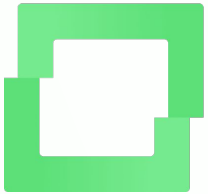


Kern AI

Building an AI-assisted dashboard for your daily newsletter

Who we are

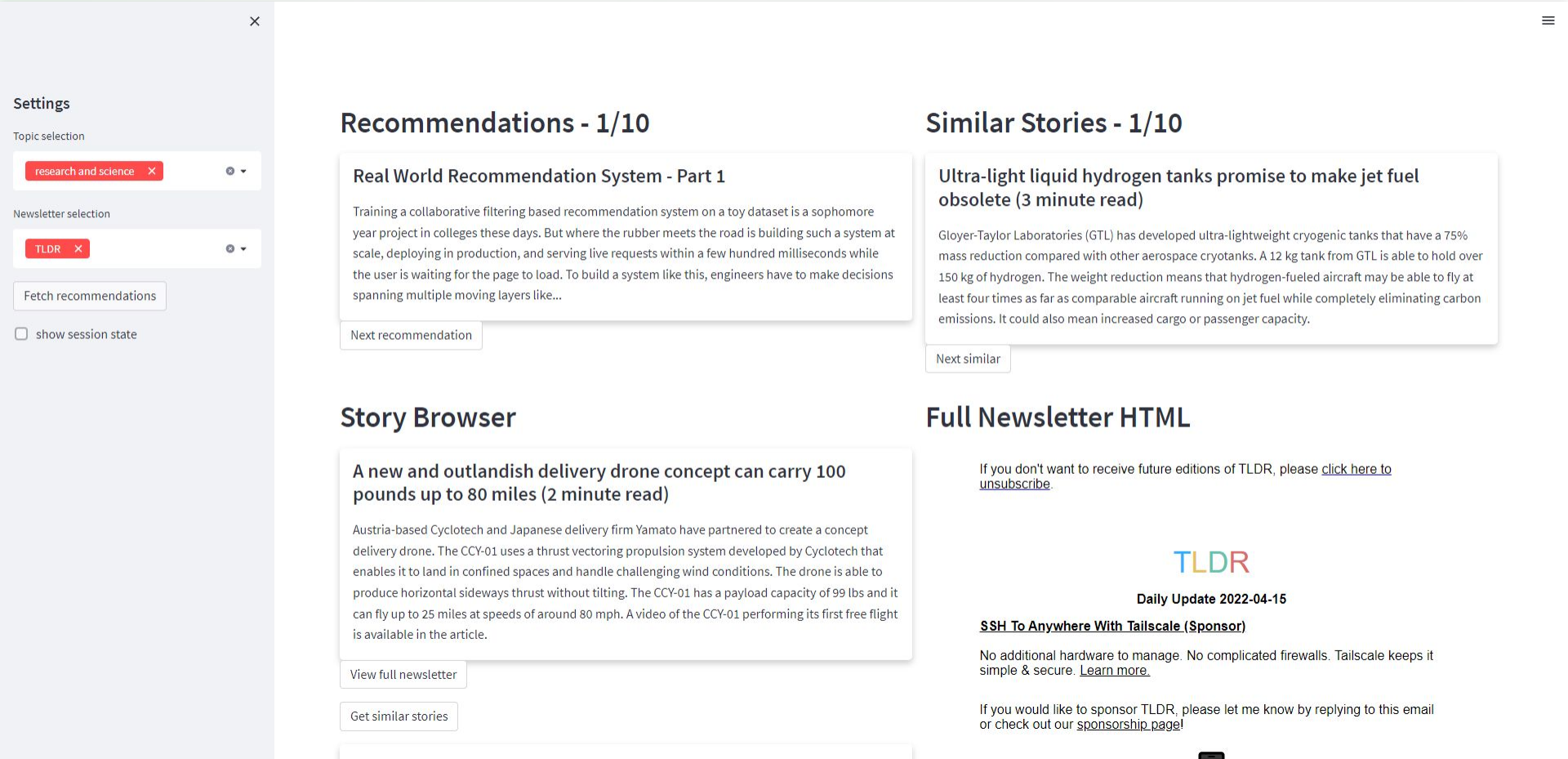
Moritz

Johannes

first time to host a workshop – please feedback us! :)

Today we want to build this:

Kern AI



Agenda for today

Kern AI

pairwise theory and practical content – all in Python!

1.	collecting data with oauth2 in GMail	15 min
2.	processing data (e.g. HTML and text parsing)	30 min
3.	building training data for topic modelling	15 min
4.	applying transfer learning with transformers	30 min
5.	**break**	10 min
6.	building the streamlit UI	40 min
7.	building a minimal backend with FastAPI	40 min

This is guided by a GitHub repository:

`https://github.com/code-kern-ai/datalift-summit`

Expectation Management

Kern AI

We (unfortunately) just have three hours.

We are building something that is not perfect, but really fun.

We are going to cut corners, but will cover the basics.

Goals:

- Show you real-life data collection
- Code a little real-life example of an NLP application
- Introduce you to fastAPI and streamlit
- Give you many possibilities for extension

Installing dependencies

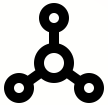
Please now go to the repository, and follow the installation instructions!

We'll have a short theoretical input for oauth2 to bridge the installation time

Data collection

Kern AI

Webpages, HTML



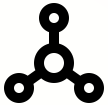
Social Media

Mail provider

Chats

Data collection

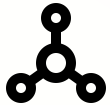
Kern

	Webpages, HTML	`selenium, requests + beautifulsoup`
	Social Media	`APIs`
	Mail provider	`oauth2`
	Chats	`Bots, APIs`

Data collection

Webpages, HTML

`selenium, requests + beautifulsoup`



Socia

APIs

always check the statement about data collection of the respective source (company, institution, ...)

!!!

Mail p

oauth2

Chats

`Bots, APIs`

Data collection

Mail provider

oauth2

Diving into oauth2

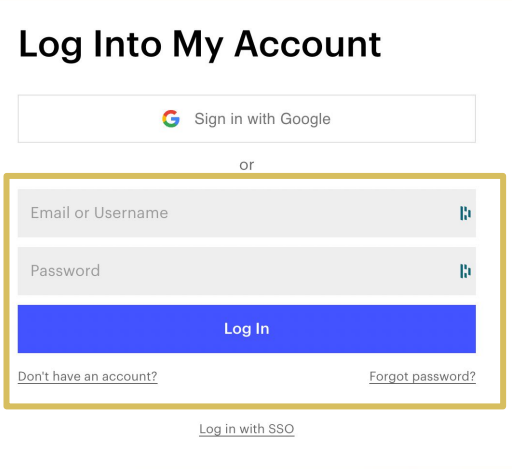
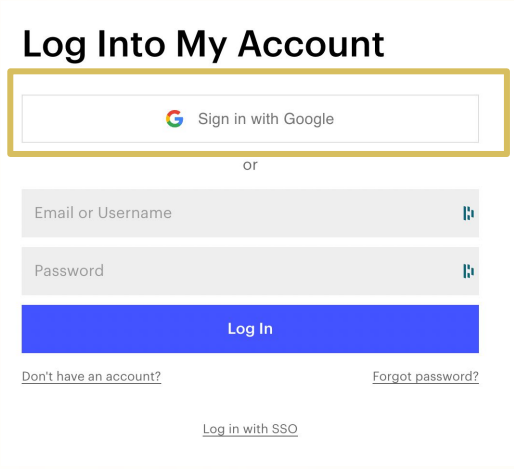
Kern

Authentication	Authorization
I prove that I am who I say I am	I prove that someone else told me that I could access something
Can literally do anything	Has scopes (e.g. reading mails, but not deleting them)
Username and Password	oauth2

e.g. Google doesn't allow 3rd party apps to authenticate as users any longer.
They can only access via authorization.

great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2

Authentication	Authorization
	

great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2

great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2

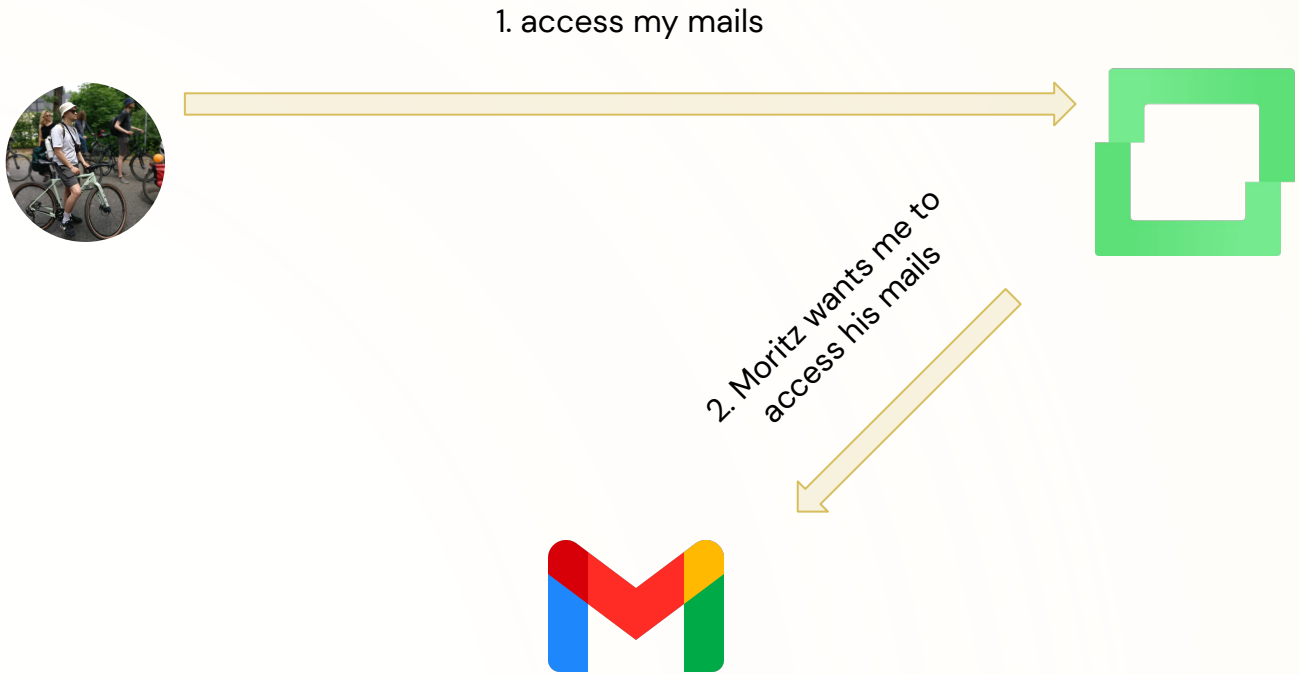
Kern

1. access my mails

great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

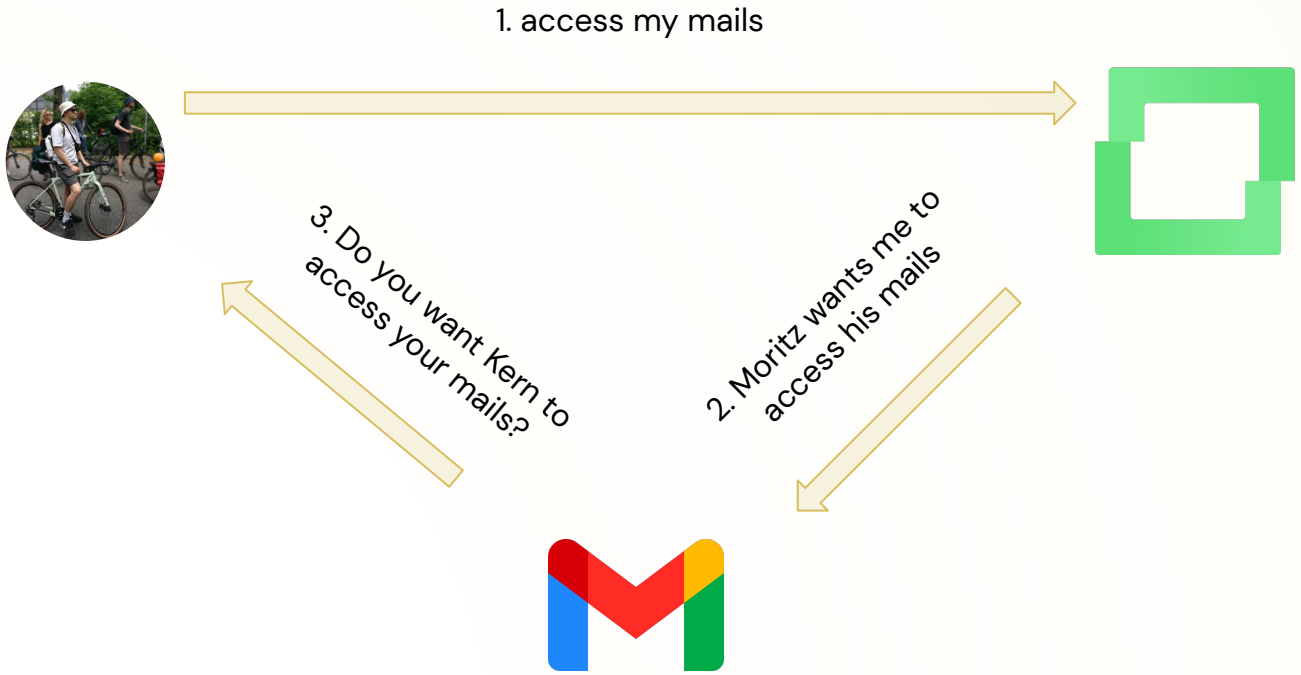
Diving into oauth2

Kern



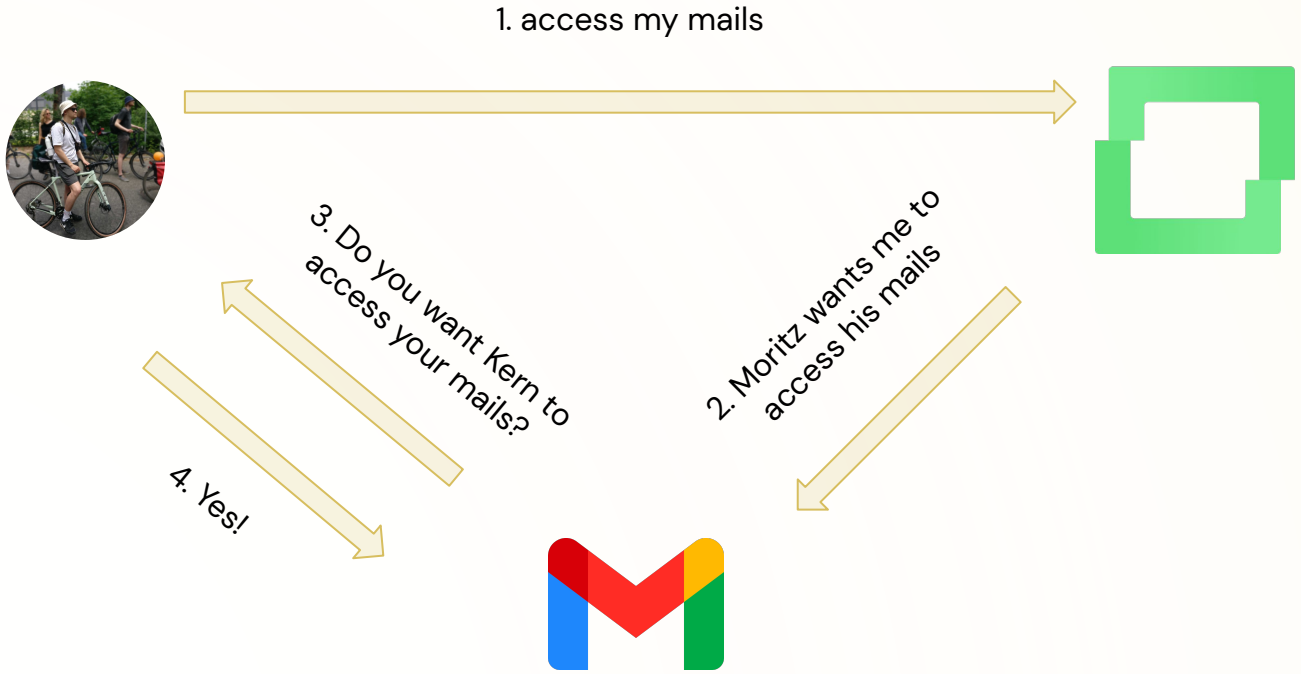
great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2



