LIVE ONLINE TRAINING

Data Engineering for Data Scientists

Build resilient pipelines to support stronger models

November 12, 2020

10:00am – 12:00pm EST

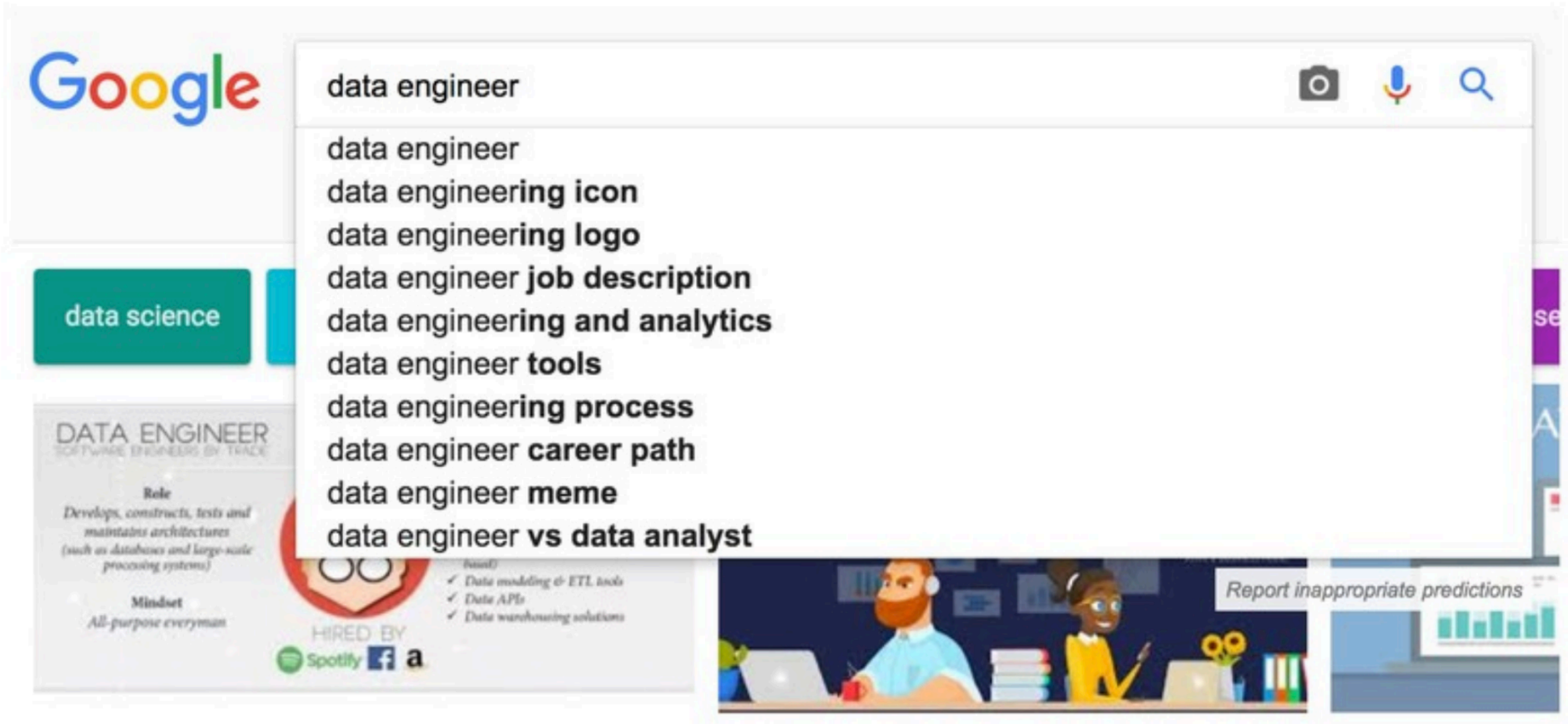
Agenda

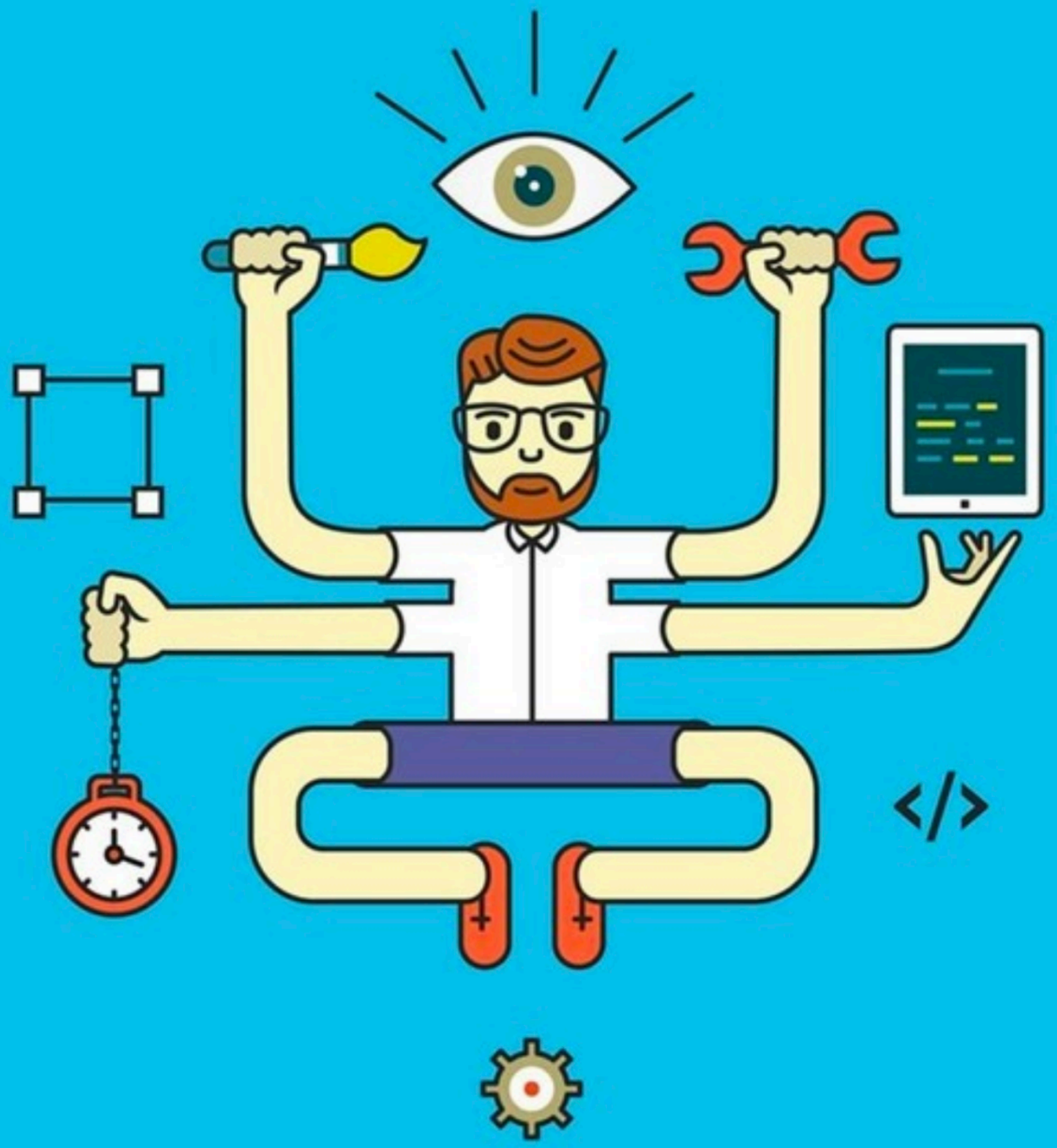
1 Extend

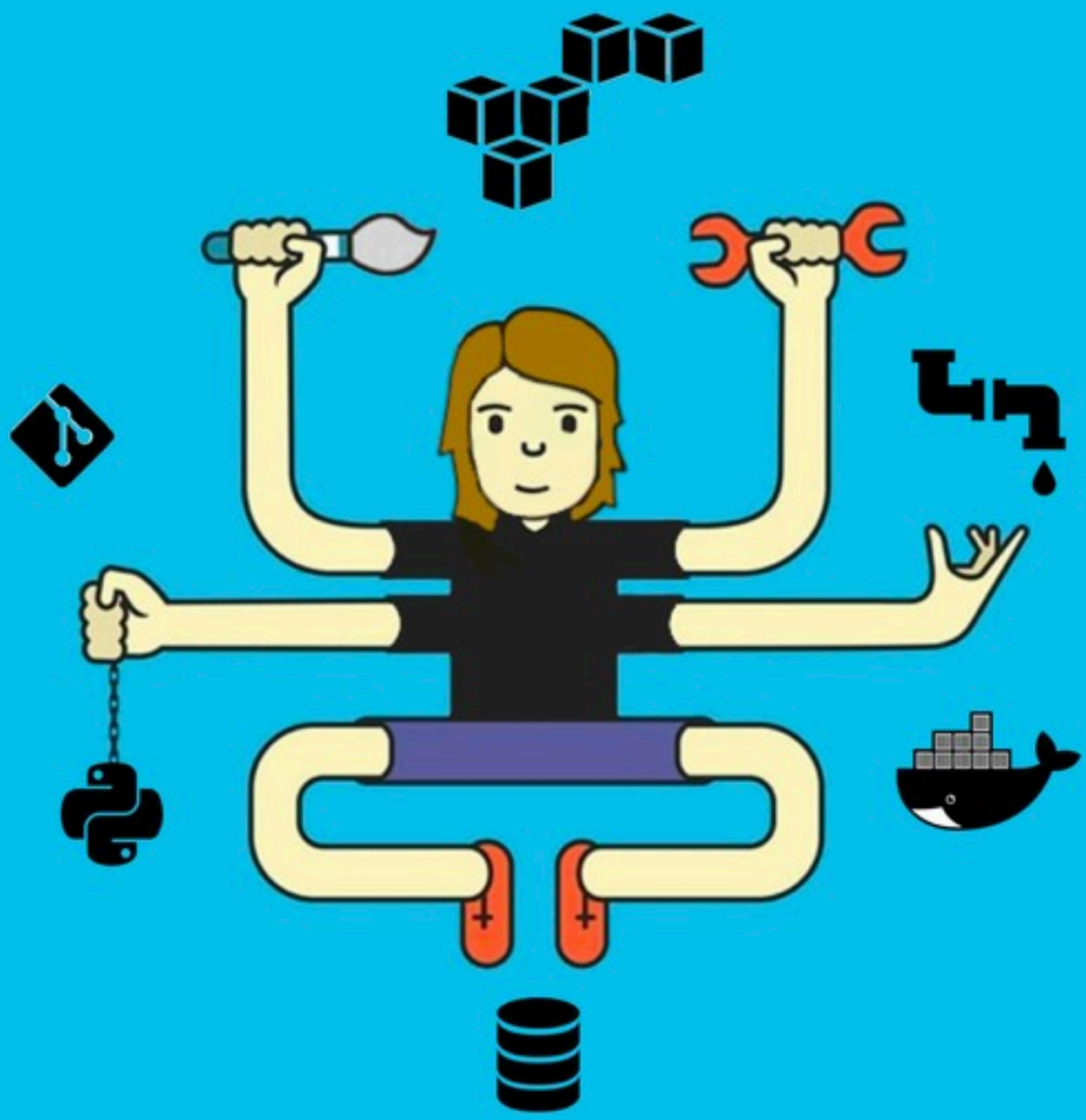
