Topic Modelling (LDA)

SCHOOL OF INFOCOMM

Types of Machine Learning

Supervised

Data points have known outcome. We train the model with data. We feed the model with correct answers. Model learns and is able predict new data's outcome.

Unsupervised

Data points have unknown outcome. Data is given to the model. Right answers are not provided to the model. The model makes sense of the data given to it.

Can reveal something you were probably not aware of in the given dataset.

Motivating Question: What are the emails about?

During her tenure as United States Secretary of State, Hillary Clinton drew controversy by using a private email server for official public communications rather than using official State Department email accounts maintained on secure federal servers. An FBI examination of Clinton's server found over 100 emails containing classified information, including 65 emails deemed "Secret" and 22 deemed "Top Secret". An additional 2,093 emails not marked classified were retroactively classified by the State Department.

The **Enron Corpus** is a large database of over 600,000 emails generated by 158 employees[1] of the Enron Corporation and acquired by the Federal Energy Regulatory Commission during its investigation after the company's collapse.[2]

Contents [show]

History [edit]

The Enron data was originally collected at Enron Corporation headquarters in Houston during two weeks in May 2002 by Joe Bartling,[3] a litigation support and data analysis contractor working for Aspen Systems, now Lockheed Martin, whom the Federal Energy Regulatory Commission (FERC) had hired to preserve and collect the vast amounts of data in the wake of the Enron Bankruptcy in December 2001. In addition to the Enron employee emails, all of Enron's enterprise database systems,[4] hosted in Oracle databases on Sun Microsystems servers, were also captured and preserved including its online energy trading platform, EnronOnline.

Topic Modelling

Large amount of data are generated and collected everyday

The goal of topic modelling is to **discover** underlying semantics structure over a large collection of text

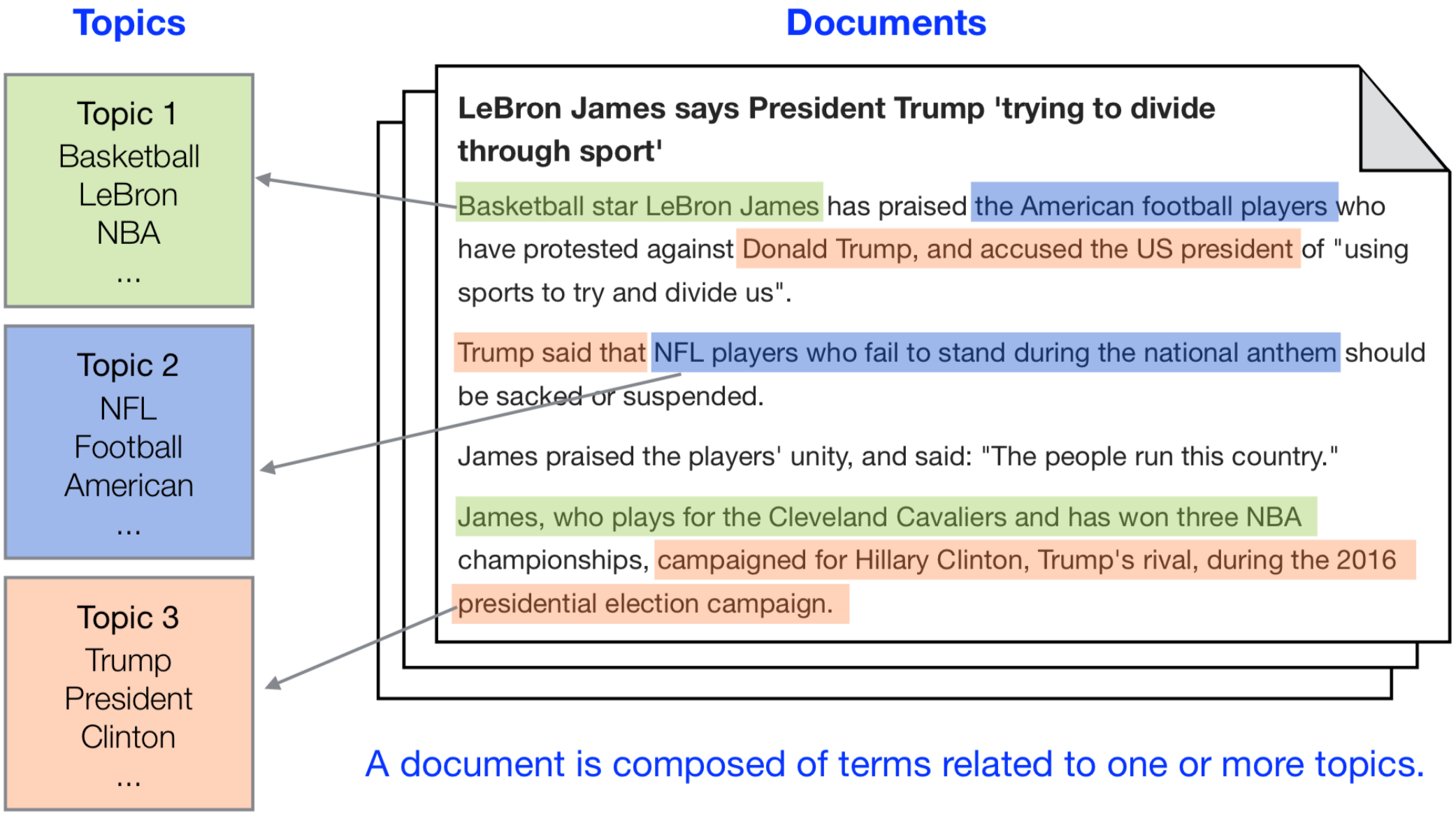
- Discovering hidden topical patterns that are present across the collection
- Annotating documents according to these topics
- Using these annotations to organize, search and summarize texts

