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

Folder:	**scripts**:
data	requirements.txt
result	app.py
models	lda.py
pyldavis	mads1.docx

Important

full.py:

Duration	: +-3hr
Goal	: run all
content	: full python code script to produce and run entire workflow from beginning to end
File to produce	: all.texts, cleaned_texts, corpus_texts, lda model(5, 10, 15, 20), pyldavis, lda_metrics
Files to save	: all.texts, cleaned_texts, corpus_texts, lda model(5, 10, 15, 20), pyldavis, lda_metrics
Memories to create	: 2.95 gb

lda.py

Duration	:
Goal	: script to run the statistics and graphics
content	: script to reopen saved models and create statistics and graphics
Files to open	: all.texts, cleaned_texts, corpus_texts, lda model(5, 10, 15, 20), pyldavis, lda_metrics
File to produce	: original visualization (most freq. tokens, article length dist.) cleaned visualization (most freq. tokens, article length dist.)
File to save	: -

 # 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)

--

 # 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.

 # 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 allow
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

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, 3249)

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.

Line 221
Create id2word to **translate numbers** to its words so its readable in LDA.
word2id['corp'] = 695
id2word[695] = corp

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.

--

 # 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 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	word_id word doc_frequency 390 classify 1534 2410 warehouse 1535 112 anchor 1535 334 cargo 1535 690 disappoint 1536 2420 way 30585 1253 leader 30537 1913 run 30321

		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	763 economic 30060 1320 make 29717 2408 want 29617

--

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	word_id word doc_frequency 102 anonymity 2144 1916 testify 2145 910 hole 2145 017 ipo 2147

					Topic 3: government, deal, official, country, election, nuclear, state, people, tell, police Topic 4: percent, market, rise, price, oil, share, stock, high, bank, trade	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**)

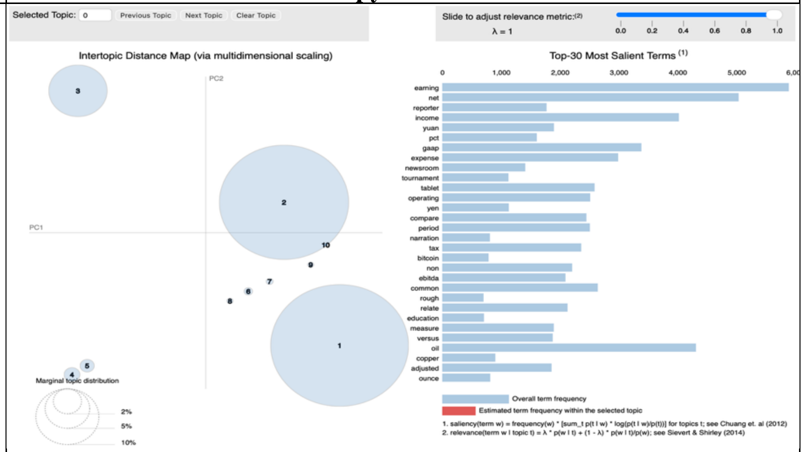
Topics: 5, Coherence: 0.5032661126255157, Perplexity: 147.93997061814554
Topics: 10, Coherence: 0.6019024176573079, Perplexity: 163.9570581333046
Topics: 15, Coherence: 0.4107777405700132, Perplexity: 152.75701092019094
Topics: 20, Coherence: 0.2566188187278901, Perplexity: 153.410583334261