2 Save

3 Schedule

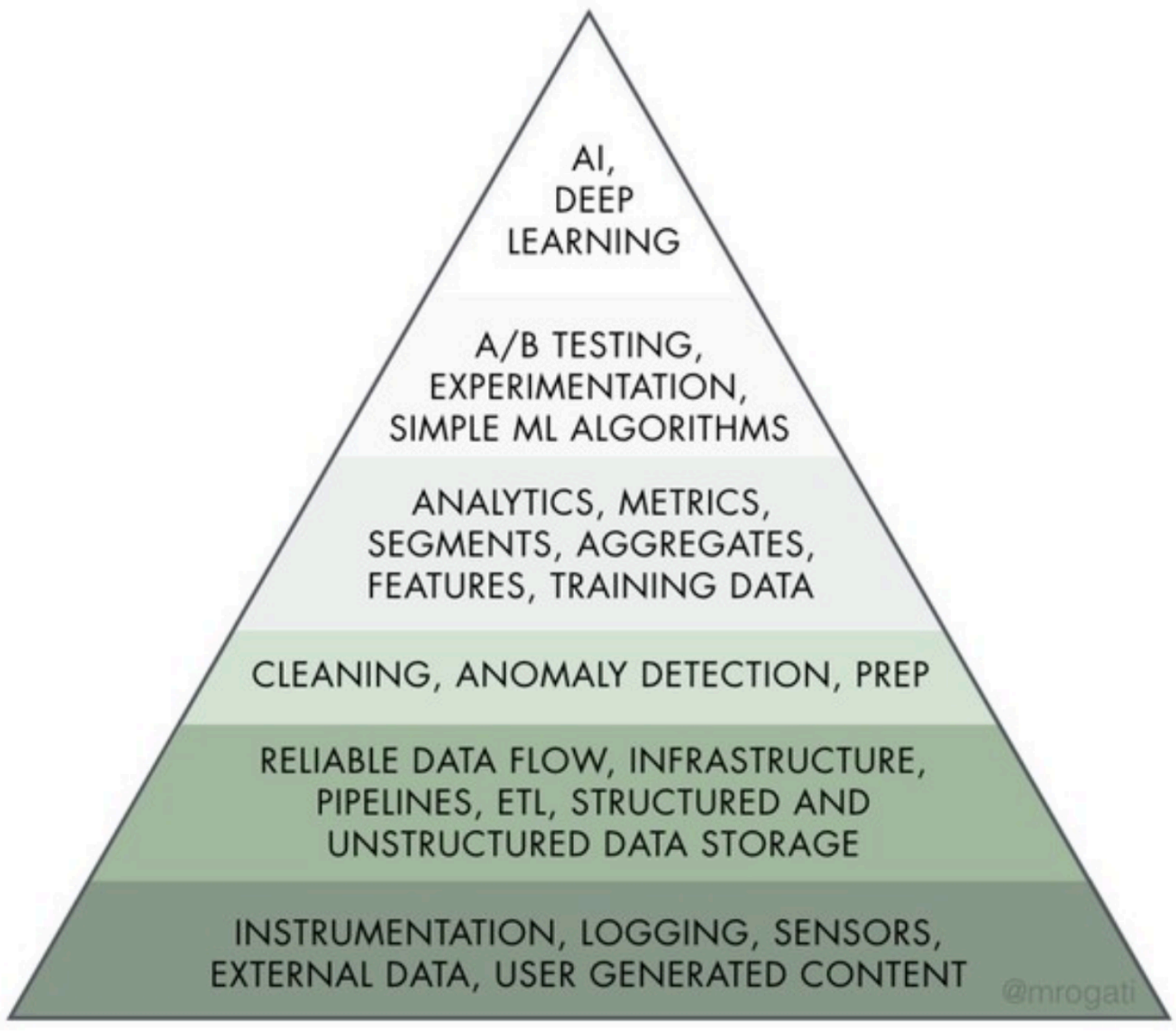
Data Engineering



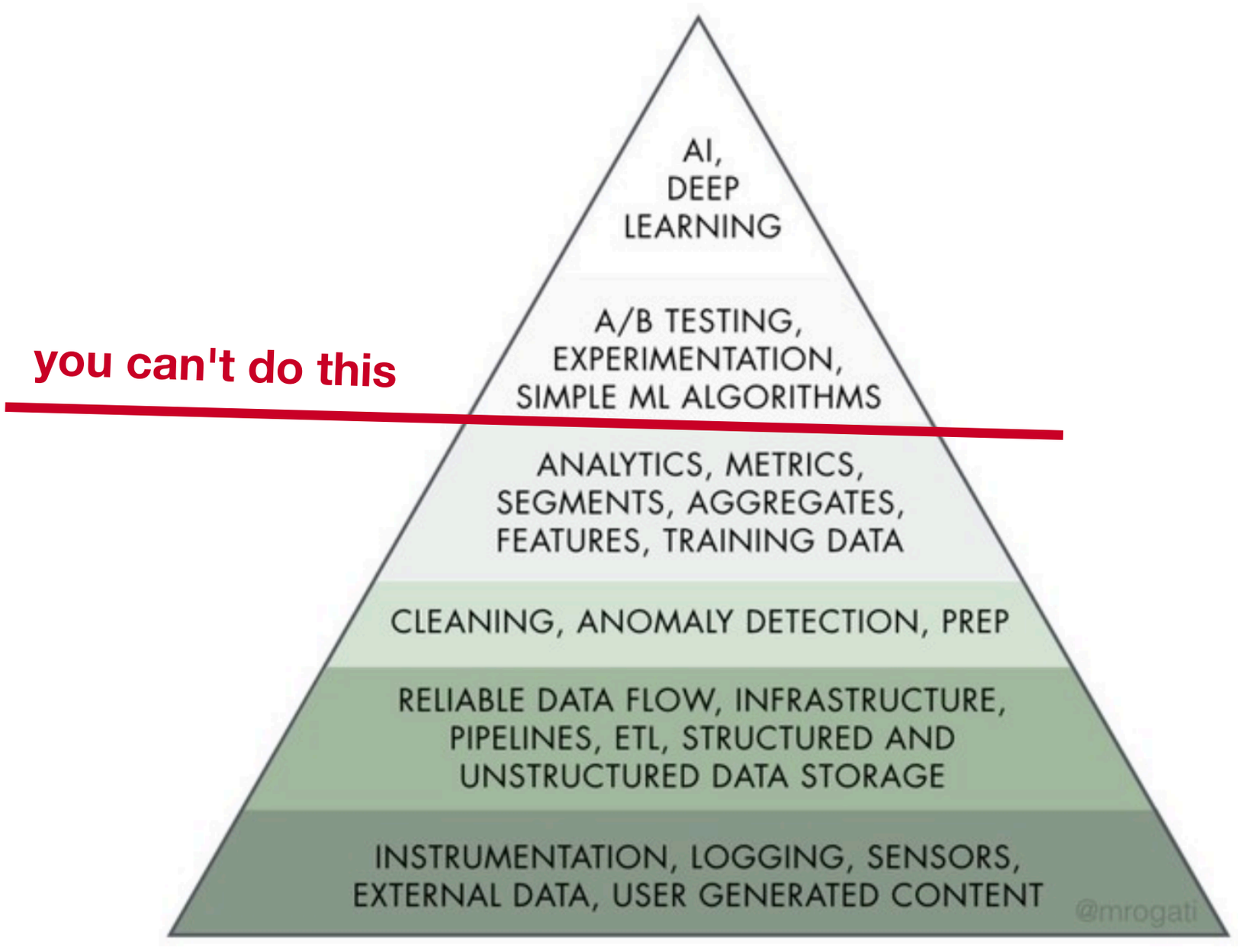




