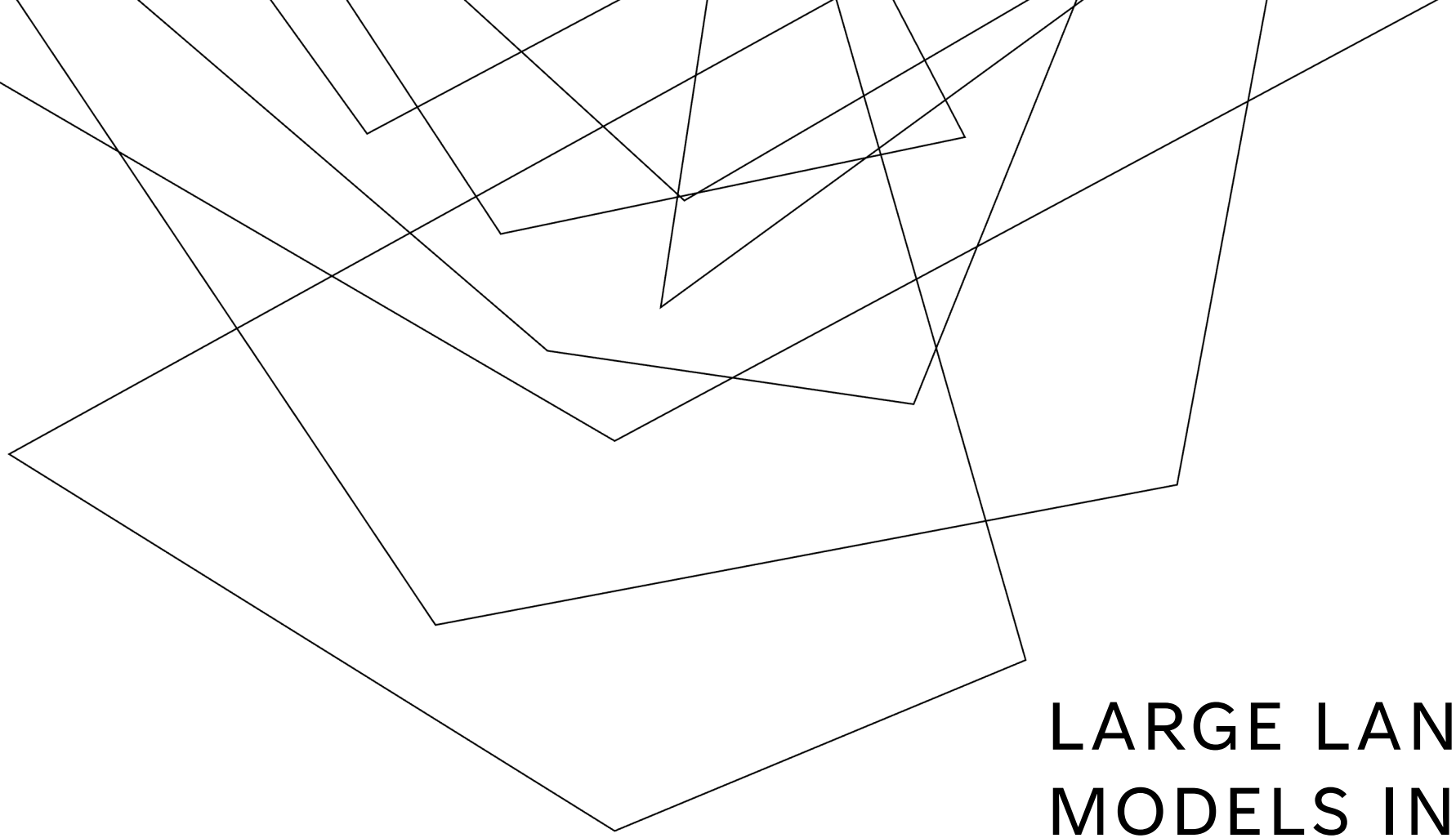


LARGE LANGUAGE MODELS IN ASSET MANAGEMENT

Lucas Bunting, CFA | October 24, 2023

AGENDA

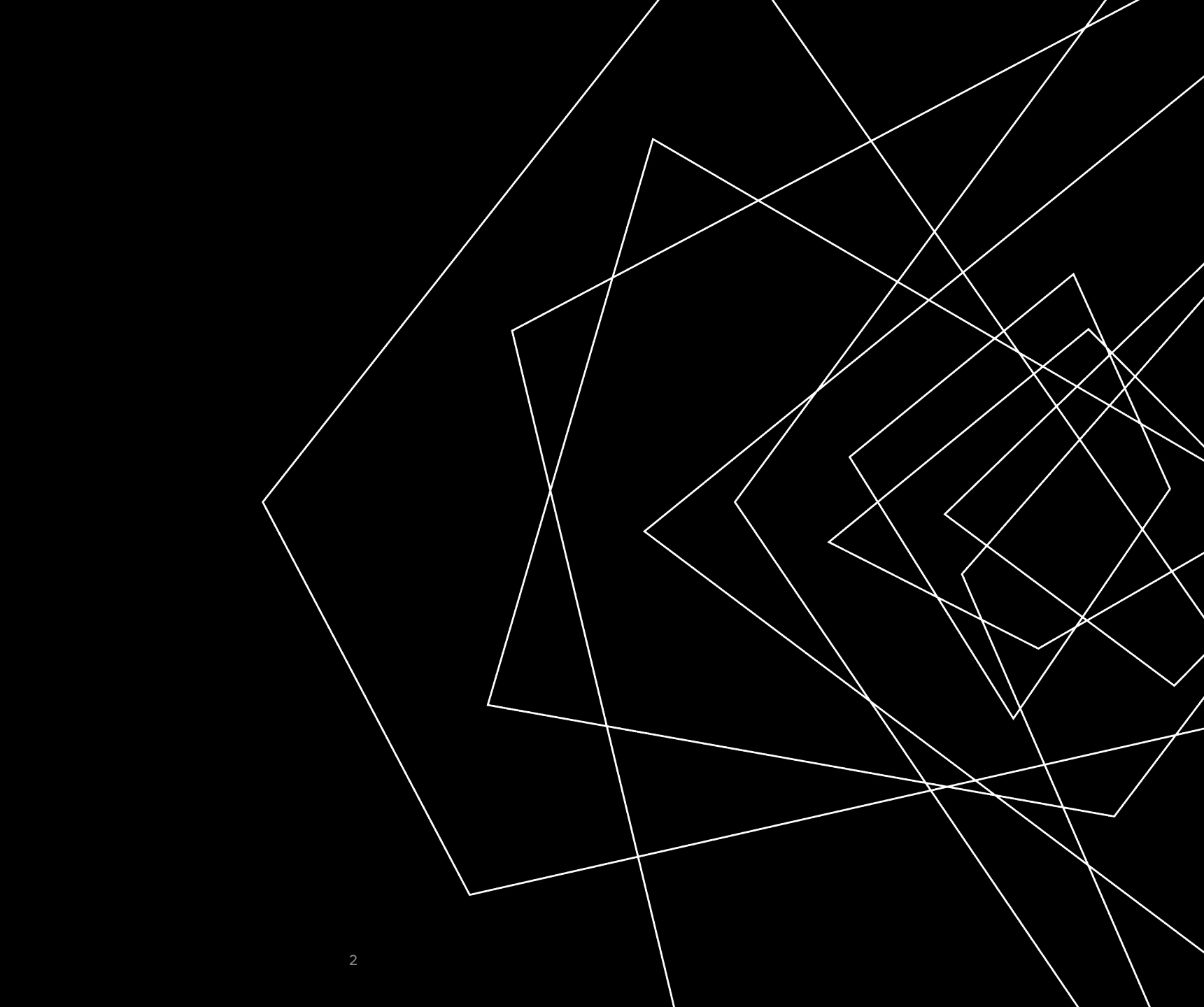
The Problem

The Workflow

Earnings Summarization

Earnings Sentiment

Best Practices & Implications



THE BOTTOM LINE AT THE TOP

This presentation is...

A practical exploration of the use of GPT 3.5 Turbo in the context of earnings season.

Meant to be a simple example that anyone with a little programming knowledge can easily implement in a short amount of time.

... not...

A lecture on the mathematics or algorithms underpinning large language models.

