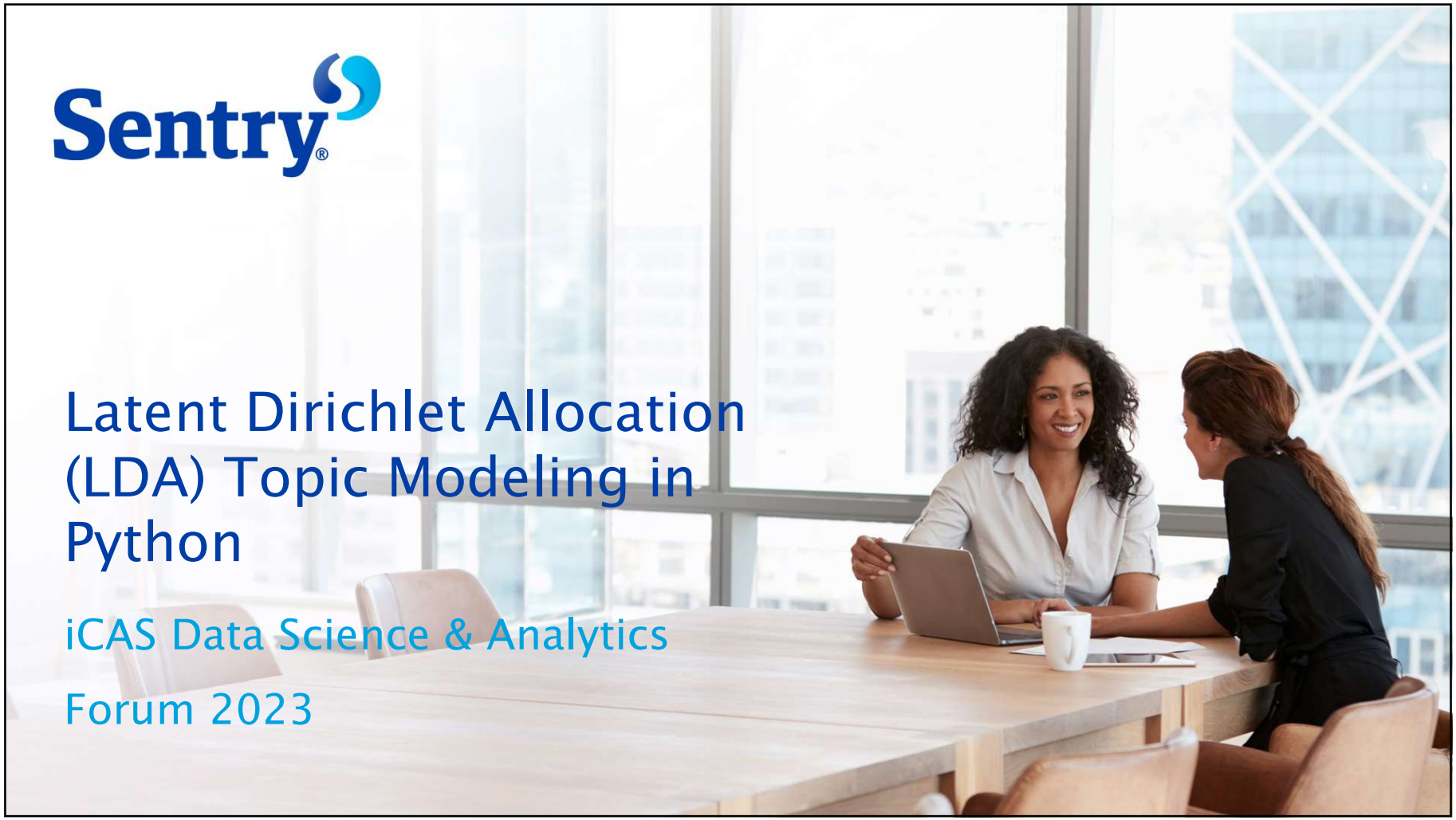

Sentry®
Latent Dirichlet Allocation (LDA) Topic Modeling in Python
iCAS Data Science & Analytics
Forum 2023

1

Agenda

What is Latent Dirichlet Allocation?

- Methodology
- Model Outputs
- Performance Metrics

Applying LDA in Practice

- Project Overview
- Data Pre-processing
- LDA Model Set-up & Evaluation
- Topic Results

What is Latent Dirichlet Allocation?

What is LDA?

- A natural language processing (NLP) technique for observing all words in a document and producing a topic distribution
- Introduced in 2003 by David Blei, Andrew Ng, and Michael I. Jordan
- A generative probabilistic model of a collection of documents that assumes each document is a mixture of topics, and each topic is characterized by a distribution of words
- Topics reside within a hidden or "latent" layer
- Probabilities are observed using Dirichlet distributions

Terminology

- **Word:** a single word denoted in a document
- **Lemma:** *main form* of a word; several words can map to a single lemma
  - Ex. {follow, followed, following, follows} map to lemma {follow}
- **Document:** a sequence of words
- **Corpus:** a collection of documents
- **Dictionary:** collection of top lemmas we want to include in the analysis, and the word-to-lemma mapping for each
- **Topic:** A list of words with the probabilities that the word belongs to that topic