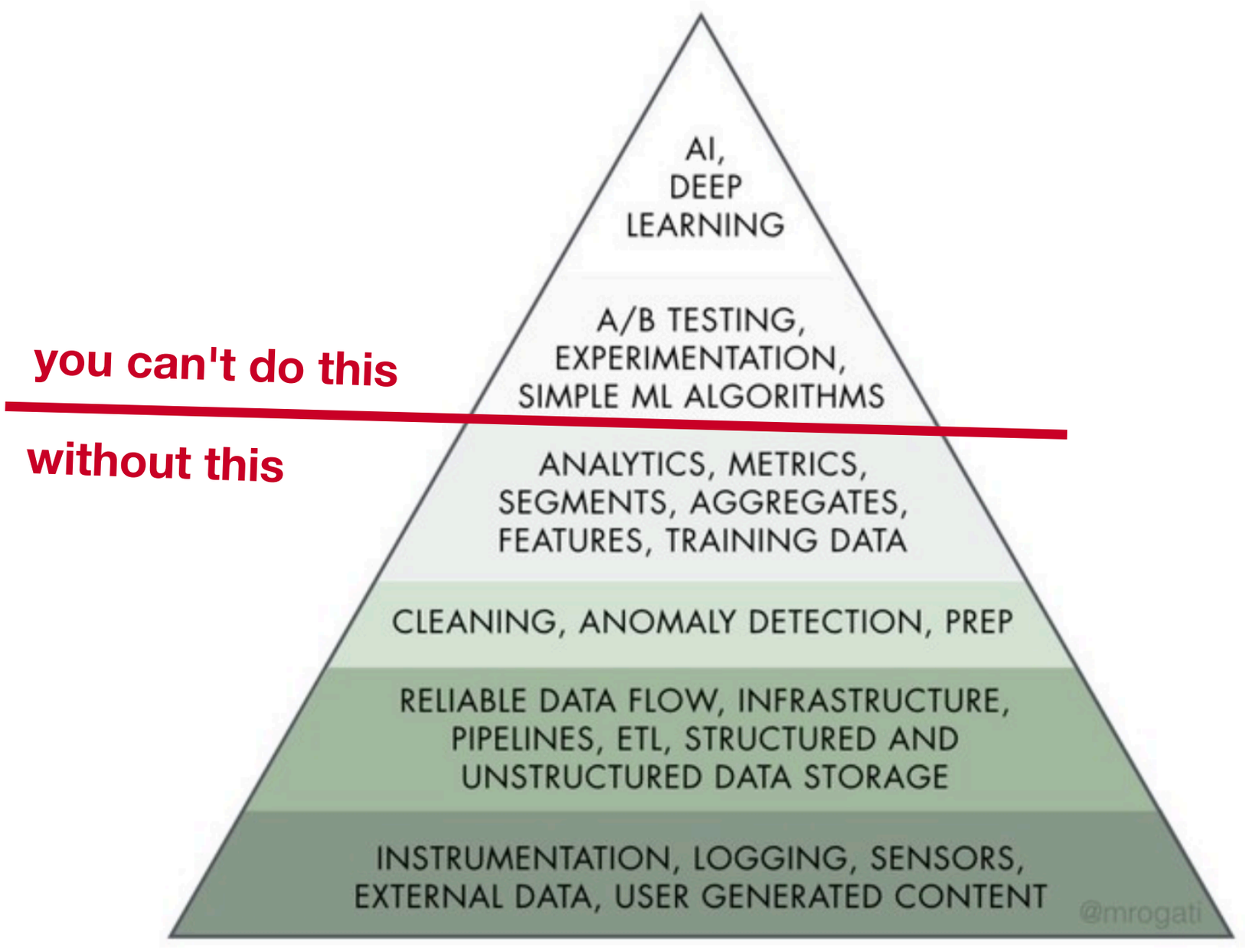
Data Hierarchy of Needs



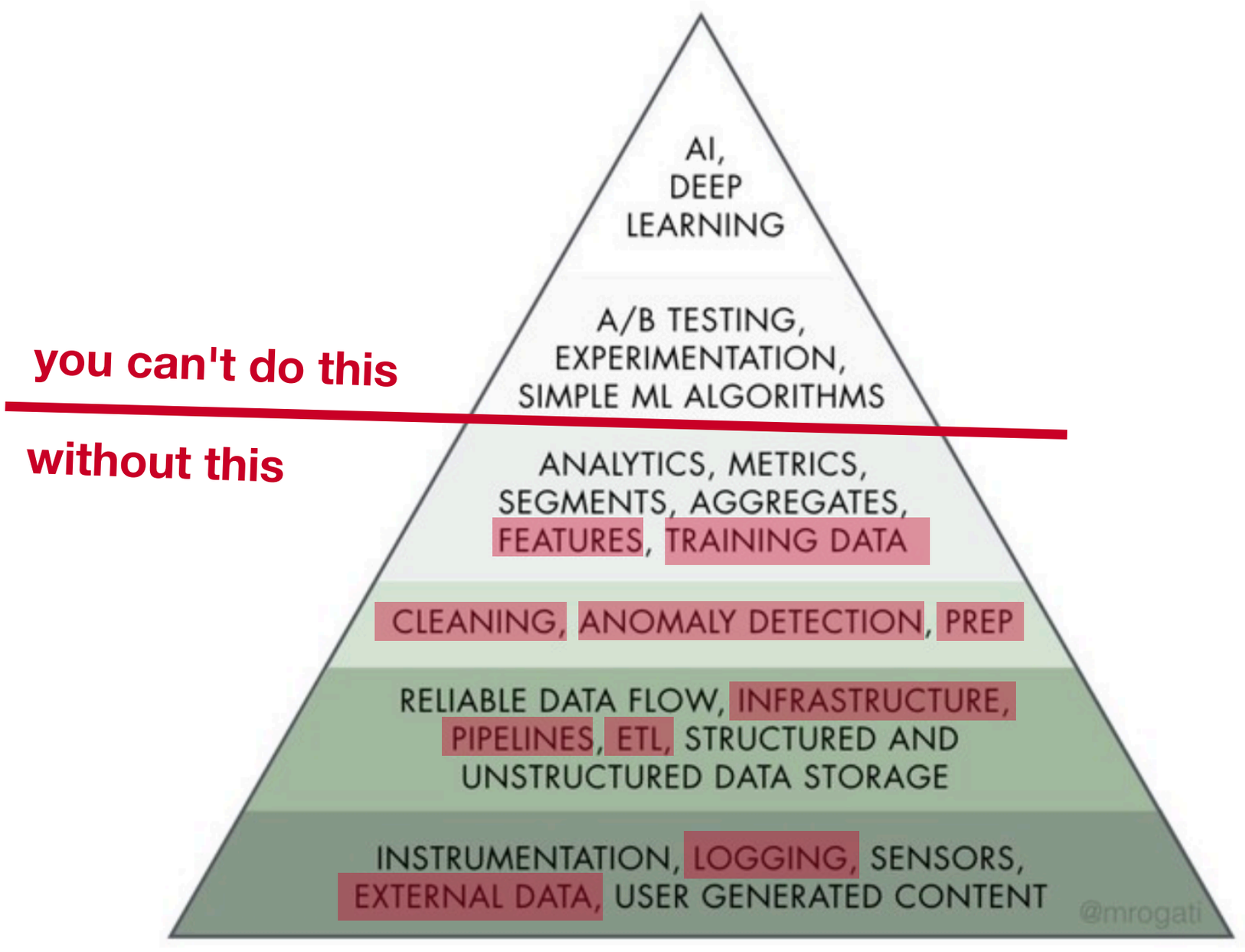
Data Hierarchy of Needs



Data Hierarchy of Needs



Data Hierarchy of Needs




ELMER'S
Washable, No Run
School Glue
#1 Teacher Brand
Safe, Non-Toxic
AP
7.625 FL OZ (225 mL)