A production-ready system for a large company or an endorsement on the use of an LLM to produce alpha signals.

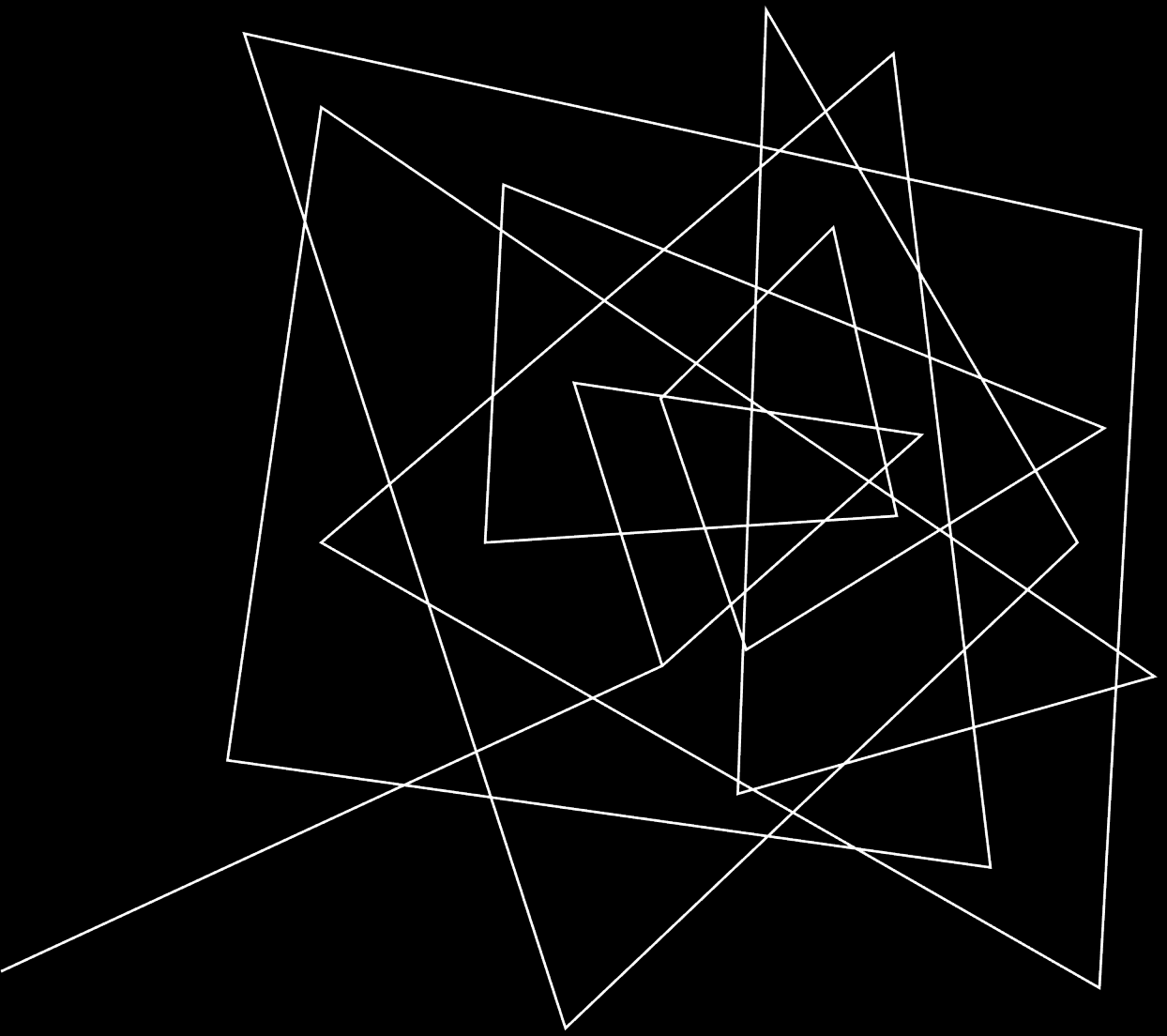
THE PROBLEM

- Deluge of unstructured text data, a fraction of which is useful
- Investors should be focusing on the future, not the past
- PMs have a responsibility to know all relevant public information of portfolio holdings and “on-deck” securities
- Analysts are responsible for knowing all information on their coverage, including developments at competitors
- Fee compression necessarily requires fundamental teams to cover more with fewer people.

