Name : Septina Winar Syahputri Semmler
Matriculation : 37010019
Module : Quantitative Methods in Information System; MADS1
Programm : Master - Applied Business on Data Science; ADS

These scripts and documentation are produced to answer the MADS1 exam with LDA topic with gensim based on Kaggle dataset.
Contain of folders and scripts with explanation:

Folder:	**scripts**:
data	requirements.txt
result	full.py
models	lda.py
pyldavis	Semmler_Exam_Mads1.pdf
demo	tfidf_vectorizer.pkl

Important!

full.py:

Duration	: +-4hr
Goal	: run all pipeline
content	: full python code script to produce and run entire workflow from beginning to end
Memories to create	: 2.95 gb
File to produce and save	: all.texts, cleaned_texts, corpus_texts, lda model(5, 10, 15, 20), pyldavis, lda_metrics

Semmler_Exam_MADS1 folder /	
Full.py	: Script to run all workflow from beginning to end
Lda.py	: Script to run topic distribution function
Requirements.txt	: Lists all Python libraries needed
Semmler_Exam_Mads1.pdf	: PDF explanation for every steps

Data folder /	
All_texts.txt	: complete raw article texts used as input for preprocessing and LDA topic modeling

Models folder /	
Gensim.log	: log file containing training progress and messages from the Gensim LDA process
Lda_10_model	: saved Gensim LDA model file containing learned 10 topic distributions
Lda_10_model.expElogbeta.npy	: NumPy array storing the learned topic–word probability matrix
Lda_10_model.id2word	: mapping word IDs to actual words used by the model during training
Lda_10_model.state	: model state file containing statistics needed to training or update the model

Pyldavis folder /	
Lda_5.html	: visualization of LDA topic model results with 5 topics
Lda_10.html	: visualization of LDA topic model results with 10 topics
Lda_15.html	: visualization of LDA topic model results with 15 topics
Lda_20.html	: visualization of LDA topic model results with 20 topics

Results folder /	
Cleaned_texts.jsonl	: contains preprocessed articles with tokens and ids in JSON format
Corpus_texts.txt	: text file from tokenized file used as input for vectorization and LDA training
Topics.json	: stored topic distributions per document in JSON format to later use in html page
Lda_metrics.json	: evaluation metrics to assess LDA model performance

Demo folder /	
Index.html	: interface script that displays the LDA demo, topic distribution viewer, visualizations, and PDF explanation
Cleaned_text.png	: visualization processed and cleaned version of the text after preprocessing steps
Heatmap.png	: heatmap visualization top words and their weights across different LDA topics
Raw_text.png	: visualization representing original, unprocessed raw text data
demo_cleaned.jsonl	: JSON file containing cleaned and tokenized 0 - 9999 articles used for the web demo
demo_corpus.txt	: text file from tokenized file from 0 - 9999 articles used for the web demo
demo_topics.json	: stored topic distributions from per 0 - 9999 articles used for the web demo

lda.py

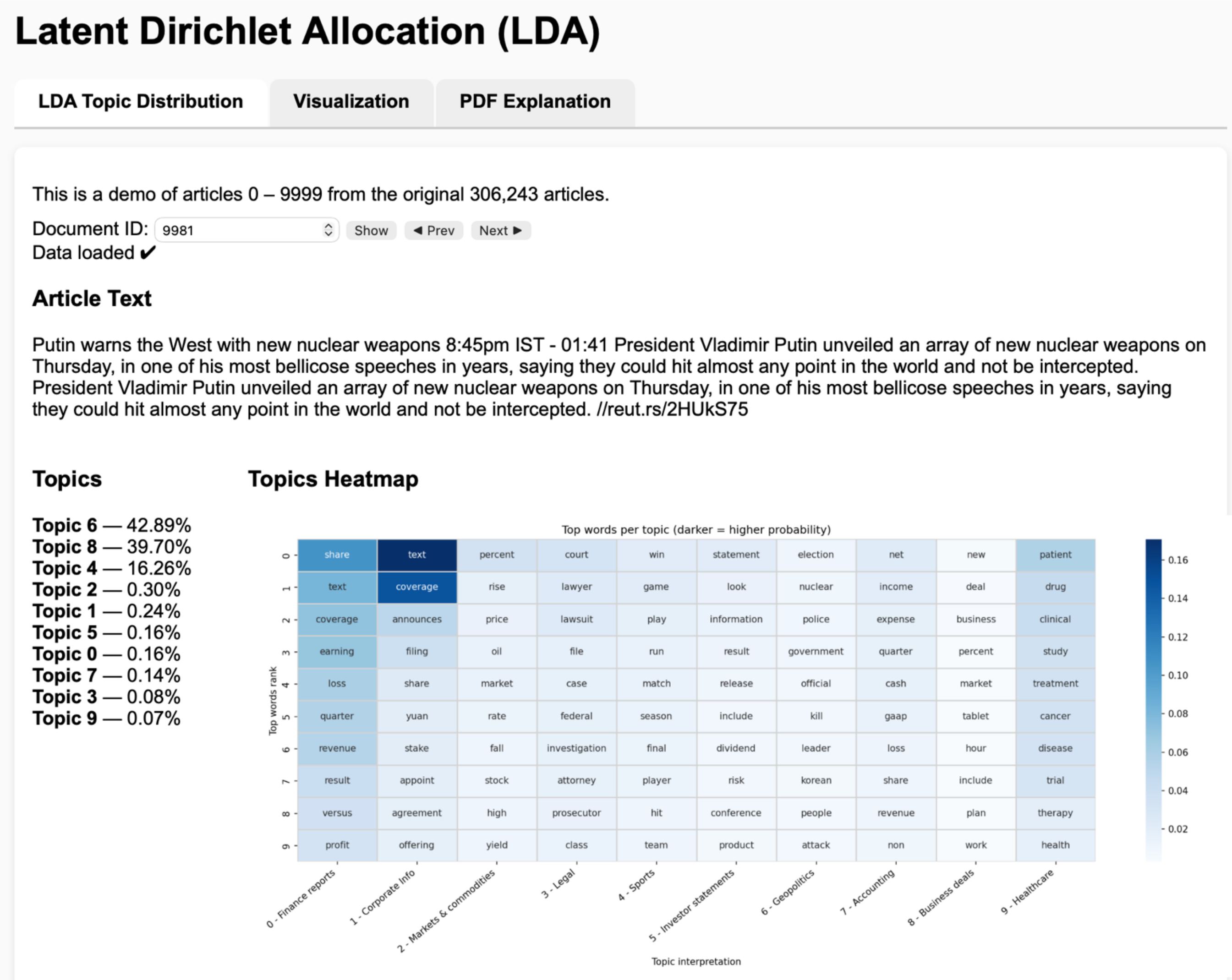
Duration	: +-
Goal	: run def **demo_get_topic** function to **find topic distribution** per document 0 – 9999 as demo
content	: script to run topic distribution function
File to produce and save	: no new file produce or save here

HTML Pages:

https://wsemmler.github.io/Exam_MADS1/

serves as an interactive web interface for exploring LDA topic modeling on a demo subset of articles. **Contains 3 tabs**:

- LDA Topic Distribution – lets users enter a document ID (0–9999 demo range) to see the article text, its topic probabilities, and a topics heatmap (heatmap.png).
- Visualization – shows static images of the raw (raw_text.png) and cleaned text (cleaned_text.png), and an interactive LDA visualization (lda_10.html) embedded in an iframe.
- PDF Explanation – embeds a PDF with a zoomed-in display for reading the project explanation.



This is a demo of articles 0 – 9999 from the original 306,243 articles.

