

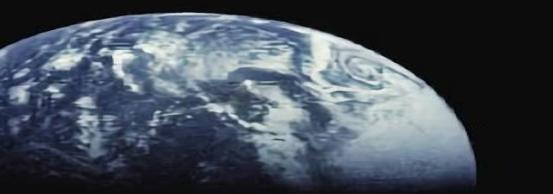


Elasticsearch Essentials: Data Loading with Python for Interplanetary Insights

Code and slides

[A black and white QR code is centered on a white rectangular background. The QR code encodes the URL of the GitHub repository mentioned in the text above.](https://github.com/JessicaGarson/Elasticsearch-Essentials-Data>Loading-with-Python-for-Interplanetary-Insights</p></div><div data-bbox=)

**This talk will walk you through how to use the
Python client as a toolkit** 



**The dataset I used was NASA's
Near Earth Object Web Service
(NeoWs), a RESTful web service
that provides near-earth
Asteroid information.**

What is an index?

An index inside Elasticsearch is a data structure where you can store your data in documents.

What you need to get started

- Python > 3.8
- A NASA API key
- Elasticsearch > 8.x

<https://wiki.python.org/moin/BeginnersGuide/Download>

<https://api.nasa.gov/>

<https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html>





```
pip install requests pandas elasticsearch notebook
```

Updating our data once



```
import requests
from getpass import getpass
import pandas as pd
from datetime import datetime, timedelta
from elasticsearch import Elasticsearch, helpers
```



```
def connect_to_nasa():
    url = "https://api.nasa.gov/neo/rest/v1/feed"
    nasa_api_key = getpass("NASA API Key: ")
    today = datetime.now()
    params = {
        "api_key": nasa_api_key,
        "start_date": today - timedelta(days=7),
        "end_date": datetime.now(),
    }
    return requests.get(url, params).json()
```



```
def connect_to_nasa():
    url = "https://api.nasa.gov/neo/rest/v1/feed"
    nasa_api_key = getpass("NASA API Key: ")
    today = datetime.now()
    params = {
        "api_key": nasa_api_key,
        "start_date": today - timedelta(days=7),
        "end_date": datetime.now(),
    }
    return requests.get(url, params).json()
```



```
response = connect_to_nasa()
```



```
df = create_df(response)  
df.head()
```

	id	neo_reference_id	name	nasa_jpl_url	absolute_magnitude_h
0	2434786	2434786	434786 (2006 PW)	https://ssd.jpl.nasa.gov/tools/sbdb_lookup.htm... e display	18.16
1	3756789	3756789	(2016 PO)	https://ssd.jpl.nasa.gov/tools/sbdb_lookup.htm... 25.10	
2	3781330	3781330	(2017 QX35)	https://ssd.jpl.nasa.gov/tools/sbdb_lookup.htm... 22.24	
3	54203000	54203000	(2018 UY37)	https://ssd.jpl.nasa.gov/tools/sbdb_lookup.htm... 22.53	
4	54317182	54317182	(2022 UG2)	https://ssd.jpl.nasa.gov/tools/sbdb_lookup.htm... 27.11	



```
df = df.drop(['links.self', 'sentry_data'], axis=1)
```



```
df.isnull().values.any()
```



```
def connect_to_elastic():
    elasticsearch_endpoint = getpass("Host Address: ")
    elasticsearch_api_key = getpass("Elastic API Key: ")
    return Elasticsearch(hosts=elasticsearch_endpoint, api_key=elasticsearch_api_key)

es = connect_to_elastic()
```





```
def connect_to_elastic():
    elasticsearch_endpoint = getpass("Host Address: ")
    elasticsearch_api_key = getpass("Elastic API Key: ")
    return Elasticsearch(hosts=elasticsearch_endpoint, api_key=elasticsearch_api_key)

es = connect_to_elastic()
```





```
index_name = "all_things_open"  
es.indices.create(index=index_name)
```



```
def doc_generator(df, index_name):
    for index, document in df.iterrows():
        yield {
            "_index": index_name,
            "_id": f"{document['id']}",
            "_source": document.to_dict(),
        }
```



```
def doc_generator(df, index_name):
    for index, document in df.iterrows():
        yield {
            "_index": index_name,
            "_id": f"{document['id']}",
            "_source": document.to_dict(),
        }
```



```
helpers.bulk(es, doc_generator(df, index_name))
```

Parallel Bulk

`parallel_bulk()` is a tool that enhances the `bulk()` API by using multiple threads to process tasks simultaneously.



```
for success, info in helpers.parallel_bulk(es, doc_generator(df, 'all_things_open'), thread_count=4):
    if not success:
        print('A document failed:', info)
```



I had a problem

When I first started working with Elasticsearch my data in my index would quickly go out of date.