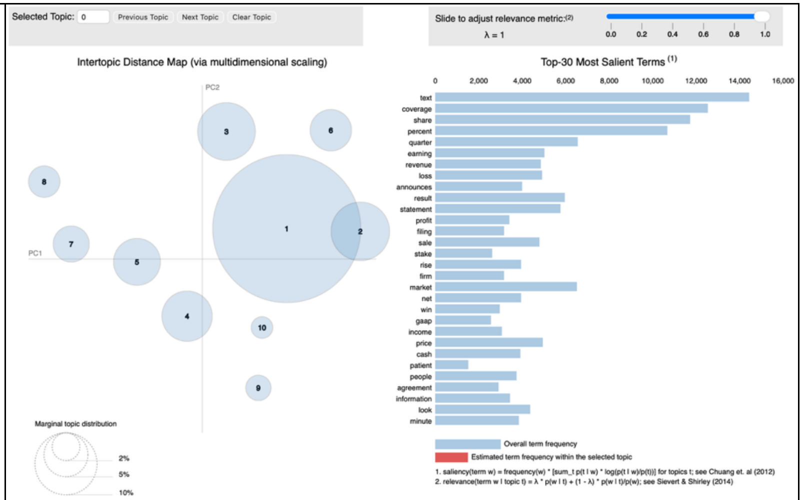
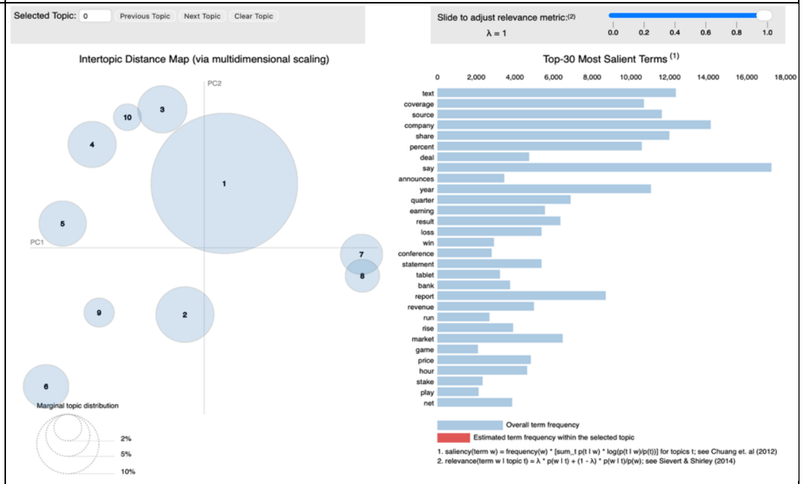
Min_df	Max_df	Topics	Coherence	Perplexity	Top Words	Words Frequency
0.007	0.7	5	0.5032661126255157	147.93997061814554	Topic 0: statement, net, income, share, expense, cash, quarter, result, look, gaap Topic 1: tablet, company, service, landscape, business, medium, product, technology, lead, customer 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, means 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.

<table>
<tr><th>min_df</th><th>max_df</th><th>Coherence</th><th>Perplexity</th><th>Topic Words</th><th>pyldavis</th></tr>
<tr><td>0.005</td><td>0.1</td><td>0.499</td><td>187.87</td><td>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</td><td>
</td></tr>
</table>

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	

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:

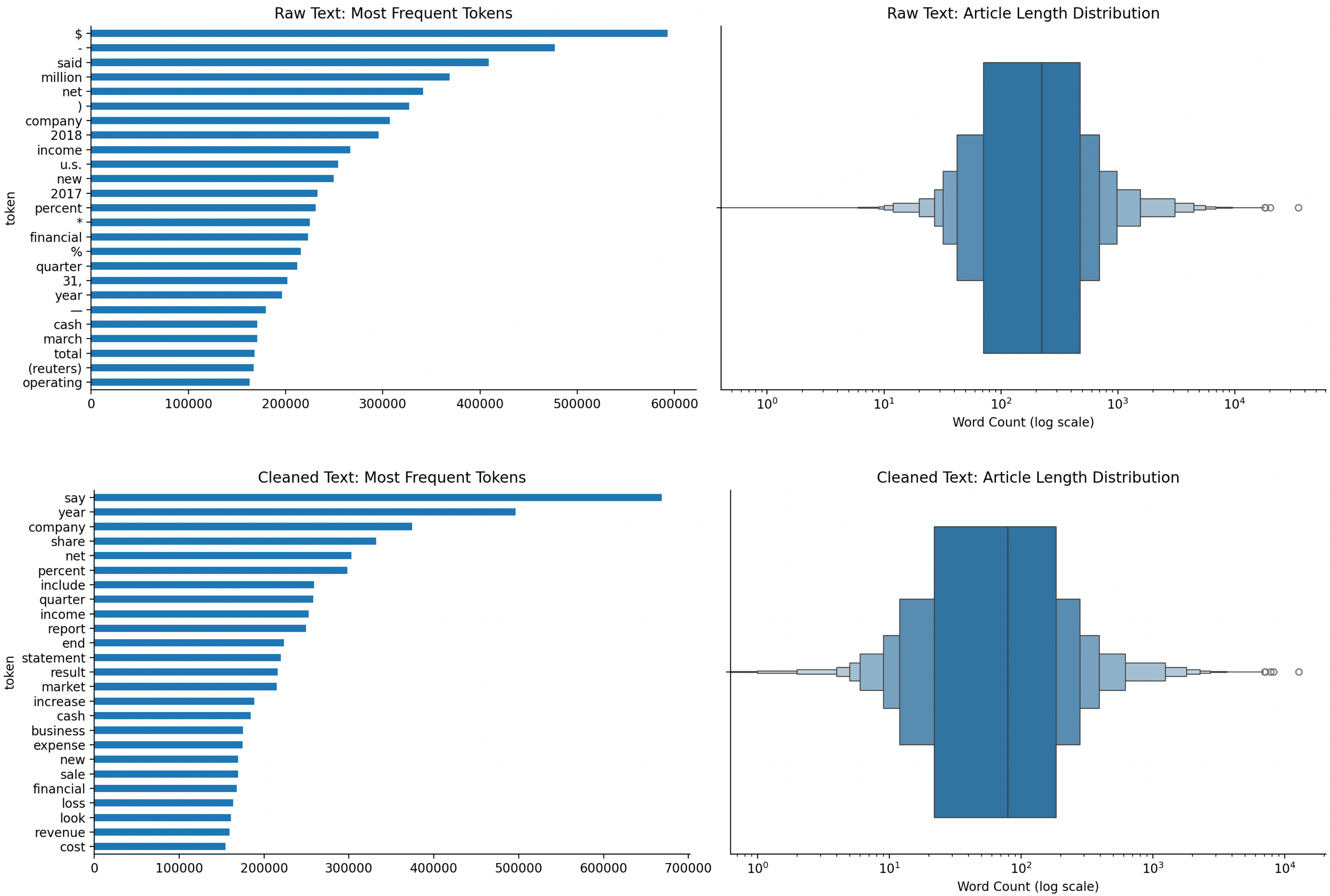
Topic	Label Intepretation
0	Finance reports
1	Corporate Info
2	Markets & commodities
3	Legal
4	Sports
5	Investor statements
6	Geopolitics
7	Accounting
8	Business deals
9	Healthcare