Document ID: 9981

Data loaded ✔

Article Text

Putin warns the West with new nuclear weapons 8:45pm IST - 01:41 President Vladimir Putin unveiled an array of new nuclear weapons on Thursday, in one of his most bellicose speeches in years, saying they could hit almost any point in the world and not be intercepted. President Vladimir Putin unveiled an array of new nuclear weapons on Thursday, in one of his most bellicose speeches in years, saying they could hit almost any point in the world and not be intercepted. //reut.rs/2HUkS75

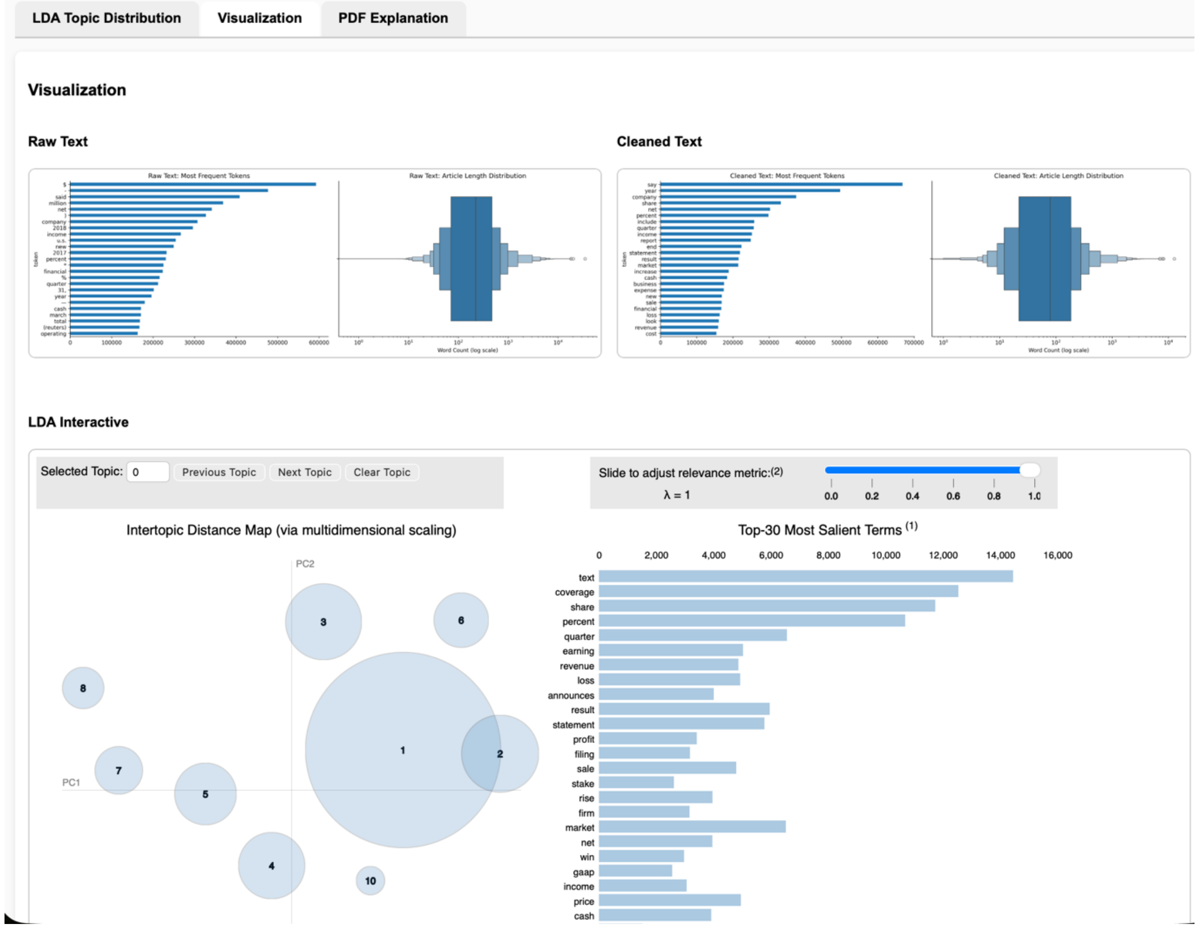
Topics

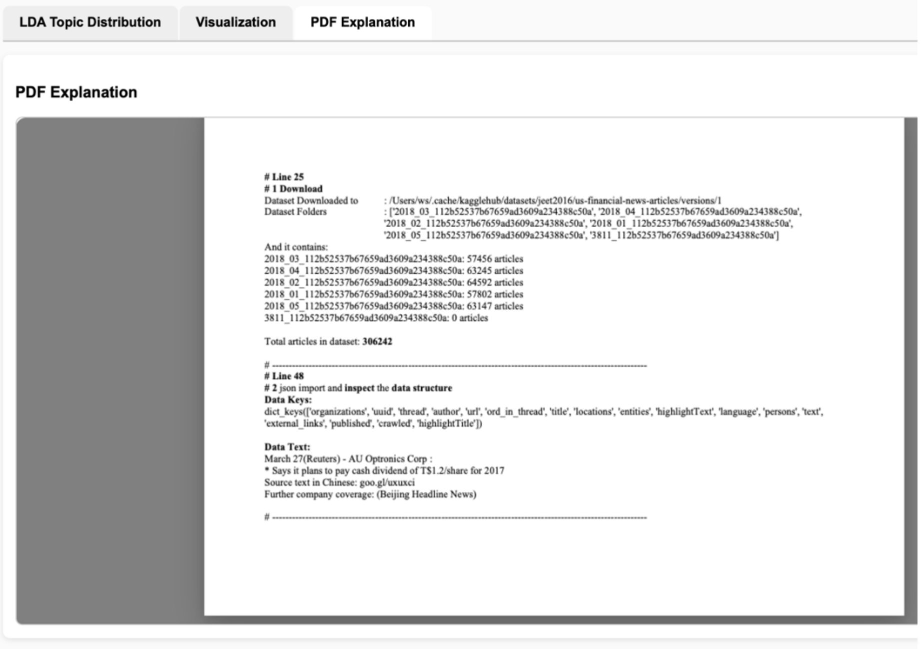
Topic 6 — 42.89%
Topic 8 — 39.70%
Topic 4 — 16.26%
Topic 2 — 0.30%
Topic 1 — 0.24%
Topic 5 — 0.16%
Topic 0 — 0.16%
Topic 7 — 0.14%
Topic 3 — 0.08%
Topic 9 — 0.07%

Topics Heatmap

Top words per topic (darker = higher probability)

	0 - Finance reports	1 - Corporate Info	2 - Markets & commodities	3 - Legal	4 - Sports	5 - Investor statements	6 - Geopolitics	7 - Accounting	8 - Business deals	9 - Healthcare
0	share	text	percent	court	win	statement	election	net	new	patient
1	text	coverage	rise	lawyer	game	look	nuclear	income	deal	drug
2	coverage	announces	price	lawsuit	play	information	police	expense	business	clinical
3	earning	filing	oil	file	run	result	government	quarter	percent	study
4	loss	share	market	case	match	release	official	cash	market	treatment
5	quarter	yuan	rate	federal	season	include	kill	gaap	tablet	cancer
6	revenue	stake	fall	investigation	final	dividend	leader	loss	hour	disease
7	result	appoint	stock	attorney	player	risk	korean	share	include	trial
8	versus	agreement	high	prosecutor	hit	conference	people	revenue	plan	therapy
9	profit	offering	yield	class	team	product	attack	non	work	health





# Line 25
# 1 Download

```
Dataset Downloaded to        : /Users/ws/.cache/kagglehub/datasets/jeet2016/us-financial-news-articles/versions/1
Dataset Folders              : ['2018_03_112b52537b67659ad3609a234388c50a', '2018_04_112b52537b67659ad3609a234388c50a',
                               '2018_02_112b52537b67659ad3609a234388c50a', '2018_01_112b52537b67659ad3609a234388c50a',
                               '2018_05_112b52537b67659ad3609a234388c50a', '3811_112b52537b67659ad3609a234388c50a']
And it contains:
2018_03_112b52537b67659ad3609a234388c50a: 57456 articles
2018_04_112b52537b67659ad3609a234388c50a: 63245 articles
2018_02_112b52537b67659ad3609a234388c50a: 64592 articles
2018_01_112b52537b67659ad3609a234388c50a: 57802 articles
2018_05_112b52537b67659ad3609a234388c50a: 63147 articles
3811_112b52537b67659ad3609a234388c50a: 0 articles

Total articles in dataset: 306242

# ------------------------------------------------------------------------------------------------
```

# Line 48
# 2 json import and **inspect** the **data structure**
Data Keys:

```
dict_keys(['organizations', 'uuid', 'thread', 'author', 'url', 'ord_in_thread', 'title', 'locations', 'entities', 'highlightText', 'language', 'persons', 'text',
'external_links', 'published', 'crawled', 'highlightTitle'])
```