Topics are collection of words that makes senses together

Common Terms for Accounting Topic

Common Terms for Democracy

Every document comprises of a mixture of topics



A document is composed of terms related to one or more topics.

What does a topic model looks like

Topics	Words / Terms
Topic 1	Basketball, Lebron, NBA
Topic 2	NFL, Football, American
Topic 3	Trump, President, Clinton
Topic N	

Topic-terms distribution

Document References	Topic 1	Topic 2	Topic 3	..Topic N
Doc 1	50%	20%	30%	0%
Doc 2	..	..	..	..

Document-topic distribution

Class Exercise: Create your topic model

Amazon said Thursday that its takeover of Whole Foods (WFM) will close on Monday, and its first order of business will be to make some items more affordable, according to a release.

"Whole Foods Market will offer lower prices starting Monday on a selection of best-selling grocery staples across its stores, with more to come," the company said in a statement.

Items that will be marked down on Monday include organic avocados, organic brown eggs, organic salmon, almond butter, organic apples and organic rotisserie chicken. Amazon said it'll keep the markdowns coming, and that Amazon Prime members will get additional discounts at Whole Foods.

"Everybody should be able to eat Whole Foods Market quality -- we will lower prices without compromising Whole Foods Market's long-held commitment to the highest standards," Jeff Wilke, CEO of Amazon Worldwide Consumer, said in a statement.

Source: CNN Money

Class Exercise: Possible Answers

Goal: break text documents down into "topics" by word:

Amazon said Thursday that its takeover of Whole Foods (WFM) will close on Monday, and its first order of business will be to make some items more affordable, according to a release.

"Whole Foods Market will offer lower prices starting Monday on a selection of best-selling grocery staples across its stores, with more to come," the company said in a statement.

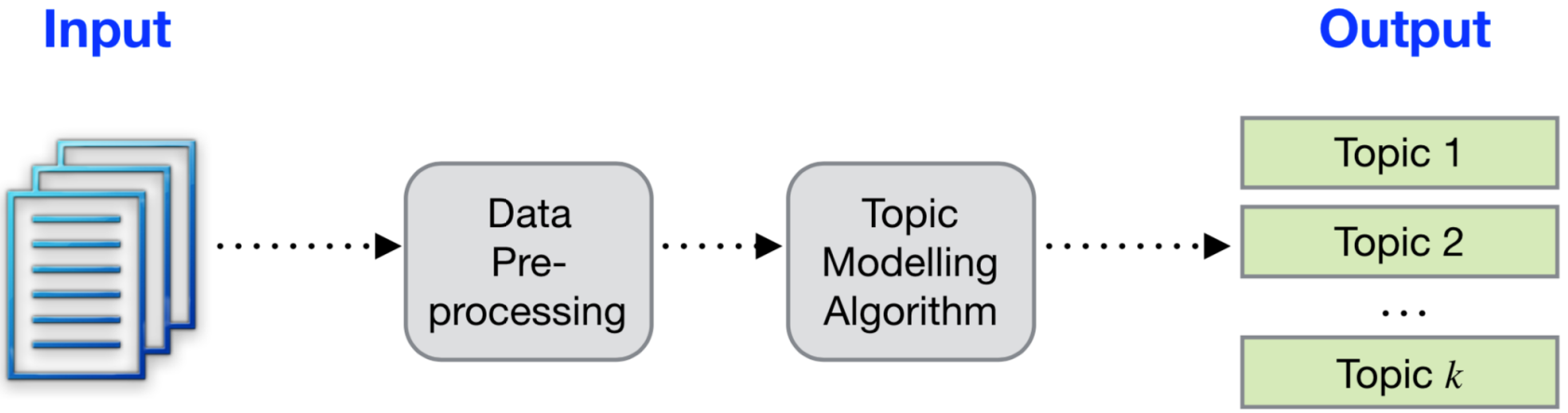
Items that will be marked down on Monday include organic avocados, organic brown eggs, organic salmon, almond butter, organic apples and organic rotisserie chicken. Amazon said it'll keep the markdowns coming, and that Amazon Prime members will get additional discounts at Whole Foods.