FANTASY
FOOTBALL

GDP
2-6-0 (14), Streak: **L3**
Waiver: 15

Team A
62.94

VS

Team B
110.30

Max
7-1-0 (2), Streak: **W2**
Waiver: 11

Hide Chart

62.94

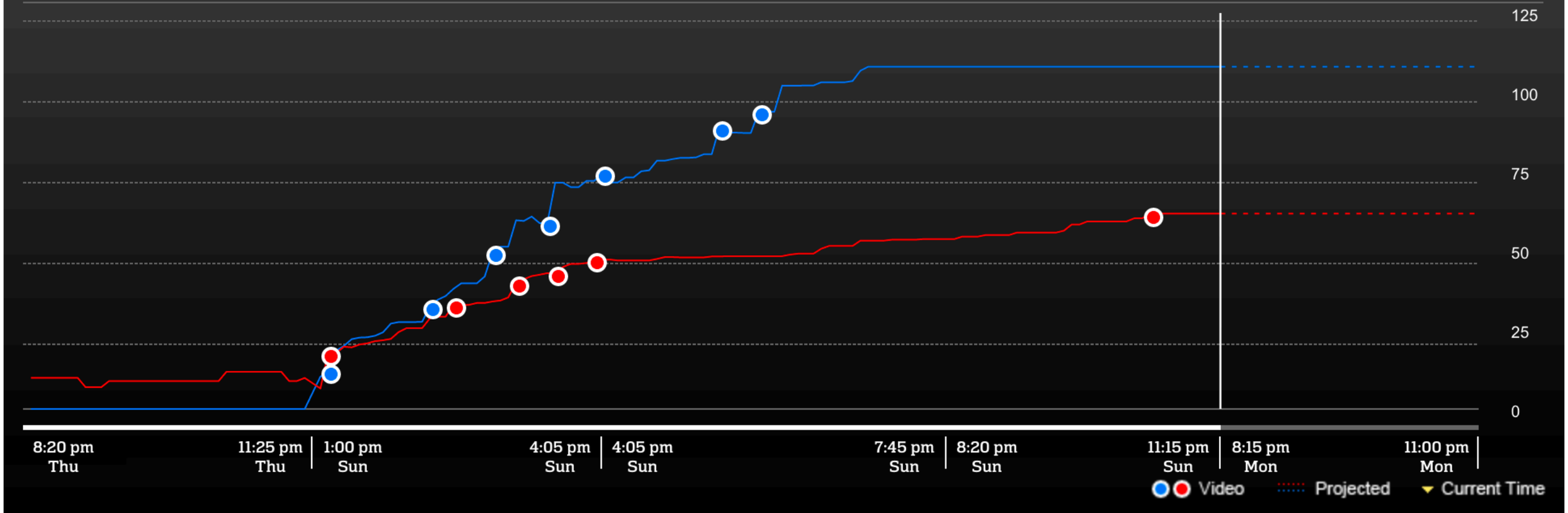
QB	RB-1	RB-2	WR-1	WR-2	R/W/ T-1	R/W/ T-2	R/W/ T-3	K	DEF
24.24	2.70	2.20	0.80	-0.10	3.10	6.90	5.10	12.00	6.00

Hide Chart

110.30

QB	RB-1	RB-2	WR-1	WR-2	R/W/ T-1	R/W/ T-2	R/W/ T-3	K	DEF
14.00	37.20	0.00	16.30	6.30	2.10	3.90	9.50	7.00	14.00

LIVE CHART



Min Remaining: 0 of 600 | Yet To Play: | In Play: | On Field: | Final: | Min Remaining: 0 of 600

1

Extend

01-model.ipynb

Q&A

Jupyter to Hydrogen

Jupyter to Hydrogen

.ipynb

- ✅ exploratory analysis
- ✅ visualizing ideas
- ✅ prototyping

- ❌ messy
- ❌ bad at versioning
- ❌ not ideal for production

Jupyter to Hydrogen

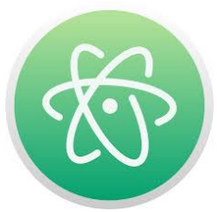
.ipynb

.py

Hydrogen

Hydrogen is an interactive coding environment that supports Python, R, JavaScript and other Jupyter kernels.

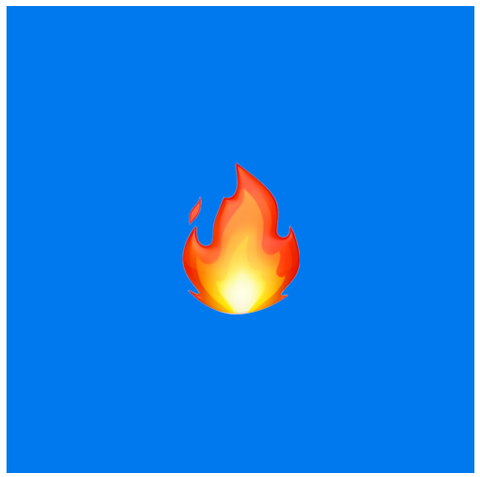
Checkout our Documentation and Medium blog post to see what you can do with Hydrogen.



https://atom.io/packages/hydrogen

02-model.py

Q&A




input
output

Python Fire 

Python Fire is a library for automatically generating command line interfaces (CLIs) from absolutely any Python object.

- Python Fire is a simple way to create a CLI in Python. [1]
- Python Fire is a helpful tool for developing and debugging Python code. [2]
- Python Fire helps with exploring existing code or turning other people's code into a CLI. [3]
- Python Fire makes transitioning between Bash and Python easier. [4]
- Python Fire makes using a Python REPL easier by setting up the REPL with the modules and variables you'll need already imported and created. [5]

Installation

To install Python Fire with pip, run: `pip install fire`

To install Python Fire with conda, run: `conda install fire -c conda-forge`

To install Python Fire from source, first clone the repository and then run: `python setup.py install`

03-predict.py

04-fire.py

```
🔥 python 04-fire.py “Aaron Rodgers”
```

Q&A

2

Save

05-database.py

```
🔥 python 05-database.py “Aaron Rodgers”
```

Monitor


Rollbar


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

Waiting for data from your Python app...

This page will update once we receive an error from your app.

Don't want to start with Python? Choose a different SDK.