Data Text:

```
March 27(Reuters) - AU Optronics Corp :
* Says it plans to pay cash dividend of T$1.2/share for 2017
Source text in Chinese: goo.gl/uxuxci
Further company coverage: (Beijing Headline News)

# ------------------------------------------------------------------------------------------------
```

Below are the steps taken to produce all files and functions written above to answer exam instructions:

1. Please download the US Financial News Articles" dataset from Kaggle (~1.2 GB)

# Line 25
# 1 Download
Dataset Downloaded to : /Users/ws/.**cache**/kagglehub/datasets/jeet2016/us-financial-news-articles/versions/1
Dataset Folders : ['2018_03_112b52537b67659ad3609a234388c50a', '2018_04_112b52537b67659ad3609a234388c50a', '2018_02_112b52537b67659ad3609a234388c50a', '2018_01_112b52537b67659ad3609a234388c50a', '2018_05_112b52537b67659ad3609a234388c50a', '3811_112b52537b67659ad3609a234388c50a']

And it contains:
2018_03_112b52537b67659ad3609a234388c50a: 57456 articles
2018_04_112b52537b67659ad3609a234388c50a: 63245 articles
2018_02_112b52537b67659ad3609a234388c50a: 64592 articles
2018_01_112b52537b67659ad3609a234388c50a: 57802 articles
2018_05_112b52537b67659ad3609a234388c50a: 63147 articles
3811_112b52537b67659ad3609a234388c50a: 0 articles

Total articles in dataset: **306242**

---

2. Familiarize yourself with Json file imports (read the docs).

# Line 48
# 2 json import and **inspect** the **data structure**
Data Keys:
dict_keys(['organizations', 'uuid', 'thread', 'author', 'url', 'ord_in_thread', 'title', 'locations', 'entities', 'highlightText', 'language', 'persons', 'text', 'external_links', 'published', 'crawled', 'highlightTitle'])

Data Text:
March 27(Reuters) - AU Optronics Corp :
* Says it plans to pay cash dividend of T$1.2/share for 2017
Source text in Chinese: goo.gl/uxuxci
Further company coverage: (Beijing Headline News)

---

3. Import the dataset to your Python IDE

# Line 63
# 3 Create function to import dataset**, called def load_all_articles**
Load all JSON articles and track progress for all articles
Decide to **save** in .txt, **called all_texts.txt,** form because spacy processing needs string format.

```
# ------------------------------------------------------------------------------------------------------------
```

4. Preprocess the dataset by using the SpaCy package

# Line 94
# 4 Clean dataset instructions:

Preprocess using the SpaCy	: nlp = spacy.load("en_core_web_sm", disable=["ner", "parser"])
use stop word lists	: t.is_stop
lower case writing	: t.lemma_.lower()
eliminate special characters	: not t.is_alpha, t.is_punct, t.is_space
numbers	: t.is_digit
up to two-letter-words	: len(lemma) <= 2
Use lemmatization or stemming techniques	: t.lemma_
optionally try out POS tagging	: allowed_pos=["NOUN", "PROPN", "ADJ", "VERB"]
Create NLP object with command	: nlp = spacy.load(en_core_web_sm_path)
create function to apply above methods	: **def clean_doc** (doc, stop_words, allowed_pos)

result observation, if we use clean function with different instruction, we found:

- 1st **try** with:

t.is_stop, t.is_digit, not t.is_alpha
t.is_punct, t.is_space, len(lemma) <= 2

Vocabulary contains **noise** like:

```
Number of repeated_letter: 1
Total repeated words: 3918
        word_id word  doc_frequency  is_noise
1164   1164      iii      3918             True
```

- 2nd **try** with:

allowed_pos=["NOUN", "**PROPN**", "ADJ", "VERB"]
t.is_stop, t.is_digit, not t.is_alpha, t.is_punct, t.is_space,
t.lemma_ == "-PRON-", len(lemma) <= 2,
t.pos_ not in allowed_pos

still contain **noise** like:

```
Number of repeated_letter: 1
Total repeated words: 3807
        word_id   word   doc_frequency   is_noise
1122  1122        iii     3807                True
```

- 3rd **try** with:

allowed_pos=["NOUN", "ADJ", "VERB"]
t.is_stop, t.is_digit, not t.is_alpha, t.is_punct, t.is_space,
t.lemma_ == "-PRON-", len(lemma) <= 2, t.pos_ not in allowed_pos,
re.fullmatch(r"(.)\1{2,}|(..+?)\1+", lemma)

```
Number of repeated_letter: 0
Total repeated words: 0
```

Decide to **use 3rd function** with addition to **remove** any **pronouns** and **remove repeated-character** words.

Test **clean_doc function**:
Original text: Apple is releasing a new iPhone model next week!!! It costs $999 and ships in 2 days.
Cleaned doc: ['apple', 'release', 'new', 'iphone', 'model', 'week', 'cost', 'ship', 'day']

--
# Line 113
# 5 create function that processes with nlp to clean and tokenize texts while removing stop words, **called def_preprocess**, that takes:

306242it [1:52:06, 45.53it/s]
Number of documents: 306242

and save the results with an ID, called **cleaned_texts.jsonl**

--

# Line 144
# 6 Create corpus
clean messy list of dictionaries format on cleaned_texts into clean list format in corpus as it easier to read and process.
save corpus as **corpus_texts.txt**
--

# 7 Test **print** result to **see difference** from messy list to clean list.

original (all_texts.txt) : March 27(Reuters) - AU Optronics Corp : * Says it plans to pay cash dividend of T$1.2/share for 2017 Source text in Chinese: goo.gl/uxuxci Further company coverage: (Beijing Headline News)
preprocessed & **cleaned** (cleaned_texts.jsonl) : {"id": 0, "tokens": ["say", "plan", "pay", "cash", "dividend", "share", "source", "text", "company", "coverage"]}
corpus (corpus_texts.txt) : say plan pay cash dividend share source text company coverage

--

# Line 156
# 8 Count statistics from all articles in corpus txt:

```
count    306242.000000
mean        154.900575
std         292.088754
min           0.000000
10%          11.000000
20%          17.000000
30%          28.000000
40%          42.000000
50%          79.000000
60%         121.000000
70%         163.000000
80%         213.000000
90%         312.000000
max       12839.000000
```

--

5. Vectorize the document articles by using the TfidfVectorizer from

# Line 172
Create matrix using **tfidf vectorizer** that:
Filtering rare and **common words** in documents
keep unique **useful words** kept after filtering

min_df : removes rare words, words appearing in too few documents
max_df : removes common words, words appearing in too many documents
The final result is a matrix where rows are documents and columns are important words
TF-IDF matrix shape: (306242, 2135)

--

# Line 189

Create dataframe from vectorizer **called df_stats**, to get ID number for each word and counts how many documents each word appears in.

```
word_id   word  doc_frequency
0          abandon    2668
1          ability       18522
1780       reuters      18105
1830       say            180686
411         company    158467
```

# Line 202

Create function called **is_noise_pattern** that marks **repeated letter** or **patterns** as **noise**.
print and show to know how many noise exists.

```
Number of repeated_letter: 0
Total repeated words: 0
```

# Line 217

converts TF-IDF matrix into a **Gensim** corpus format, so we get numeric format to prepare to run LDA model.

---

5. Create a corpus and define the corresponding id2word mapping.

# Line 221

Create id2word to **translate numbers** to its words so its readable in LDA.
word2id['corp'] = 695
id2word[695] = corp

---

6. Perform an LDA topic modeling algorithm by using the Gensim

#Line 227

Run LDA model with genism corpus using different number of topics. Save each model then print top words for each topic so can directly analyzed model result.

---

7. Evaluate your models by using perplexity and coherence scores as well as the LDAviz package

# Line 272

Calculate coherence, **perplexity** and create **pyldavis** from saved model.
Coherence: Measures how “interpretable” your topics are. Higher is better.
Perplexity: Measures how well the model predicts the words in documents. Lower is better.

pyldavis save in html format, cause code run in vscode, so the visualization must be opened in a browser instead being displayed inline with jupyter.