You cannot read, let alone comprehend, all information available on your security universe.

Generated by DALL-E 3

“A faucet gushing out books, documents, and digital bits, with a person attempting to catch and organize them using various tools.”



THE WORKFLOW

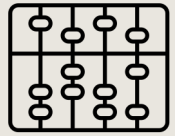
A high-level overview of the data pipeline

LARGE LANGUAGE MODELS FOR EARNINGS SEASON

Carefully constructed prompt.

Earnings reports across entire universe.

LLM hyperparameters (e.g. temperature, max response length, etc.)



Large Language Model (Local/API)

Transcript summarization

Responses

1010
1010

Sentiment estimation

THE WORKFLOW: PROMPT

`'role':'system'`

The "persona" the model adopts.

`'role':'user'`

The question posed to the model.

`'role':'assistant'`

The model's response. Can be an actual response or part of context.

A few considerations:

- What are the cost and time constraints?
- Is the `user` prompt specific enough, or is calibration required?
- What are the best hyperparameters for the use case?
- If my text is longer than the context window, how should I break up the text and reaggregate the responses?

https://platform.openai.com/playground

USE CASE #1:
EARNINGS TRANSCRIPT SUMMARIZATION

EARNINGS SUMMARIZATION: PROMPT, ONE-SHOT LEARNING

‘role’:‘system’

The “persona” the model adopts.

- Give it the persona of “intelligent stock research analyst…”
- Describe the structure of the earnings transcript
  - Management prepared remarks
  - Analyst Q&A
- Delimit the text, and ensure it only uses the text provided.

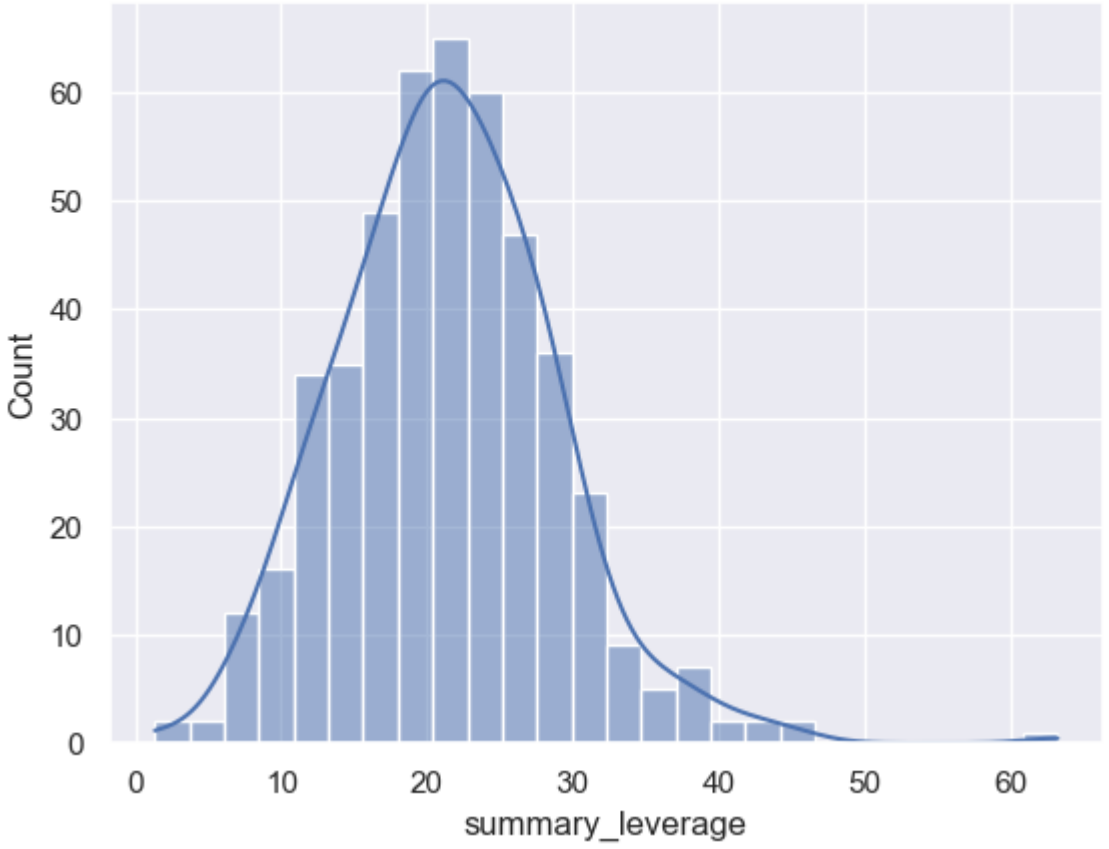
‘role’:‘user’

The question posed to the model.

