Creando un conjunto de datos usando una API con Python

Karan Bhanot 22 de septiembre de 2018 · 5 min de lectura

"Persona que usa una computadora portátil" de rawpixel en Unsplash

Siempre que comenzamos un proyecto de Machine Learning, lo primero que necesitamos es un conjunto de datos. Si bien hay muchos conjuntos de datos que puede encontrar en línea con información variada, a veces desea extraer datos por su cuenta y comenzar su propia investigación. Aquí es cuando una API proporcionada por un sitio web puede acudir al rescate.

> *An application program interface (API) is code that allows two software programs to communicate with each other. The API defines the correct way for a developer to write a program that requests services from an operating system (OS) or other application. — TechTarget*

API is actually a very simple tool that allows anyone to access information from a given website. You might require the use of certain headers but some APIs require just the URL. In this particular article, I'll use the Twitch API provided by Free Code Camp Twitch API Pass-through. This API route does not require any client id to run, thus making it very simple to access Twitch Data. The whole project is available as a Notebook in the Create-dataset-using-API repository.

Import Libraries

As part of accessing the API content and getting the data into a .CSV file, we'll have to import a number of Python Libraries.

1. **requests** library helps us get the content from the API by using the `get()` method. The `json()` method converts the API response to JSON format for easy handling.
2. **json** library is needed so that we can work with the JSON content we get from the API. In this case, we get a dictionary for each Channel's information such as name, id, views and other information.
3. **pandas** library helps to create a dataframe which we can export to a .CSV file in correct format with proper headings and indexing.

Understand the API

We first need to understand what all information can be accessed from the API. For that we use the example of the channel *Free Code Camp* to make the API call and check the information we get.

```
import numpy as np
import pandas as pd
import requests
import json

url = "https://wind-bow.glitch.me/twitch-api/channels/freecodecamp"
JSONContent = requests.get(url).json()
content = json.dumps(JSONContent, indent = 4, sort_keys=True)
print(content)
```

API.py hosted with ♡ by GitHub view raw

This prints the response of the API

To access the API response, we use the function call `requests.get(url).json()` which not only gets the response from the API for the `url` but also gets the JSON format for it. We then dump the data using `dump()` method into `content` so that we can view it in a more presentable view. The output of the code is as follows:

```
{
    "_id": 79776140,
    "_links": {
        "chat": "https://api.twitch.tv/kraken/chat/freecodecamp",
        "commercial": "https://api.twitch.tv/kraken/channels/freecodecamp/commercial",
        "editors": "https://api.twitch.tv/kraken/channels/freecodecamp/editors",
        "features": "https://api.twitch.tv/kraken/channels/freecodecamp/features",
        "follows": "https://api.twitch.tv/kraken/channels/freecodecamp/follows",
        "self": "https://api.twitch.tv/kraken/channels/freecodecamp",
        "stream_key": "https://api.twitch.tv/kraken/channels/freecodecamp/stream_key",
        "subscriptions": "https://api.twitch.tv/kraken/channels/freecodecamp/subscriptions",
```

```
        "teams": "https://api.twitch.tv/kraken/channels/freecodecamp/teams",
        "videos": "https://api.twitch.tv/kraken/channels/freecodecamp/videos"
    },
    "background": null,
    "banner": null,
    "broadcaster_language": "en",
    "created_at": "2015-01-14T03:36:47Z",
    "delay": null,
    "display_name": "FreeCodeCamp",
    "followers": 11770,
    "game": "Creative",
    "language": "en",
    "logo": "https://static-cdn.jtvnw.net/jtv_user_pictures/freecodecamp-profile_image-d9514f2df
    "mature": false,
    "name": "freecodecamp",
    "partner": false,
    "profile_banner": "https://static-cdn.jtvnw.net/jtv_user_pictures/freecodecamp-profile_banne
    "profile_banner_background_color": null,
    "status": "Some GoLang Today #go #golang #youtube",
    "updated_at": "2018-09-19T23:01:33Z",
    "url": "https://www.twitch.tv/freecodecamp",
    "video_banner": "https://static-cdn.jtvnw.net/jtv_user_pictures/freecodecamp-channel_offline_
    "views": 216340
}
```