Summarize result:
try few **min_df** and **max_df** on vectorizer phase and analyze result via topics, coherence, perplexity and pyldavis, with some consideration:

min_df=**0.005**,
max_df=**0.1**,
TF-IDF matrix shape: (306242, **2478**)
Topics: 5, Coherence: 0.5510697270148179, Perplexity: 188.03084284371695
Topics: 10, Coherence: 0.4988689810923795, Perplexity: 187.86852873120364
Topics: 15, Coherence: 0.35803461277094334, Perplexity: 189.6625620638587
Topics: 20, Coherence: 0.3081267357563046, Perplexity: 190.07720526377403

Min_df	Max_df	Topics	Coherence	Perplexity	Top Words	
0.005	0.1	5	0.5510697270148179	188.03084284371695	Topic 0: oil, bank, euro, analyst, dollar, index, yield, bond, economy, inflation Topic 1: net, income, expense, gaap, non, operating, customer, development, common, acquisition	word_id word doc_frequency 390 classify 1534 2410 warehouse 1535 112 anchor 1535

					Topic 2: official, election, nuclear, vote, leader, meeting, minister, police, sanction, rule Topic 3: tablet, browser, win, game, play, run, landscape, match, season, final Topic 4: earning, announces, versus, filing, gaap, shares, adjusted, common, stake, standard	334 cargo 1535 690 disappoint 1536 2420 way 30585 1253 leader 30537 1913 run 30321 763 economic 30060 1320 make 29717 2408 want 29617
		10	0.4988689810923795	187.86852873120364	Topic 1: yuan, newsroom, copper, ounce, bpd, automaker, miner, aluminium, brent, renminbi Topic 2: pressure, prevent, presentation, preserve, presidency, president, presidential, press, presence, prevail Topic 3: pct, yen, sentiment, bell, turmoil, exporter, drugmaker, bounce, greenback, bullish Topic 4: win, euro, official, run, meeting, nuclear, leader, election, talk, hit Topic 5: narration, rough, reporter, url, cut, awareness, hide, copy, code, pop Topic 6: sex, harassment, diversified, defendant, education, network, sexual, workplace, assault, misconduct Topic 7: earning, net, income, gaap, expense, tablet, operating, compare, period, tax Topic 8: tournament, golf, championship, medal, hole, shot, olympic, course, par, champion Topic 9: oil, bank, announces, shareholder, conference, fund, customer, agreement, analyst, production	

```
# ---------------------------------------------------------------------------------------------------------------
```

min_df=**0.007**,
max_df=**0.3**,
TF-IDF matrix shape: (306242, **2125**)
Topics: 5, Coherence: 0.5472399470706638, Perplexity: 154.38071199536444
Topics: 10, Coherence: 0.5988408308686612, Perplexity: 163.02532641952456
Topics: 15, Coherence: 0.3317684052278175, Perplexity: 158.10634502433203
Topics: 20, Coherence: 0.2687469102952111, Perplexity: 159.14331103590376

Min_df	Max_df	Topics	Coherence	Perplexity	Top Words	Words Frequency
0.007	0.3	5	0.5472399470706638	154.38071199536444	Topic 0: statement, net, result, share, information, income, include, look, quarter, expense Topic 1: text, coverage, share, earning, announces, loss, quarter, revenue, result, versus Topic 2: tablet, win, game, landscape, play, medium, hour, run, time, season Topic 3: government, deal, official, country, election, nuclear, state, people, tell, police Topic 4: percent, market, rise, price, oil, share, stock, high, bank, trade	word_id word doc_frequency 102 anonymity 2144 1916 testify 2145 910 hole 2145 017 ipo 2147 544 destination 2148 953 include 85928 1226 new 81151 1136 market 79075 2012 update 76871 1931 time 75599 1703 share 73448
		10	0.5988408308686612	163.02532641952456	Topic 0: share, text, coverage, earning, loss, quarter, revenue, result, versus, profit Topic 1: text, coverage, announces, filing, share, yuan, stake, appoint, agreement, offering Topic 2: percent, rise, price, oil, market, rate, fall, stock, high, yield Topic 3: court, lawyer, lawsuit, file, case, federal, investigation, attorney, prosecutor, class Topic 4: win, game, play, run, match, season, final, player, hit, team Topic 5: statement, look, information, result, release, include, dividend, risk, conference, product Topic 6: election, nuclear, police, government, official, kill, leader, korean, people, attack Topic 7: net, income, expense, quarter, cash, gaap, loss, share, revenue, non Topic 8: new, deal, business, percent, market, tablet, hour, include, plan, work Topic 9: patient, drug, clinical, study, treatment, cancer, disease, trial, therapy, health	

```
# ------------------------------------------------------------------------------------------------------------
min_df=0.007,
max_df=0.7,
TF-IDF matrix shape: (306242, 2130)
Topics: 5, Coherence: 0.5032661126255157, Perplexity: 147.93997061814554
Topics: 10, Coherence: 0.6019024176573079, Perplexity: 163.9570581333046
Topics: 15, Coherence: 0.4107777405700132, Perplexity: 152.75701092019094
Topics: 20, Coherence: 0.2566188187278901, Perplexity: 153.410583334261
```

Min_df	Max_df	Topics	Coherence	Perplexity	Top Words	Words Frequency
0.007	0.7	5	0.5032661126255157	147.93997061814554	Topic 0: statement, net, income, share, expense, cash, quarter, result, look, gaap Topic 1: tablet, company, service, landscape, business, medium, product, technology, lead, customer Topic 2: text, coverage, source, company, share, earning, announces, loss, quarter, revenue Topic 3: say, report, year, people, government, win, tell, official, country, deal Topic 4: percent, say, year, market, price, rise, oil, report, stock, trade	word_id word doc_frequency 102 anonymity 2144 911 hole 2145 1920 testify 2145 1018 ipo 2147 1744 slowdown 2148 1653 say 180686 365 company 152657 2124 year 144537 1574 report 143617 1762 source 107285
		10	0.6019024176573079	163.9570581333046	Topic 0: net, gaap, quarter, income, share, expense, revenue, loss, cash, adjusted Topic 1: statement, look, information, result, risk, include, patient, share, dividend, release Topic 2: conference, webcast, replay, available, result, website, information, dial, financial, quarter Topic 3: police, kill, attack, military, israeli, iranian, court, prosecutor, arrest, investigation Topic 4: stake, share, percent, deal, bank, company, loan, fund, private, buy Topic 5: say, report, year, company, new, hour, deal, government, country, tell Topic 6: tablet, landscape, service, medium, company, wide, business, technology, team, experience Topic 7: percent, rise, price, oil, market, rate, stock, fall, year, index Topic 8: text, coverage, source, company, share, announces, earning, loss, quarter, result Topic 9: game, win, play, match, run, season, final, player, team, score	

Observation:

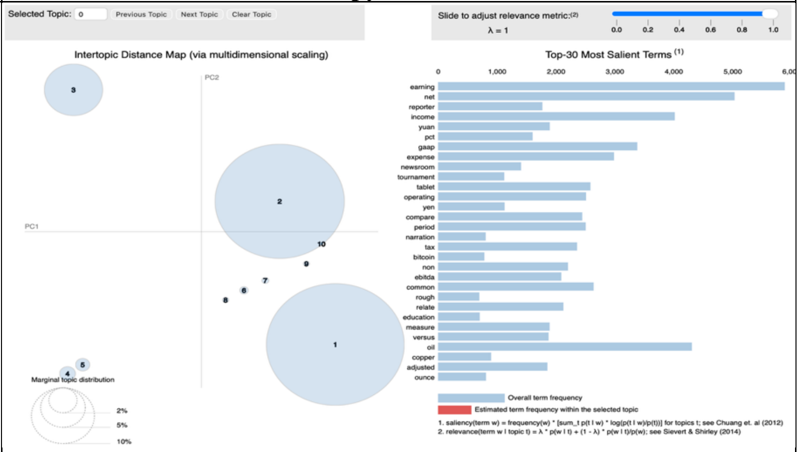
from all 3 model trial’s coherence peaked at 5 topics and gradually decreased, with a significant drop after 10 topics, shows the meaning becomes unclear.