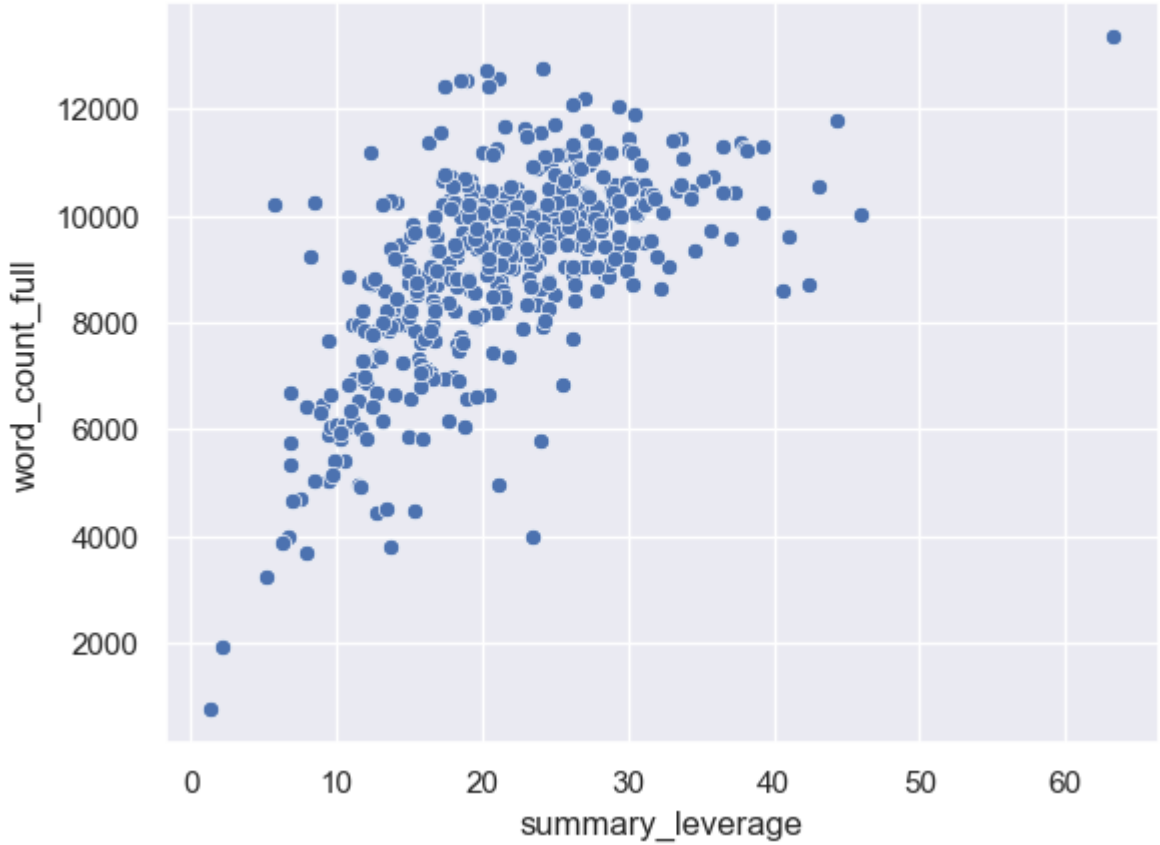
“Your job is to summarize the following transcript for the quarter {quarter} {year} earnings release by {company_name}. Pay specific attention to discussed items that impact sales, margins and earnings. Identify what the sell side analysts focused on the most in the Q&A portion, if there is one. Here is the transcript:
"""{transcript}"""”

EARNINGS SUMMARIZATION: 500 SUMMARIES

```
# histogram of leverage
sns.histplot(comparison, x='summary_leverage', kde=True);
```




```
# scatter plot between starting number of words and amount of leverage gained
sns.scatterplot(comparison, x='summary_leverage', y='word_count_full');
```



USE CASE #2:
EARNINGS SENTIMENT EXTRACTION

SENTIMENT ANALYSIS: PROMPT, FEW-SHOT LEARNING

‘role’:‘system’

The “persona” the model adopts.

- Mostly same as before: “intelligent stock research analyst…”
- I used the summaries as inputs to save cost, so I told it that was the format of the data it would receive.

‘role’:‘user’

The question posed to the model.

‘role’:‘assistant’

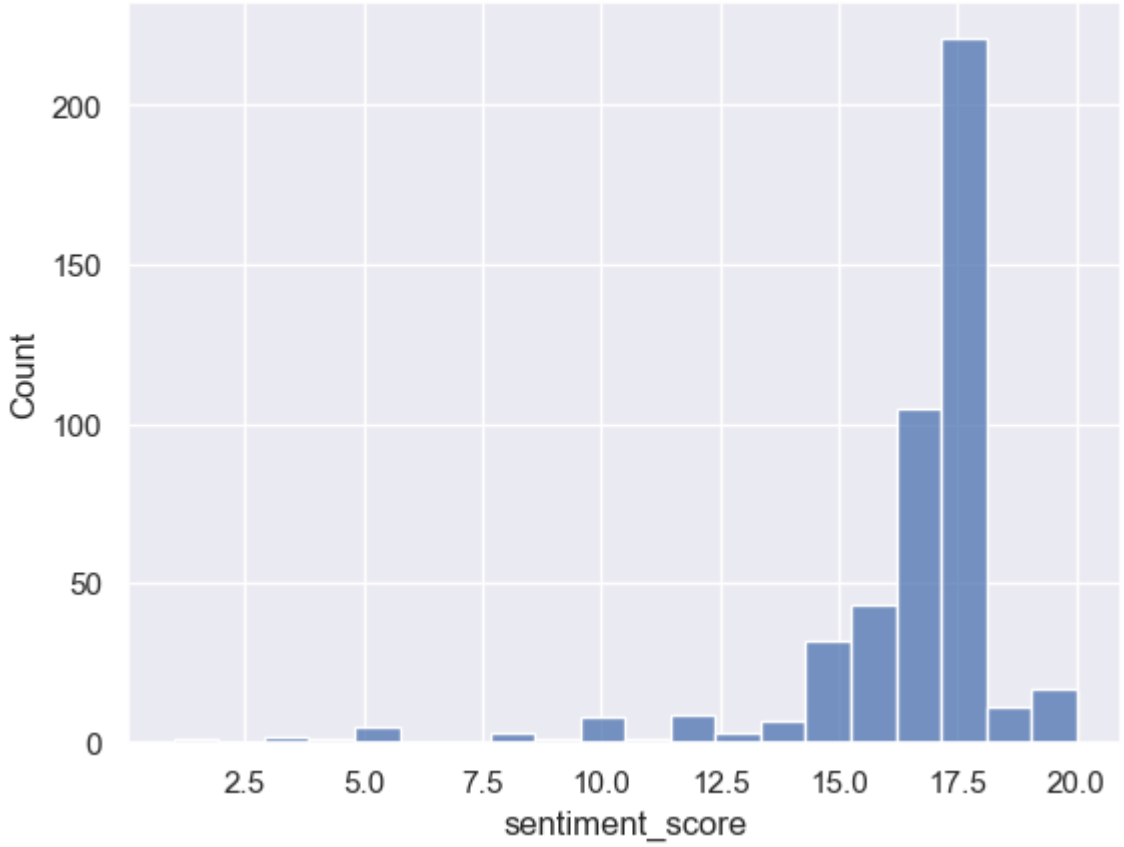
The model’s response. Can be an actual response or part of context.

```
{"role": "user", "content": calibration_init + f"""{poor_sentiment}""" },
{"role": "assistant", "content": "2"},
{"role": "user", "content": "That is a good response. I have another one for you to analyze." + calibration_init + f"""{positive_sentiment}"""’ },
{"role": "assistant", "content": "19"},
{"role": "user", "content": "That is a good response. I have another one for you to analyze."},
{"role": "assistant", "content": "Thank you. Ready when you are."}
```