LDA Graphical Model Representation

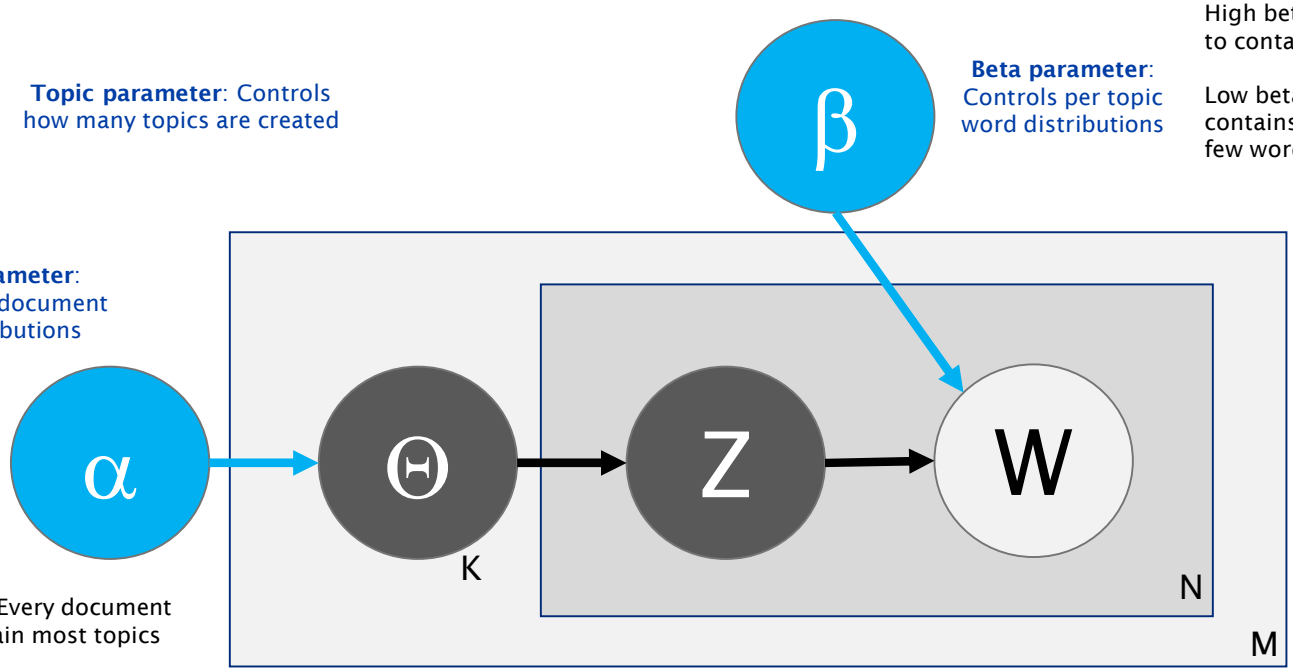
Topic parameter: Controls how many topics are created

Beta parameter: Controls per topic word distributions

High beta = Each topic likely to contain most words

Low beta = Each topic contains mixture of just a few words

Alpha parameter: Controls per document topic distributions



High alpha = Every document likely to contain most topics

Low alpha = Every document likely to represent just a few topics

N words
M documents
K Topics

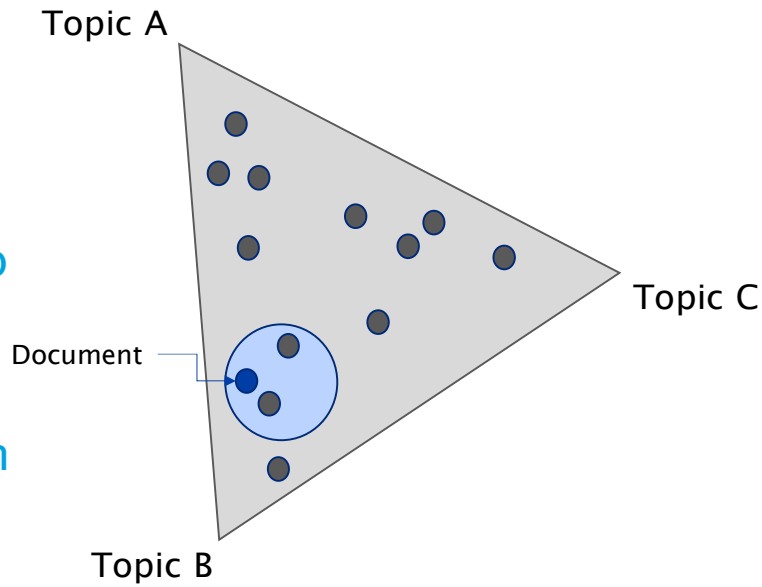
Topic distribution for document M_i

Topic for word w_j in document M_i

Word in document M_i

LDA Space

- LDA space is a simplex (a triangle of arbitrary dimensions)
- Dimensionality of the space depends on the number of topics
- Each topic is a corner of the simplex
- Not easy to visualize with many topics
- The more a topic is represented in a document, the closer the document is to that topic's corner
- Can use Jensen-Shannon Divergence to measure distance between documents in LDA space to find similar documents



The Math

$$P(W, Z, \theta, \varphi, \alpha, \beta) = \prod_{i=1}^{K} P(\varphi_i ; \beta) \prod_{j=1}^{M} P(\theta_j ; \alpha) \prod_{t=1}^{N} P(Z_{j,t} | \theta_j) P(W_{j,t} | \varphi_{Z_{j,t}})$$