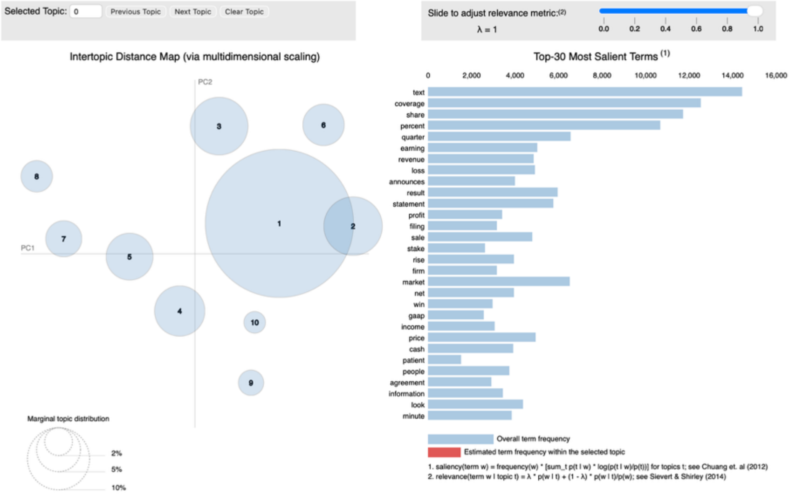
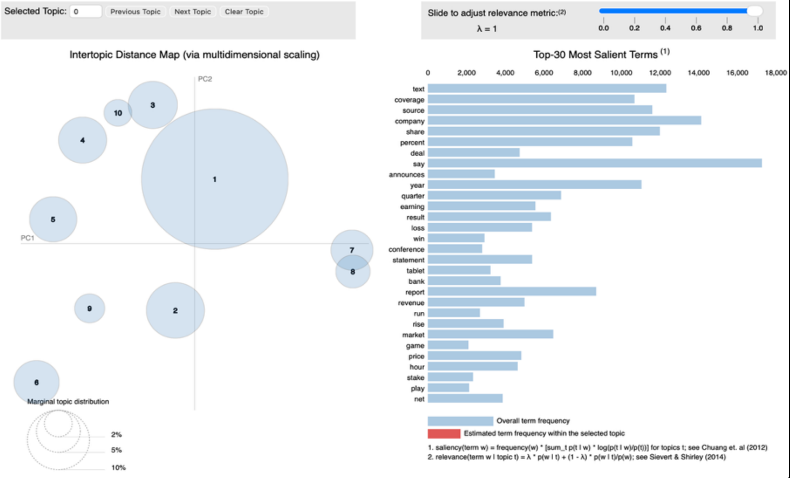
Perplexity showed minimal improvement beyond 10 topics, suggesting no substantial gain in statistical fit.

Therefore, **10 topics** provides the best trade-off between interpretability and model complexity

---

After we chose 10 topics, we **filtered different min_df** and **max_df** values **to** find the ones that **produce clear** and understandable **topics**. We also checked the results using pyLDAvis to make sure the topics are well separated and easy to interpret.

min_df	max_df	Coherence	Perplexity	10 Topic Words	pyldavis
0.005	0.1	0.499	187.87	Topic 1: yuan, newsroom, copper, ounce, bpd, automaker, miner, aluminium, brent, renminbi Topic 2: pressure, prevent, presentation, preserve, presidency, president, presidential, press, presence, prevail Topic 3: pct, yen, sentiment, bell, turmoil, exporter, drugmaker, bounce, greenback, bullish Topic 4: win, euro, official, run, meeting, nuclear, leader, election, talk, hit Topic 5: narration, rough, reporter, url, cut, awareness, hide, copy, code, pop Topic 6: sex, harassment, diversified, defendant, education, network, sexual, workplace, assault, misconduct Topic 7: earning, net, income, gaap, expense, tablet, operating, compare, period, tax Topic 8: tournament, golf, championship, medal, hole, shot, olympic, course, par, champion Topic 9: oil, bank, announces, shareholder, conference, fund, customer, agreement, analyst, production	 Selected Topic: 0 Previous Topic Next Topic Clear Topic Intertopic Distance Map (via multidimensional scaling) PC2 PC1 1 2 3 4 5 6 7 8 9 10 Marginal topic distribution 2% 5% 10% Slide to adjust relevance metric:(2) λ = 1 0.0 0.2 0.4 0.6 0.8 1.0 Top-30 Most Salient Terms (1) 0 1,000 2,000 3,000 4,000 5,000 6,00 earning net reporter income yuan pct gaap expense newsroom tournament tablet operating yen compare period narration tax bitcoin non ebitda common rough relate education measure versus oil copper adjusted ounce Overall term frequency Estimated term frequency within the selected topic 1. saliency(term w) = frequency(w) * [sum_t p(t I w) * log(p(t I w)/p(t))] for topics t; see Chuang et. al (2012) 2. relevance(term w I topic t) = λ * p(w I t) + (1 - λ) * p(w I t)/p(w); see Sievert & Shirley (2014)

0.007	0.3	0.599	163.03	Topic 0: share, text, coverage, earning, loss, quarter, revenue, result, versus, profit Topic 1: text, coverage, announces, filing, share, yuan, stake, appoint, agreement, offering Topic 2: percent, rise, price, oil, market, rate, fall, stock, high, yield Topic 3: court, lawyer, lawsuit, file, case, federal, investigation, attorney, prosecutor, class Topic 4: win, game, play, run, match, season, final, player, hit, team Topic 5: statement, look, information, result, release, include, dividend, risk, conference, product Topic 6: election, nuclear, police, government, official, kill, leader, korean, people, attack Topic 7: net, income, expense, quarter, cash, gaap, loss, share, revenue, non Topic 8: new, deal, business, percent, market, tablet, hour, include, plan, work Topic 9: patient, drug, clinical, study, treatment, cancer, disease, trial, therapy, health	
0.007	0.7	0.602	163.96	Topic 0: net, gaap, quarter, income, share, expense, revenue, loss, cash, adjusted Topic 1: statement, look, information, result, risk, include, patient, share, dividend, release Topic 2: conference, webcast, replay, available, result, website, information, dial, financial, quarter Topic 3: police, kill, attack, military, israeli, iranian, court, prosecutor, arrest, investigation Topic 4: stake, share, percent, deal, bank, company, loan, fund, private, buy Topic 5: say, report, year, company, new, hour, deal, government, country, tell Topic 6: tablet, landscape, service, medium, company, wide, business, technology, team, experience Topic 7: percent, rise, price, oil, market, rate, stock, fall, year, index Topic 8: text, coverage, source, company, share, announces, earning, loss, quarter, result Topic 9: game, win, play, match, run, season, final, player, team, score	

Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0 0.2 0.4 0.6 0.8 1.0
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
1 2 3 4 5 6 7 8 9 10
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms (1)
0 2,000 4,000 6,000 8,000 10,000 12,000 14,000 16,000
text
coverage
share
percent
quarter
earning
revenue
loss
announces
result
statement
profit
filing
sale
stake
rise
firm
market
net
win
gaap
income
price
cash
patient
people
agreement
information
look
minute
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