>> Insert text for sentiment analysis <<

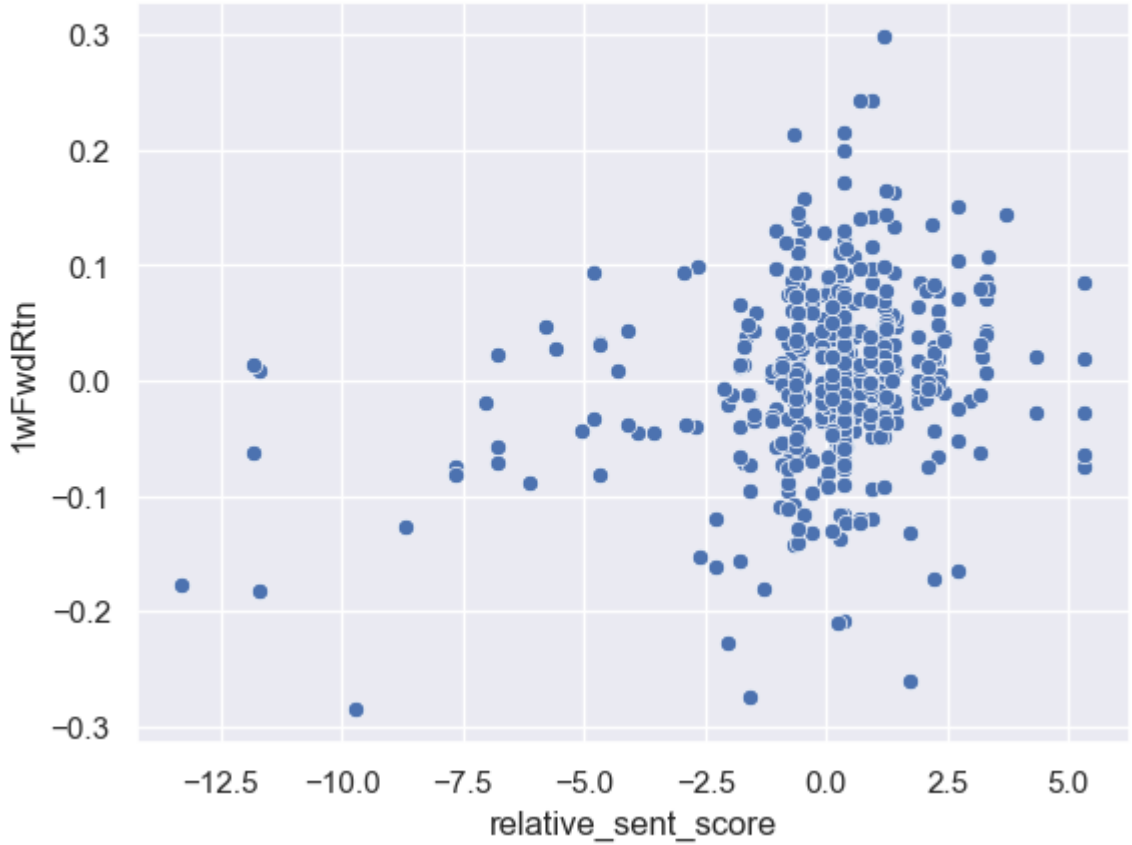
EARNINGS SENTIMENT

```
sns.histplot(df.sentiment_score, bins=20);
```




```
sns.scatterplot(df_rel, x='relative_sent_score', y='1wFwdRtn')
```

```
<Axes: xlabel='relative_sent_score', ylabel='1wFwdRtn'>
```