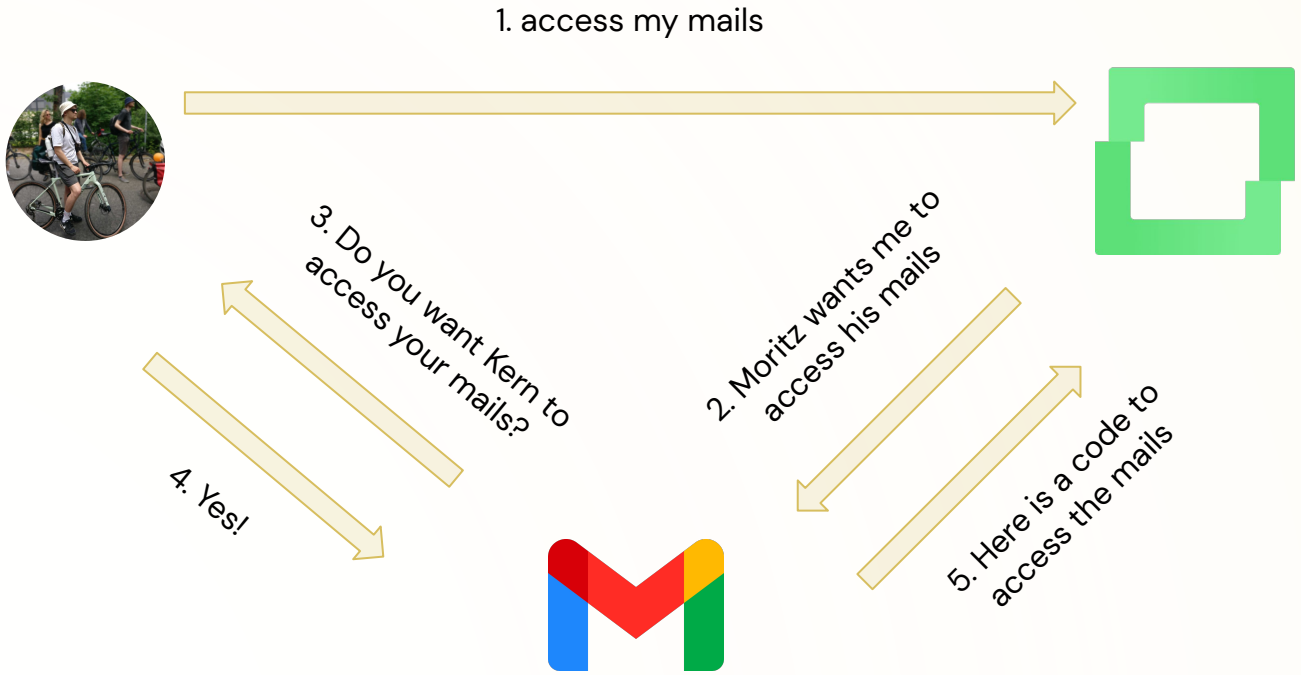
great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2



great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Diving into oauth2



great resource: https://www.youtube.com/watch?v=g6IAGvBZDkE&ab_channel=JasonGoodison

Practical part 1

Processing data

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> <html xmlns="http://www.w3.org/1999/xhtml">
<head> <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> <title>Data Science Weekly - Issue 440</title></head> <body style="margin: 0;padding:
0;background-color: #;"> <table align="center" border="0" cellpadding="0" cellspacing="0" height="100%" width="100%" id="bodyTable" style="mso-table-lspace:
0pt;mso-table-rspace: 0pt;margin: 0;padding: 0;background-color: #;border-collapse: collapse !important;height: 100% !important;width: 100% !important;"> <tr> <td
align="center" height="100%" valign="top" width="100%" id="bodyCell" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;margin: 0;padding: 20px;border-collapse: collapse
!important;height: 100% !important;width: 100% !important;"> <table border="0" cellpadding="0" cellspacing="0" id="templateContainer" style="mso-table-lspace:
0pt;mso-table-rspace: 0pt;width: 600px;border-collapse: collapse !important;"> <tr> <td align="center" valign="top" style="mso-table-lspace: 0pt;mso-table-rspace:
0pt;border-collapse: collapse !important;"> <!-- BEGIN PREHEADER --> <table border="0" cellpadding="0" cellspacing="0" width="100%" id="templatePreheader"
style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;border-top: 0;border-bottom: 0;border-collapse: collapse !important;"> <tr> <td class="preheaderContent"
style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;color: #34495e;font-family: Tahoma;font-size: 9px;line-height: 125%;padding-top: 10px;padding-bottom: 10px;text-align:
left;border-collapse: collapse !important;">Curated news,
articles and jobs related to Data Science.
 Keep up with all the latest developments</td> <!-- --> <td width="220"
class="preheaderContent" style="padding-left: 20px;mso-table-lspace: 0pt;mso-table-rspace: 0pt;color: #34495e;font-family: Tahoma;font-size: 9px;line-height:
125%;padding-top: 10px;padding-bottom: 10px;text-align: left;border-collapse: collapse !important;"><span
style="font-family:tahoma,verdana,segoe,sans-serif;">Email not displaying correctly?
 <a
href="https://datascienceweekly.us3.list-manage.com/track/click?u=71a2b2a38789d4d25b738462f&id=3d2f3344bc&e=7dcf46431f" target="_blank" style="color: #34495e;font-weight:
bold;text-decoration: none;">View it in your browser. </td> <!-- --> </tr> </table> <!-- END PREHEADER --> </td> </tr> <tr> <td align="center"
valign="top" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;border-collapse: collapse !important;"> <!-- BEGIN HEADER --> <table border="0" cellpadding="0"
cellspacing="0" width="100%" id="templateHeader" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;border-top: 10px solid #000000;border-bottom: 5px solid
#000000;border-collapse: collapse !important;"> <tr> <td valign="top" class="headerContent" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;color: #000000;font-family:
Helvetica;font-size: 20px;font-weight: bold;line-height: 100%;padding-top: 20px;padding-bottom: 20px;text-align: center;border-collapse: collapse !important;"><h1
style="display: block;font-family: Georgia;font-size: 26px;font-style: normal;font-weight: bold;line-height: 100%;letter-spacing: normal;margin-top: 0;margin-right:
0;margin-bottom: 10px;margin-left: 0;text-align: center;color: #34495e !important;"><img align="none" height="253"
src="https://gallery.mailchimp.com/71a2b2a38789d4d25b738462f/images/twitter_DataScienceWeekly.png" width="319" style="border: 0;height: auto;line-height: 100%;outline:
none;text-decoration: none;"></h1> <h2 style="display: block;font-family: Tahoma;font-size: 20px;font-style: normal;font-weight: bold;line-height: 100%;letter-spacing:
normal;margin-top: 0;margin-right: 0;margin-bottom: 10px;margin-left: 0;text-align: center;color: #34495e !important;">Issue #440

 April 28 2022</h2> </td> </tr>
</table> <!-- END HEADER --> </td> </tr> <tr> <td align="center" valign="top" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;border-collapse: collapse !important;">
<!-- BEGIN BODY --> <table border="0" cellpadding="0" cellspacing="0" width="100%" id="templateBody" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;border-top:
0;border-bottom: 0;border-collapse: collapse !important;"> <tr> <td valign="top" class="bodyContent" style="mso-table-lspace: 0pt;mso-table-rspace: 0pt;color:
#000000;font-family: Helvetica;font-size: 16px;line-height: 150%;padding-top: 40px;padding-bottom: 40px;text-align: left;border-collapse: collapse !important;"><h2
style="line-height: 20px;display: block;font-family: Tahoma;font-size: 20px;font-style: normal;font-weight: bold;letter-spacing: normal;margin-top: 0;margin-right:
0;margin-bottom: 10px;margin-left: 0;text-align: center;color: #34495e !important;">Editor Picks</h2>
<a
href="https://datascienceweekly.us3.list-manage.com/track/click?u=71a2b2a38789d4d25b738462f&id=0b657fd1e9&e=7dcf46431f" target="_blank" style="color: #FF0000;font-weight:
normal;text-decoration: none;">Beyond interpretability: developing a language to shape our relationships with AI
 AI will continue becoming more complex, bigger,
and smarter. Wouldn’t it be nice if we could ask it questions to learn how and why it makes its predictions? Unfortunately, we don’t have a good language to
communicate with AI yet...[This post is based on the 2022 ICLR Keynote]...   <a
href="https://datascienceweekly.us3.list-manage.com/track/click?u=71a2b2a38789d4d25b738462f&id=8330e19e35&e=7dcf46431f" target="_blank" style="color: #FF0000;font-weight:
normal;text-decoration: none;">DALL·E 2 and The Origin of Vibe Shifts
 The point of this essay isn’t about predicting next year’s design trends.
To me the more interesting thing is to understand the ecological process that generates those trends, seeing the true signaling function of visual design, and learning
why some corporate status signaling is so effective and why some isn’t. Projecting further out, it’s about trying to picture a world where it’s cheap
and easy for anyone to generate just about any kind of image they want...To answer these questions, we’re going to tap the most well-developed pool of knowledge on
the use of costly signals and their evolution over time: biology....   <a
href="https://datascienceweekly.us3.list-manage.com/track/click?u=71a2b2a38789d4d25b738462f&id=a74165086a&e=7dcf46431f" target="_blank" style="color: #FF0000;font-weight:
normal;text-decoration: none;">Learning with not Enough Data Part 3: Data Generation
 Part 3 of “what if you don’t have enough training data”
series - touch base on creating more synthetic data by data augmentation or model generation, as well as some ideas on how to work with noisy labels (given synthetic data
might not be fully correct)...   <hr>   <h2 style="line-height: 20px;display: block;font-family: Tahoma;font-size: 20px;font-style:

Processing data

newsletter	date	headline	...
datascienceweekly	2022-04-28 23:21:02+00:00	Beyond interpretability: developing a language to shape our relationships with AI	
datascienceweekly	2022-04-28 23:21:02+00:00	DALL·E 2 and The Origin of Vibe Shifts	
datascienceweekly	2022-04-28 23:21:02+00:00	It's Our Moral Obligation to Make Data More Accessible	
datascienceweekly	2022-04-28 23:21:02+00:00	Scientists Publish Breakthrough Study in Oreo-Splitting Research	
Box of Amazing	2022-05-08 06:01:12+00:00	Refind – Get smarter every day	
Box of Amazing	2022-05-08 06:01:12+00:00	Inside Elon Musk's Big Plans for Twitter	
Box of Amazing	2022-05-08 06:01:12+00:00	Influencer culture is everywhere — even in academia	

Processing data

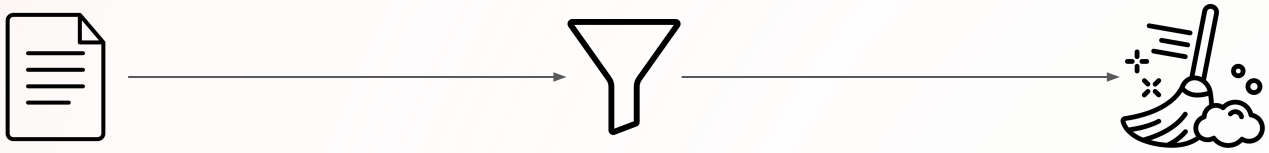
the “typical” steps to preprocess raw data

- loading: what data types do we have, how can we integrate them into one file?
- filtering: which newsletter sources do we want to keep for which time period?
- cleansing: what part of the HTML do we want to keep?
- splitting: how many stories does one newsletter contain?

for processing HTML data, we highly recommend `BeautifulSoup`!
(you’ll see that in the practical part)

processing for general data preparation and processing for ML can differ!

Practical part 2



Read and look at the data

Filter the newsletters

Clean the HTMLs

Building training data

SL use case =
algorithms + labeled data

> When a system isn't performing well, many teams instinctually **try to improve the code**. But for many practical applications, it's more effective instead to **focus on improving the data**.
>
> Andrew Ng
> Ex-Head Google Brain/Baidu AI, Founder Landing.AI Coursera, Professor Stanford

f(x) = y

model-centric

f(**x**) = **y**

data-centric

Building training data

Scientists Publish Breakthrough
Study in Oreo-Splitting Research

Building training data

Kern AI

Building training data

Label Studio

doccano

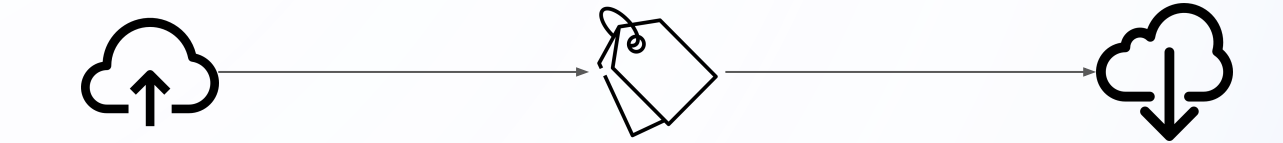
Kern AI

different open-source labeling tool options:

- Label Studio is the most diverse with custom HTML templates for labeling UIs
- Doccano is a great alternative, simple usage as Label Studio
- Kern AI is the most programmatic approach, focusing on NLP and semi-structured data (OS-release on 17.07)

All of them are great tools. Pick which you like the most 😄

Practical part 3



Upload the data to kern

Label the data

Download the data

Applying transfer learning

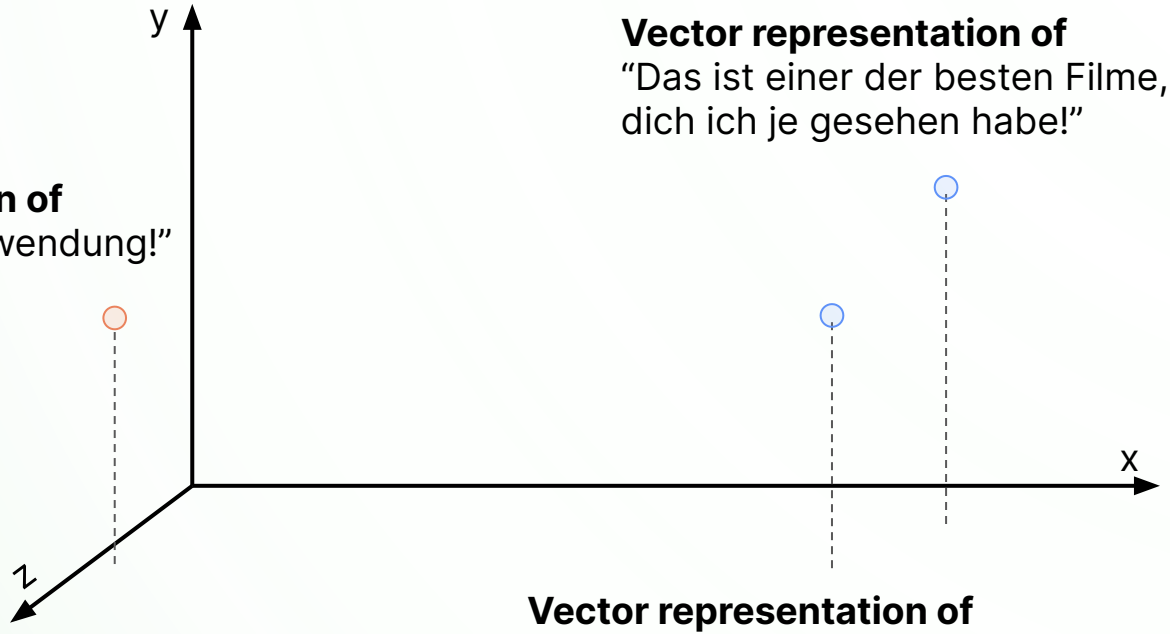
integration of large-scale language models

- goal: achieve great results with rather small amounts of training data

1. pull a transformer model from Hugging Face
2. create vector representations for your sentences (encoding/embedding)
3. train a logistic regression (or some other simple model) that takes these vectors as input

We will not finetune the model as this would take too much time!