Installation

To send errors to Rollbar from your Python application you should use the pyrollbar SDK. Install pyrollbar with pip:

```
pip install rollbar
```

COPY

Send a Message

You'll need your project's server-side access token to initialize the pyrollbar SDK. Sending a message to the Rollbar server is as simple as:

```
import rollbar

rollbar.init('49e86c1178e346899c2f29e3744420a9')
rollbar.report_message('Rollbar is configured correctly')
```

COPY

A few seconds after you execute this code, the message should appear on your project's "Items" page.

remove password

Search

- Repositories 244
- Code 50M+
- **Commits 508K**
- Issues 280K
- Packages 1
- Marketplace 0
- Topics 0
- Wikis 34K
- Users 0

Advanced search Cheat sheet

Showing 445,215 available commit results

Sort: **Recently committed**

removed password ···

Verified c64833c

reharr4 committed to reharr4/blamazon 2 hours ago

Onmibus updates (#7) ···

Verified b1b80c9

stevector committed to stevector/internet-button-playground 4 hours ago ✔

Merge pull request #534 in EDD/edd-django from ~WCMORRELL/edd:feature... ···

8aa8978

wmorrell committed to JBEI/edd 7 hours ago

File restructuring ···

8735106

caitlin authored and **PilotBob** committed to CassandraWerewolf/CassandraWeb 7 hours ago

removed password

removed password from bamazonCustomer.js

master

reharr4 committed 2 hours ago Verified 1 parent 7e38084 commit c64833c915cc0f604cb0eeab

Showing 1 changed file with **2 additions** and **2 deletions**.

4 bamazonCustomer.js

```
@@ -9,7 +9,7 @@ var connection = mysql.createConnection({
9  9          // username
10 10         user: "root",
11 11         // password
12    -       password: "L!sb0n13",
   12 +       password: "",
13 13         database: "bamazon_db"
14 14     });
15 15
@@ -94,4 +94,4 @@ connection.connect(function (err) {
94 94     //         })
95 95     //     })
96 96     // }
97    - connection.end();
   97 + connection.end();
```


PSSST...!


Secrets

python-dotenv |

Reads the key,value pair from `.env` file and adds them to environment variable. It is great for managing app settings during development and in production using 12-factor principles.

> Do one thing, do it well!

Installation

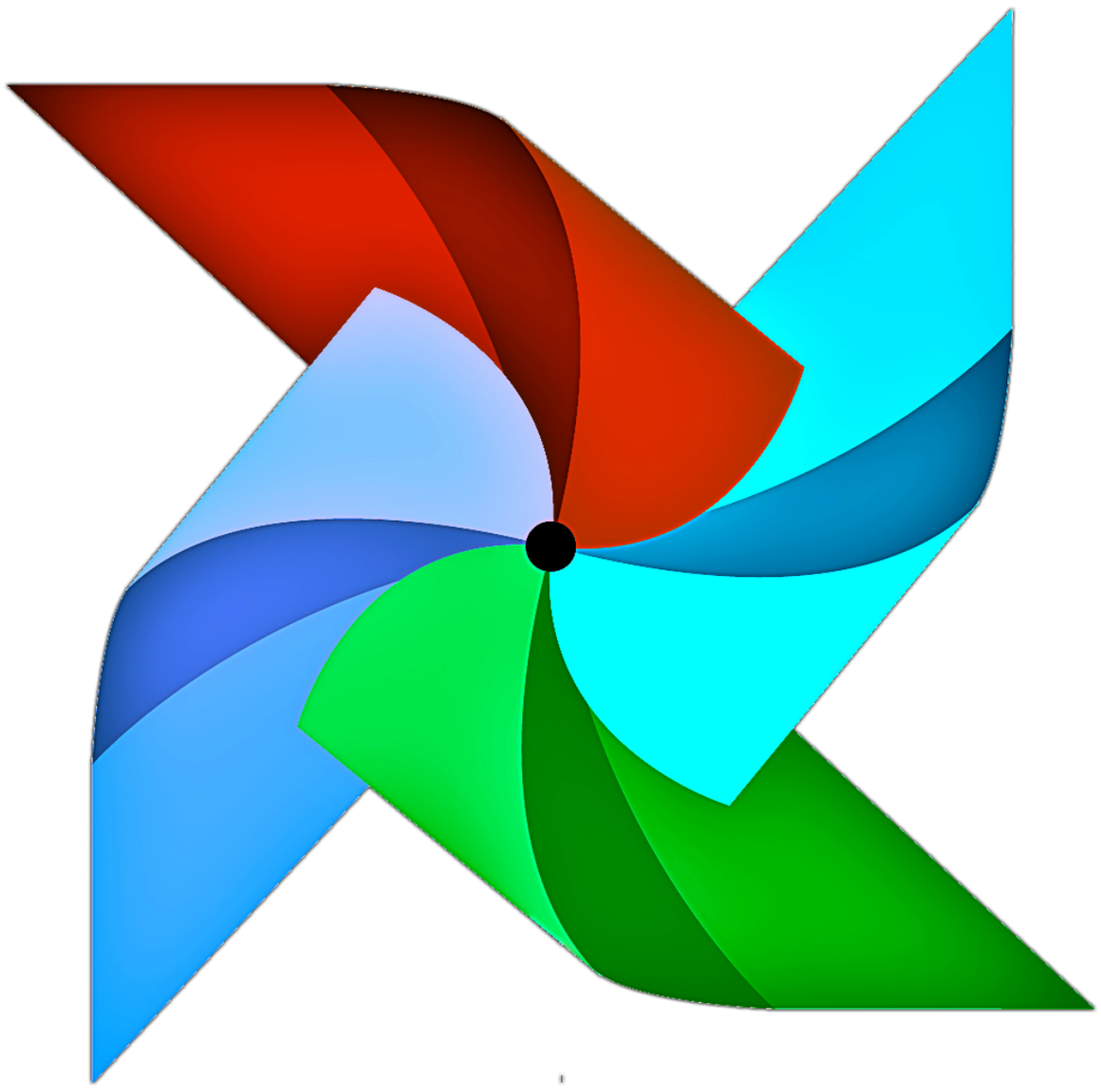
```
pip install -U python-dotenv
```

06-rollbar.py

Q&A

3

Schedule



Airflow

Apache Airflow

Apache Airflow (or simply Airflow) is a platform to programmatically author, schedule, and monitor workflows.