Suppose we have D documents and we assume K topics:

- For each topic 1…k:
  - Draw a multinomial over words $\varphi \sim Dir(\beta)$
- For each document 1…d:
  - Draw a multinomial over topics $\Theta \sim Dir(\alpha)$
  - For each word w_{N_d}:
    - Draw a topic $Z_{N_d} \sim Mult(\Theta_D)$ with $Z_{N_d} \varepsilon [1 \ldots K]$
    - Draw a word $w_{N_d} \sim Mult(\varphi)$

D = total # of documents

$\mathbf{N_d}$ = total # of lemmas in a given document

α = Dirichlet prior on the per-document topic distribution

β = Dirichlet prior on the per-topic word distribution

Θ_i = topic distribution for document i

φ_i = word distribution for topic k

$\mathbf{z_{ij}}$ = topic for the jth word in document i

$\boldsymbol{w_{ij}}$ = specific word

Model Outputs

- Keywords for each topic and the weight of each keyword within the topic
- Topic(s) that each document belongs to with percentage
  - Example – 50% Topic A, 30% Topic B, 20% Topic C
  - Since this is a probability distribution, will always add up to 100%
- Topic(s) each word belongs to and its phi value
  - Phi value = probability the word belongs to that particular topic

Model Performance Metrics

- Model Perplexity – The normalized log-likelihood of a hold-out test set. (i.e., how well does the model represent or reproduce the statistics of the hold-out data)
  - Perplexity and human judgment are often not correlated
  - Optimizing for perplexity often not yield interpretable topics
- Topic Coherence – Measures the degree of semantic similarity between high scoring words in a topic
  - Uses methodology developed by Michael Order, Andreas Both, Alexander Hinneburg
  - Multiple variations of coherence measures available in Python Gensim package

Applying LDA in Practice

Project: Claims Self-Service Opportunities

- Goal – Identify reasons why claim participants call claim reps to prioritize build out of self-service and/or proactive communication functionality
- Phone call transcripts were not available, but voicemail transcripts were able to be accessed
  - ~40% of claim rep calls are sent to voicemail, likely credible sample of types of calls
  - Voicemails short messages that should contain relatively focused topics
- Used gensim package in Python to create LDA Topic Models
- Used pyLDAvis to visualize topics

Overview of Data

Voicemail Data

- Sample of ~110,000 claims voicemail transcripts
- Some pre-processing done to remove PII and identify claim numbers
  - Could identify claim number in transcript for ~60% of transcripts

Claim Data

- Joined claim information to voicemail transcript data
  - Loss date, FNOL date, claim type, line of business, etc.
- Joined caller's participant role (when available)
  - Vehicle owner, insured, counsel, repair facility, etc.

Final Data Selected for LDA

- Personal lines auto claims transcripts: ~43,000 records

Data Preprocessing and Building Dictionary

Text Pre-Processing

- Convert voicemail transcripts into list of tokens
  - Lowercase, tokenize, and de-accent words in a document
- Remove stop words
  - Words like “the”, “and”, “hi”, etc.
- Create bigram model and add found bigrams to list of tokens
  - Detects two words frequently together that can be used as additional tokens in the LDA model
- Lemmatize tokens keeping only nouns, adjectives, verbs, and adverbs
- Create dictionary, corpus, and term document frequency list

Text Pre-Processing – Python Code

```
############### Tokenize words ###############
def words(transcripts):
    for transcript in transcripts:
        yield(gensim.utils.simple_preprocess(str(transcript), deacc=True))
        # deacc=True removes punctuations

Transcript_Data_List = Transcript_Data.Transcript.values.tolist()
Transcript_Data_Words = list(words(Transcript_Data_List))
print(Transcript_Data_Words[:1])
print('Done')
```

```
############### Bigram Model ###############
#Bigrams are two words frequently occurring together in the document.
bigram=gensim.models.Phrases(Transcript_Data_Words, min_count=5, threshold=100)
# Faster way to get a sentence clubbed as a bigram
bigram_mod = gensim.models.phrases.Phraser(bigram)
print('Done')
```

Text Pre-Processing – Python Code Cont'd

```
# Define functions for stopwords, bigrams and lemmatization
def remove_stopwords(texts):
    return [[word for word in simple_preprocess(str(doc)) if word not in stop_words] for doc in texts]

def make_bigrams(texts):
    return [bigram_mod[doc] for doc in texts]

def lemmatization(texts, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):
    """https://spacy.io/api/annotation"""
    texts_out = []
    for sent in texts:
        doc = nlp(" ".join(sent))
        texts_out.append([token.lemma_ for token in doc if token.pos_ in allowed_postags])
    return texts_out

# Remove Stop Words
Transcript_Data_Words_Nostops=remove_stopwords(Transcript_Data_Words)

# Form Bigrams
Transcript_Words_Bigrams=make_bigrams(Transcript_Data_Words_Nostops)

# Initialize spacy 'en' model, keeping only tagger component (for efficiency)
nlp=spacy.load("en_core_web_sm", disable=['parser', 'ner'])

# Do lemmatization keeping only noun, adj, vb, adv
Transcript_lemmatized = lemmatization(Transcript_Words_Bigrams, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV'])
print(Transcript_lemmatized[:1])
print('Done')
```