Oh wait, I know how
to solve this!

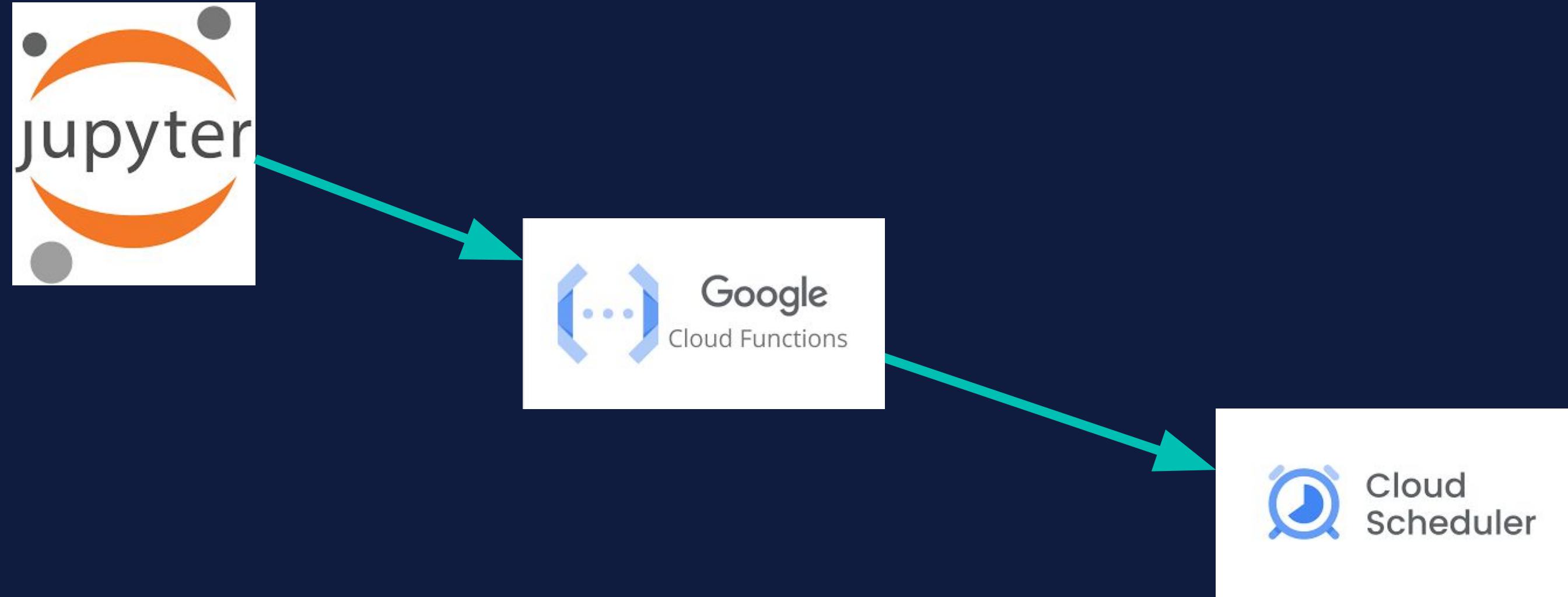
You can do this a lot of different ways

- Python scheduling libraries like APScheduler or schedule
- GCP Cloud Functions and Cloud Scheduler
- AWS Lambda
- Azure Function Apps
- Railway cron jobs
- Render cron jobs
- GitHub Actions



Using Google Cloud Platform

Solution overview



<https://cloud.google.com/scheduler>

<https://cloud.google.com/functions>

<https://www.elastic.co/search-labs/blog/keeping-your-elasticsearch-index-current-with-python-and-google-cloud-platform-functions>