When workflows are defined as code, they become more maintainable, versionable, testable, and collaborative.

Use Airflow to author workflows as directed acyclic graphs (DAGs) of tasks. The Airflow scheduler executes your tasks on an array of workers while following the specified dependencies. Rich command line utilities make performing complex surgeries on DAGs a snap. The rich user interface makes it easy to visualize pipelines running in production, monitor progress, and troubleshoot issues when needed.

Getting started

Please visit the Airflow Platform documentation (latest **stable** release) for help with installing Airflow, getting a quick start, or a more complete tutorial.

Documentation of GitHub master (latest development branch): ReadTheDocs Documentation

For further information, please visit the Airflow Wiki.

Tutorial

This tutorial walks you through some of the fundamental Airflow concepts, objects, and their usage while writing your first pipeline.

Example Pipeline definition

Here is an example of a basic pipeline definition. Do not worry if this looks complicated, a line by line explanation follows below.

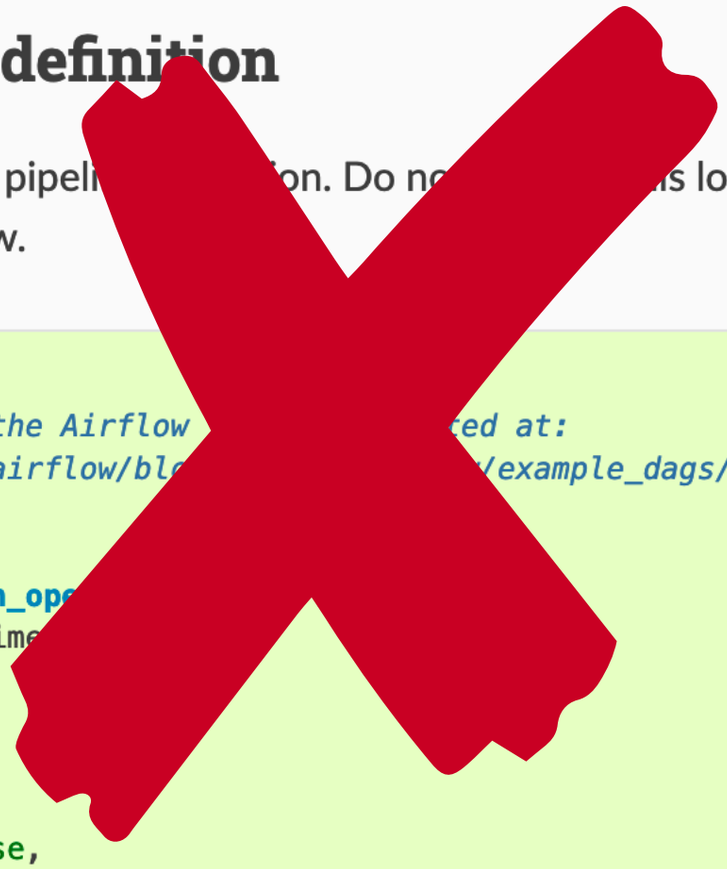
```
"""
Code that goes along with the Airflow tutorial located at:
https://github.com/apache/airflow/blob/master/airflow/example_dags/tutorial.py
"""
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from datetime import datetime, timedelta


default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2015, 6, 1),
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
    # 'queue': 'bash_queue',
    # 'pool': 'backfill',
    # 'priority_weight': 10,
    # 'end_date': datetime(2016, 1, 1),
}

dag = DAG('tutorial', default_args=default_args, schedule_interval=timedelta(days=1))
```

Tutorial

This tutorial walks you through some of the fundamental Airflow concepts, objects, and their usage while writing your first pipeline.

Example Pipeline definition

Here is an example of a basic pipeli on. Do n s looks complicated, a line by line explanation follows below.



```
"""
Code that goes along with the Airflow                ted at:
https://github.com/apache/airflow/bl                /example_dags/tutorial.py
"""
from airflow import DAG
from airflow.operators.bash_ope
from datetime import datetime


default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2015, 6, 1),
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
    # 'queue': 'bash_queue',
    # 'pool': 'backfill',
    # 'priority_weight': 10,
    # 'end_date': datetime(2016, 1, 1),
}

dag = DAG('tutorial', default_args=default_args, schedule_interval=timedelta(days=1))
```

live coding

```
cd DE4DS

# airflow needs a home
export AIRFLOW_HOME=`pwd`/airflow

pip install -U apache-airflow

# initialize the database
airflow initdb
```

airflow/dags/football.py

💨 `airflow test football fetch <date>`

```
airflow test basketball fetch <date>

# start the web server

airflow webserver -p 8080

# start the scheduler (new window)

export AIRFLOW_HOME=`pwd`/airflow
airflow scheduler
```

https://github.com/maxhumber/hickory

```
pip install hickory
```

```
hickory schedule 07-continuous.py --every=day@9:30am
hickory status
hickory kill 07-continuous.py
```

Homework

💻 **add t3 to save predictions to the db!**

Q&A

That's all Folks!