Applying transfer learning

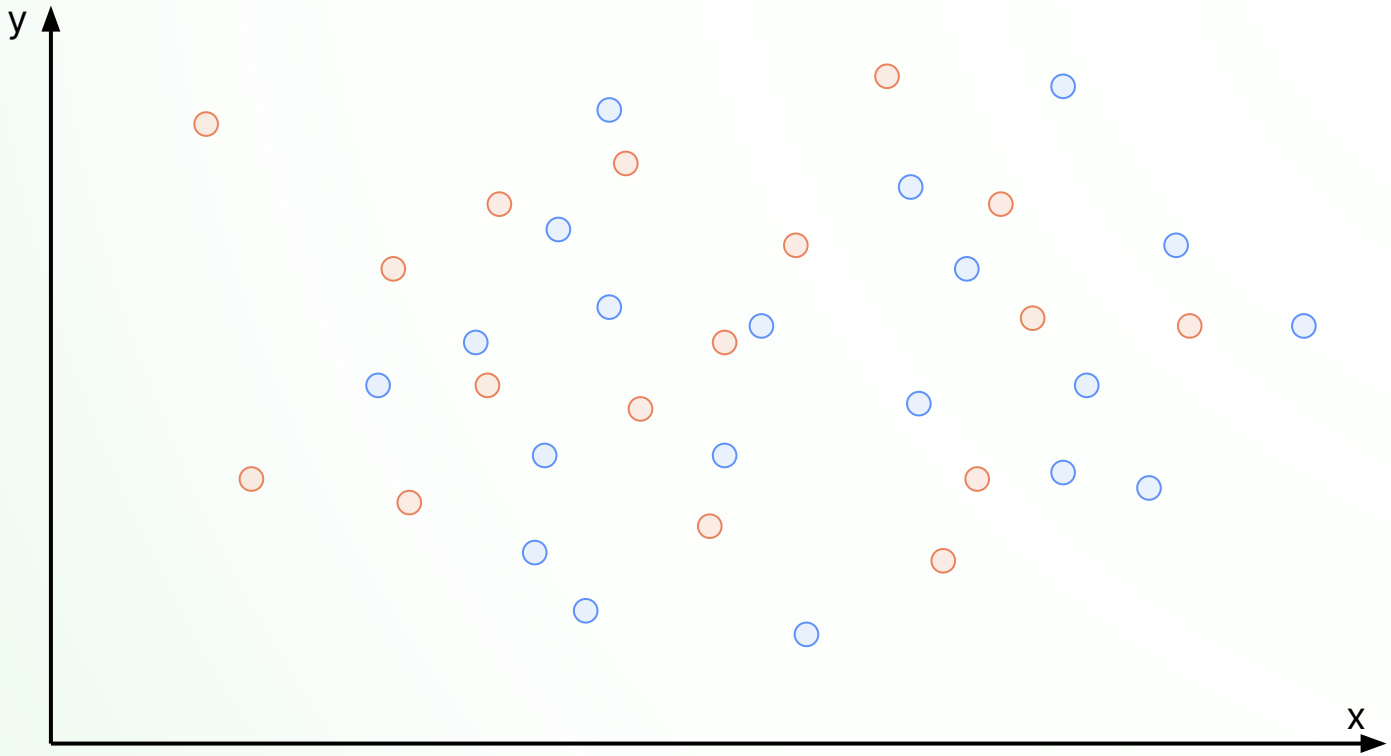


Vector representation of
“Was eine Zeitverschwendung!”

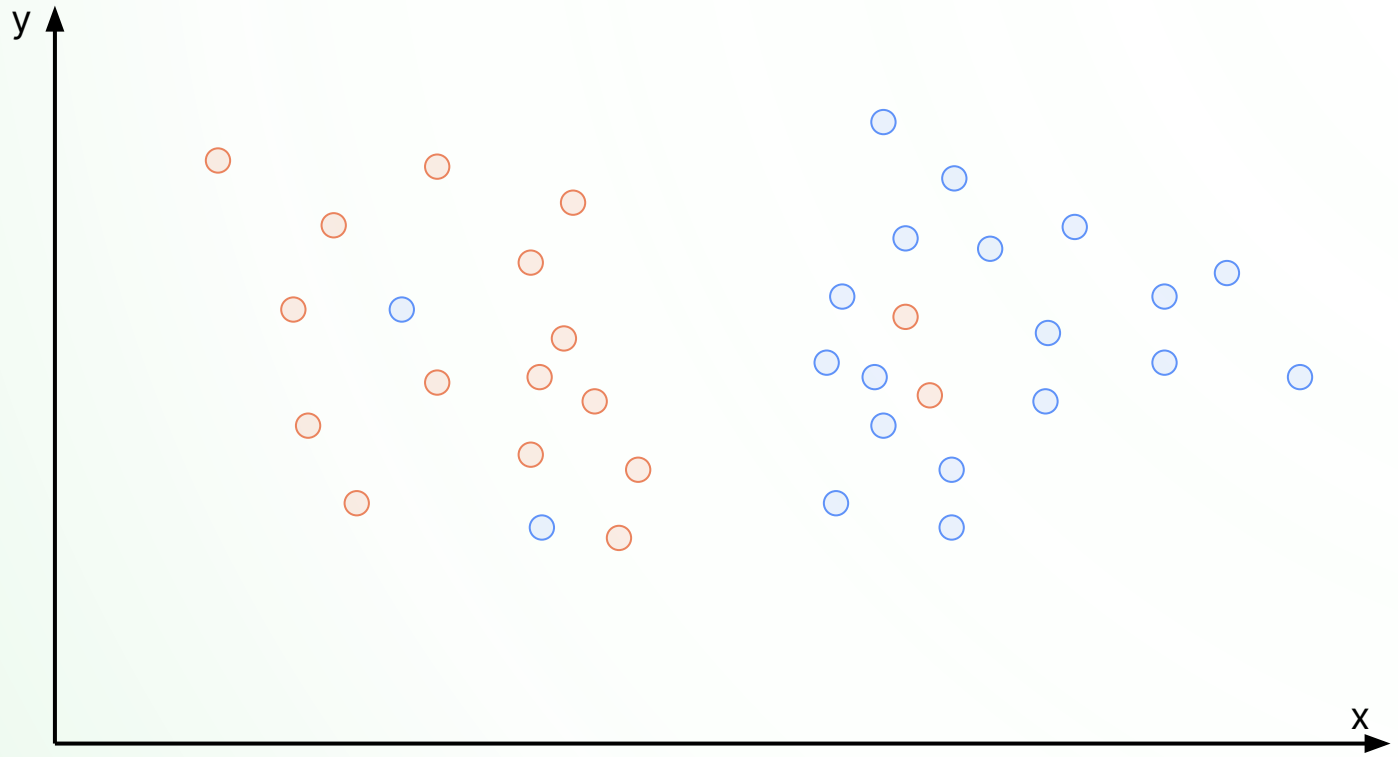
Vector representation of
“Das ist einer der besten Filme, dich ich je gesehen habe!”

Vector representation of
“Tolle Schauspieler, und der Plot war genial!”

