Конспект по теме "Поиск дубликатов"

Ручной поиск дубликатов

Часто при анализе данных возникают дубликаты. Если не найти дубликаты, то анализ данных может привести к некорректным результатам.

Дубликаты можно искать двумя способами.

Способ 1. Ранее вы уже знакомились с методом `duplicated()`. В сочетании с методом `sum()` он возвращает количество дубликатов. Если выполнить метод `duplicated()` без суммирования на экране будут отображены все строки. Там, где есть дубликаты, будет значение `True`, где дубликата нет - `False`.

Способ 2. Вызвать метод `value_counts()`, возвращающий уникальные значения с их частотой. Его применяют к объекту *Series*. Результат работы метода — список пар «значение-частота», отсортированный по убыванию. Значит, интересующие нас дубликаты будут в начале списка.

Ручной поиск дубликатов с учетом регистра

Дубликаты в строковых данных требуют особого внимания, поскольку регистр имеет значение: заглавная `'A'` и строчная `'a'` с точки зрения python - разные символы, но имеют одинаковое значение - буква *A*.

Чтобы учесть такие дубликаты, все символы в строке приводят к **нижнему регистру** вызовом метода `lower()`.

В pandas символы приводят к **нижнему регистру** методом с похожим синтаксисом: `str.lower()`.

Стемминг

При разделении строк по категориям, проверка вхождения определённых подстрок в строку не всегда может дать корректный результат. Один из

приемлемых вариантов для решения такой задачи - стемминг.

Стемминг - это процесс нахождения основы заданного слова, называемой **стемом**.

В Python для стемминга есть специальная библиотека NLTK. Он содержит стеммеры - специальные объекты, содержащие правила определения стемов. Метод `stem()` применяется для извлечения стема для всех слов в строке.

```
from nltk.stem import SnowballStemmer
russian_stemmer = SnowballStemmer('russian')
russian_stemmer.stem(text)
```

Лемматизация. Библиотека PyMystem

Теория

Стемминг — не единственный алгоритм для поиска слов, записанных в разных формах. Более продвинутый процесс — **лемматизация**, или приведение слова к его словарной форме **(лемме)**.

В русском языке формы записи в словаре (**леммы**) следующие:

- для существительных — именительный падеж, единственное число;
- для прилагательных — именительный падеж, единственное число, мужской род;
- для глаголов, причастий, деепричастий — глагол в инфинитиве несовершенного вида.

Практика

Одна из библиотек с функцией **лемматизации** на русском языке — *pymystem3*, разработана сотрудниками Яндекса. Для многих случаев это гораздо лучше основы слова, которую возвращает *NLTK*. *pymystem3* по умолчанию выдает **список лемматизированных слов** (слов, сведённых к лемме), а *NLTK* — строку.

```
# pymystem3 импортируется так:
from pymystem3 import Mystem
m = Mystem()
lemmas = m.lemmatize(text)
```

Для подсчёта встречаемости значений в списке используется специальный контейнер Counter из модуля collections.

```
from collections import Counter
print(Counter(lst))
```