Лабораторная работа
Исследовательский анализ данных

Выполнила: Короткова Инга Сергеевна

2020 год

Исследовательский анализ данных.

Цель работы – получить навыки работы с библиотеками **Pandas**, **Numpy**.

Задачи:

- Установить необходимые библиотеки
- Импортировать библиотеки
- Загрузить набор данных
- Изучить существующие функции и проделать агрегации

Импорт

импортируем необходимые библиотеки для работы с данными.

```
import numpy as np
import pandas as pd
```

```
# распакуем архив с csv в текущую папку, несжатый csv долго заливается в colab
!unzip data.zip
```

```
Archive:  data.zip
  inflating: data.csv
```

Используем pandas библиотеку для считывания датафрейма.

```
# Используем функцию read_csv для считывания данных из файла csv
data = pd.read_csv('data.csv').drop(columns=['Unnamed: 0', 'id', 'key']).set_index(['artists', 'name'])
```

выведем первые 5 строк (по умолчанию 5).

```
data.head()
```

artists	name	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	liveness	loudness	mode	popularity	release
['Dennis Day']	Clancy Lowered the Boom	0.732	0.819	180533	0.341	0	0.000000	0.160	-12.441	1	8	
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve	0.982	0.279	831667	0.211	0	0.878000	0.665	-20.096	1	5	
['John McCormack']	The Wearing of the Green	0.996	0.518	159507	0.203	0	0.000000	0.115	-10.589	1	6	

для вывода n строк достаточно указать в скобках, сколько именно требуется вывести

```
data.head(8)
```

artists	name	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	liveness	loudness	mode	popularity	releas
['Dennis Day']	Clancy Lowered the Boom	0.732	0.819	180533	0.341	0	0.000000	0.1600	-12.441	1	8	
['Sergei Rachmaninoff',	Piano Concerto											

выведем последние 5 строк датафрейма

```
1 data.tail()
```

artists	name	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	liveness	loudness	mode	popularity	release_date
['Kelly Clarkson']	Born to Die	0.6430	0.481	205787	0.3680	0	0.000000	0.125	-8.310	0	58	2020-03-13
['JoJo']	Man	0.6700	0.661	173760	0.5800	1	0.000055	0.117	-7.718	1	63	2020-03-13
['S.J Morgan']	Rivers	0.9790	0.502	160125	0.0355	0	0.867000	0.106	-26.940	1	66	2020-02-28
['Childish Gambino']	0.00	0.6720	0.174	179387	0.0466	0	0.196000	0.420	-18.458	1	62	2020-03-22
		0.957	0.418	166693	0.193	0	0.000002	0.2290	-10.096	1	4	

Выведем статистические характеристики для каждого численного признака

```
1 data.describe()
```

	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	liveness	loudness	mode	popu
count	168592.000000	168592.000000	1.685920e+05	168592.000000	168592.000000	168592.000000	168592.000000	168592.000000	168592.000000	168592.
mean	0.501360	0.533648	2.327016e+05	0.488577	0.071516	0.169476	0.205151	-11.358180	0.709446	31.
std	0.377993	0.175919	1.223921e+05	0.267346	0.257685	0.315383	0.175896	5.670176	0.454019	21.
min	0.000000	0.000000	5.108000e+03	0.000000	0.000000	0.000000	0.000000	-60.000000	0.000000	0.
25%	0.097800	0.412000	1.721600e+05	0.265000	0.000000	0.000000	0.098200	-14.388000	0.000000	13.
50%	0.515000	0.543000	2.091330e+05	0.480000	0.000000	0.000264	0.134000	-10.466000	1.000000	34.
75%	0.896000	0.662000	2.637070e+05	0.709000	0.000000	0.111000	0.259000	-7.135000	1.000000	48.
max	0.996000	0.988000	5.403500e+06	1.000000	1.000000	1.000000	1.000000	3.855000	1.000000	100.

Выведем статистику тому, какие типы данных содержит датафрейм и их количестве (удобно судить о наличии пропусков)

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 168592 entries, ("['Dennis Day']", 'Clancy Lowered the Boom') to ("['Alina Baraz', '6LACK']", 'Morocco (feat. 6LACK)')
Data columns (total 15 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   acousticness      168592 non-null  float64
 1   danceability      168592 non-null  float64
 2   duration_ms       168592 non-null  int64  
 3   energy            168592 non-null  float64
 4   explicit          168592 non-null  int64  
 5   instrumentalness  168592 non-null  float64
 6   liveness          168592 non-null  float64
 7   loudness          168592 non-null  float64
 8   mode              168592 non-null  int64  
 9   popularity        168592 non-null  int64  
 10  release_date      168592 non-null  object 
 11  speechiness       168592 non-null  float64
 12  tempo             168592 non-null  float64
 13  valence           168592 non-null  float64
 14  year              168592 non-null  int64  
dtypes: float64(9), int64(5), object(1)
memory usage: 28.1+ MB
```

Если требуется вывести только типы данных:

```
1 data.dtypes
```

```
acousticness        float64
danceability        float64
duration_ms           int64
energy              float64
explicit              int64
instrumentalness    float64
liveness            float64
loudness            float64
mode                  int64
popularity            int64
```

Выведем размеры датафрейма (кол-во строк/колонок)

```
tempo               float64
```

```
1 data.shape
```

```
(168592, 15)
```

Вывести количество значений в датафрейме (количество признаков * количество записей)

```
1 data.size
```

```
2528880
```

Использование агрегации средним ('mean') по году ('year') колонки 'duration_ms' и вывод первых пяти записей (head) в **секундах** с сортировкой по убыванию длительности.