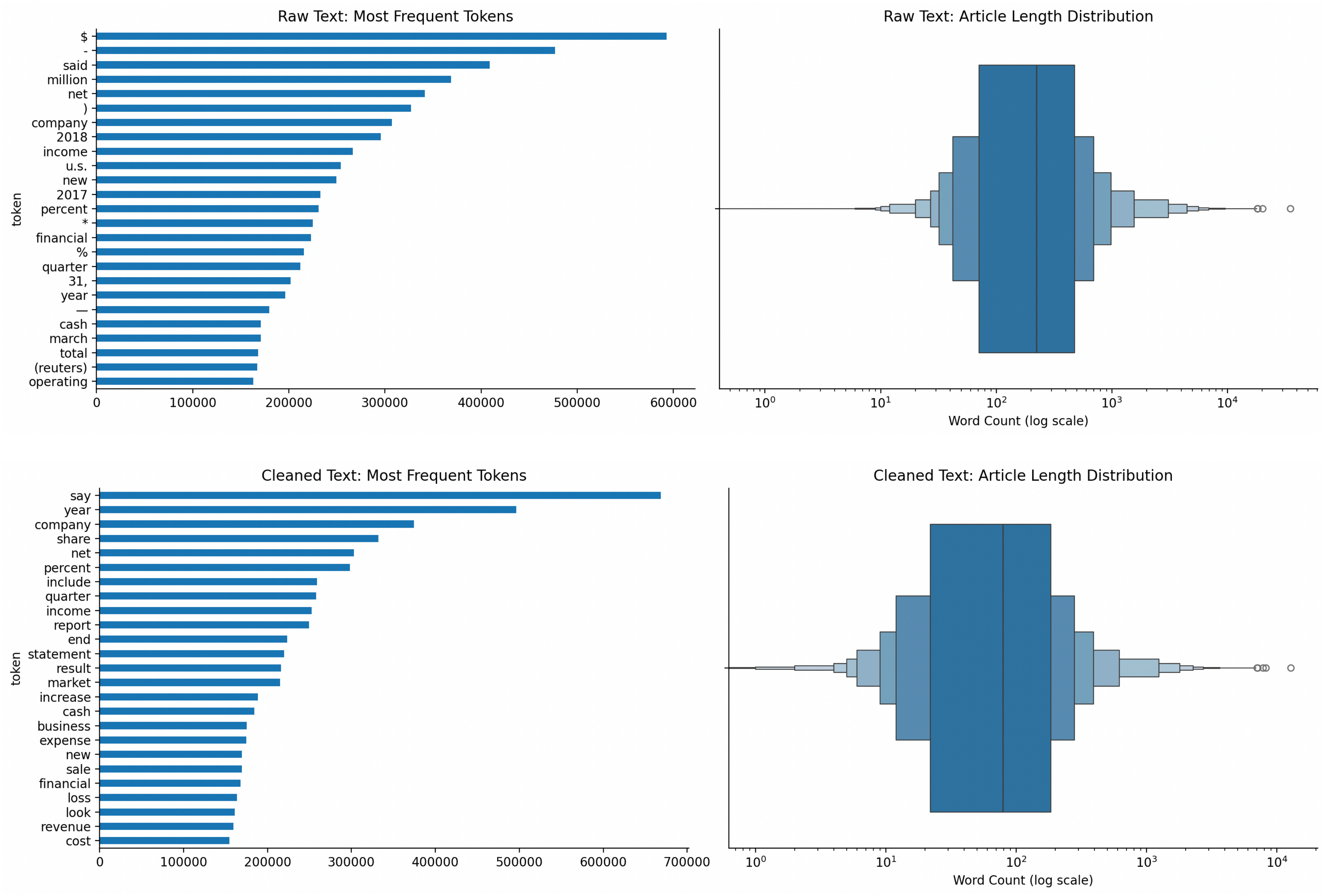
Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0 0.2 0.4 0.6 0.8 1.0
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
1 2 3 4 5 6 7 8 9 10
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms (1)
0 2,000 4,000 6,000 8,000 10,000 12,000 14,000 16,000 18,000
text
coverage
source
company
share
percent
deal
say
announces
year
quarter
earning
result
loss
win
conference
statement
tablet
bank
report
revenue
run
rise
market
game
price
hour
stake
play
net
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

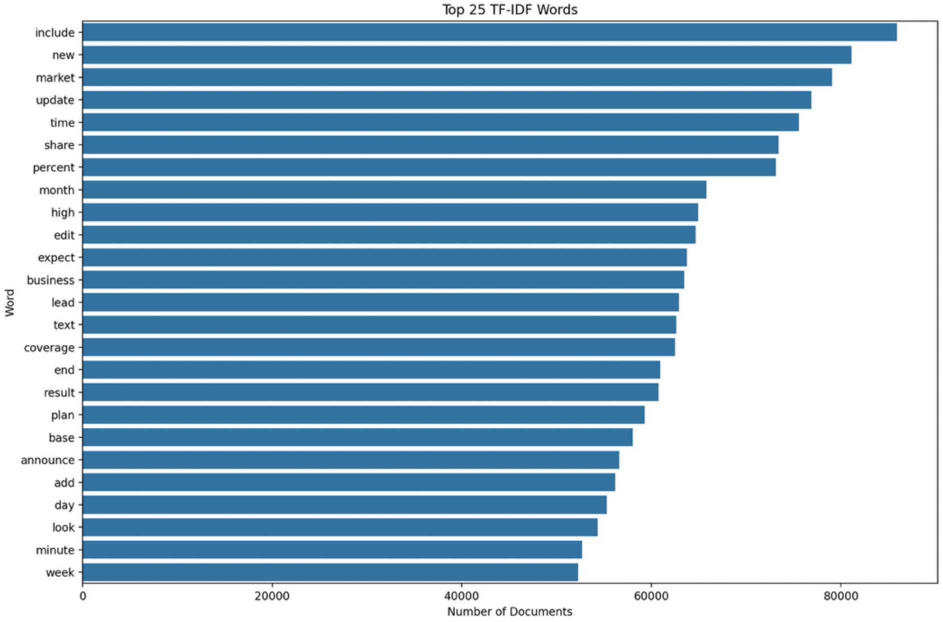
Observation:
second topic set with **min_df: 0.007** and **max_df: 0.3**, is considered better because the words in each topic clearly match one main theme. For example, some topics focus on sports, some on healthcare, and some on finance results. This makes the topic distribution easy to understand and label. The themes are clear and meaningful, so the results are more useful for analysis. Possible label given:

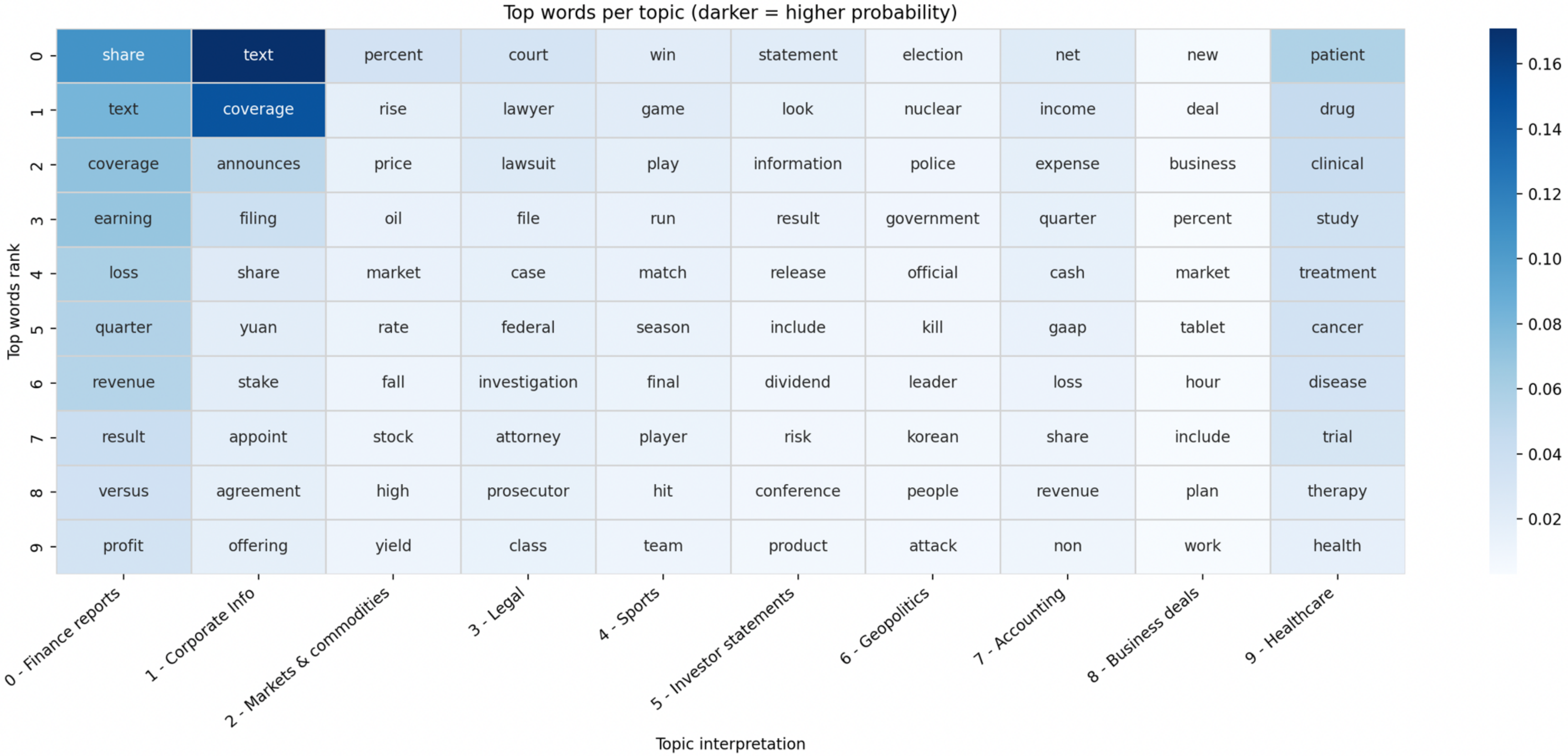
7. Provide topic labels

Topic	Label Intepretation
0	Finance reports
1	Corporate Info
2	Markets & commodities
3	Legal
4	Sports
5	Investor statements
6	Geopolitics
7	Accounting
8	Business deals
9	Healthcare