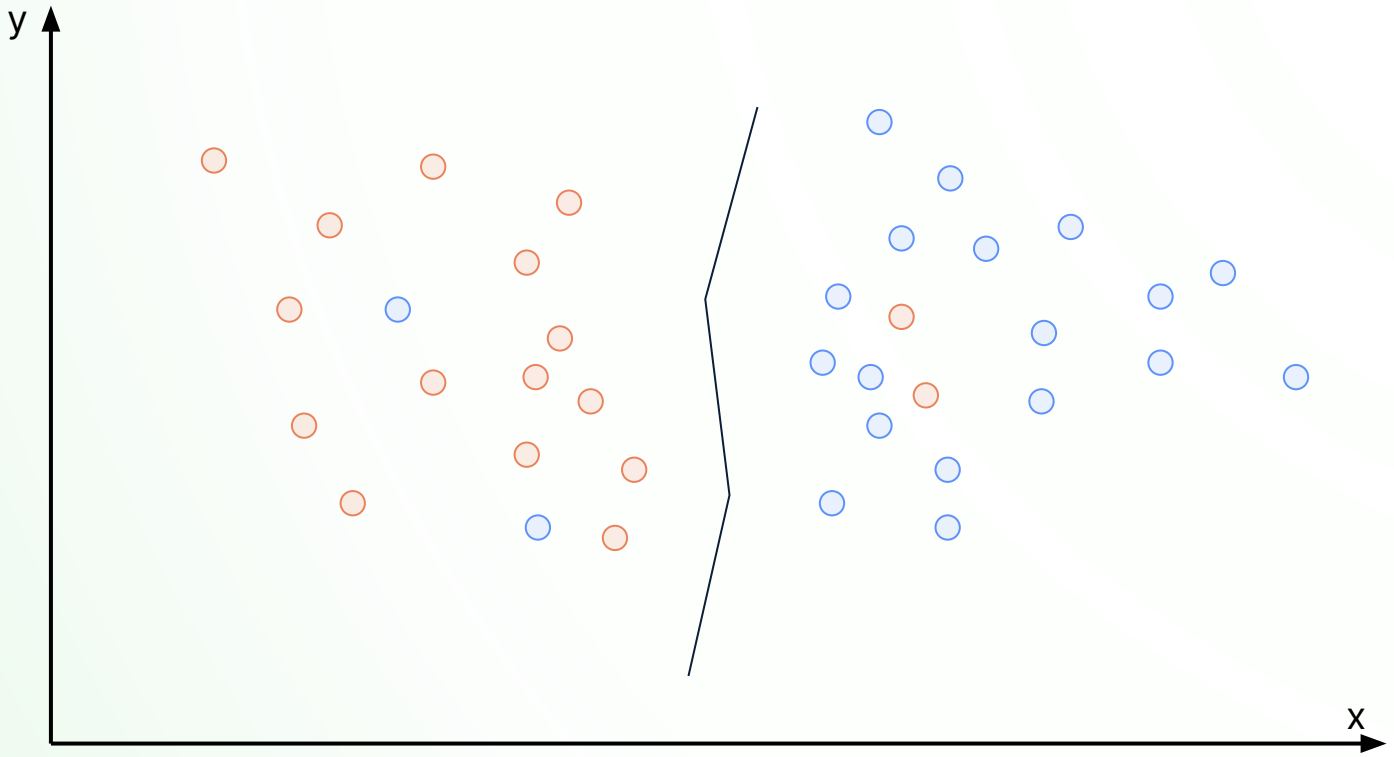
Regular encoder



Transformer model



Transformer model



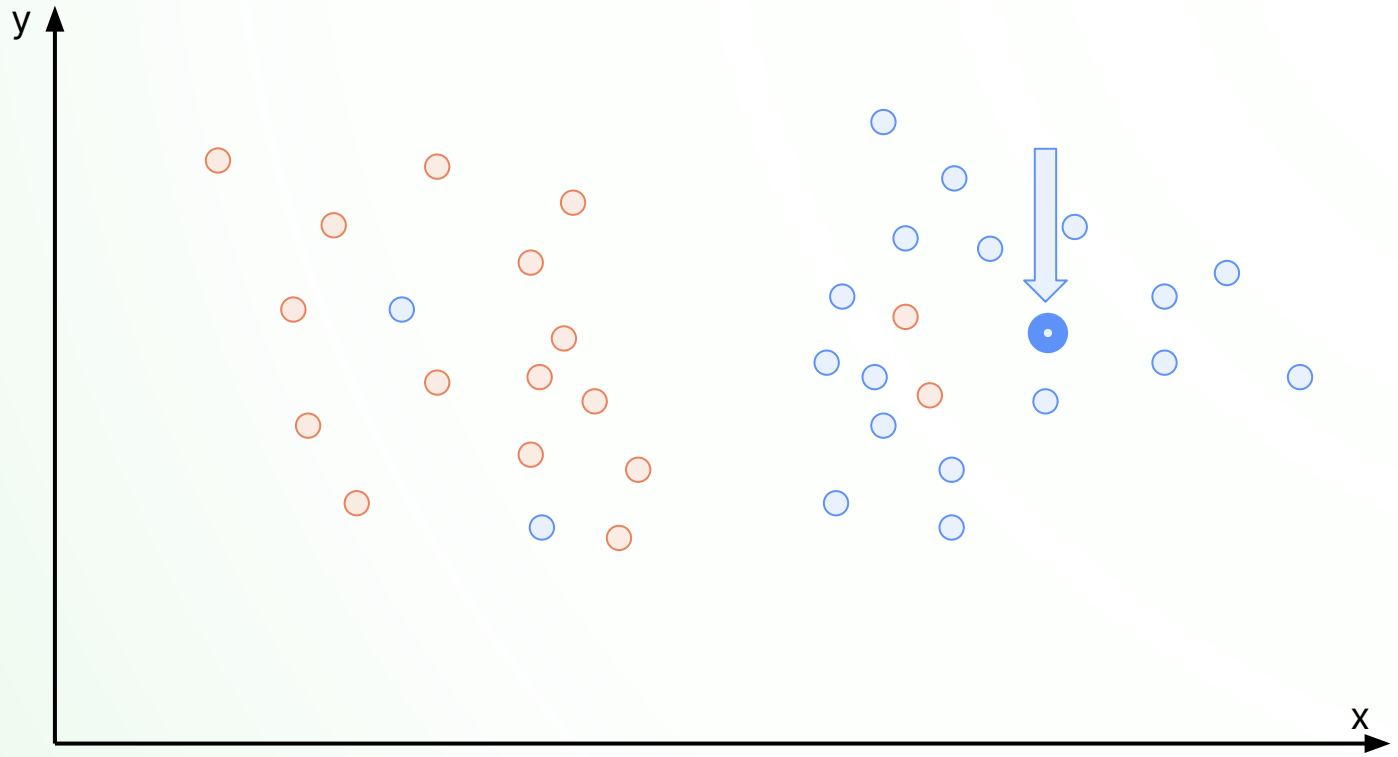
Making recommendations

Kern AI

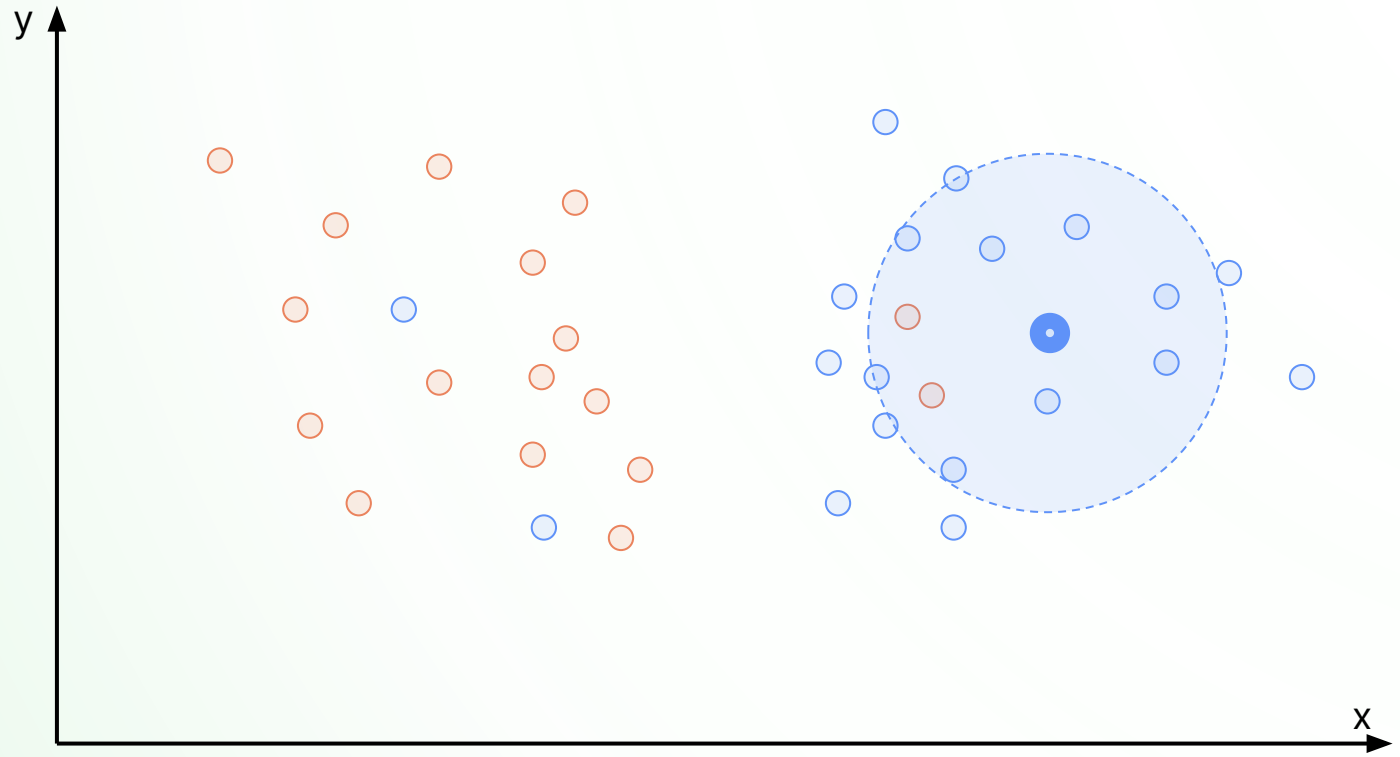
very basic idea:

- collect all interesting records and their embeddings
- compute the mean of this embedding as a representative embedding
- derive the k-nearest-neighbors of that representative embedding for suggestions

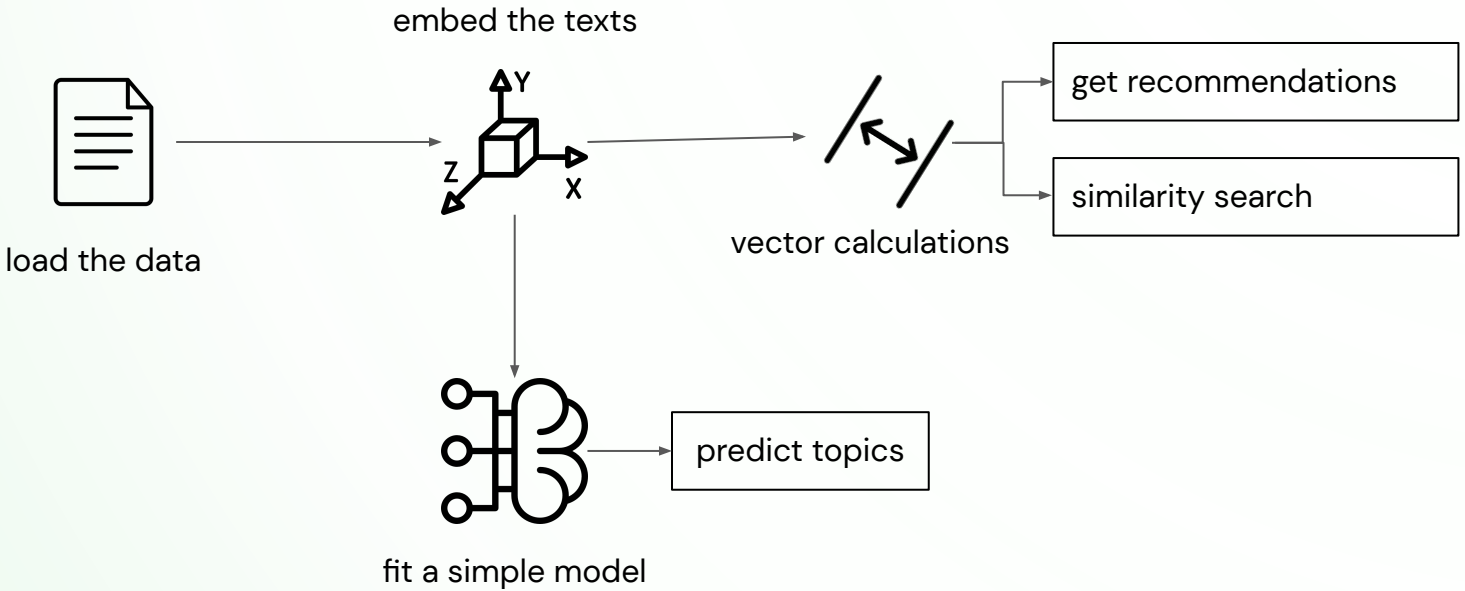
Making recommendations



Making recommendations



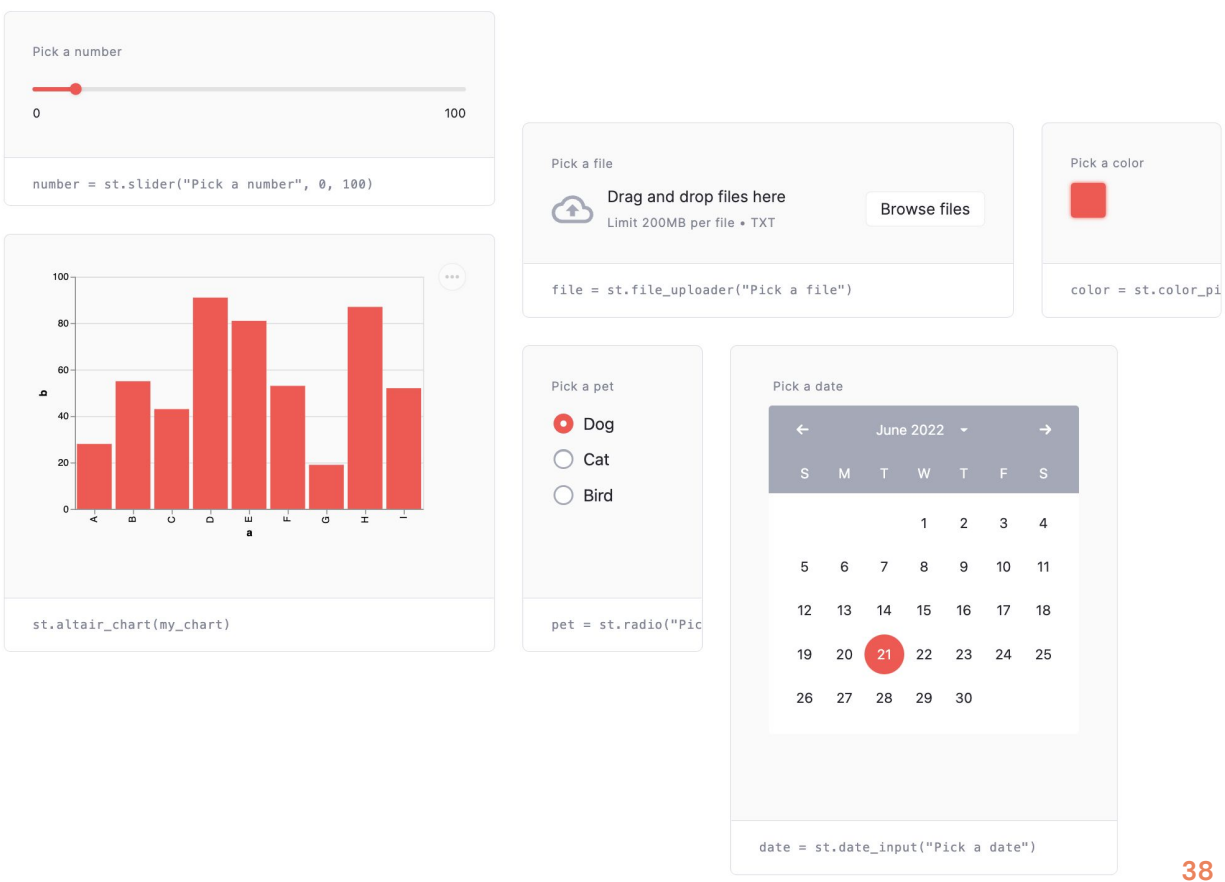
Practical part 4



Building the frontend

small + easy-to-build UI

- communicate results
- improve usability
- all in Python!



https://streamlit.io/

Building the frontend

graphical user interface

Building the frontend

HTML: <h1>

styled <input>

call backend

```python
import streamlit as st
import requests

st.title("Prediction UI")

input_ = st.text_input(
    "Try out your new machine learning model in this user interface. Just type some text below."
)

if st.button("Predict!"):
    if input_ is not None:
        # Get request output from the fastapi
        response = requests.post(
            "http://localhost:7531/predict", json={"text": [input_]}
        )
        if response.status_code == 200:
            st.markdown(response.json(), unsafe_allow_html=True)
```

Streamlit main concepts

Kern AI

Data flow

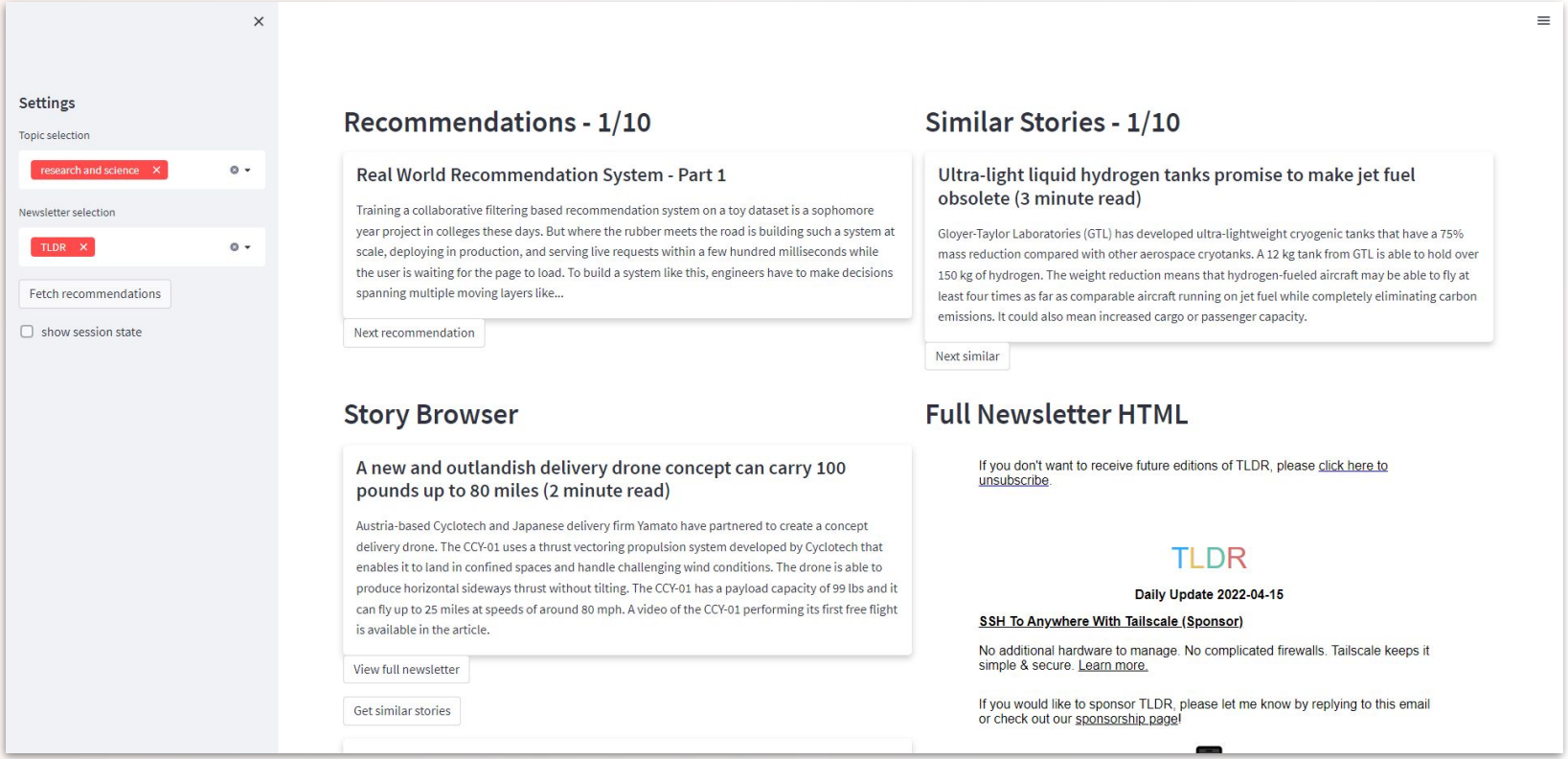
Caching

```python
import streamlit as st

@st.cache  # 👈 This function will be cached
def my_slow_function(arg1, arg2):
    # Do something really slow in here!
    return the_output
```

Session State

```python
import streamlit as st

def get_session_state_value(key : str):
    # Initialization
    if key not in st.session_state:
        return None
    else:
        return st.session_state[key]
```

Practical part 5



Finishing with the backend

deploy the model we've built as a micro web-service

- enable fast development to production
- integrate with other services (e.g. UI)
- keep code simple

FastAPI

https://fastapi.tiangolo.com/

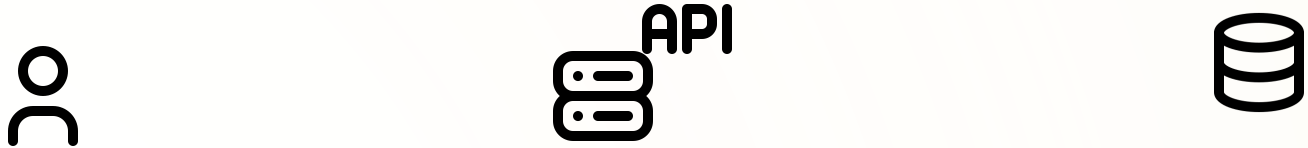
Finishing with the backend

Kern

REST API with HTTP methods

- POST: is used to submit an entity to the specified resource
- GET: requests a representation of the specified resource. Requests using GET should only retrieve data
- PUT: replaces all current representations of the target resource with the request payload
- DELETE: deletes the specified resource.
- ...