response.json hosted with ♡ by GitHub view raw

Response for API

If we look closely at the output, we can see that there is a lot of information that we have received. We get the id, links to various other sections, followers, name, language, status, url, views and much more. Now, we can loop through a list of channels, get information for each channel and compile it into a dataset. I will be using a few properties from this list including *_id*, *display_name*, *status*, *followers* and *views*.

Create the dataset

Now that we are aware of what to expect from the API response, let's start with compiling the data together and creating our dataset. For this blog, we'll consider a list of channels that I collected online.

We will first start by defining out list of channels in an array. Then for each channel, we'll use the API to get its information and store each channel's information inside another array `channels_list` using the `append()` method till we get all information collected together in one place. The request response is in JSON format, so to access any key value pair we simply write the key's name within square brackets after the `JSONContent` variable. Now, we use the pandas library to convert this array into a pandas Dataframe using the method `DataFrame()` provided in pandas library. A dataframe is a representation of the data in a tabular form similar to a table, where data is expressed in terms of rows and columns. This dataframe allows fast manipulation of data using various methods.

```
# List of channels we want to access
channels = ["ESL_SC2", "OgamingSC2", "cretetion", "freecodecamp", "storbeck", "habathcx", "Robot
            "ninja", "shroud", "Dakotaz", "esltv_cs", "pokimane", "tsm_bjergsen", "boxbox", "wtc
           "kinggothalion", "amazhs", "jahrein", "thenadeshot", "sivhd", "kingrichard"]

channels_list = []
# For each channel, we access its information through its API
for channel in channels:
    JSONContent = requests.get("https://wind-bow.glitch.me/twitch-api/channels/" + channel).json
    if 'error' not in JSONContent:
        channels_list.append([JSONContent['_id'], JSONContent['display_name'], JSONContent['stat
```

```
                                JSONContent['followers'], JSONContent['views']])

dataset = pd.DataFrame(channels_list)
dataset.sample(5)
```

Pandas Dataframe creation.py hosted with ♡ by GitHub view raw

Create dataframe using Pandas

El `sample()` método pandas muestra filas seleccionadas al azar del marco de datos. En este método, pasamos el número de filas que deseamos mostrar. Aquí, mostremos 5 filas.

	0	1	2	3	4
2	90401618	cretetion	Logging some Miles With My Pack	2812	37083
15	51950404	wtcN	Bıktım başlık yazmaktan | sümüklü | instagram....	834344	35332694
5	6726509	habathcx	Massively Effective	24	2622
21	27686136	SivHD	Siv HD messing around in WoW. the original FAM...	993961	37815131
7	82534701	noobs2ninjas	Doing some work and felt like streaming. Progr...	974	55127

dataset.sample (5)

En una inspección más cercana, vemos que el conjunto de datos tiene dos problemas menores. Abordémoslos uno por uno.