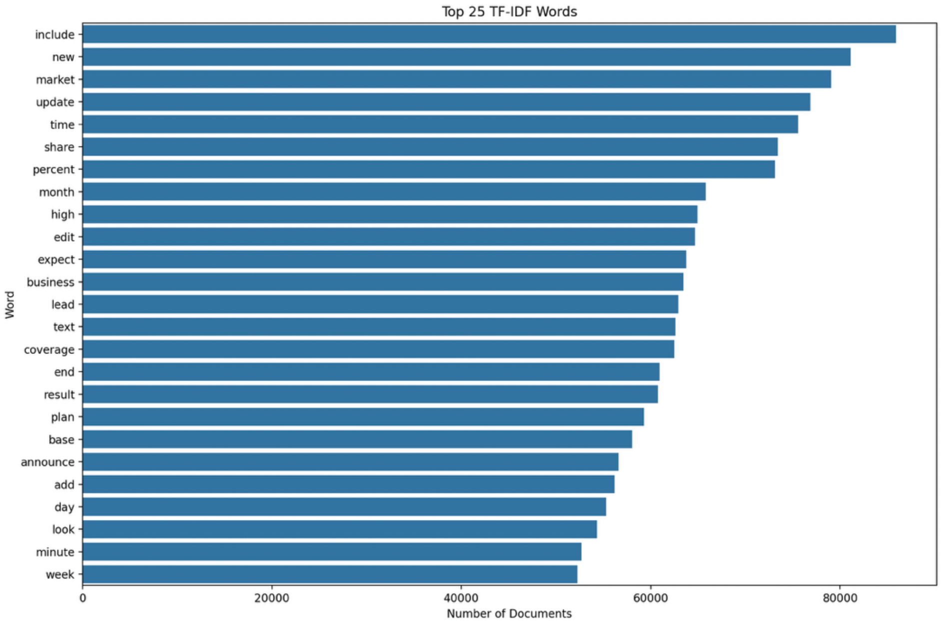
\# --

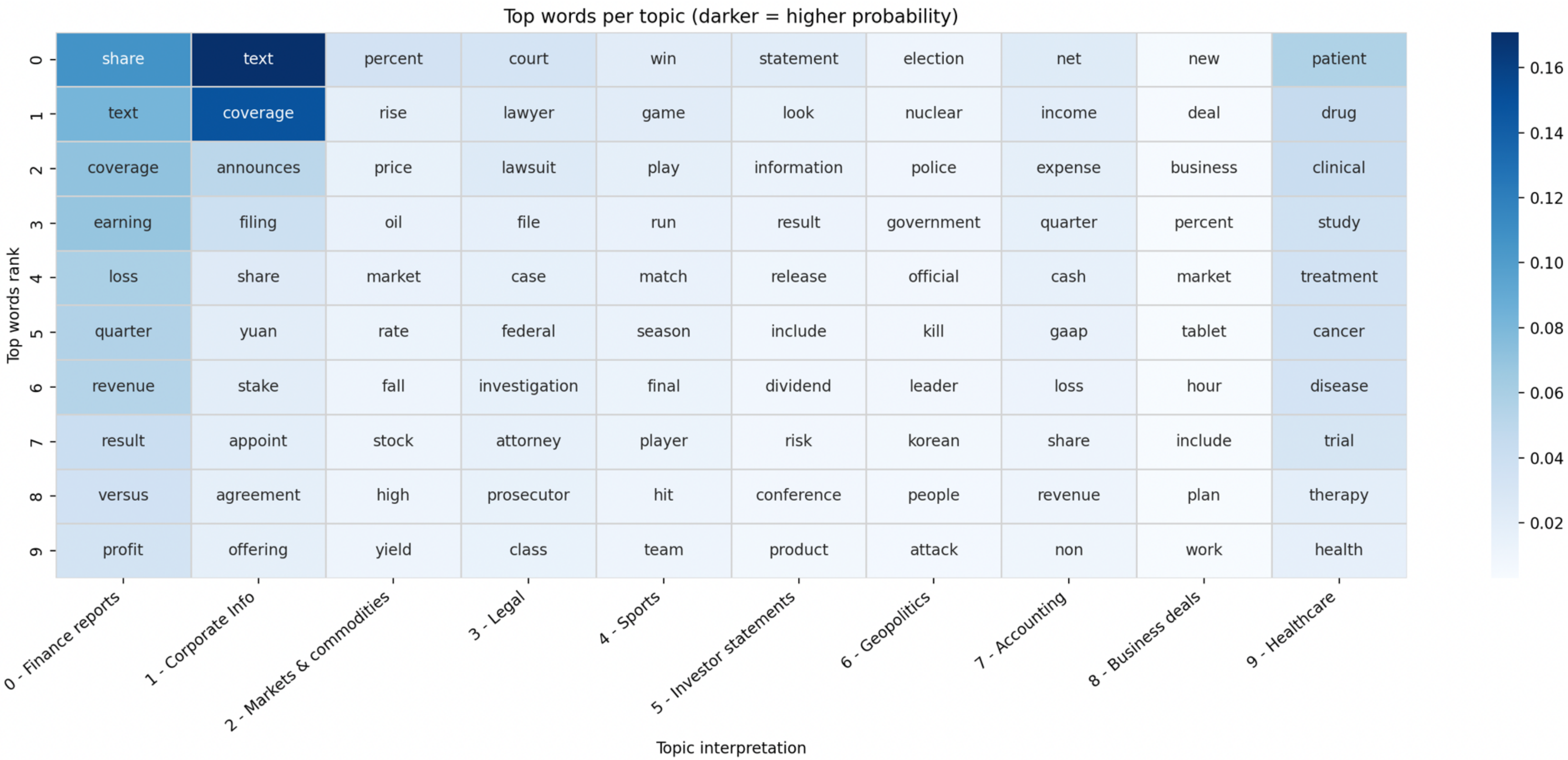
 # Line 320

\# 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	: to compare


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
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
Word Count (log scale)