Finishing with the backend



Finishing with the backend

POST

add image

Finishing with the backend

POST

add image

API

Python

store to DB

Finishing with the backend

POST

add image

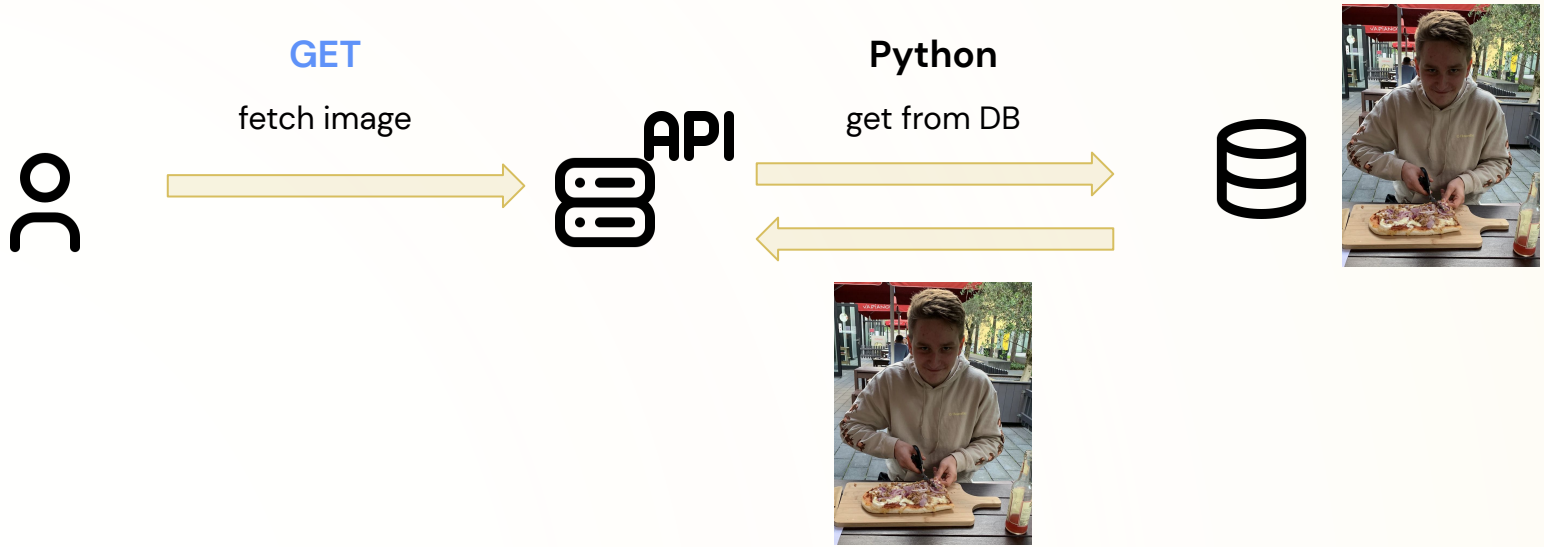
Python

store to DB

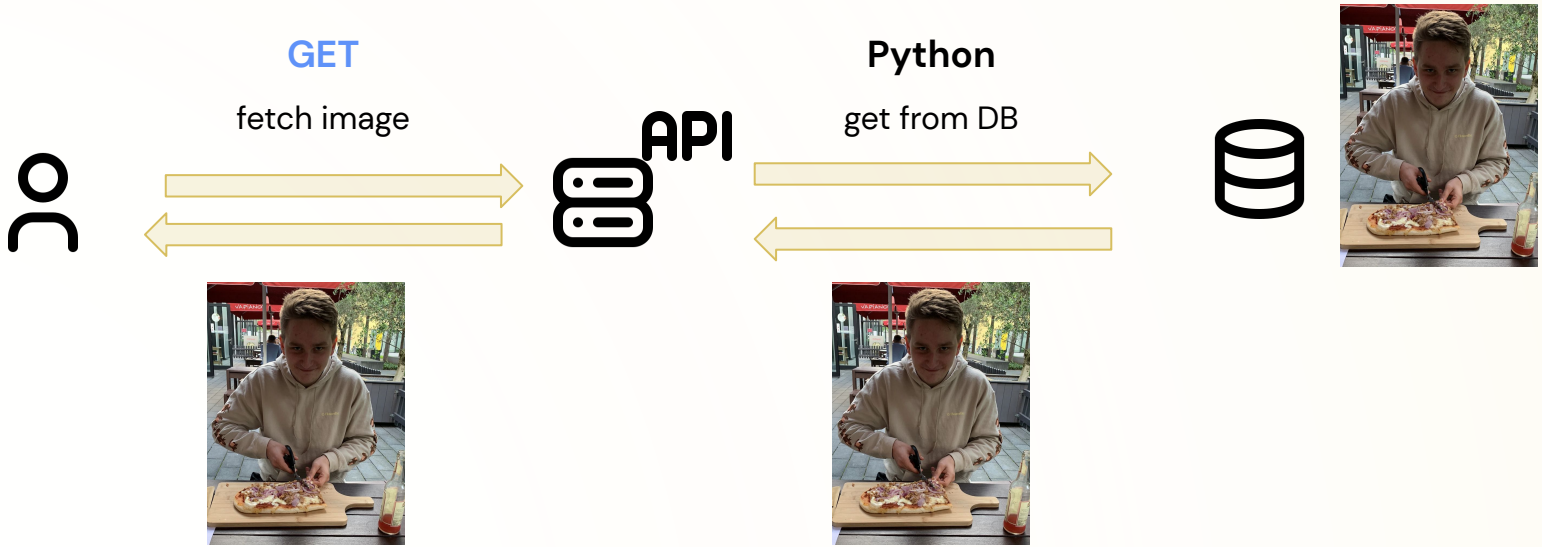
Finishing with the backend

Finishing with the backend

Finishing with the backend



Finishing with the backend



Finishing with the backend

Kern

```python
from fastapi import FastAPI

app = FastAPI()


@app.get("/")
async def root():
    return {"message": "Hello World"}
```

That’s a minimalistic but complete web API!

Finishing with the backend

Kern

can also take as input parameters, e.g. as query params or request body

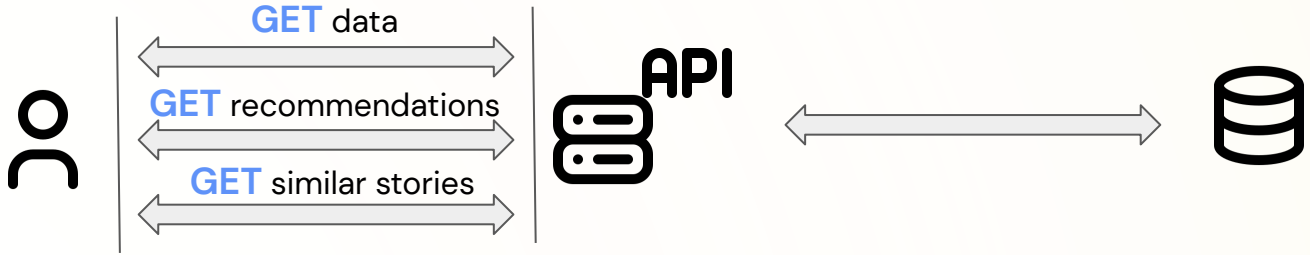
endpoint URI

e.g. http://localhost:8000/

```python
@app.get("/")
def root():
    return {"message": f"Hello World"}
```

GET value, e.g. as JSON

Practical part 6



Possible improvements

increasing complexity

Streamlit / fastAPI

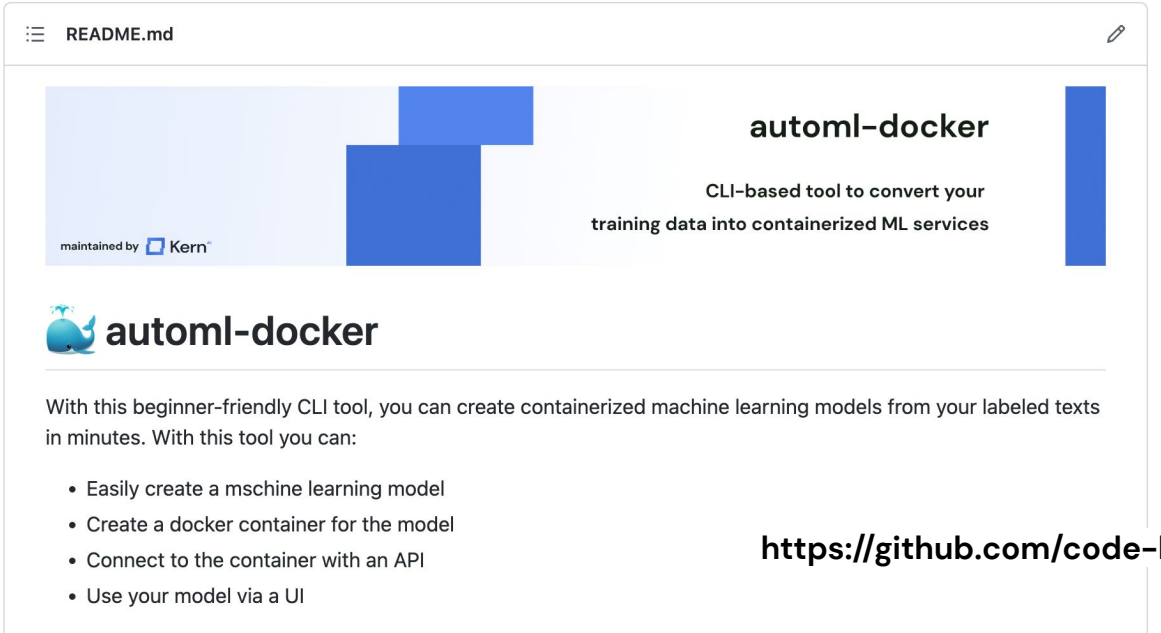
- add hyperlinks to the articles in the Story Browser
- add more filter settings (e.g. “last 7 days”, “no sponsored posts”, “skip empty entries”, …)
- make entries removable
- add more attributes to the cards, e.g. date, newsletter, topic
- provide label functionality for interesting articles to update recommendations
- add pagination in the front- and backend
- make it “online” with real incoming mails from your inbox

Data Science

- reduce embedding dimensions to 2D for an explorable newsletter map
- play around with different recommendation methods and see what works best
- try more sophisticated topic modeling with BERTopic

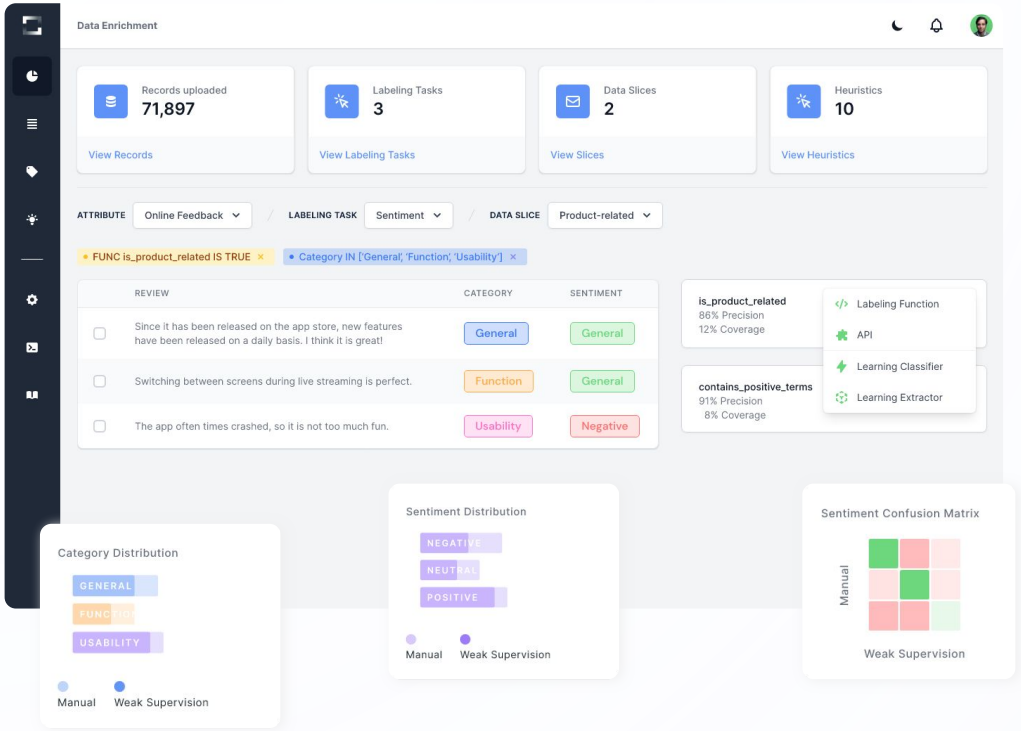
Check out our open-source `automl-docker`

GitHub repo containing ML training, containerized FastAPI backend and streamlit UI for custom Natural Language Classifiers.



https://github.com/code-kern-ai/automl-docker

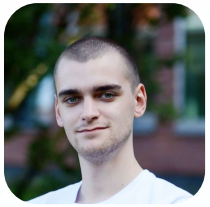
Data-centric IDE for NLP tasks on 17.07.2022



Make sure to check out our **open-source** release! 🤓

Register on our website www.kern.ai/pages/open-source for our raffle
(win a GeForce RTX 3090 Ti, Kern AI T-Shirts, ...)

Thanks for having us!



Moritz Feuerpfeil

moritz.feuerpfeil@kern.ai

https://www.linkedin.com/in/moritz-feuerpfeil/

Johannes Hötter

johannes.hoetter@kern.ai

https://www.linkedin.com/in/johanneshötter/

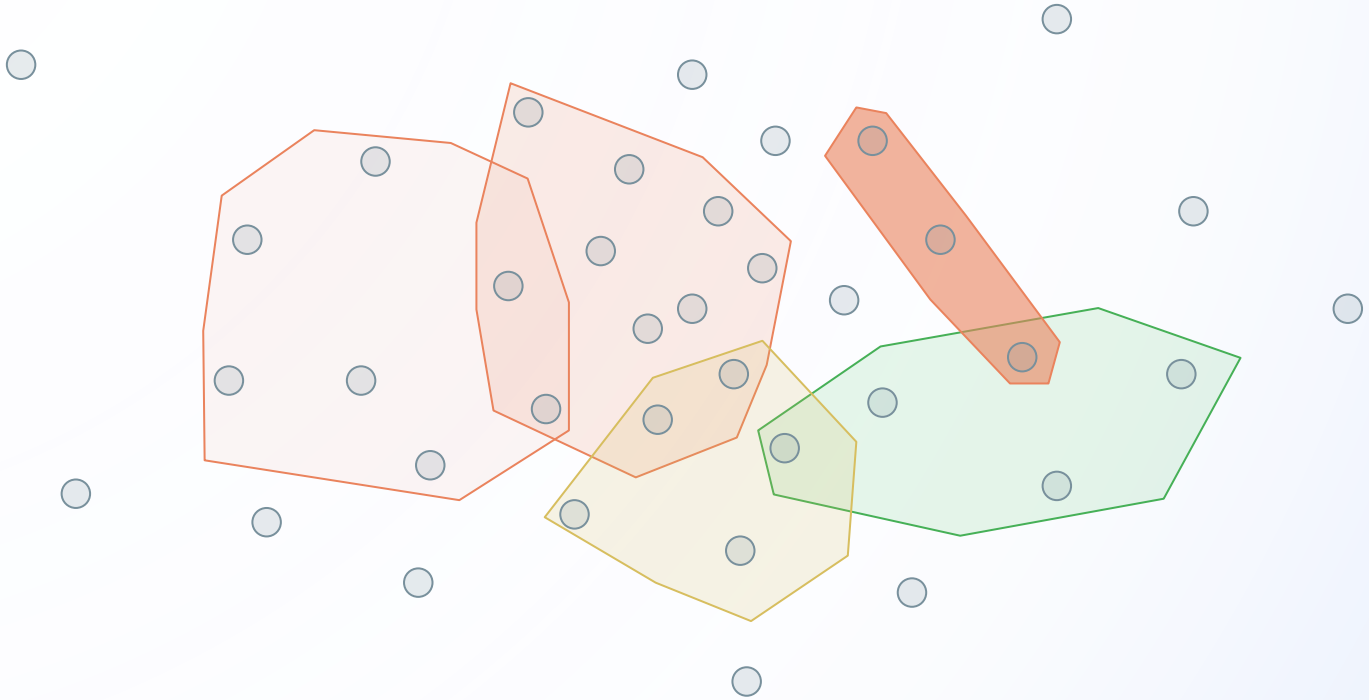
Backup Slides

Building training data

Kern AI

	Heuristic #1	Heuristic #2	...	Heuristic #N	**Weakly supervised**
Record #1	News	none		News	News 78.2%
Record #2	none	Politics		Sports	Politics 48.1%
...					
Record #N	none	none		News	News 32.6%

Building training data



Building training data

Kern AI

Majority Vote	requires no labeled data, *baseline*
Stochastic Gradient Descent-based	unsupervised learning, labeling functions as feature matrix
Closed form solutions	magnitude faster than SGD approach, requires e.g. triplets as features
Hidden Markov Model	for sequence labeling (e.g. named entity recognition), learns label dependencies
Weighted Majority Vote	requires *few* labeled reference data, fast and precise

Building training data

Kern AI

- Labeling functions

```
def starts_with_digit(record):
    if record["headline"].text[0].is_digit:
        return "Clickbait"
```

- Distant supervision (lookup values)
- Active (transfer) learning modules
- Zero-shot classifiers
- Unexperienced labelers (e.g. crowdlabeling)
- 3rd party systems, legacy systems, ...