Step 1

Filter functions									
	Environment	Name ↑	Last deployed	Region	Recommendation	Trigger	Runtime	Memory allocated	Executed function
<input type="checkbox"/>	Cloud Run function (1st gen)	function-1	May 12, 2022, 2:07:38 PM	us-central1		Topic: Cats	Python 3.9	256 MB	hello_pubsub
<input type="checkbox"/>	Cloud Run function (1st gen)	function-2	Feb 14, 2024, 5:26:24 PM	us-central1		Topic: Celtics	Python 3.9	256 MB	hello_pubsub
<input type="checkbox"/>	Cloud Run function	function-3	Feb 15, 2024, 5:24:10 PM	northamerica-northeast1		HTTP	Python 3.12	256 MB	hello_pubsub
<input type="checkbox"/>	Cloud Run function	nasa	Jul 26, 2024, 8:39:17 PM	northamerica-northeast1		Topic: asteroids	Python 3.12	256 MiB	hello_pubsub

Step 2

1 Configuration — 2 Code

✓ nasa Cloud Run function
northamerica-northeast1

Trigger

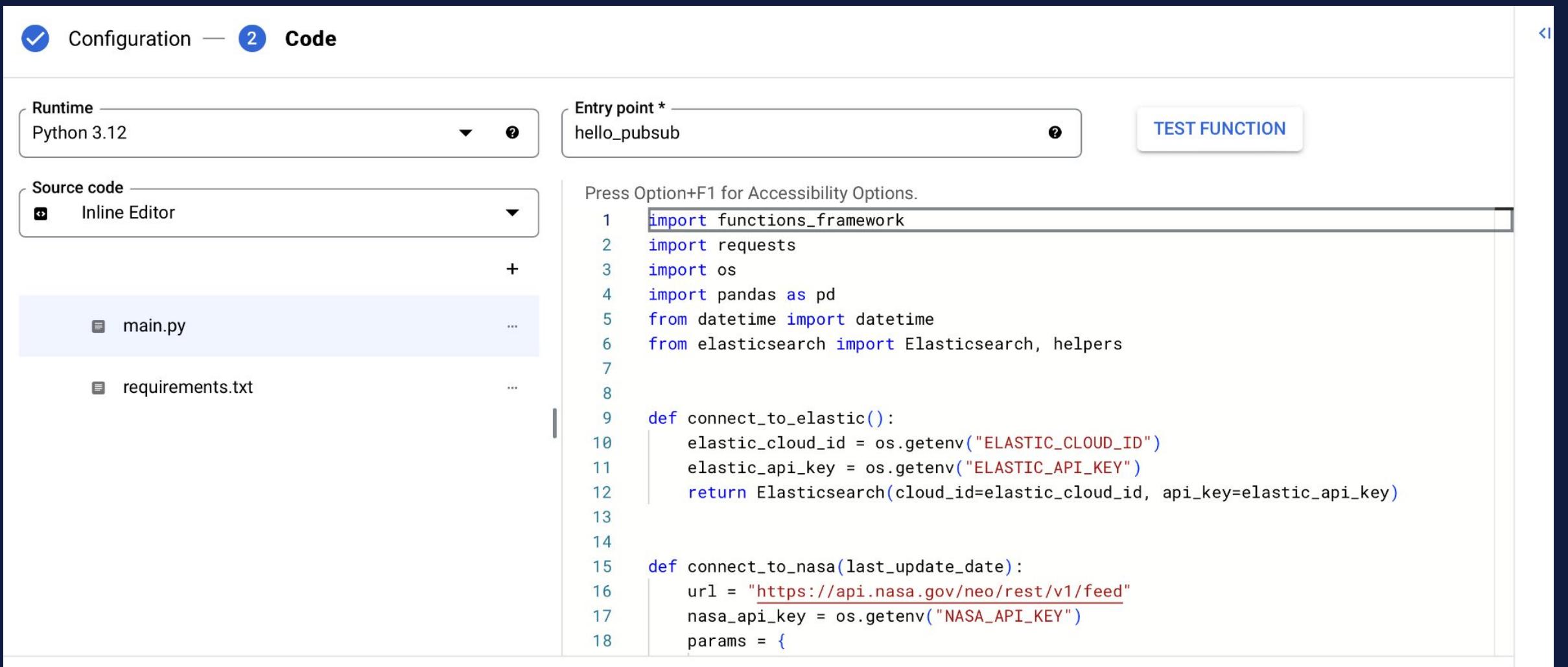
Trigger type
Cloud Pub/Sub

Cloud Pub/Sub topic *
projects/ncav-293119/topics/asteroids

Retry on failure [?](#)