Top 25 TF-IDF Words
Word
include
new
market
update
time
share
percent
month
high
edit
expect
business
lead
text
coverage
end
result
plan
base
announce
add
day
look
minute
week
0
20000
40000
60000
80000
Number of Documents


Top words per topic (darker = higher probability)
Top words rank
0
1
2
3
4
5
6
7
8
9
share
text
coverage
earning
loss
quarter
revenue
result
versus
profit
text
coverage
announces
filing
share
yuan
stake
appoint
agreement
offering
percent
rise
price
oil
market
rate
fall
stock
high
yield
court
lawyer
lawsuit
file
case
federal
investigation
attorney
prosecutor
class
win
game
play
run
match
season
final
player
hit
team
statement
look
information
result
release
include
dividend
risk
conference
product
election
nuclear
police
government
official
kill
leader
korean
people
attack
net
income
expense
quarter
cash
gaap
loss
share
revenue
non
new
deal
business
percent
market
tablet
hour
include
plan
work
patient
drug
clinical
study
treatment
cancer
disease
trial
therapy
health
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
Topic interpretation
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02

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 50000 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 ligter 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

LDA Topic Distribution

Document ID: 49998 Show ◀ Prev Next ▶

Data loaded ✔

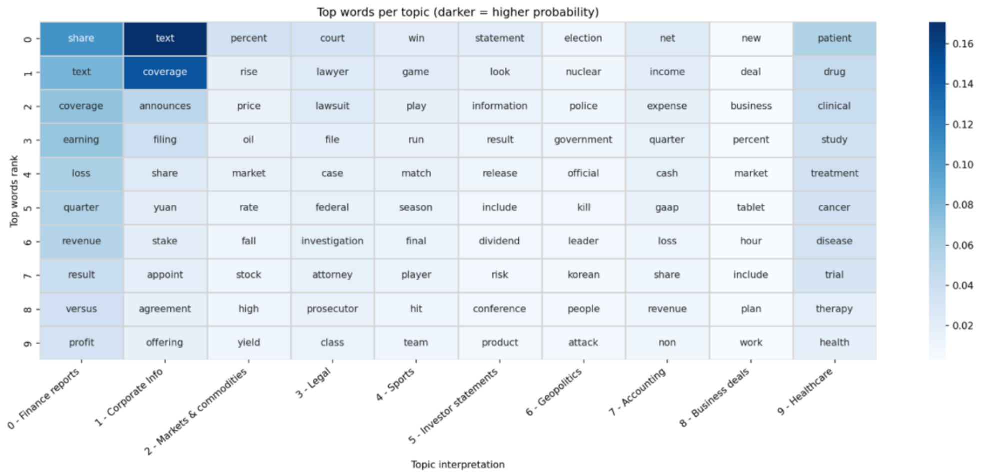
Article Text

March 2 (Reuters) - Jaguar Health Inc: * JAGUAR HEALTH - ON FEB 26, CO ENTERED SECURITIES PURCHASE AGREEMENT WITH CHICAGO VENTURE PARTNERS - SEC FILING * JAGUAR HEALTH - PURSUANT TO AGREEMENT CO ISSUED TO CVP PROMISSORY NOTE IN AGGREGATE PRINCIPAL AMOUNT OF $2.2 MILLION FOR PURCHASE PRICE OF $1.6 MILLION * JAGUAR HEALTH - ALSO ENTERED SECURITY AGREEMENT, PURSUANT TO WHICH CVP WILL RECEIVE SECURITY INTEREST IN SUBSTANTIALLY ALL OF CO'S ASSETS Source text: (bit.ly/2HZ8UZU) Further company coverage:

Topics

Topic 9 — 93.46%
Topic 4 — 4.12%
Topic 7 — 2.32%
Topic 3 — 0.03%
Topic 2 — 0.02%
Topic 5 — 0.01%
Topic 6 — 0.01%
Topic 1 — 0.01%
Topic 8 — 0.01%
Topic 10 — 0.01%

Topics Heatmap



Processing documents: 100%|██| 306242/306242 [00:30<00:00, 10103.20it/s]