускает исключения. Чтобы увидеть их, вернитесь к строкам или нанесите отдельные длины на ось.

В опыте ниже используется учебный набор длительностей, знакомый по EDA. Измените последнее наблюдение и проследите среднее. В Python эту же операцию выполняли бы повторным вычислением `mean` после изменения столбца. График показывает причину изменения числа, а сводная команда ускоряет расчёт.

Проверьте свой прогноз



Пять поездок: 10, 12, 12, 16, 100 мин. Среднее: 30,0 мин; медиана: 12 мин. Каждый столбик — одна поездка.

Проверьте, почему одно необычное наблюдение сдвигает среднее. Это учебные поездки; они не являются SMS.

Интерактивный эксперимент — в электронной версии урока.

Для полноценного отчёта о сообщениях гистограмма длин должна иметь подписанные оси: длина текста и число сообщений в интервале. Если рисуете две группы, используйте одинаковые границы интервалов. Иначе одинаковая высота столбцов может скрывать разные диапазоны. Рядом укажите размер каждой группы; три спам-сообщения недостаточны для устойчивой оценки формы распределения.

Профи. **Что groupby считает на самом деле: среднее против медианы**

Команда `groupby("label")` сама ничего не вычисляет: она лишь раскладывает строки по корзинам `ham` и `spam`. Сводка появляется, когда к корзинам применяют функцию. Метод `agg` принимает список функций и выдаёт таблицу: по строке на класс, по столбцу на функцию.

Сравним среднее и медиану у обычных сообщений. Длины 22, 30, 31, 51, 133; сумма 267, среднее 267/5 = 53,4, медиана 31. Разница в 22 символа появилась из-за одной строки длиной 133: без неё среднее четырёх оставшихся 134/4 = 33,5, а медиана 30,5. Среднее реагирует на каждое значение, медиана только на порядок. У спама длины 42, 69, 105: среднее 72, медиана 69; здесь разницы почти нет, потому что нет выброса.

Типичная ошибка: сравнить среднее одной группы с медианой другой, взятые из разных команд. Чтобы её исключить, считайте все сводки одним вызовом и смотрите на столбец `count`: он напоминает, что три спама слишком мало для выводов о форме распределения.

Три сводки одним вызовом

Цель: Добавьте среднее в сводку и извлеките среднюю длину обычных сообщений.

```
import pandas as pd
from io import StringIO

csv = """label,length
ham,31
spam,42
ham,30
spam,69
ham,51
spam,105
ham,133
ham,22
"""
df = pd.read_csv(StringIO(csv))
summary = df.groupby("label")["length"].agg(["count", "median"])  # TODO: добавьте "mean"
ham_mean = 0  # TODO: возьмите среднее обычных сообщений из summary
print(summary)
print(f"Средняя длина ham: {ham_mean:.1f}")
```

5. Повторить исследование с начала

Лаборатория содержит полный вход и все вычисления. Нажатие запуска заново читает CSV и создаёт таблицу. Так можно проверить, что результат не опирается на забытый фильтр от предыдущего запуска. В стартовом варианте доля спама подменена нулём. Замените эту строку вычислением по столбцу классов.

История

Ранние версии pandas разрабатывались для анализа табличных и временных данных. О проекте pandas. Здесь библиотека экономит повторяющиеся операции, но решение о смысле столбцов принимает исследователь.

Сначала добейтесь верной доли, затем добавьте печать числа пропусков в столбце длины и медиан по классам. На бумаге у вас уже есть ожидаемые значения. Если результат отличается, сначала проверьте размеры и фильтры. Не меняйте исходные значения ради совпадения с ожидаемым ответом.