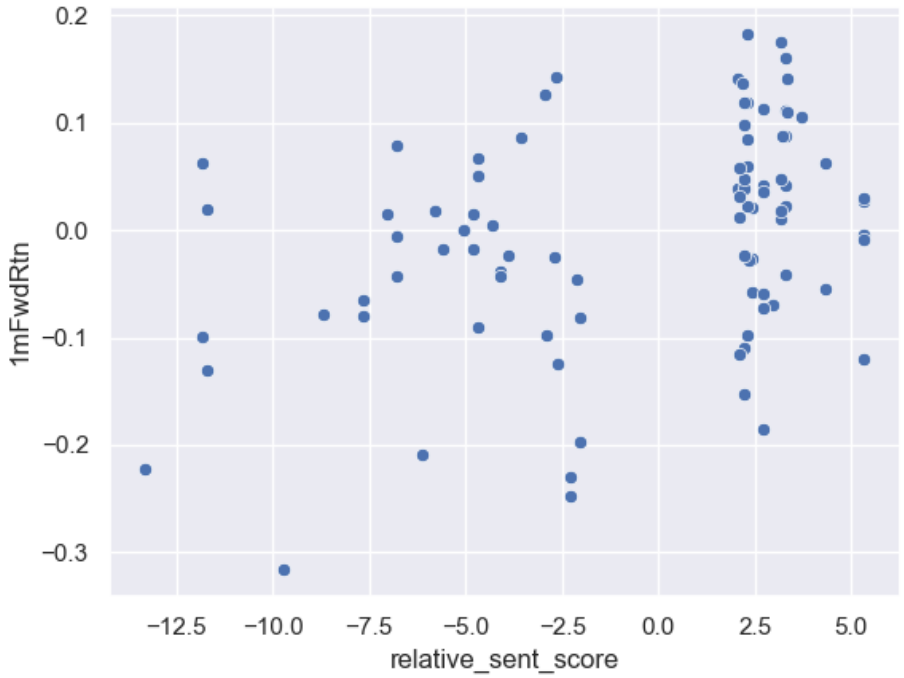


EARNINGS SENTIMENT

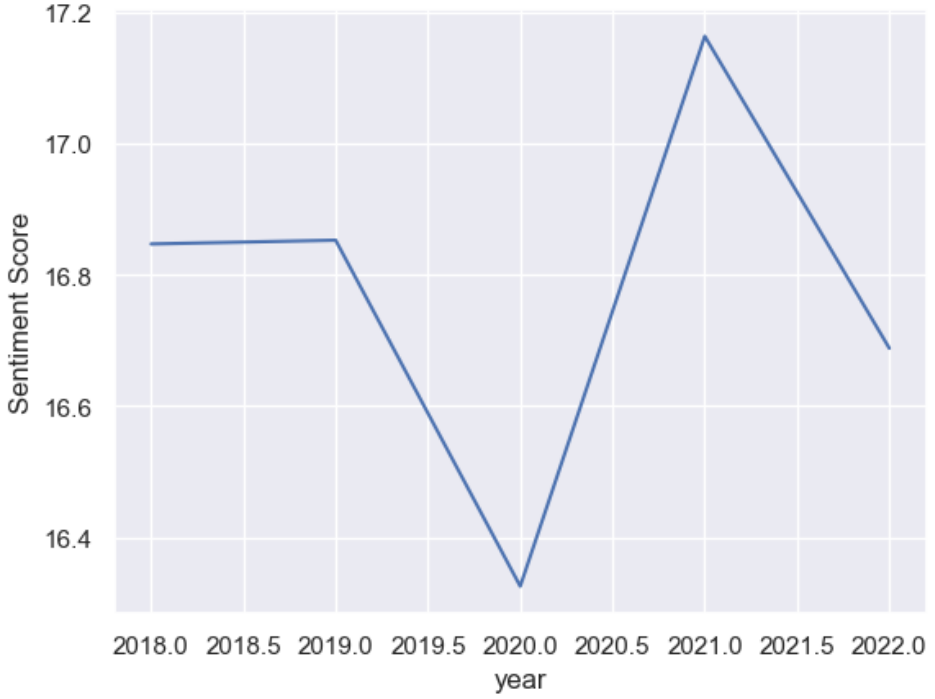
- To limit the impact of noise, we remove the middle chunk and view the tails.
- > 65% of these filtered observations were classified correctly.

```
df_masked = df_rel[(df_rel.relative_sent_score <= -2) | (df_rel.relative_sent_score >= 2)]
sns.scatterplot(df_masked, x='relative_sent_score', y='1mFwdRtn')
```

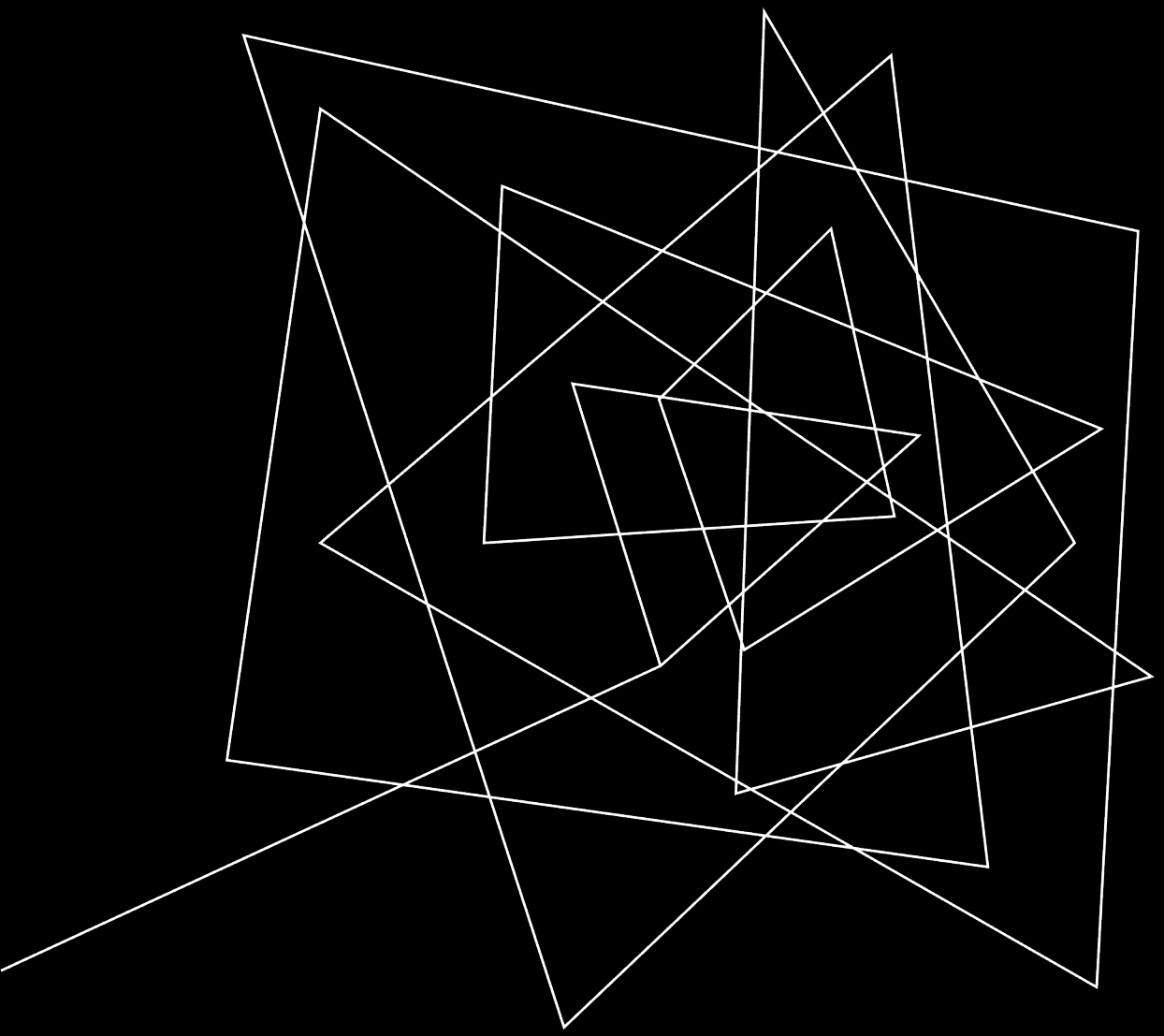
```
<Axes: xlabel='relative_sent_score', ylabel='1mFwdRtn'>
```




```
df.groupby('year')['sentiment_score'].agg(np.mean).plot(ylabel='Sentiment Score');
```



- Warning: This is not robust to lookahead bias, as the mean used to compute relative sentiment score uses the whole period.
- You decide: Are LLMs reliable for this task?



BEST PRACTICES & IMPLICATIONS

A few things to keep in mind while developing your own workflow.

KEEP IT SIMPLE, KEEP IT CHEAP

Most tasks can be done with the simple setup I showed.

You do not need vector embeddings unless you have an ocean of text.

Embedding technology is rapidly changing; you will be forced to restructure multiple times.