Text Pre-Processing – Python Code Cont'd

```
############### Corpus and Dictionary ###############
# Create Dictionary
id2word = corpora.Dictionary(Transcript_lemmatized)

# Create Corpus
texts = Transcript_lemmatized

# Term Document Frequency
corpus = [id2word.doc2bow(text) for text in texts]
# View
print(corpus[:1])
print('Done')
```

Raw Transcript Data to Lemmatized Data

Examples are illustrative only and do not represent actual voicemails

Voicemail ID	Original Transcript	Lemmatized Token List
1	Hello. This is X calling regarding claim number XXXXXXX. regarding for the Uh, I was giving you guys a call to find out what's going on. I called the appraiser, um, trying to figure out we get paid on this file. If you guys can give me a call back. Thank you.	['call', 'collision', 'regard', 'claim', 'number', 'regard', 'give', 'call', 'find', 'go', 'call', 'time', 'appraiser', 'try', 'figure', 'get', 'pay', 'file', 'give', 'call', 'thank']
2	uh hi. This is X calling for claim XXXXXXXX. My phone number is (XXX) XXX XXXX. Uh I am calling regarding uh Honda Civic with Century insurance policy of XXXXXXXX. This vehicle struck my parked car while while it was unoccupied on Saturday and I'm just trying to get this taken care of. I filed the claim yesterday and I would like to hear back. thank you very much. Bye bye.	['calling', 'claim', 'phone', 'number', 'call', 'regard', 'chrysler', 'century', 'century', 'insurance', 'policy', 'vehicle', 'strike', 'park', 'car', 'unoccupied', 'try', 'get', 'taken_care', 'file', 'claim', 'yesterday', 'love', 'hear', 'thank', 'much']
3	Hi X, this is X. You were helping me with my claim regarding the white Ford Explorer. Um The claim number is XXXXXXXX My phone number is XXXXXXXXXX. I was calling in regards to the check that I was supposed to have received 3 to 5 business days after sending in the title of the car. Um It's been about a 2 weeks now and I still haven't received it. So I was just wondering if you would be able to help me. Thank you so much.	['help', 'claim', 'regard', 'white', 'claim', 'number', 'phone', 'number', 'call', 'regard', 'check', 'suppose', 'receive', 'business', 'day', 'send', 'title', 'car', 'week', 'still', 'receive', 'wonder', 'able', 'help', 'thank', 'much']
4	Hello, This is X calling with X. It was giving you a call regarding claim number. XXXXXXXX. Um, I'm trying to get proof of payment for this customers vehicle is complete upfront. Um, can you please give me a call back at XXX XXX XXXX. Thank you.	['call', 'give', 'call', 'regard', 'claim', 'number', 'try', 'get', 'proof', 'payment', 'customer', 'vehicle', 'complete', 'upfront', 'give', 'call', 'thank']
5	Good afternoon. This is X claim number XXXXXXX. Uh I wanted to see with the new estimate that came in uh for the supplement. Is it possible to get covered for the rental car rental? Um As my car is in a repair shop, I need a vehicle. I was wondering if um that would be covered. So please give me a call tomorrow. XXX XXX XXXX.	['afternoon', 'claim', 'number', 'want', 'new', 'estimate', 'come', 'supplement', 'possible', 'get', 'cover', 'rental', 'car', 'rental', 'car', 'repair', 'shop', 'need', 'vehicle', 'wonder', 'cover', 'give', 'call', 'tomorrow']

LDA Model Set-up & Evaluation

LDA Model Parameters

```
# Build LDA model
LDA_Model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                            id2word=id2word,
                                            num_topics=13,
                                            random_state=100,
                                            chunksize=1000,
                                            passes=10,
                                            alpha='auto',
                                            #eta=0.3,
                                            per_word_topics=True
                                               )
```

- **Corpus** – term document frequency output
- **Id2word** – mapping from word IDs to words
- **num_topics** – number of topics that should be created
- **random_state** – seed for reproducibility
- **chunksize** – number of documents to consider at once (affects memory consumption)
- **passes** – number of passes through corpus during training
- **alpha** – scalar for a symmetric prior over document-topic distribution
- **eta** (beta) – scalar for a symmetric prior over topic-word distribution
- **per_word_topics** – if true allows for extraction of the most likely topics given a word

Tuning Hyperparameters

- Hyperparameters alpha and beta (called 'eta' in gensim)
  - Both alpha and beta can be set to 'symmetric', 'asymmetric', 'auto', or an exact number of your choosing. The default for both is a symmetric distribution of 1 divided by the number of topics.
  - For 'auto' the model tries to learn the "best values" for the hyperparameters as it is training on more and more data, but it does take longer to compute. We also found that the auto option sometimes seemed to overfit
- Number of topics most important parameter besides alpha and beta

Tuning Hyperparameters

- What works best? The standard answer "it depends on the data" applies
- Initially, we focused on the Coherence metric

```
# Compute Coherence Score
Coherence_Model_LDA = CoherenceModel(model=LDA_Model, texts=Transcript_lemmatized, dictionary=id2word, coherence='c_v')
Coherence_LDA = Coherence_Model_LDA.get_coherence()
print('\nCoherence Score: ', Coherence_LDA)
print('Done')
```

- Used pyLDAvis to visualize topics
  - Reviewed topic terms interpretability
  - Reviewed topic overlap
  - Edited stop words
- …and ran many, many models