# Line 320
# Create visualization
File to produce : from **original** documents **before** processed Most Frequency Tokens, Article Length Distribution
: from **cleaned** documents **after** processed Most Frequency Tokens, Article Length Distribution
File to save : in demo folder; raw_text.png, cleaned_text.png, heatmap.png
Goal : compare how much noises removed after preprocessed
compare how are words distribution after cleaned


Raw Text: Most Frequent Tokens
token
$
-
said
million
net
)
company
2018
income
u.s.
new
2017
percent
*
financial
%
quarter
31,
year
—
cash
march
total
(reuters)
operating
0
100000
200000
300000
400000
500000
600000
Raw Text: Article Length Distribution
10^0
10^1
10^2
10^3
10^4
Word Count (log scale)
Cleaned Text: Most Frequent Tokens
token
say
year
company
share
net
percent
include
quarter
income
report
end
statement
result
market
increase
cash
business
expense
new
sale
financial
loss
look
revenue
cost
0
100000
200000
300000
400000
500000
600000
700000
Cleaned Text: Article Length Distribution
10^0
10^1
10^2
10^3
10^4
Word Count (log scale)





Top words per topic (darker = higher probability)

Top words rank	0 - Finance reports	1 - Corporate Info	2 - Markets & commodities	3 - Legal	4 - Sports	5 - Investor statements	6 - Geopolitics	7 - Accounting	8 - Business deals	9 - Healthcare
0	share	text	percent	court	win	statement	election	net	new	patient
1	text	coverage	rise	lawyer	game	look	nuclear	income	deal	drug
2	coverage	announces	price	lawsuit	play	information	police	expense	business	clinical
3	earning	filing	oil	file	run	result	government	quarter	percent	study
4	loss	share	market	case	match	release	official	cash	market	treatment
5	quarter	yuan	rate	federal	season	include	kill	gaap	tablet	cancer
6	revenue	stake	fall	investigation	final	dividend	leader	loss	hour	disease
7	result	appoint	stock	attorney	player	risk	korean	share	include	trial
8	versus	agreement	high	prosecutor	hit	conference	people	revenue	plan	therapy
9	profit	offering	yield	class	team	product	attack	non	work	health

# Line 457
Check **vocabulary mappings** to **see difference:**

word2id : [('plan', 1354)]
(vectorizer.vocabulary)
modelid2word : [('plan', 1354)]
{word: idx for idx, word in (model10.id2word).items()}

print(word2id == modelid2word) : True

Even though they look the same, they are **not** guaranteed to represent the **same** vocabulary system.
We must use the **dictionary from** the **trained LDA model** because topic inference depends on the exact word-to-ID mapping learned during training.
This ensures the words are translated into the exact numbers the model knows.

# Line 471
Create get_topic_by id function
It looks up a document by its ID, turns its words into numbers the LDA model understands, and shows which topics the document is about.
It can show sorted topics distribution for one document or multiple documents.
This **function tells** us **what topics** a **document belongs** to.

#Line 493
run **per id**
sorted topics for **doc 17**:
Doc 17 top topics → Topic 2: 0.7152, Topic 3: 0.1749, Topic 9: 0.0807, Topic 7: 0.0089, Topic 5: 0.0043, Topic 6: 0.0041, Topic 1: 0.0041, Topic 8: 0.0037, Topic 4: 0.0022, Topic 10: 0.0019

#Line 499
run for **multiple id**
Doc 0 sorted topics → Topic 2: 0.8875, Topic 9: 0.0753, Topic 7: 0.0089, Topic 3: 0.0080, Topic 5: 0.0043, Topic 6: 0.0041, Topic 1: 0.0041, Topic 8: 0.0037, Topic 4: 0.0022, Topic 10: 0.0019

Doc 10 sorted topics → Topic 8: 0.9232, Topic 9: 0.0766, Topic 7: 0.0001, Topic 3: 0.0000, Topic 2: 0.0000, Topic 5: 0.0000, Topic 6: 0.0000, Topic 1: 0.0000, Topic 4: 0.0000, Topic 10: 0.0000

Doc 999 sorted topics → Topic 9: 0.8641, Topic 6: 0.1343, Topic 7: 0.0004, Topic 3: 0.0003, Topic 2: 0.0003, Topic 5: 0.0002, Topic 1: 0.0002, Topic 8: 0.0002, Topic 4: 0.0001, Topic 10: 0.0001

--
#Line 508
Create **topics.json** that use a trained topic model to calculate the probability of each topic for every document, keep the top topics, and save those results into a JSON file for later analysis.

--
#Line 529
FOR HTML
Sample topics.json to only has 10000 files to make it lighter instead of 306243 files, in order to display in html.

--
#Line 545
store each line with an ID in a dictionary, optionally limit the number of entries (e.g., first 50,000), and then save the structured data into a JSON file for easier processing or analysis later.

--
#Line 561
Sample corpus text and cleaned text in order to be lighter to display in html page.

--

#Line 586
Create get_topic_by id function to run with **demo files**
It looks up a document by its ID, turns its words into numbers the LDA model understands, and shows which topics the document is about.
It can show sorted topics distribution for one document or multiple documents.
This **function tells** us **what topics** a **document belongs** to.

#Line 606
run **per id**

```
Doc 17 top topics → Topic 2: 0.7152, Topic 3: 0.1749, Topic 9: 0.0807, Topic 7: 0.0089, Topic 5: 0.0043, Topic 6: 0.0041, Topic 1: 0.0041,
Topic 8: 0.0037, Topic 4: 0.0022, Topic 10: 0.0019
```

#Line 612
run for **multiple id**

```
Doc 0 top topics → Topic 2: 0.8875, Topic 9: 0.0753, Topic 7: 0.0089, Topic 3: 0.0080, Topic 5: 0.0043, Topic 6: 0.0041, Topic 1: 0.0041,
Topic 8: 0.0037, Topic 4: 0.0022, Topic 10: 0.0019
Doc 10 top topics → Topic 8: 0.9232, Topic 9: 0.0766, Topic 7: 0.0001, Topic 3: 0.0000, Topic 2: 0.0000, Topic 5: 0.0000, Topic 6: 0.0000,
Topic 1: 0.0000, Topic 4: 0.0000, Topic 10: 0.0000
Doc 999 top topics → Topic 9: 0.8641, Topic 6: 0.1343, Topic 7: 0.0004, Topic 3: 0.0003, Topic 2: 0.0003, Topic 5: 0.0002, Topic 1: 0.0002,
Topic 8: 0.0002, Topic 4: 0.0001, Topic 10: 0.0001
```

---

8. How can the derived topic labels be used in a Robo Advisory solution to improve future asset allocation processes?

Robo Advisory (RA) Definition

Robo advisors are basically computer programs that give investment advice and manage your money automatically. They figure out what kind of investments best suited by asking about goals and financial situation (Gaspar & Oliveira, 2024). Basically, they replace some traditional human asset managers with algorithm-based decisions (Becker et al., 2021).

RA Application

Robo advisors ask clients questions to figure out their risk tolerance and check if their answers make sense. Then algorithm matches each person to a risk profile, which decides how their money splits between things like stocks, bonds, and cash. For example, Schwab's "conservative" profile puts about 20% in stocks and the rest in safer investments, so each RA has its own way of setting these profiles (Wong, 2020).

RA currently used to automatically manage investments and create portfolios for clients using algorithms and quantitative financial models. They help investors by assessing risk levels, selecting suitable assets, and rebalancing portfolios without much human involvement. These systems are mainly applied in digital wealth management services to provide low-cost and automated investment solutions (Beketov et al., 2018).