Interface to collect noisy labels;

Relevance of each heuristic can be derived from e.g. manually labeled reference data

Building training data

green

```
def lkp_orgs(record):
    for chunk in record[“details”].noun_chunks:
        if any([chunk.text in trie or trie in chunk.text for trie in [“M3”, “...”]]):
            yield “ORG”, chunk.start, chunk.end
```

red

```
def window_cue_search(record):
    for chunk in record[“details”].noun_chunks:
        left_bound = max(chunk.sent.start, chunk.start - (window // 2) +1)
        right_bound = min(chunk.sent.end, chunk.end + (window // 2) + 1)
        window_doc = record[“details”][left_bound: right_bound]
        if any([term in window_doc.text for term in [“visits”, “...”]]):
            yield “PERSON”, chunk.start, chunk.end
```

yellow

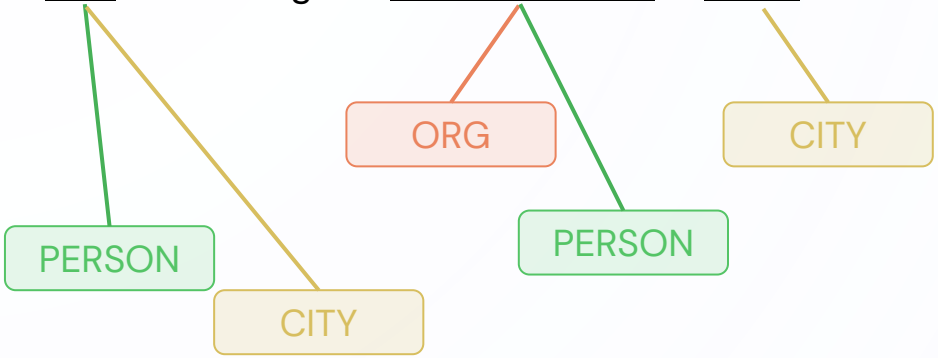
```
def model_prediction(record):
    for prediction in my_model(record[“details”]):
        label, start_idx, end_idx = prediction
        yield label, start_idx, end_idx
```

spaCy

further heuristics could be:

- spaCy lookups
- `requests / API`
- zero-shot models
- token-constraints

Building training data



selection depends on algorithm!

```
{
  "person": "Max",
  "org": "Datalift summit",
  "city": "Berlin"
}
```

Building training data

Building training data

Kern AI

1. **pick a random sample**
2. get the most diverse (distant) sample from labeled samples

Building training data

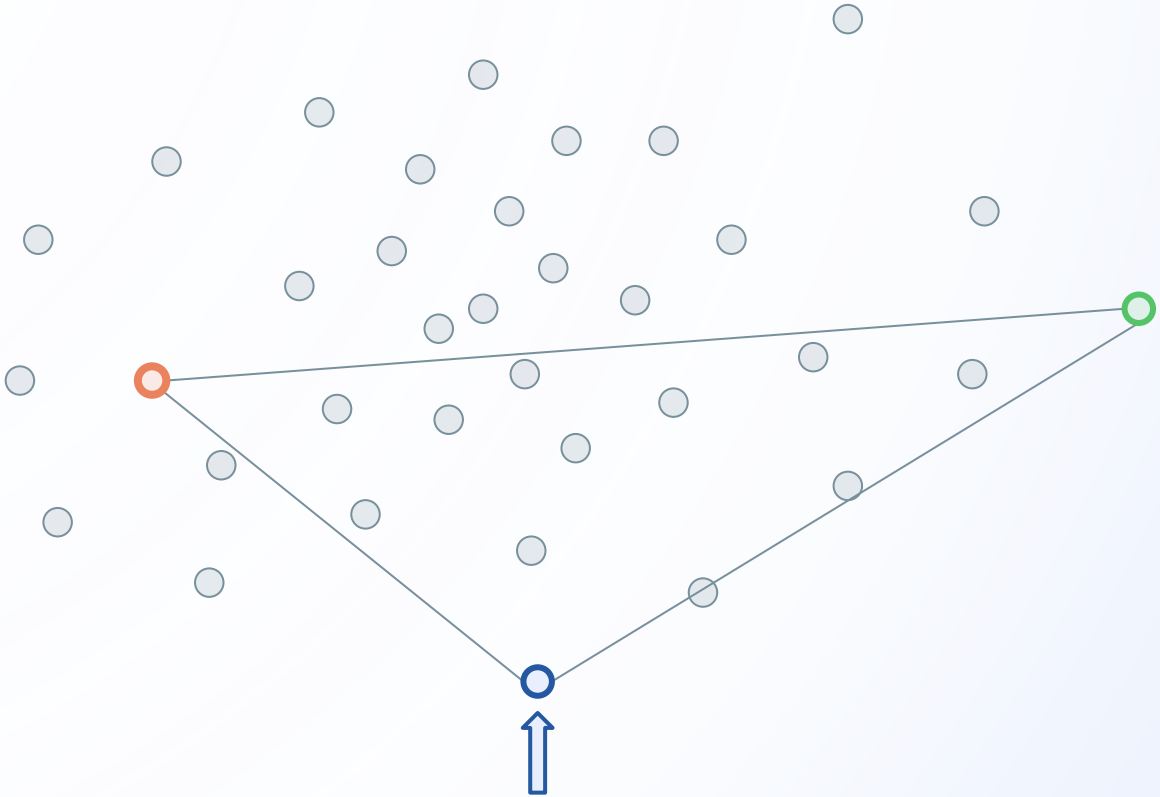
Kern AI

1. pick a random sample
2. get the most diverse (distant) sample from labeled samples

Building training data

Kern AI

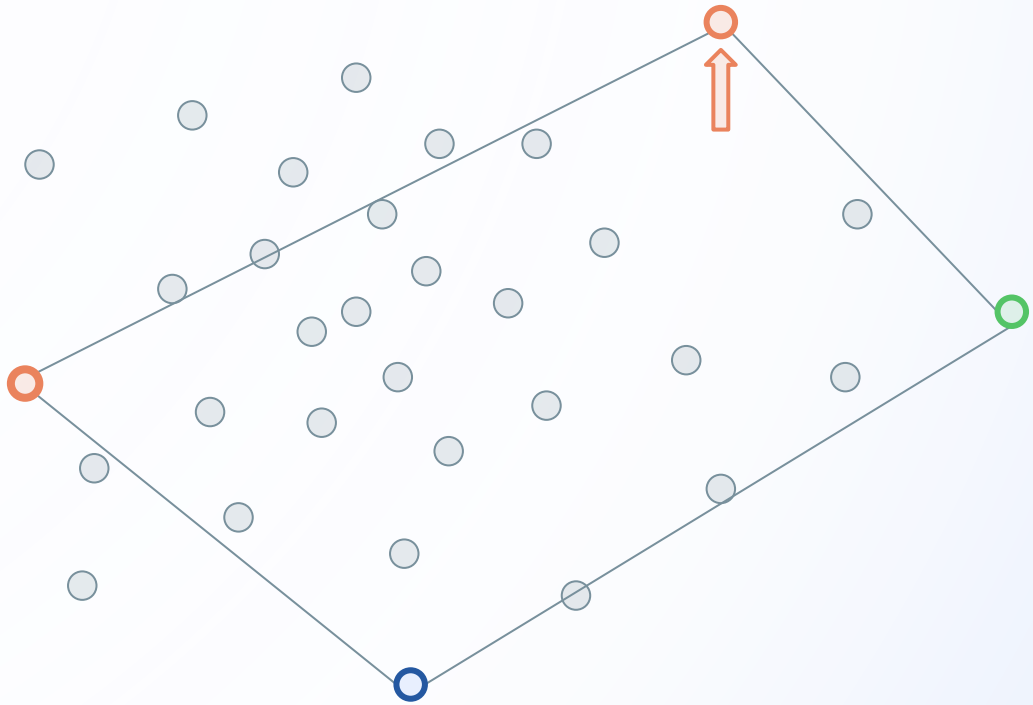
1. pick a random sample
2. get the most diverse (distant) sample from labeled samples



Building training data

Kern AI

1. pick a random sample
2. get the most diverse (distant) sample from labeled samples



Building training data

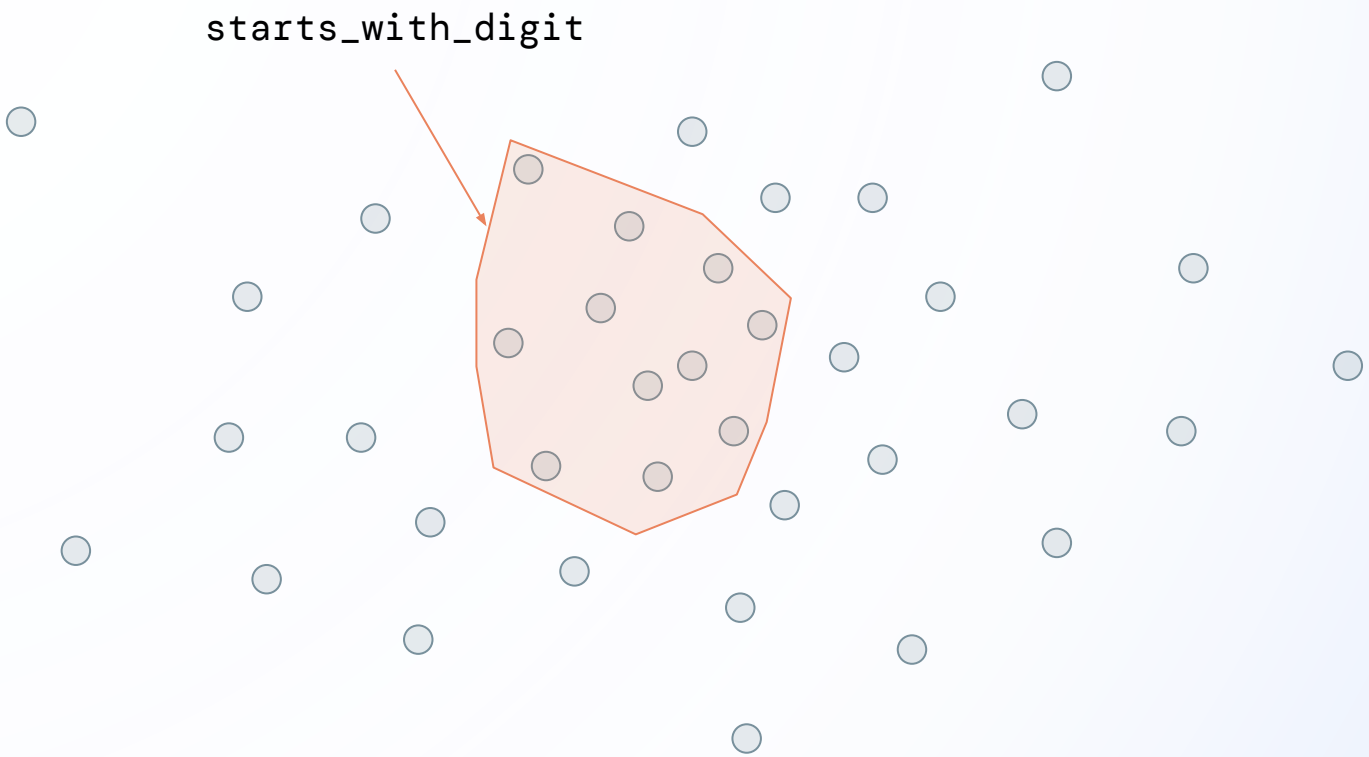
Kern AI

Building training data

Kern AI

1. **carve obvious samples**
2. label hard samples
3. label outliers

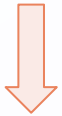
Building training data



Building training data

```
def starts_with_digit(record):
    if record["headline"].text[0].is_digit:
        return "Clickbait"
```

Precision 83%
Coverage 2.5%



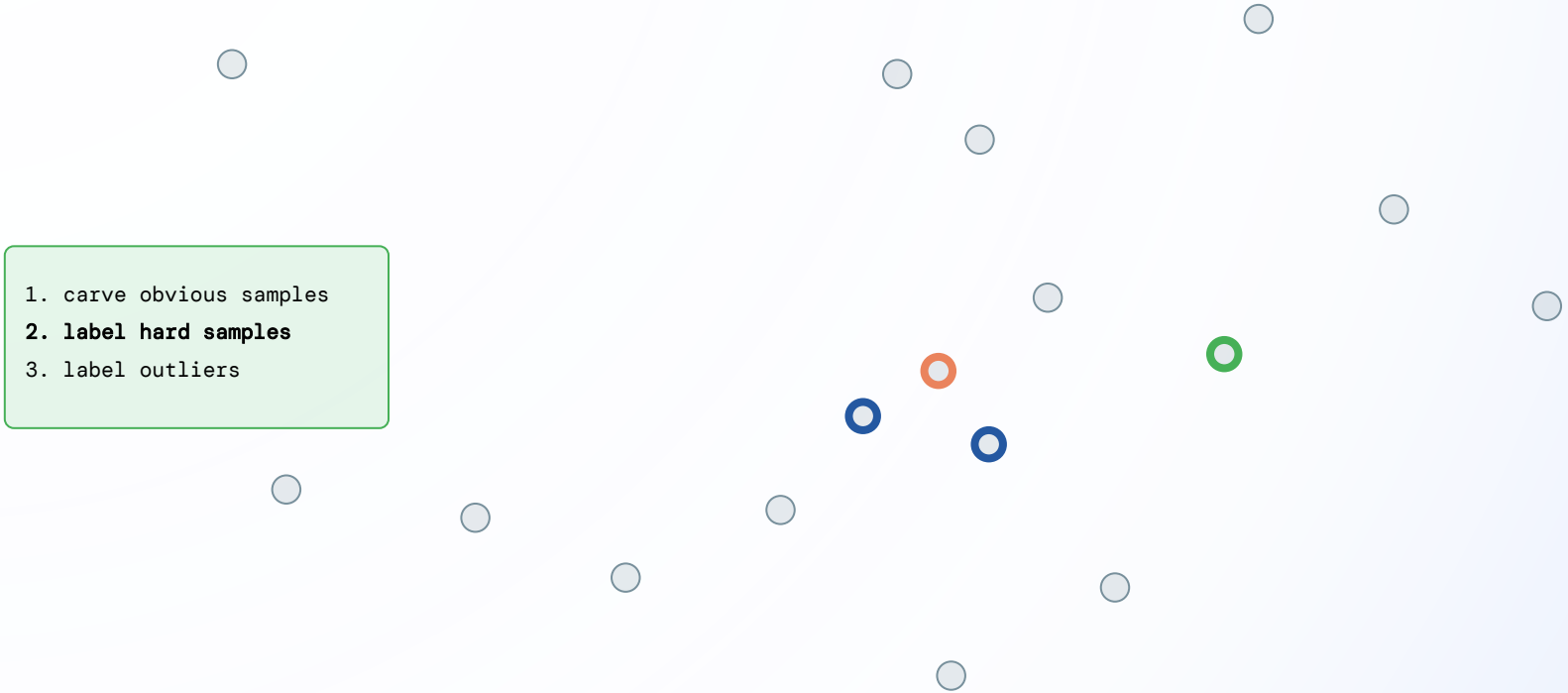
analyze filter where `starts_with_digit == "Clickbait"`

```
def starts_with_digit(record):
    if record["headline"].text[0].is_digit
    and record["sentiment"] > 0.7:
        return "Clickbait"
```