Tuning Hyperparameters

- Started with a grid search approach, looping through different combinations, looking for highest Coherence
- Reviewed topics in pyLDAvis for top Coherence models
- Initially, liked a draft with 7 topics, but after reviewing with stakeholders agreed that more topics were desired
- Final model chosen had lower Coherence, but more topics that were interpretable and distinct from each other
- "Winning" model used alpha = auto, beta = symmetric, topics = 13, and had Coherence = 0.5042

Validation_Set	Topics	Alpha	Beta	Coherence
75% Corpus	9	0.34	1	0.590252213
75% Corpus	9	0.01	0.67	0.56708697
75% Corpus	10	0.67	1	0.56678607
75% Corpus	10	0.34	0.67	0.56542286
75% Corpus	10	0.01	0.67	0.564641798
75% Corpus	10	0.34	1	0.558610027
75% Corpus	7	0.34	0.67	0.555730214
75% Corpus	10	0.01	0.34	0.554257578
75% Corpus	9	0.01	0.34	0.54917702
75% Corpus	10	0.34	0.34	0.536206629
75% Corpus	8	0.34	0.67	0.53253265
75% Corpus	9	0.34	0.34	0.527283105
75% Corpus	9	0.34	0.67	0.527220787
75% Corpus	7	0.34	0.34	0.526368068
75% Corpus	9	0.67	1	0.525038443
75% Corpus	8	0.34	0.34	0.523669254
75% Corpus	9	0.01	0.01	0.520818033
75% Corpus	7	0.01	0.67	0.520258445
75% Corpus	7	0.01	1	0.51987567
75% Corpus	8	0.34	1	0.518647014
75% Corpus	10	1	0.67	0.517457933
75% Corpus	9	1	1	0.513690436
75% Corpus	7	0.34	1	0.512527518
75% Corpus	10	1	1	0.50922247
75% Corpus	10	0.67	0.67	0.505574766
75% Corpus	8	0.01	0.67	0.503395281
75% Corpus	8	0.67	1	0.503310269
75% Corpus	10	1	0.34	0.503285777
75% Corpus	9	0.67	0.67	0.501942944
75% Corpus	10	0.01	0.01	0.501028854

Partial results from grid search

Visualizing Topics

- pyLDAvis is a topic model visualization interface that allows the user to look closely at the make up of topics and the words associated with topics
- Ben Mabey created a port of the R package originally created by Carson Sievert and Kenny Shirley

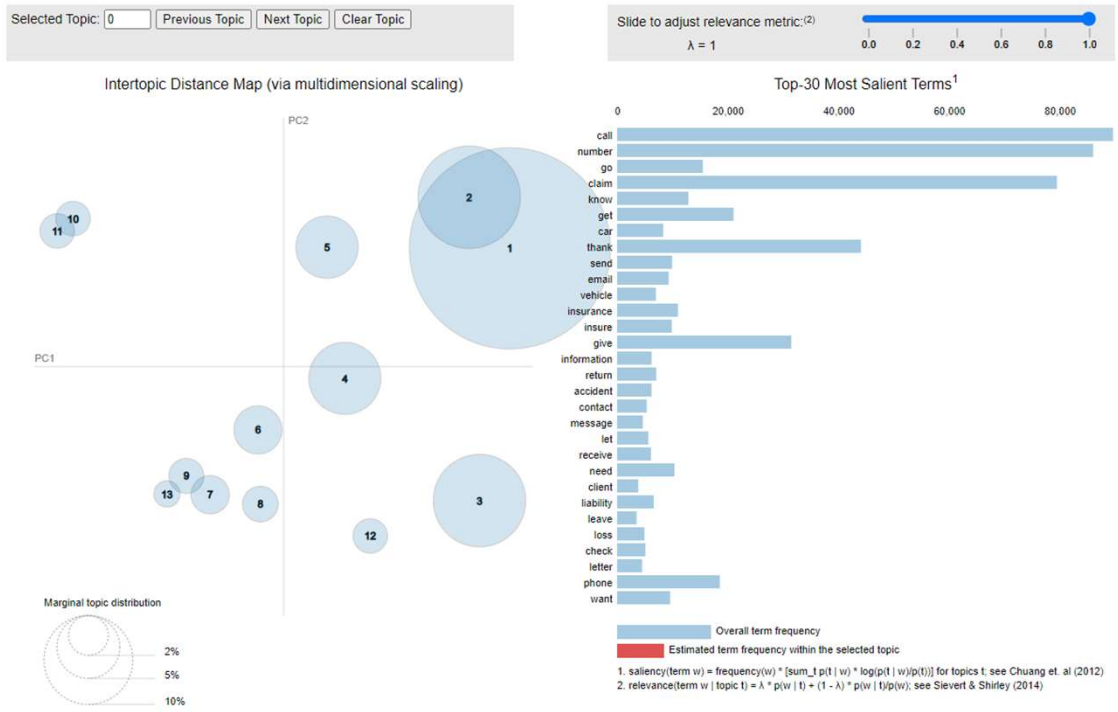
```
# Visualize the topics
pyLDAvis.enable_notebook()
num_topics=13
LDAvis_data_filepath = os.path.join('filename'+str(num_topics))
# # this is a bit time consuming - make the if statement True
# # if you want to execute visualization prep yourself

if 1 == 1:
    LDAvis_prepared = gensimvis.prepare(LDA_Model, corpus, id2word)
    with open(LDAvis_data_filepath, 'wb') as f:
        pickle.dump(LDAvis_prepared, f)
# load the pre-prepared pyLDAvis data from disk
with open(LDAvis_data_filepath, 'rb') as f:
    LDAvis_prepared = pickle.load(f)
pyLDAvis.save_html(LDAvis_prepared, 'filename'+ str(num_topics) +'.html')
LDAvis_prepared
#print('Done')
```