"Everybody should be able to eat Whole Foods Market quality -- we will lower prices without compromising Whole Foods Market's long-held commitment to the highest standards," Jeff Wilke, CEO of Amazon Worldwide Consumer, said in a statement.

Topics:

- Business
- Prices
- Food

Topic Modelling Process



- Unstructured text documents
- No labels, no annotation

- Stemming
- Stop words removal
- vectorization

- LDA
- NMF
- ISA
- ...

- A set of k topics based on related terms
- Associations from documents to topics

Algorithm: Latent Dirichlet Allocation (LDA)

LDA is one of the generative, probabilistic algorithms we can choose to convert between word and topic spaces.

It assumes a structured process of writing:

- Choose one topic to write about
- Choose words that are about that topic
- Add those words to the document
- Repeat

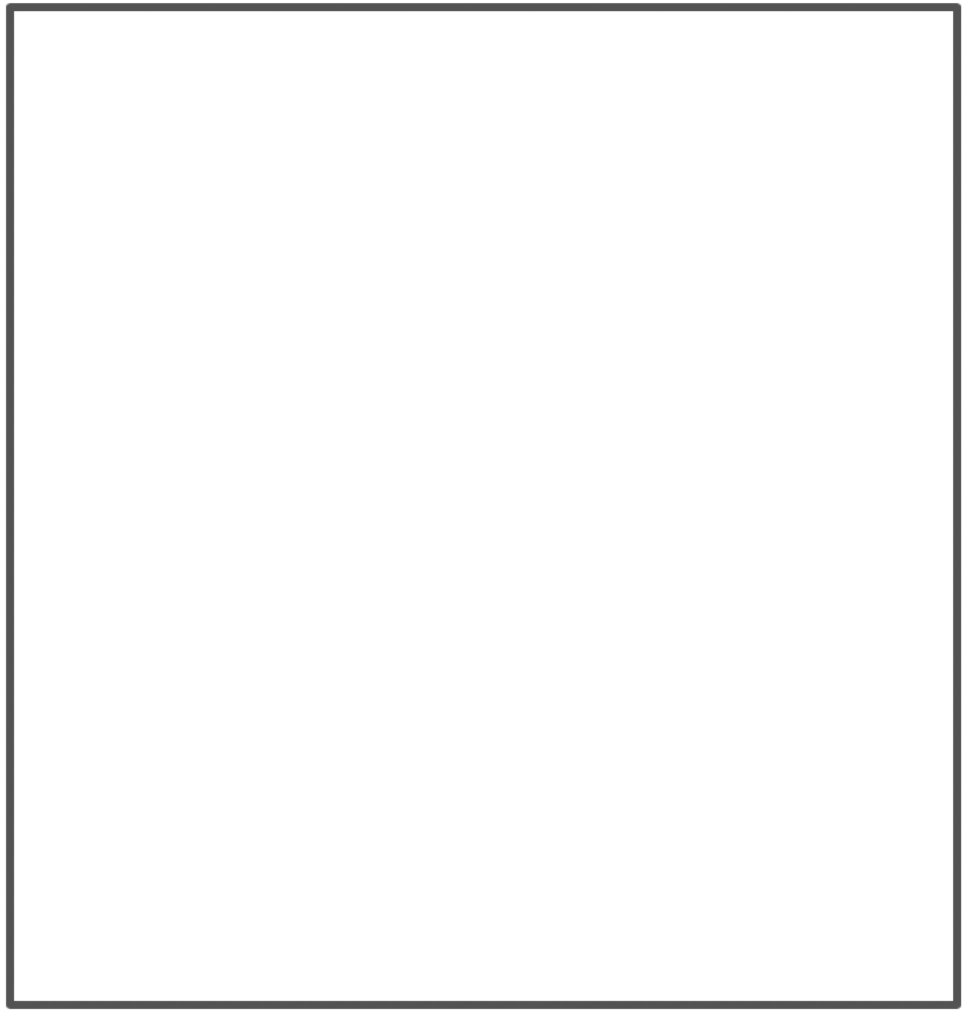
Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%