1. **Encabezados:** Actualmente, los encabezados son números y no reflejan los datos que representa cada columna. Puede parecer menos importante con este conjunto de datos porque solo tiene unas pocas columnas. Sin embargo, cuando explore conjuntos de datos con cientos de columnas, este paso será realmente importante. Aquí, definimos las columnas usando el `columns()` método proporcionado por pandas. En este caso, definimos explícitamente los encabezados, pero en ciertos casos, puede elegir las claves como encabezados directamente.
2. **Valores ninguno / nulo / en blanco:** algunas de las filas tendrán valores perdidos. En tales casos, tendremos dos opciones. Podemos eliminar la fila completa donde cualquier valor está en blanco o podemos ingresar algún valor cuidadosamente seleccionado en los espacios en blanco. Aquí, la `status` columna tendrá `None` en algunos casos. Eliminaremos estas filas utilizando el método `dropna(axis = 0, how = 'any', inplace = True)` que elimina filas con valores en blanco en el propio conjunto de datos. Luego, cambiamos el índice de los números de 0 a la longitud del conjunto de datos usando el método `RangeIndex(len(dataset.index))`.

```
conjunto de datos . columnas  = [ 'Id' , 'Nombre' , 'Estado' , 'Seguidores' , 'Vistas' ]
conjunto de datos . dropna ( axis  =  0 , how  =  'any' , inplace  =  True )
conjunto de datos . índice  =  pd . RangeIndex ( len ( conjunto de datos . Index ))
```

Headings-index.py alojado con ♡ por GitHub ver crudo

Agregar encabezados de columna y actualizar índice

Exportar conjunto de datos

Nuestro conjunto de datos ya está listo y se puede exportar a un archivo externo. Usamos el `to_csv()` método. Definimos dos parámetros. El primer parámetro se refiere al nombre del archivo. El segundo parámetro es un

booleano que representa si la primera columna del archivo exportado tendrá el índice o no. Ahora tenemos un archivo .CSV con el conjunto de datos que creamos.

	Id	Name	Status	Followers	Views
0	30220059	ESL_SC2	RERUN: ShoWTimE vs. GuMiho [PvT] - Group D - I...	233265	75380804
1	71852806	OgamingSC2	Olimoleague Weekly #134	79265	38973783
2	90401618	cretetion	Logging some Miles With My Pack	2812	37083
3	79776140	FreeCodeCamp	Some GoLang Today #go #golang #youtube	11771	216347
4	6726509	habathcx	Massively Effective	24	2622
5	54925078	RobotCaleb	Sabering	35	6684
6	82534701	noobs2ninjas	Doing some work and felt like streaming. Progr...	974	55127
7	19571641	Ninja	Duos with Travis Scott! See how you could squ...	11343822	320594181
8	37402112	shroud	gaminxS @Shroud for updates!	4352191	206944462
9	39298218	dakotaz	TSM Dakotaz - instagram/twitter: @dakotaz	2625586	59728959
10	31239503	ESL_CSGO	RERUN: IEM Shanghai 2018	2475125	296624596
11	44445592	pokimane	See you guys soon! Follow my twitter & insta f...	2170913	43709758
12	38421618	TSM_Bjergsen	Monday stream as per usualllllll!	1440737	91181335
13	38881685	boxbox	Mr. Challenger Climb I don't FEel so good	1389852	83498885
14	51950404	wtcN	Bıktım başlık yazmaktan | sümüklü | instagram....	834344	35332694
15	19070311	A_Seagull	cozy	864185	24793514
16	43830727	KingGothalion	Raid. Day. Today.	875741	34100062
17	43356746	AmazHS	✧AMAZ✧ It's me! =D | !MTGA #sponsored , Ravnic...	894551	93073898
18	6768122	Jahrein	60 Parsecs uzay versiyonu | instagram.com/jahr...	935101	36168882
19	21130533	Nadeshot	The Return to Fortnite | 100 Thieves	948418	42710352
20	27686136	SivHD	Siv HD messing around in WoW. the original FAM...	993961	37815131
21	66691674	KingRichard	Fall Skirmish Practice #bushbandits @KingRicha...	1075328	17037876

Dataset.csv

En este artículo, discutimos un enfoque para crear nuestro propio conjunto de datos utilizando la API de Twitch y Python. Puede seguir un enfoque similar para acceder a la información a través de cualquier otra API. Pruébelo usando la API de GitHub .

. . .

Siéntase libre de compartir sus pensamientos y contácteme con cualquier pregunta que pueda tener.

Ciencia de los datos | Pitón | Aprendizaje automático | Raspado web | API