[MORE OPTIONS](#)

Step 3



The screenshot shows the AWS Lambda function configuration interface. The top navigation bar has a checkmark icon next to 'Configuration' and a '2' icon next to 'Code'. The 'Code' tab is selected. On the left, there are dropdown menus for 'Runtime' (Python 3.12) and 'Entry point' (hello_pubsub). Below these is a 'Source code' section with an 'Inline Editor' dropdown and a '+' button. Underneath are two files listed: 'main.py' and 'requirements.txt'. The right side shows the actual Python code for 'main.py' with syntax highlighting for imports and functions. A note at the top says 'Press Option+F1 for Accessibility Options.'

```
1 import functions_framework
2 import requests
3 import os
4 import pandas as pd
5 from datetime import datetime
6 from elasticsearch import Elasticsearch, helpers
7
8
9 def connect_to_elastic():
10     elastic_cloud_id = os.getenv("ELASTIC_CLOUD_ID")
11     elastic_api_key = os.getenv("ELASTIC_API_KEY")
12     return Elasticsearch(cloud_id=elastic_cloud_id, api_key=elastic_api_key)
13
14
15 def connect_to_nasa(last_update_date):
16     url = "https://api.nasa.gov/neo/rest/v1/feed"
17     nasa_api_key = os.getenv("NASA_API_KEY")
18     params = {
```

Step 4

The screenshot shows the AWS Lambda Function Configuration interface. The top navigation bar has a checkmark icon and the text "Configuration — 2 Code". The "Code" tab is selected. On the left, there are dropdown menus for "Runtime" (Python 3.12) and "Source code" (Inline Editor). Below these are two files listed: "main.py" and "requirements.txt". A plus sign (+) button is available to add more files. On the right, the "Entry point" is set to "hello_pubsub". Below it is a "TEST FUNCTION" button. A note says "Press Option+F1 for Accessibility Options." followed by a list of requirements:

```
1 functions-framework==3.*  
2 requests==2.31.0  
3.elasticsearch==8.12.0  
4 pandas==2.1.4
```

Step 5

- Define the schedule

Region

northamerica-northeast1

Description

Frequency *

0 8 * * *



Schedules are specified using unix-cron format. E.g. every minute: "* * * * *", every 3 hours: "0 */3 * * *", every Monday at 9:00: "0 9 * * 1". [Learn more](#)



- Minute and Hour:

- At 8:00 AM

Timezone *

Eastern Daylight Time (EDT)



- Jobs in set in timezones affected by Daylight Saving Time can run outside of cadence during DST change. Using a UTC timezone can avoid the problem. [Learn more](#)

CONTINUE

Step 6

SCHEDULER JOBS		APP ENGINE CRON JOBS							
<input type="checkbox"/>	Name 	Status of last execution	Region	State	Description	Frequency	Target	Last run	Next run
<input type="checkbox"/>	Cats	 Success	us-central1	Enabled	Daily Cat Facts	0 */12 *** (America/Detroit)	Topic : projects/ncav-293119/topics/Cats	Oct 27, 2024, 12:00:00 PM	Oct 28, 2024, 12:00:00 AM
<input type="checkbox"/>	Celtics_UPdate	 Success	northamerica-northeast1	Enabled	Celtics_Sched	0 8 *** (America/New_York)	Topic : projects/ncav-293119/topics/Celtics	Oct 27, 2024, 8:00:00 AM	Oct 28, 2024, 8:00:00 AM
<input type="checkbox"/>	nasa	 Success	northamerica-northeast1	Enabled		0 8 *** (America/New_York)	Topic : projects/ncav-293119/topics/asteroids	Oct 27, 2024, 8:00:00 AM	Oct 28, 2024, 8:00:00 AM

Using an ingest pipeline for Elasticsearch can be a natural next step for optimizing data uploads into an index, mainly if you deal with large volumes or complex data transformations.