Precision 92%
Coverage 1.8%

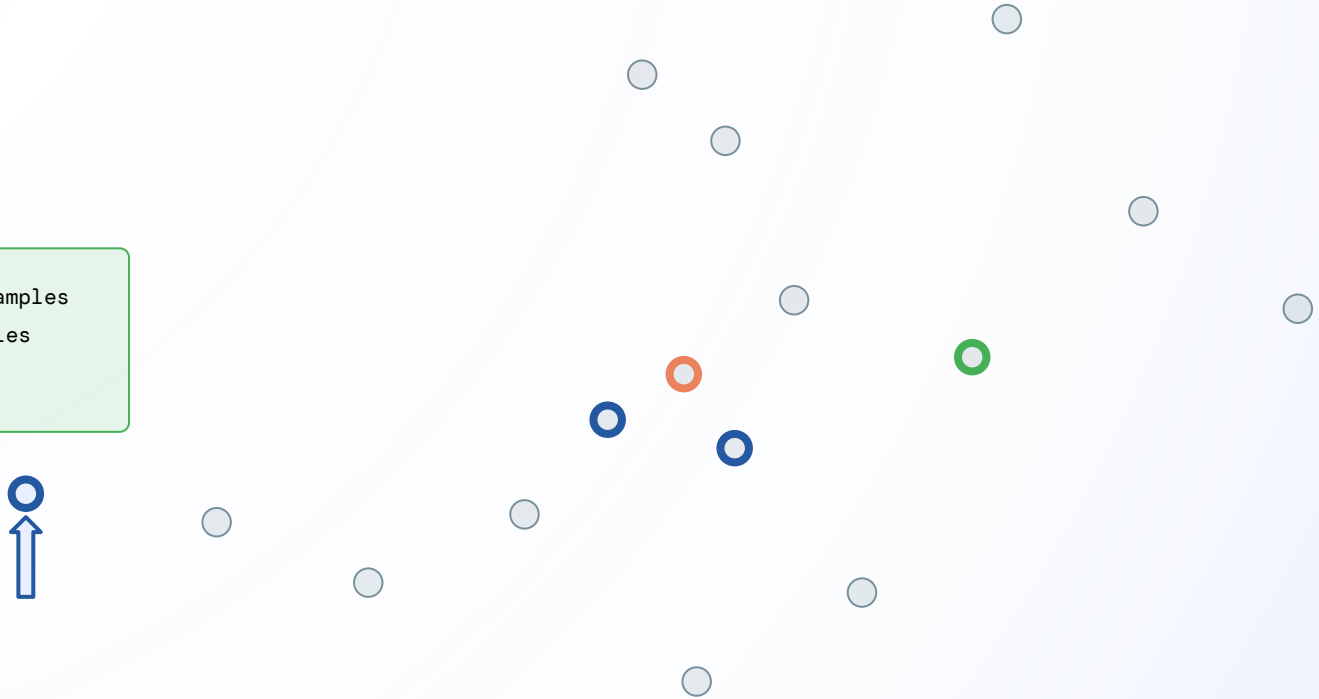
Building training data

Building training data

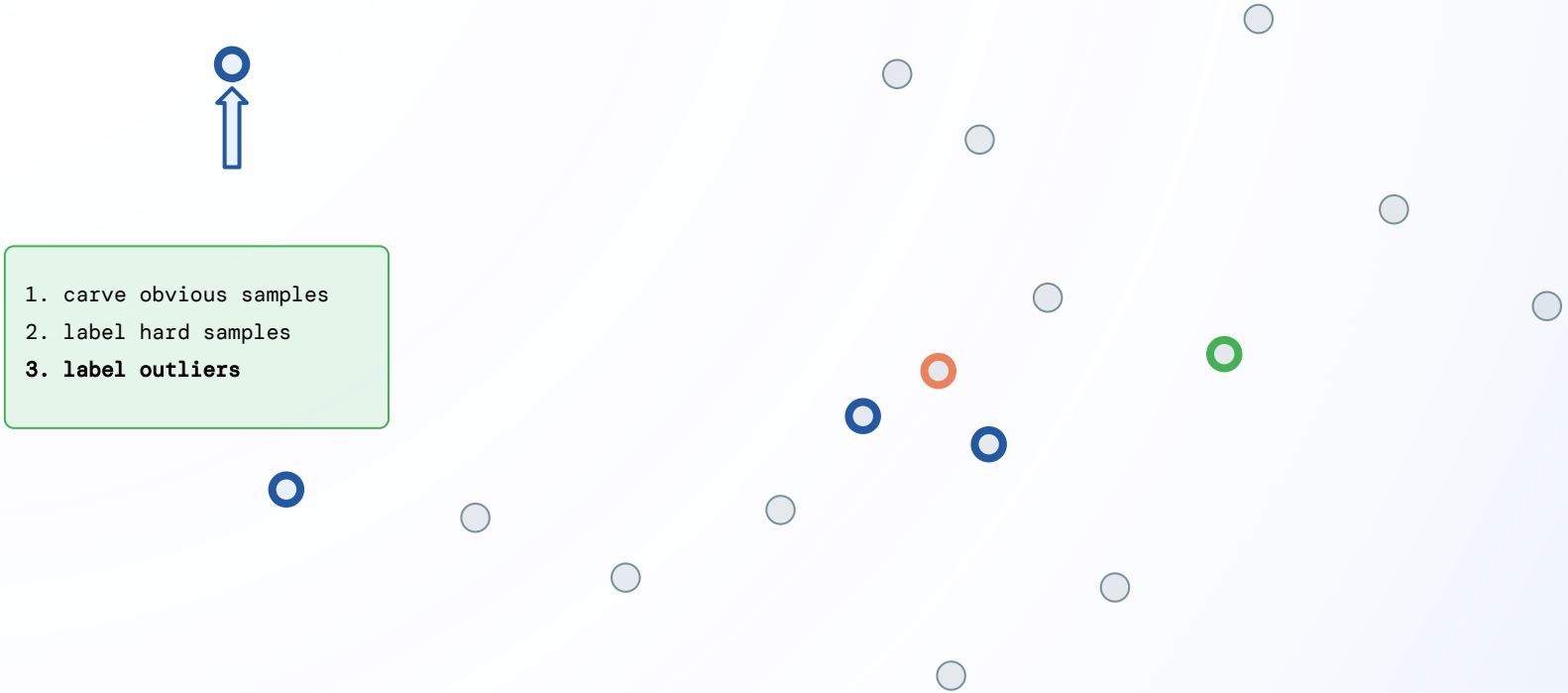


Building training data

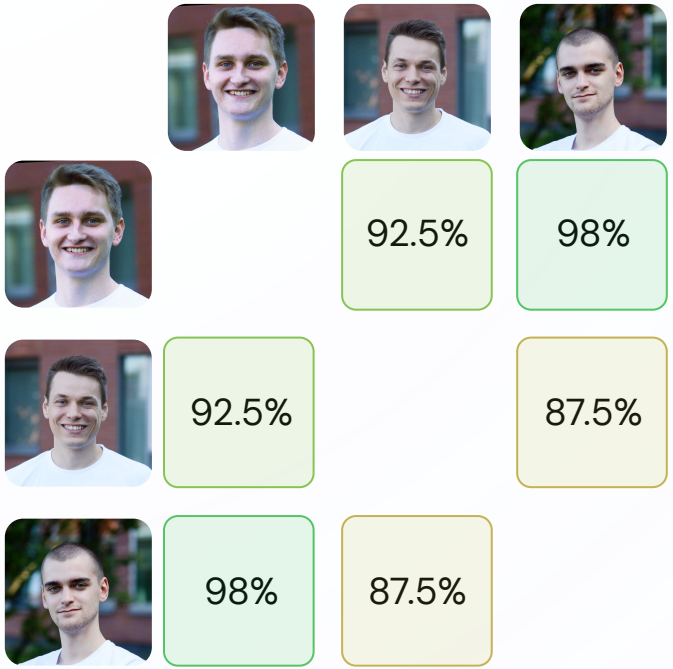
1. carve obvious samples
2. label hard samples
3. **label outliers**



Building training data



Building training data




```
def starts_with_digit(record):
    if record["headline"].text[0].is_digit:
        return "Clickbait"
```

95% precision

75% precision

Setting oauth2 up for Gmail

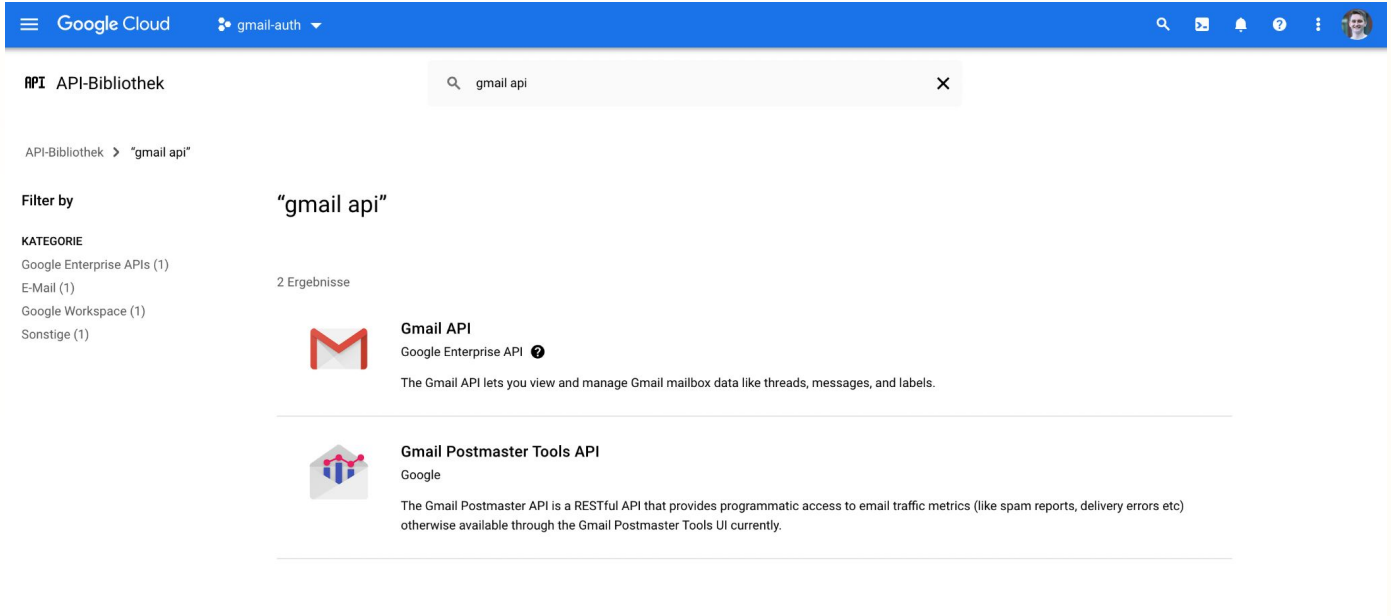
Kern

requires App in Google Cloud Platform

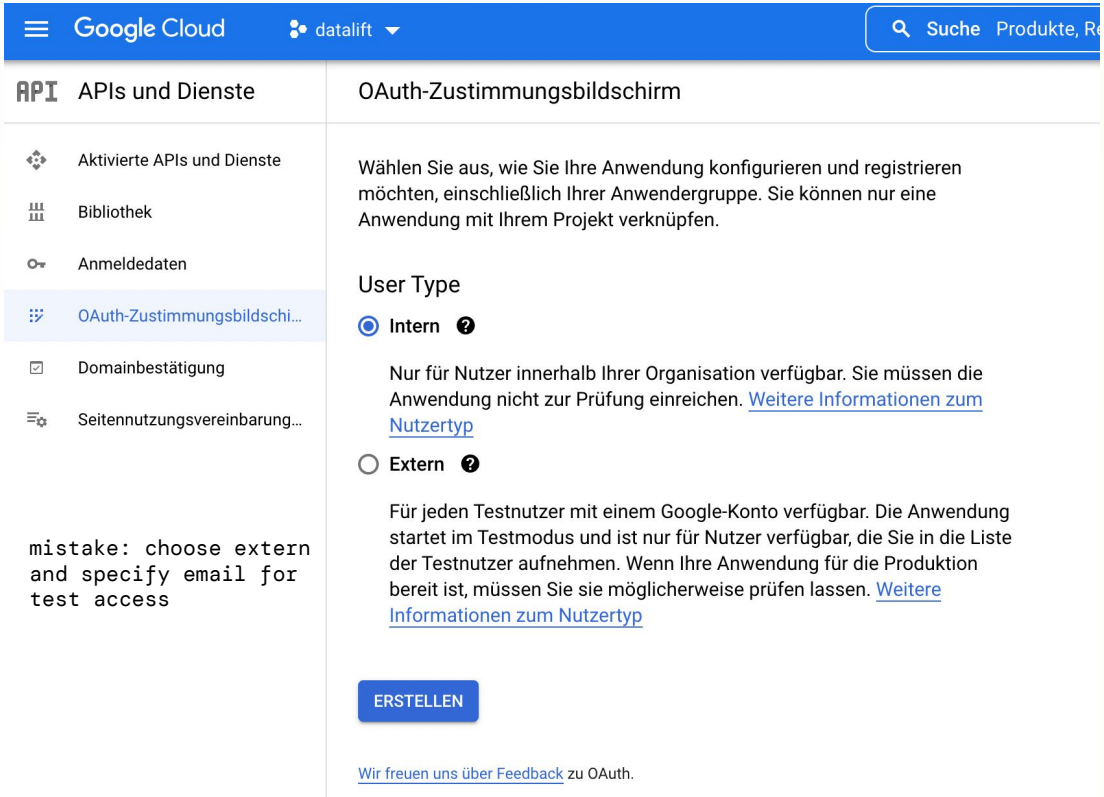
1. set up a GCP account
2. set up a new project and go to API page
3. activate GMail API in GCP via the library
4. create an oauth2 consent screen
5. set up credentials (oauth2 client ids) and download them as json file
6. write a little python script to integrate Gmail using the newly created oauth2 consent screen