Visualizing Topics

pyLDAvis

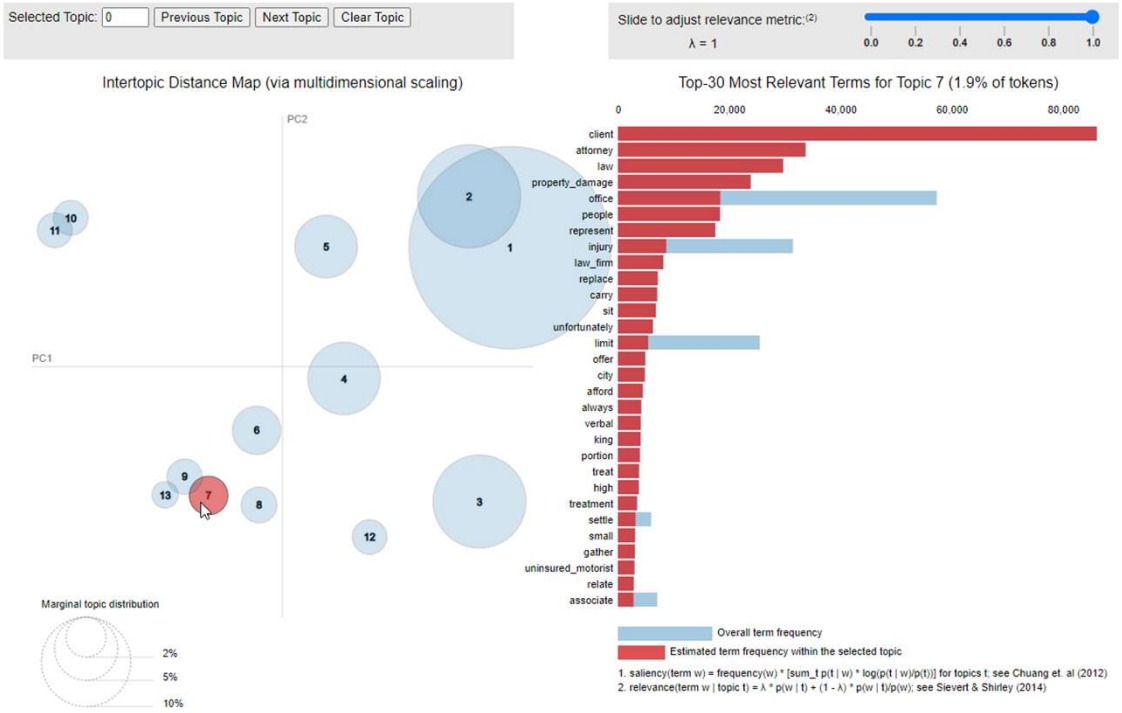
- Each bubble represents a topic. Larger bubbles mean a higher % of documents are about that topic.
- The farther the bubbles are away from each other, the more different they are.
- Documents are often tagged with multiple topics, so the bubbles are not mutually exclusive
- Blue bars represent the overall frequency of each word in the corpus



Visualizing Topics

pyLDAvis

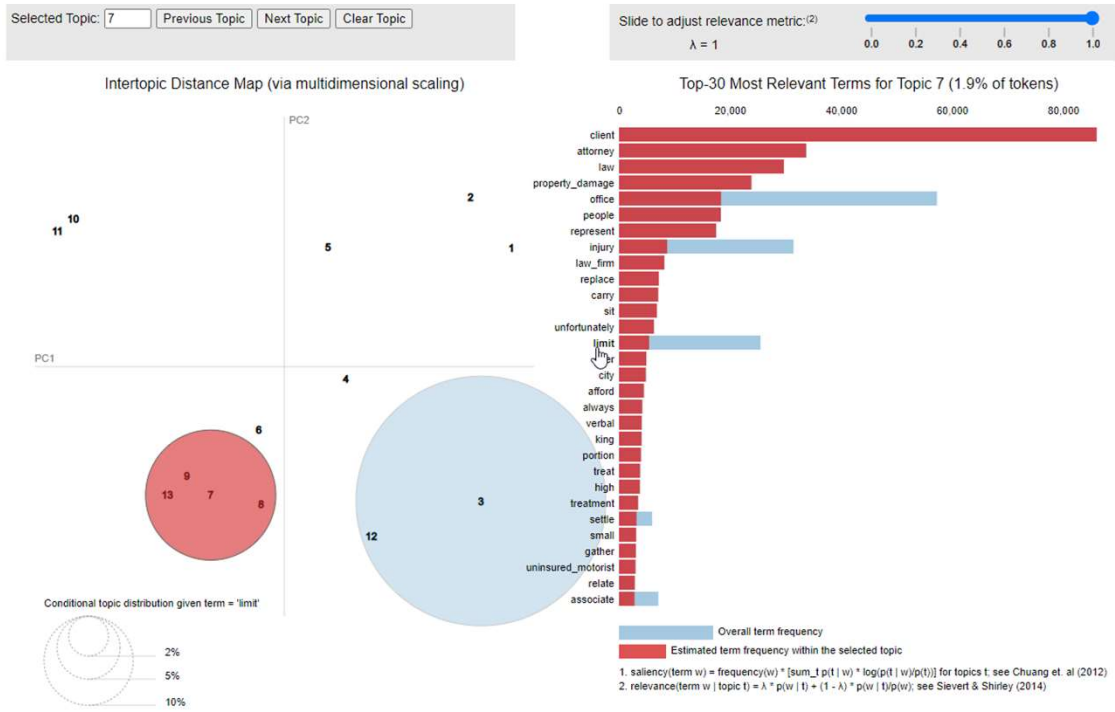
- Red bars gives the estimated number of times a word was generated by a given topic
- Words with the longest red bars were used the most by documents belonging to the selected topic
- Relevance metric slider controls how words are sorted
  - "1" sorts words by their frequency in topic
  - "0" sorts words by how much their frequency is above baseline