Also, fine-tuning a model is a supervised learning process, i.e. it can be extremely costly and tedious to label your data and may defeat the purpose of using the LLM.

This is extremely cheap, even for an individual.

Left: Transcript Summary

Middle and Right: Two versions of Sentiment

Total: < $40

MORE COMPUTE

Ask for reasoning.

- If the prompt includes direction to explain each step of a process, you effectively give the model more compute to likely generate a more robust answer.
- “Briefly summarize how to bake a cake.” vs. "Explain in detail the steps involved in baking a chocolate cake, including the ingredients, quantities, procedure, and the science behind each step.“
  - First prompt response: recipe ~150 words
  - Second prompt response: a dissertation on cake baking ~500 words

Call a function.

- The code generating abilities of LLMs are generally well-regarded.
- If you give a well-detailed prompt, you can have it generate code, or specify certain functions you have defined on your machine.
- Quick flow:
  - Unstructured text data comes in, give it to LLM to analyze.
  - Have LLM generate the name of the function that you want called on that text.
    - Stores in directory
    - Email to group
    - Run through sentiment analysis
    - Etc.

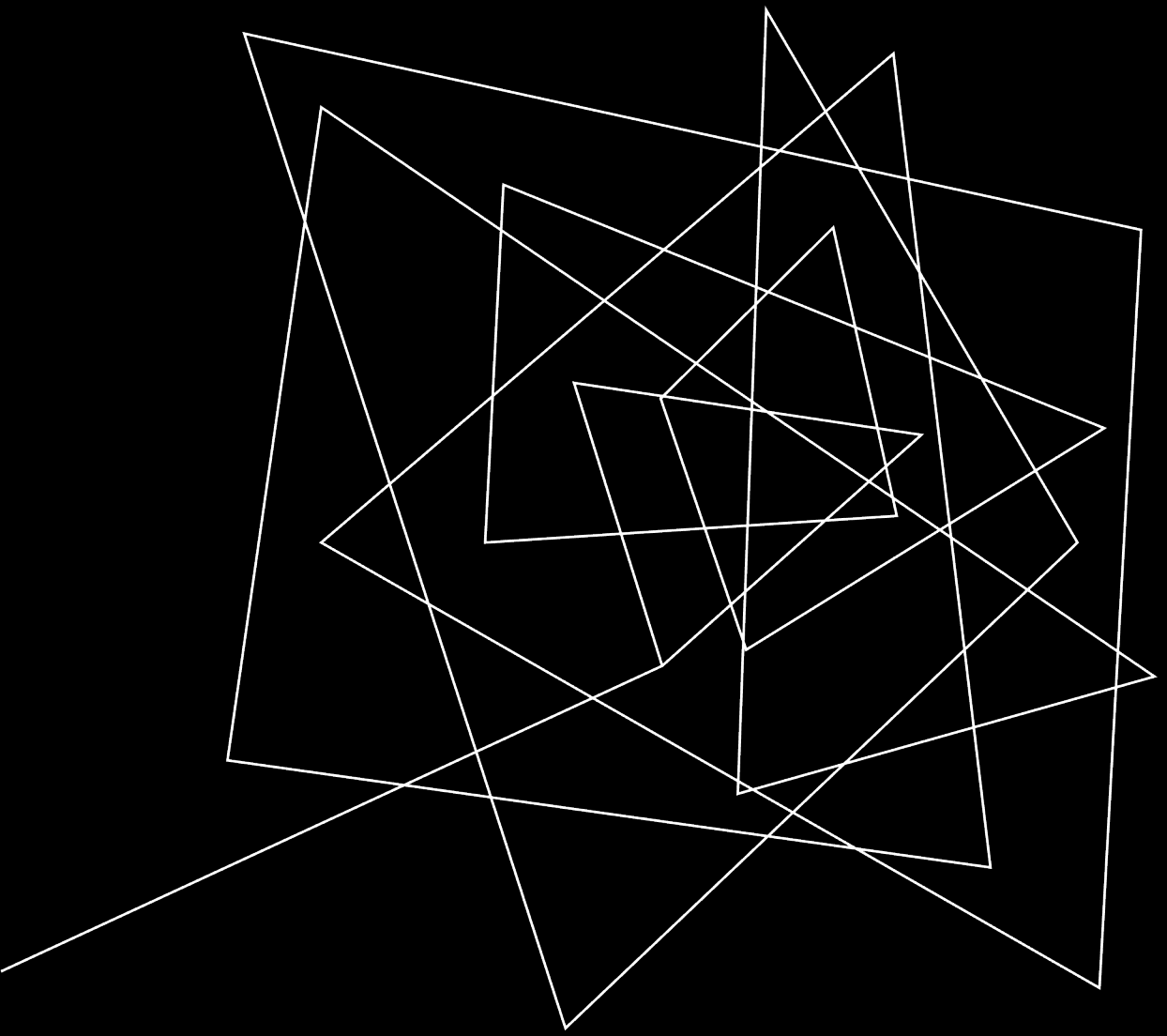
A FEW TAKEAWAYS FOR OUR INDUSTRY

- The barriers to entry for NLP dataset providers have been decimated.
  - All investors now have cheap access to datasets that will be unique and potentially useful.
  - At the very least, you have something to compare when evaluating alternative datasets.
- AI is hyped, and potentially overhyped/misunderstood by management teams.
  - You understand a few simple use cases.
  - As you interview management teams, you may be able to detect what is reliably attainable at this stage in the game.
- A roadmap of “riskiness” is likely the best way forward for your company.
  - Rank projects in order of risk (failure, data availability, cost/time)
  - As you go from less to more risky, the complexity of the workflow will increase, but having experience tackling easier problems is helpful.

SUMMARY

- LLMs are easy to use.
- LLMs are infinitely customizable.
- Simple is better for now.
- Summarization is a great use case.
- Sentiment analysis could be useful to some.
- We are in the earliest innings of this technology. Keep an eye on it, don’t bet the farm.

Generated by DALL-E 3; “An ocean made up of binary code depicted in watercolor, where a small boat struggles to navigate through the waves of information.”



APPENDIX

PRECAUTIONS

“AI gives you infinite interns”

Jesse Blocher, Director of Graduate Studies, Data Science Institute @ Vanderbilt

- If you ask the question poorly, don’t be surprised when you get something weird in return.
- Scalability is the obvious advantage, but you don’t want to scale crap.
- Data is important, as is the way you ask the question.