Первый отчёт pandas

Цель: Вычислите долю спама из столбца label.

```
import pandas as pd
from io import StringIO

csv = """source_row,label,length
371,ham,31
5384,spam,42
1353,ham,30
577,spam,69
264,ham,51
3291,spam,105
5040,ham,133
81,ham,22
"""
df = pd.read_csv(StringIO(csv))
share = 0  # TODO: вычислите долю строк с label == "spam"
print(df.head())
print(f"Строк: {len(df)}; доля спама: {share:.3f}")
print(df.groupby("label")["length"].median())
```

Сохраните три вывода: в выборке восемь строк, спама три, медианы длин равны 31 и 69. Добавьте ограничение о способе отбора. Эти числа годятся для проверки нашего кода; переносить их на все сообщения нельзя. При дальнейшем обучении модели отдельно отложите проверочные примеры. Их ответы не должны подсказывать, какие признаки выбрать или какие пороги настроить.

Эпик. Перенос на полный корпус: оценка вместо уверенности

Полный SMS Spam Collection содержит 5574 сообщения, из них 747 спам: базовая доля около 0,134. В нашей восьмёрке доля спама 0,375, а среди длинных 0,5. Ученик делает вывод: «в корпусе половина длинных сообщений — спам». Проверим этот вывод оценкой, не загружая корпус.

Доля спама среди длинных зависит от двух величин: какая часть спама длинная и какая часть обычных сообщений длинная. Пусть длинными окажутся 90 % спама и 40 % обычных. Тогда длинного спама примерно $0{,}9 \cdot 747 \approx 672$, длинных обычных $0{,}4 \cdot 4827 \approx 1931$, и доля спама среди длинных равна $672/2603 \approx 0{,}26$. Даже при очень «длинном» спаме половина не получается: обычных сообщений в шесть с половиной раз больше, и их меньшая доля длинных всё равно даёт больше строк.

Заметка

Если среднее логической маски непонятно, используйте два шага: `spam_count = (df["label"] == "spam").sum()`, затем `share = spam_count / len(df)`. Получится тот же расчёт с явным знаменателем.

Подумай

Добавьте обычное сообщение длиной тысяча символов. Медиана обычной группы и её среднее изменятся по-разному. Сначала запишите прогноз, затем пересчитайте обе характеристики.

Доля внутри фильтра определяется базовыми частотами классов и тем, как фильтр отбирает каждый класс. Выборка из восьми строк с подобранной вручную пропорцией 5 к 3 не несёт сведений о базовой частоте, и аккуратность в pandas это не исправит. Перед переносом числа на другие данные спросите: какие там доли классов и почему фильтр отбирает их так же.

> В корпусе 4827 обычных сообщений и 747 спам-сообщений. Длинными оказались 30 % обычных и 80 % спама. Какова доля спама среди длинных сообщений? Ответ с точностью до сотых.

Где это встретится дальше

Каждое преобразование таблицы должно иметь понятный смысл, а каждый вывод — явный знаменатель и описание выборки.

Прочитать CSV и проверить размеры, типы и пропуски.

Использовать маску и группировку для расчёта долей и медиан.

Повторить отчёт с исходных данных и указать ограничения выборки.

Домашняя работа

1. В выборке обычные SMS имеют длины 31, 30, 51, 133 и 22 символа, спам — 42, 69 и 105. Найдите число сообщений длиннее 50 символов и долю спама среди них.
2. Пять обычных SMS имеют длины 31, 30, 51, 133 и 22 символа. Вычислите медиану их длин.
3. ★ Из корпуса SMS вручную выбрали восемь сообщений для демонстрации разных случаев: пять обычных и три спама. Объясните, почему по такой выборке нельзя оценивать долю спама во всём корпусе.

Источники

- pandas: первые шаги (https://pandas.pydata.org/docs/getting_started/intro_tutorials/index.html).
- pandas: сводная статистика (https://pandas.pydata.org/docs/getting_started/intro_tutorials/06_calculate_statistics.html).
- SMS Spam Collection, UCI (https://archive.ics.uci.edu/dataset/228/sms+spam+collection).

Повторить эту тему