Visualizing Topics

pyLDAvis

- When a topic is selected, hovering over a word in the right panel will change the left panel to show other topics in which that word is prominent



28

Topic Results

Topic Results

Topic Number	Topic Name	Top 30 Keywords	% Documents
1	Generic	call, number, claim, thank, give, phone, name, regard, day, back, get, much, return, reach, want, try, information, soon, accident, speak, reference, morning, question, time, contact, update, message, hear, possible, last	100%
2	Estimate/Picture	get, need, say, try, work, talk, take, make, sure, tell, really, right, estimate, come, picture, already, actually, able, hold, help, time, person, way, kind, today, guess, little, start, collision, company	78%
3	Other Insurance Companies	insurance, insure, liability, coverage, loss, policy, look, reference, insurance_co_1, extension, insured, date, status, file, follow, want, confirm, claim, adjuster, insurance_co_2, reach, mutual, handle, verify, end, state, decision, accept, provide, first_notice	59%
4	Payments/Documents	send, email, receive, letter, check, address, payment, fax, copy, request, issue, mail, release, demand, denial, respond, response, spell, amount, sign, supplement, due, include, easy, document, state, form, century, directly, title	25%
5	Follow Up	go, know, let, want, ahead, still, wait, week, month, card, agent, recently, proceed, hear, least, course, paper, representative, fill, town, pay, around, court, lawyer, add, explain, plan, almost, feel, run	15%
6	Damages	vehicle, accident, damage, driver, happen, report, involve, statement, fault, incident, side, owner, deductible, front, police, passenger, video, clear, rear_ende, park, party, pull, stop, steal, information, rear, license, permission, photo, motor	6%
7	Medical Provider	service, patient, bill, medical, hospital, charge, anytime, open, date, account, record, translator, locate, member, med_pay, behalf, birth, inquire, health, event, billing, chiropractic, reference, doctor, main, provider, country, renter, build, hotel	4%
8	Legal	client, attorney, law, property_damage, office, people, represent, injury, law_firm, replace, carry, sit, unfortunately, limit, offer, city, afford, always, verbal, king, portion, treat, high, treatment, settle, small, gather, uninsured_motorist, relate, associate	3%
9	FNOL	new, contact, notice, information, file, list, frozen_first, insure, loss, return, obtain, representation, first, option, info, appear, benefit, report, lunch, injure, spanish_speaker, voice, frozen, golf, usaa, line, fourth, segregation, rd, interpreter	3%
10	Car Rental/Tow	car, rental, hit, cover, tow, enterprise, deny, set, damage, inspection, light, storage, green, bumper, download, allow, supervisor, run, reservation, stay, cut, potentially, recovery_unit, arrange, due, zack, spend, evidence, stick, traffic	3%
11	Insurance Company X	leave, message, voicemail, insurance_co_X, spanish, husband, left, business, next_step, base, lane, safe, word, power, fast, working, exchange, hand, pocket, period, indicate, pursue, wanting, documentation, somewhere, shortly, willing, concerned, anymore,	2%
12	Requesting Further Info	car, fix, text, pick, next, daughter, schedule, appointment, wife, week, away, cancel, road, drop, lady, job, evening, cost, head, crash, mother, vacation, extend, drivable, seat, step, middle, visit, facility, home	2%
13	Auto Repair/Lienholder	repair, shop, body, money, cause, approve, problem, windshield, paint, house, lien_hold, approval, appraisal, drive, pain, tire, fee, credit, lien_holder, opportunity, rather, bear, salvage, giving, value, mechanic, deliver, repairable, lake, waste	1%

Topic – Other Insurance Companies

Examples are illustrative only and do not represent actual voicemails

Transcript	Topic	Document Topic %
Okay. Hello, this is X insurance calling regarding a liability decision as well as to verify if there were any injuries in your insurance vehicle. And to confirm if the policy is active on the date of loss for the claim number XXXXXXXX. The claim number is XXXXXXXX. The adjuster could be reached at XXX XXX XXXX. Thank you. And have a nice day	Other Insurance Companies	47.5%
Hi X. This is X with X Insurance. Um I'm calling on claim number XXXXXXXX for insured X . Uh Date of loss was September 5th of this year. Um The policy number is XXXXXXXX. I'm just calling to verify that um that policy was active on the data loss um for X. Um I also have a driver of X. So I also need to verify if they were um included on the policy or excluded. You can call me back at XXX XXX XXXX.	Other Insurance Companies	47.2%
Hi X. Uh My name is X with uh X insurance. And I was just calling on your claim number XXXXXXX. And I just wanted to get a status on this claim. Just to confirm if there was coverage for this data loss that occurred on uh March nine of this year. The driver of uh your insured driver is a uh excuse me. X. And I was just confirming that he was insured with um X at the time of this loss. Uh if you could give me a call, my phone number is (XXX) XXX XXXX. In reference to my claim number of XXXXXXXX.	Other Insurance Companies	46.3%