Depending on the size and frequency of data updates, consider batching data to reduce the number of API calls and to enhance performance.

Let me know if this talk inspires you to build anything. I'm [@JessicaGarson](#) on most platforms.

My email is jess.garson@elastic.co

Code and slides

[A black and white QR code is centered on a white rectangular background. The QR code encodes the URL of the GitHub repository mentioned in the text above.](https://github.com/JessicaGarson/Elasticsearch-Essentials-Data>Loading-with-Python-for-Interplanetary-Insights</p></div><div data-bbox=)

RESOURCES FOR DEVELOPERS | BY DEVELOPERS LIKE YOU!

Search Labs



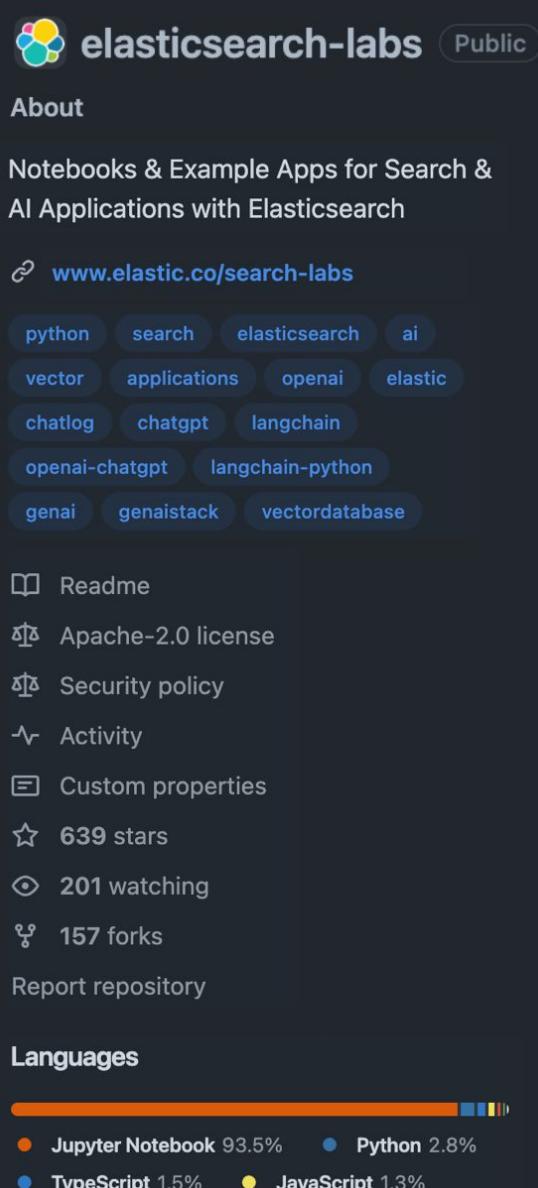
ML Research December 1, 2023

RAG evaluation metrics: A journey through metrics

Explore RAG evaluation metrics like BLEU score, ROUGE score, PPL, BARTScore, and more. Discover...

By: Quentin Herreros, Thomas Veasey and Thanos Papaoikonomou

elastic.co/search-labs



About
Notebooks & Example Apps for Search & AI Applications with Elasticsearch
www.elastic.co/search-labs

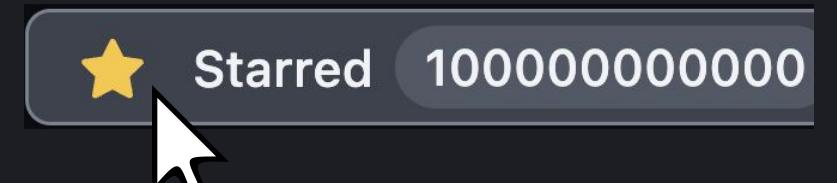
python search elasticsearch ai
vector applications openai elastic
chatlog chatgpt langchain
openai-chatgpt langchain-python
genai gennai vectordatabase

Readme
Apache-2.0 license
Security policy
Activity
Custom properties
639 stars
201 watching
157 forks
Report repository

Languages

Jupyter Notebook	93.5%	Python	2.8%
TypeScript	1.5%	JavaScript	1.3%

github.com/elastic/elasticsearch-labs



Tutorials
GitHub Example Code
Integrations
Blogs



Thank you!