Topic labels in RA

Topic modelling, like LDA, is a method used to find hidden topics in a large group of documents automatically (Blei et al., 2003). It has been widely used in finance to study things like stock movements, market volatility from news, analyst reports, company disclosures, and even financial fraud detection. Researchers have also improved LDA with newer versions that can capture relationships between topics and investor sentiment, helping analyze social media, news articles, and economic trends more accurately (Ji R and Han Q, 2022).

Topic modelling can be challenging because researchers need to decide things like the number of topics and the best model settings. In this study, LDA was used to analyze advisor notes by turning documents into a bag-of-words format and repeatedly calculating the probability of words belonging to different topics until the model stabilizes. To choose the best number of topics, the researchers tested different values and used a coherence score to measure how good and meaningful the topics were (Pagliaro et al., 2021).

Asset Allocations process

Asset allocation is the process of dividing investments among different kinds of asset (Morningstar, 2011):

- First, pick the asset classes and estimate their future returns, risks, and how they move together.
- Then, an algorithm decides what percentage of your money goes into each asset.
- Finally, forecasting of how portfolio might do over time, like what it could be worth in three years if returns are really low.

For example, an investor can view a prediction of what the portfolio value would be three years from now if its returns were in the bottom 5% of the projected range during this period.

With the rise of fully automated investment systems known as Robo Advisors (RA), advances in computing power and data storage have made their development possible. Technologies such as ETFs, Big Data, and digital financial platforms allow robo advisors to manage large amounts of assets efficiently. The traditional tasks of asset management can be divided into five main categories (Beketov et al., 2018):

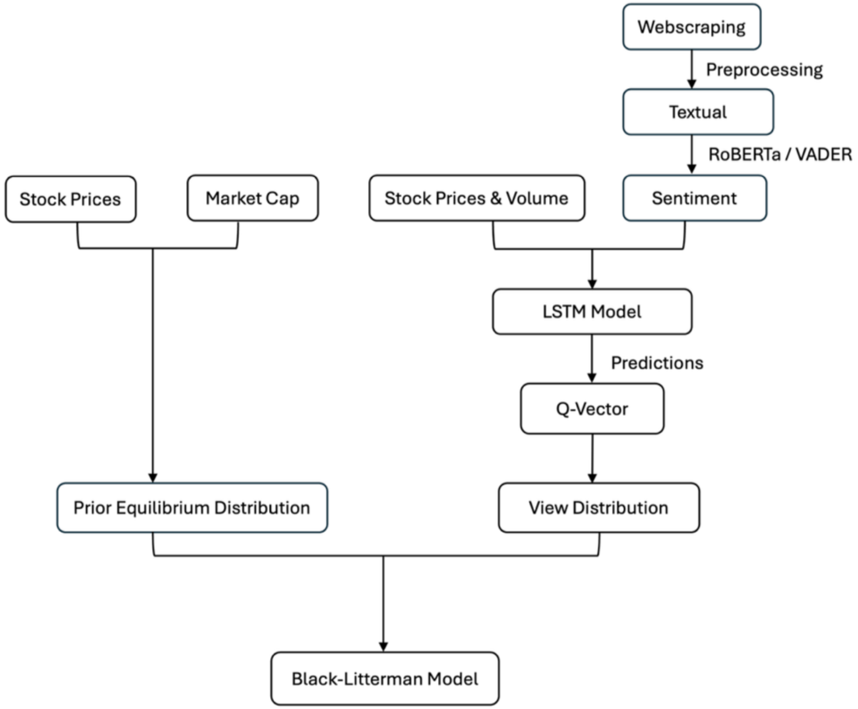
- Asset universe selection: stocks chosen based on personal choice
- Investor profile identification: based on questionnaire
- Asset allocation or portfolio optimization: based on analytical solution
- Monitoring and rebalancing: based on RA adjustment once a month/quarter
- Performance review and reporting: based on RA report machine

RA use machine learning and algorithms to analyze investments, like comparing bonds or predicting performance. However, most ML-based solutions in the study were partial tools, not full RA, so they don’t manage the entire asset allocation process (Becker et al., 2021). LDA can read lots of financial reviews and automatically group related topics or aspects. A RA could use these topic labels to understand market sentiment or which factors investors care about most. By feeding this information into its machine learning models, RA could make smarter, more informed decisions about future asset allocation (Fatima-Zahrae et al., 2021).

Framework

Possible framework to apply Robo Advisory in Asset Allocation:
Predict **future prices based** on news **sentiment** and historical **prices**

Figure 3 - Flowchart



Source:
https://research-api.cbs.dk/ws/portalfiles/portal/108051705/1857311_Master_Thesis_Thomas_Bach_Andersen.pdf

Goal:
Check if **information** from **social media** can **help** make **better investment** decisions?

Possible steps done:

1. Data Extraction: webscraping data from sources to capture public opinion and collecting market data prices.
2. Preprocessing: Raw text is cleaned by removing rare, common stop words and punctuation. It is then tokenized into smaller pieces.
3. Sentiment Scoring: Two different models, VADER and RoBERTa, analyze the cleaned text to assign it a sentiment score, showing if the sentiment positive or negative.

4. LSTM Model: input sentiment scores and stock prices into an LSTM to recognizes patterns over time to generate future price predictions.
5. Generate Market Expectation: calculate a Prior Equilibrium Distribution, which represents the average market expectation for returns.
6. Generate AI Prediction: Convert LSTM's price predictions into a mathematical form called a Q-Vector and View Distribution.
7. Portfolio Optimization: Use Black-Litterman Model to combine market's expectations with the AI's predictions.

Opinion:

1. This framework is suitable to apply in Robo Advisory to help in Asset Allocation because:

   The **LSTM predicts** future stock **prices based** on **sentiment** and historical **prices**.

   The **Black-Litterman** model as portfolio optimization **turns** these **predictions** into **allocation weights** to **suggest** how much **money** to invest.
2. This framework could possibly answer the questions in RA application:

   "where should I allocate my certain asset this month?" **based** on **future** prices **prediction**.
3. **LDA** topic labels **could** be **inserted before** run **LSTM** model.

   possible improvement after input LDA: **model** could **learn** which **topic occurs** in news when **prices high** or **low**?

References

Andersen, T. B. (n.d.). [Master’s thesis]. Copenhagen Business School. https://research-api.cbs.dk/ws/portalfiles/portal/108051705/1857311_Master_Thesis_Thomas_Bach_Andersen.pdf

Becker, A., et al. (2021). Machine learning in automated asset management processes 4.1. Die Unternehmung, 75(3), 411–430. https://doi.org/10.5771/0042-059X-2021-3-411

Beketov, M. A., Lehmann, K., & Wittke, M. (2018). Financial risk and computational economics. Computational Economics. Springer. https://link.springer.com/content/pdf/10.1057/s41260-018-0092-9.pdf

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. Journal of Machine Learning Research, 3, 993–1022. https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Fatima-Zahrae, M., et al. (2021). Application of latent Dirichlet allocation (LDA) for clustering financial tweets. E3S Web of Conferences, 297, Article 01071. https://doi.org/10.1051/e3sconf/202129701071

Gaspar, R. M., & Oliveira, M. (2024). Robo advising and investor profiling. FinTech, 3, 102–115. https://doi.org/10.3390/fintech3010007

Ji, R., & Han, Q. (2022). Understanding heterogeneity of investor sentiment on social media: A structural topic modeling approach. Frontiers in Artificial Intelligence, 5, Article 884699. https://doi.org/10.3389/frai.2022.884699

Morningstar. (2011). Morningstar asset allocation optimization methodology. Morningstar, Inc. https://morningstardirect.morningstar.com/clientcomm/MorningstarAssetAllocationOptimizationMethodology.PDF

Pagliaro, M., et al. (2021). Investor behavior modeling by analyzing financial advisor notes: A machine learning perspective. arXiv. https://arxiv.org/abs/2107.05592

Wong, T. (2020). Short paper: Robo-advisor methodologies. Duke University. https://dukespace.lib.duke.edu/server/api/core/bitstreams/6a629ec1-c1a7-4ac2-8017-e02079d48f59/content