Topic – Car Rental/Tow

Examples are illustrative only and do not represent actual voicemails

Transcript	Topic	Document Topic %
Hi X, this is X. My claim number is XXXXXXXX. Um I was wondering if you guys cover the rental car. Um X insurance is saying I have to pay out of pocket for the rental car, but they're reimbursing me. So I was wondering if I could go through you guys, I did find a rental car place That does have a rental car for $35 a day. Um if you could please give me a call back, at (XXX) XXX XXXX. Bye.	Car Rental/Tow	25.1%
Hello. Hi X. My name is X. I'm calling in regards to claim number XXXXXXXX. I'm calling uh because I uh I was finally able to find a place that has openings because everyone is so booked out. They, said they could I could bring the car over there Tuesday or the day after. Uh I just want to give the rental car near their place. I car. Um So please give me a call on how we're going to be with the uh rental car. Uh XXX XXX XXXX. I'm sorry. XXX XXX XXXX. Thank you.	Car Rental/Tow	23.8%
This X. The claim number XXXXXXXX. my telephone number is XXX XXX XXXX. I've got my car set up to be repaired. Uh And I was wondering what the process. If y'all had a rental car repair Call that your convenience. XXX XXX XXXX. Again, this is X Claim number XXXXXXXX. Uh and I'm just trying to set up a rental car. I've got the repair Uh for the 2019 Toyota set up for November second. So I was trying to see what uh what the procedure is to request a rental car. Thank you.	Car Rental/Tow	23.0%

Project Summary

Project Summary

- LDA model did allow for categorization of voicemail transcripts & overall better understanding of calls to claim reps
- Results not always as detailed as we would like
  - Needed to review transcripts after identifying high level topic to determine details of reasons for call
- Possible future improvements
  - Remove more generic terms from stop list
  - Build models split by claim participant
  - Build models split by time in claim lifecycle the call occurred

Questions & Thank you

References

References

"API - Gensim¶." *API - Gensim*, 2016, https://tedboy.github.io/nlps/api_gensim.html.

Blei, David M, et al. "Latent Dirichlet Allocation - Journal of Machine Learning Research." *Journal of Machine Learning Research*, Jan. 2003, https://jmlr.org/papers/volume3/blei03a/blei03a.pdf.

Fairless, Andrew. "Parameter Testing." *Andrew Fairless PhD*, https://afairless.com/the-peanuts-project/topic-modeling/parameter-testing/.

Fg, Pawang. "Latent Dirichlet Allocation." Edited by Seth Anika, *GeeksforGeeks*, GeeksforGeeks, 6 June 2021, https://www.geeksforgeeks.org/latent-dirichlet-allocation/.

Kapadia, Shashank. "Evaluate Topic Models: Latent Dirichlet Allocation (LDA)." *Towards Data Science*, 19 Aug. 2019, https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0.

Kapadia, Shashank. "Topic Modeling in Python: Latent Dirichlet Allocation (LDA)." *Towards Data Science*, 14 Apr. 2019, https://towardsdatascience.com/end-to-end-topic-modeling-in-python-latent-dirichlet-allocation-lda-35ce4ed6b3e0.

Knispelis, Andrius. "What Is Latent Dirichlet Allocation (LDA) - Latent Dirichlet Allocation (LDA) Definition from MarketMuse Blog." *MarketMuse Blog*, 8 June 2021, https://blog.marketmuse.com/glossary/latent-dirichlet-allocation-definition/#:~:text=Latent%20Dirichlet%20Allocation%20(LDA)%20is,known%20as%20a%20latent%20layer.

References Cont'd

Mabey, Ben, and Paul English. "Welcome to PyLDAvis's Documentation!¶." *Welcome to PyLDAvis's Documentation! - PyLDAvis 2.1.2 Documentation*, 6 Feb. 2018, https://pyldavis.readthedocs.io/en/latest/.

Rehurek, Radim. "Gensim: Topic Modelling for Humans." *Radim Rehurek: Machine Learning Consulting*, 21 Dec. 2022, https://radimrehurek.com/gensim/index.html.

Roder, Michael, et al. *Exploring the Space of Topic Coherence Measures*. AKSW Research Group @ Univerity of Leipzig, 2015, http://svn.aksw.org/papers/2015/WSDM_Topic_Evaluation/public.pdf.

"TMO Guide (3) – Pyldavis – WE1S." *Topic Model Observatory Guide Section 3*, What Every 1 Says - A 4Humanities Project, 30 May 2019, https://we1s.ucsb.edu/research/we1s-tools-and-software/topic-model-observatory/tmo-guide/tmo-guide-pyldavis/.

Zvornicanin, Enes. "Topic Modeling and Latent Dirichlet Allocation (LDA)." *DataScience+*, 30 Jan. 2022, https://datascienceplus.com/topic-modeling-and-latent-dirichlet-allocation-lda/.