Set up a new project in GCP

Activate GMail via the library

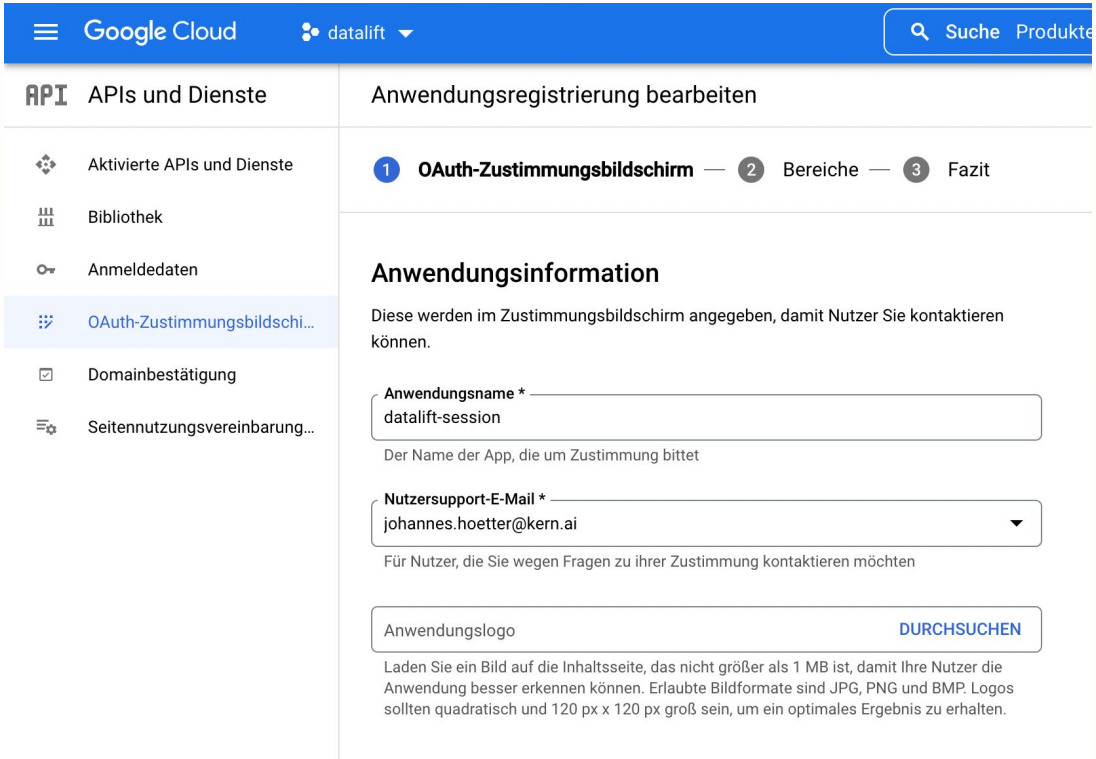


Create a consent screen

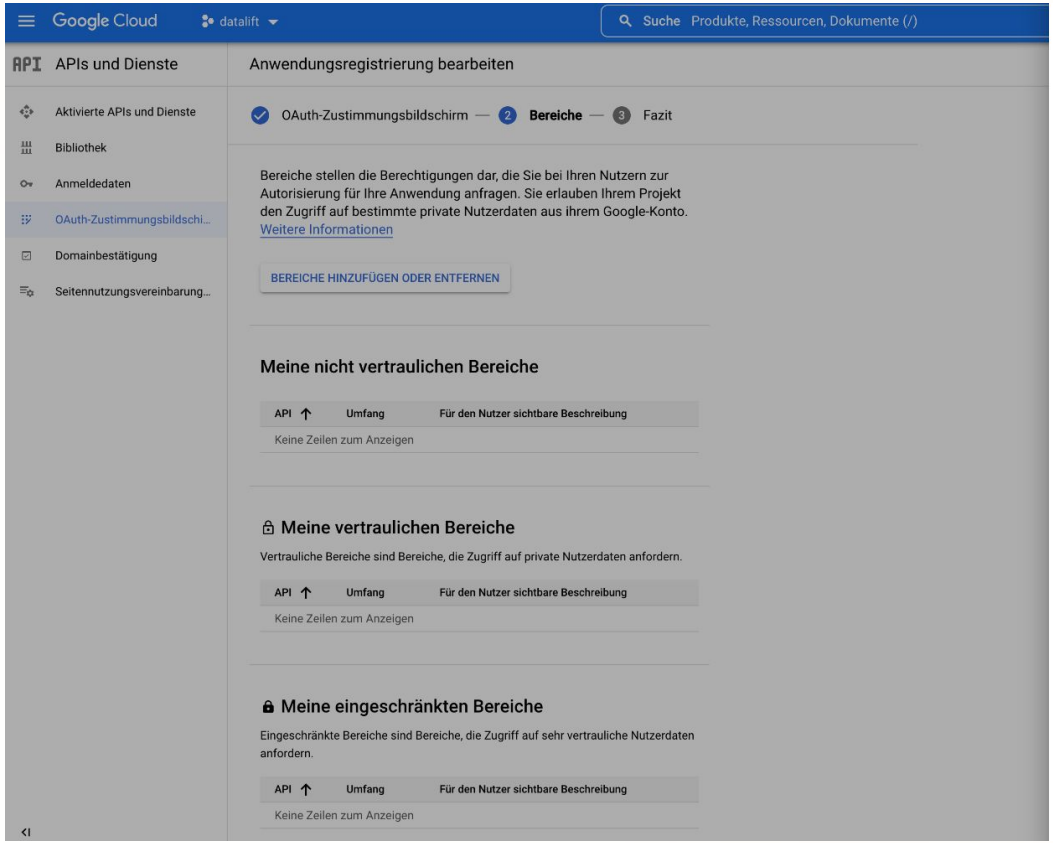


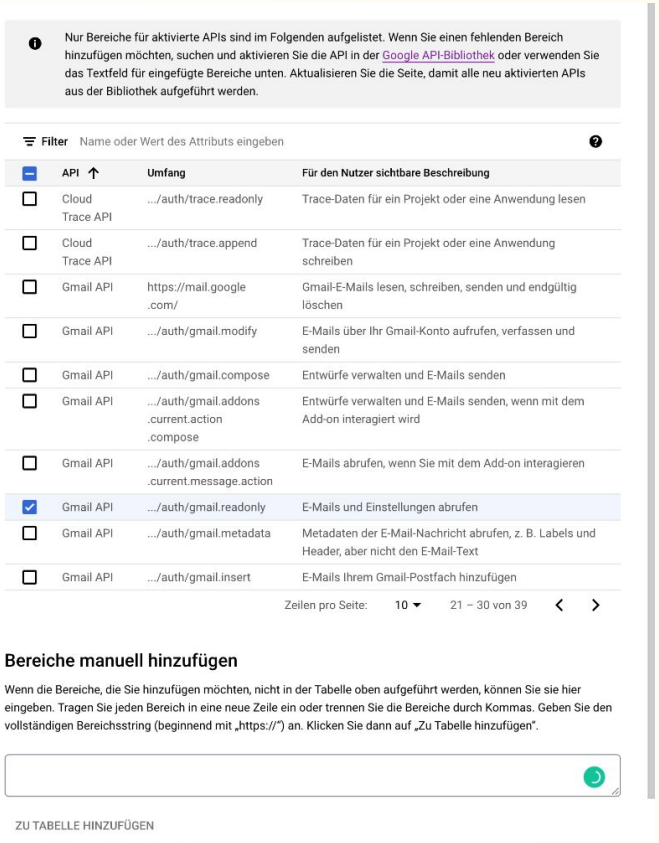
mistake: choose extern and specify email for test access

Create a consent screen

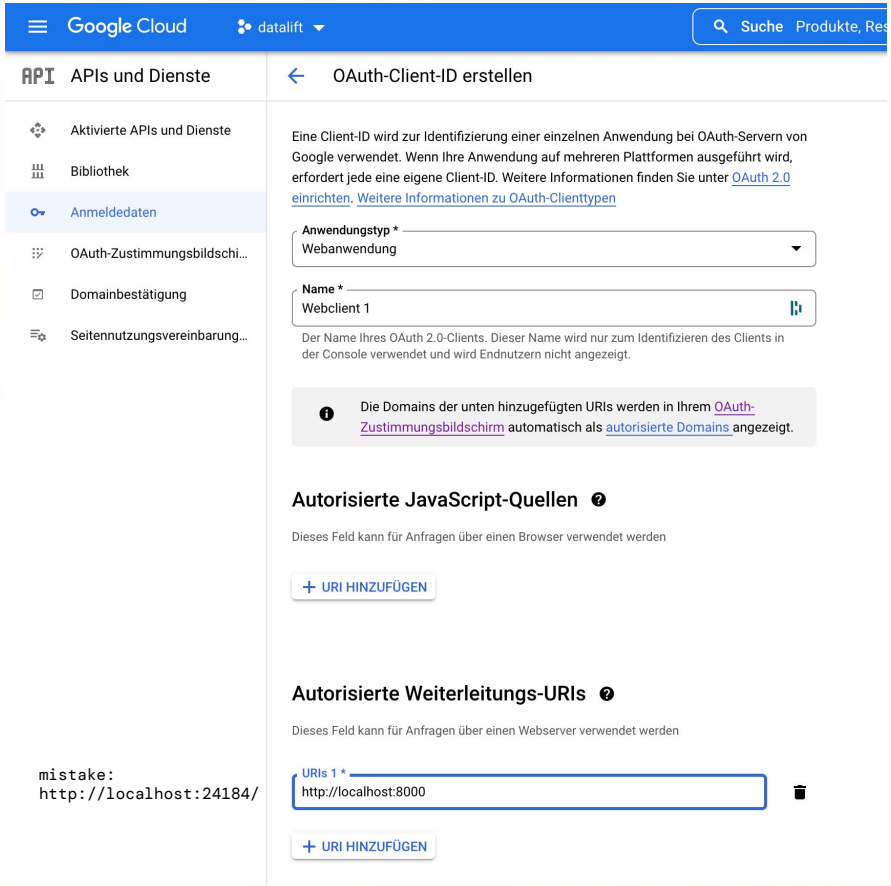


Create a consent screen



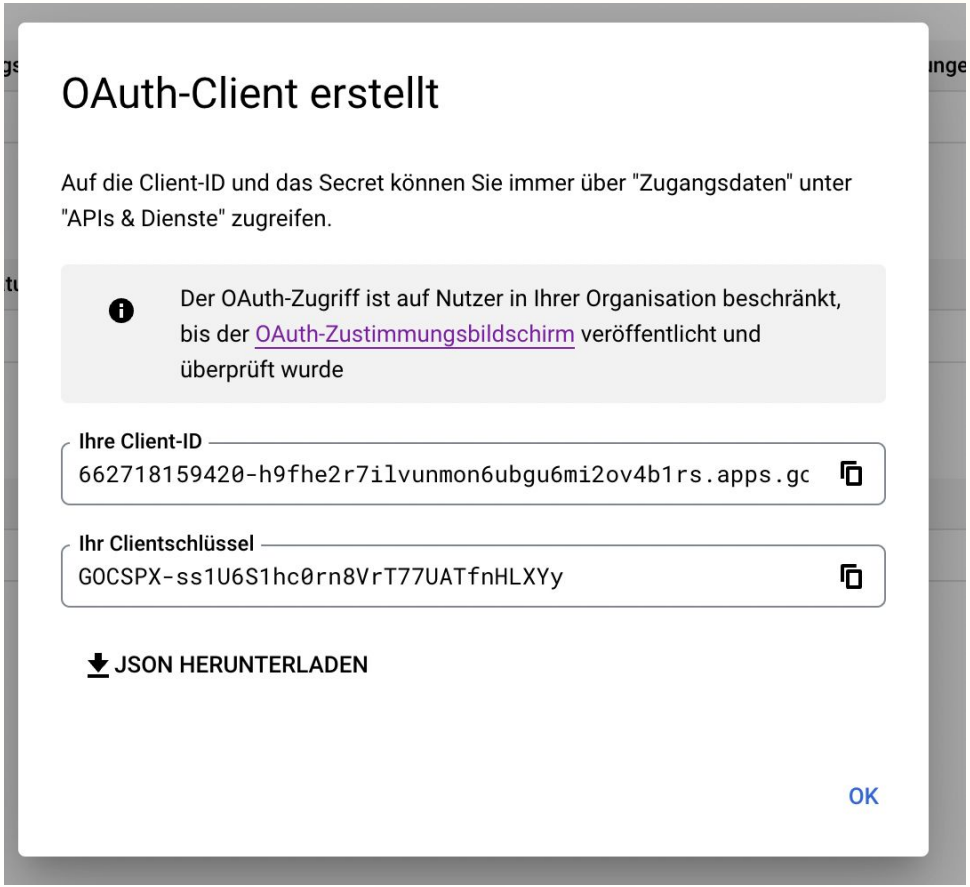


Creating credentials




```
mistake:
http://localhost:24184/
```

Creating credentials



Integrating via Python

Kern AI

app.py 2 | authentication.py

authentication.py > ...

```python
from google.auth.transport.requests import Request
from google_auth_oauthlib.flow import InstalledAppFlow

creds = None

def get_auth_creds(scopes):
    global creds
    if not creds or not creds.valid:
        if creds and creds.expired and creds.refresh_token:
            creds.refresh(Request())
        else:
            auth_with_google_and_store_creds(scopes)
    return creds


def auth_with_google_and_store_creds(scopes):
    flow = InstalledAppFlow.from_client_secrets_file(
        "client_secrets.json",
        scopes=scopes,
    )

    # port must be the port from "redirect_uris" in "client_secrets.json"
    flow.run_local_server(port=24184, prompt="consent", authorization_prompt_message="")

    global creds
    creds = flow.credentials

```

check if new creds are needed

credentials from GCP

authorize on local server

great resource: https://www.youtube.com/watch?v=vQQEaSnQ_bs&t=1079s&ab_channel=CoreySchafer

Integrating via Python

Kern AI

app.py 2 × authentication.py

app.py > ...

```python
from googleapiclient.errors import HttpError
from googleapiclient.discovery import build
from authentication import get_auth_creds
from tqdm import tqdm


SCOPES = ["https://www.googleapis.com/auth/gmail.readonly"]

if __name__ == "__main__":
    creds = get_auth_creds(SCOPES)

    mail_data = []
    try:
        # find docs for service here:
        # https://developers.google.com/resources/api-libraries/documentation/gmail/v1/python/latest/index.html
        service = build("gmail", "v1", credentials=creds)

        api_result = service.users().messages().list(userId="me", q=query).execute()
        messages = api_result["messages"]
        result_size_estimate = api_result["resultSizeEstimate"]
        for idx, message_meta in enumerate(tqdm(messages, total=result_size_estimate)):


            message_dict = {}
            message_id = message_meta["id"]
            message = (
                service.users().messages().get(userId="me", id=message_id).execute()
            )
            payload = message["payload"]
            headers = payload["headers"]
            for header in headers:
                key = header["name"]
                if key in ["Delivered-To", "Cc", "From", "Date", "Subject"]:
                    value = header["value"]
                    message_dict[key] = value
            message_dict["Snippet"] = message["snippet"]
            mail_data.append(message_dict)
            break

    except HttpError as error:
        print(f"An error occurred: {error}")

    # once everything is collected, store it to the project id; directly via DB or via API?
    print(mail_data)

```

get credentials

authorize and build API

fetch data and convert into target format

great resource: https://www.youtube.com/watch?v=vQQEaSnQ_bs&t=1079s&ab_channel=CoreySchafer