```
1 sub_df = data.groupby('year').aggregate({'duration_ms':'mean'}) // 1000
2
3 sub_df.sort_values(by = 'duration_ms', ascending = False).head()
```

year	duration_ms
1976	265.0
1946	262.0
1971	259.0
1990	259.0
1977	257.0

Использование агрегации суммой ('sum') по году ('year') колонки 'popularity' и вывод первых пяти записей (head)сортировкой по убыванию популярности.

```
1 sub_sum_df = data.groupby('year').aggregate({'popularity':'sum'})
2
3 sub_sum_df.sort_values(by = 'popularity', ascending = False).head()
```

year	popularity
2018	134936
2019	132332
2017	129248
2015	119371
2016	117178

Похоже в 2018 году были самые популярные песни (вывод неточный, исходя из того, что нет описания как собирались данные).

Используем **.loc** и **.iloc** для вывода конкретных записей и столбцов датафрейма

```
1 data.iloc[0:5,0:3]
```

artists	name	acousticness	danceability	duration_ms
['Dennis Day']	**Clancy Lowered the Boom**	0.732	0.819	180533
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve**	0.982	0.279	831667
['John McCormack']	**The Wearing of the Green**	0.996	0.518	159507
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve**	0.982	0.279	831667

```
1 data.loc[("['Dennis Day']"), ['duration_ms', 'danceability']]
```

name	duration_ms	danceability
Clancy Lowered the Boom	180533	0.819
How Can You Buy Killarny	196307	0.241
Galway Bay	177067	0.278
St. Patrick's Day Parade	167027	0.661

```
# строки 0, 3, 5 и столбцы 0, 3, последний
data.iloc[[0, 3, 5], [0, 3, -1]]
```

artists	name	acousticness	energy	year
['Dennis Day']	**Clancy Lowered the Boom**	0.732	0.341	1921
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve**	0.982	0.211	1921
['Phil Regan']	**Come Back To Erin**	0.957	0.212	1921

Отрицательные индексы для нумерации с конца.

-1 - последний элемент

```
data.iloc[:,-3:-1].head()
```

artists	name	tempo	valence
['Dennis Day']	**Clancy Lowered the Boom**	60.936	0.9630
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve**	80.954	0.0594
['John McCormack']	**The Wearing of the Green**	66.221	0.4060
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve**	80.954	0.0594
['Phil Regan']	**When Irish Eyes Are Smiling**	101.665	0.2530

Тип объекта определяется с помощью метода type()

Выбирая несколько колонок датафрейма - получаем датафрейм (pandas.core.frame.DataFrame)

```
type(data.iloc[:,-3:-1])
```

```
pandas.core.frame.DataFrame
```

Выбирая одну колонку - получаем серию (pandas.core.series.Series)

```
data.iloc[:,0].head()
```

```
artists                                                 name
['Dennis Day']                                          Clancy Lowered the Boom                                       0.732
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']  Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve    0.982
['John McCormack']                                      The Wearing of the Green                                      0.996
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']  Piano Concerto No. 3 in D Minor, Op. 30: III. Finale. Alla breve    0.982
['Phil Regan']                                          When Irish Eyes Are Smiling                                   0.957
Name: acousticness, dtype: float64
```

```
type(data.iloc[:,0])
```

```
pandas.core.series.Series
```

Создадим серию

```
pd.Series([1,2,3])
```

```
0    1
1    2
2    3
dtype: int64
```

```
pd.Series(['abc', 'def', 'foo', 'bar'])
```

```
0    abc
1    def
2    foo
3    bar
dtype: object
```

Создадим две серии, запишем их в переменные.

```
my_series_1 = pd.Series([1,2,3], index=['Tom', 'Tim', 'Sam'])
my_series_1
```

```
Tom    1
Tim    2
Sam    3
dtype: int64
```

```
my_series_2 = pd.Series([4,5,6], index=['Tom', 'Tim', 'Sam'])
my_series_2
```

```
Tom    4
Tim    5
Sam    6
dtype: int64
```

Соеденим две серии в датафрейм

```
pd.DataFrame({'col_1':my_series_1, 'col_2':my_series_2})
```

	col_1	col_2
Tom	1	4
Tim	2	5
Sam	3	6

Серия или датафрейм

Есть возможность выбрать одну колонку получив не серию, а датафрейм

```
type(data['year'])
```

```
pandas.core.series.Series
```

```
type(data[['year']])
```

```
pandas.core.frame.DataFrame
```

Работа с признаками

- Преобразование колонки release_date к типу datetime.
- Оставляем из даты только год

```
data.release_date = pd.to_datetime(data.release_date, yearfirst=True)
data.release_date = data.release_date.dt.year
```

```
data.head(3)
```

		acousticness	danceability	duration_ms	energy	explicit	instrumentalness	liveness	loudness	mode	popularity	release
artists	**name**											
['Dennis Day']	**Clancy Lowered the Boom**	0.732	0.819	180533	0.341	0	0.000	0.160	-12.441	1	8	
['Sergei Rachmaninoff', 'James Levine', 'Berliner Philharmoniker']	**Piano Concerto No. 3 in D Minor, Op. 30: III**	0.982	0.279	831667	0.211	0	0.878	0.665	-20.096	1	5	

Заключение.

В этой работе познакомились с основыми возможностями пакета Pandas.

Были получены навыки работы с пакетом Pandas, а именно:

- как корректно обращаться к данным
- делать выборки из датафрейма
- группировать данные
- создавать pandas dataframe и series
- ознакомились с тем, что все данные в колонках обладают своим типом (int, float, str и т.д.) и пакет pandas обладает различными функциями, для отображения различных статистик.