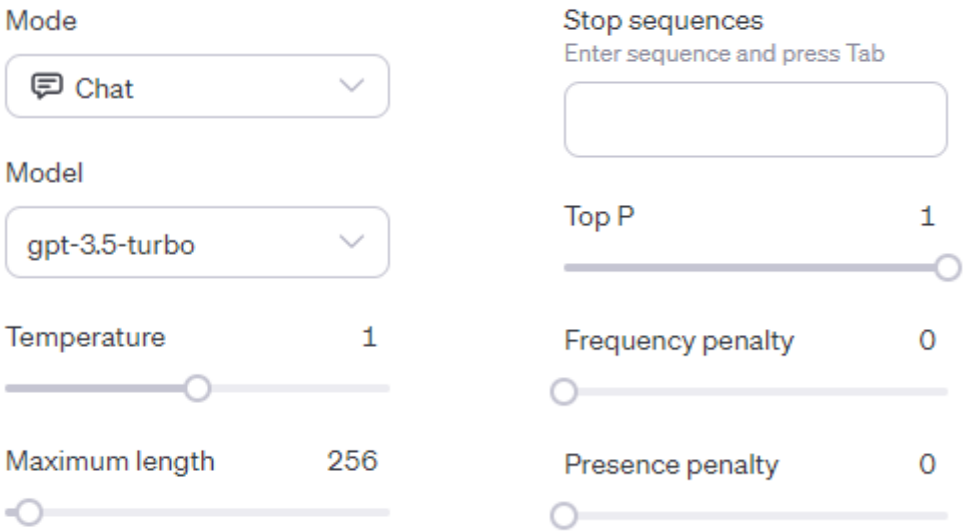
LLMs are more useful to those who know what “good” looks like.

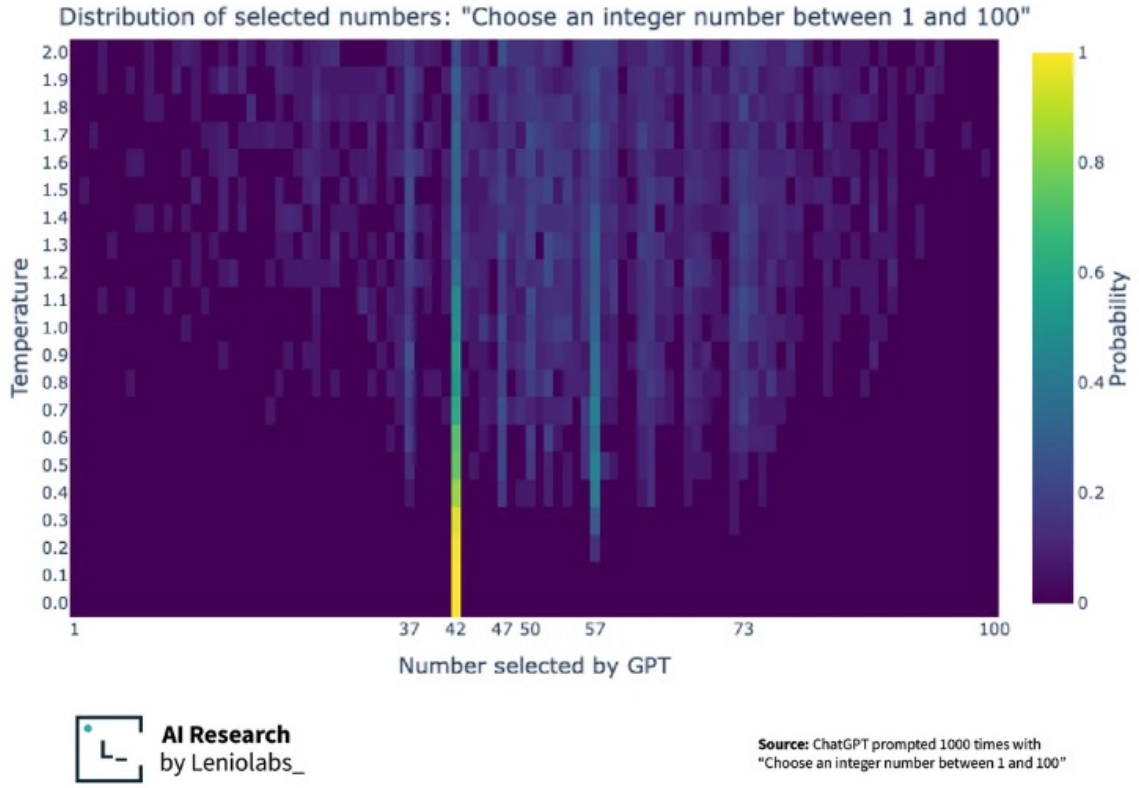
- LLMs can return code (it’s just text data). It does not mean it will achieve what you want or that it will even work.
- If you don’t know how to program, you will have a heck of a time debugging code you receive from an LLM. If you know the language well, it can level up your game.

“Everything on the internet is true.”

- If you take a response from a model at face value, you are indirectly assuming everything written on the internet is true.
- Related note: Contrary to popular belief, even the largest LLMs haven’t seen the whole internet.

HYPERPARAMETERS





AI Research
by Leniolabs_

Source: ChatGPT prompted 1000 times with "Choose an integer number between 1 and 100"

SECURITY CONCERNS

Our commitments

https://openai.com/enterprise-privacy

Ownership: You own and control your data

- ✓ We do *not* train on your data from ChatGPT Enterprise or our API Platform
- ✓ You own your inputs and outputs (where allowed by law)
- ✓ You control how long your data is retained (ChatGPT Enterprise)

Control: You decide who has access

- ✓ Enterprise-level authentication through SAML SSO
- ✓ Fine-grained control over access and available features
- ✓ Custom models are yours alone to use, they are not shared with anyone else

Security: Comprehensive compliance

- ✓ We've been audited for SOC 2 compliance
- ✓ Data encryption at rest (AES-256) and in transit (TLS 1.2+)
- ✓ Visit our Trust Portal to understand more about our security measures

- Additionally, cloud players like Microsoft have added copyright protection to the outputs of their model, so if any third party sues you for copyright infringement, Microsoft defends you.
  Microsoft announces new Copilot Copyright Commitment for customers - Microsoft On the Issues