Now we roll a dice to decide which topic we start with: **Technology**

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer

Now we roll a dice to decide which topic we start with: **Technology**

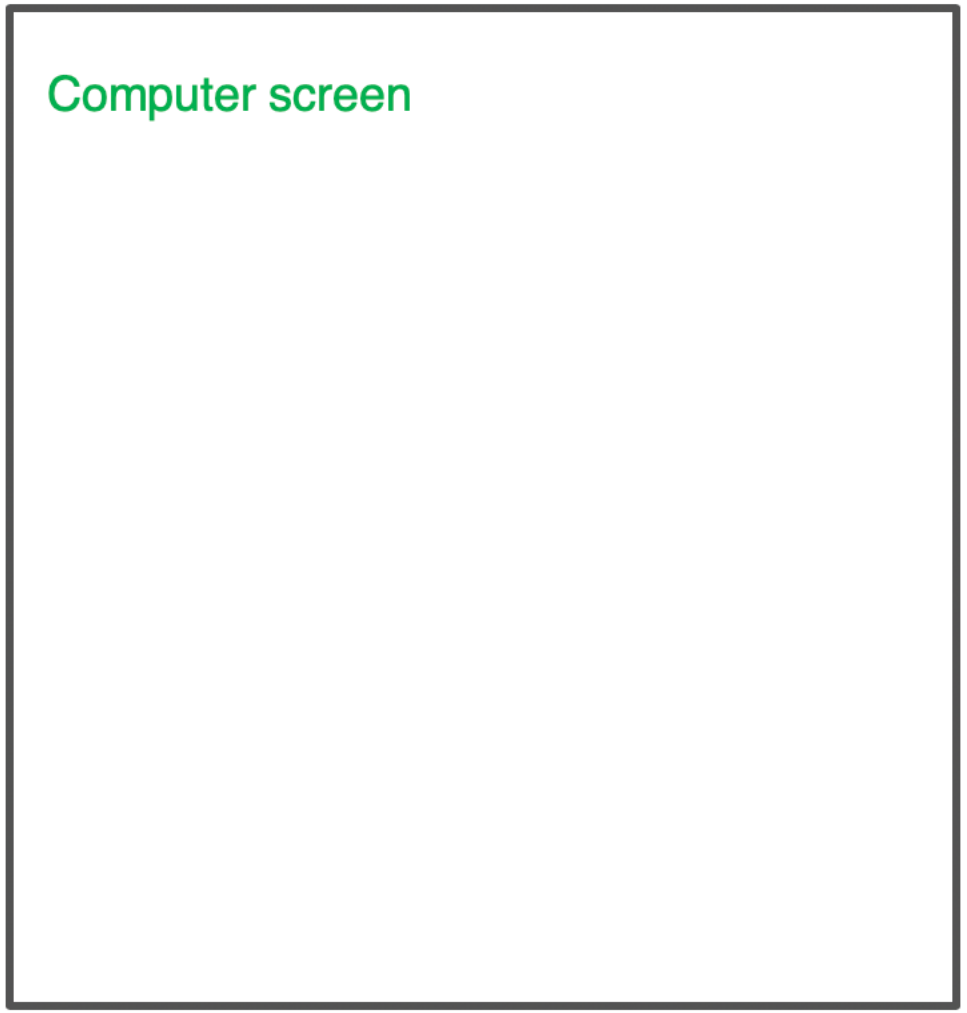
Now we roll a new dice within the technology topic to see which word and we get: **Computer**

Now we repeat the process for the next word.

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%



Now we roll a dice to decide which topic we start with: **Technology**

Now we roll a new dice within the technology topic to see which word and we get: **screen**

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the

Now we roll a dice to decide which topic we start with: **Technology**

Now we roll a new dice within the technology topic to see which word and we get: **the**

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar

Now we roll a dice to decide which topic we start with:

Music

Now we roll a new dice within the technology topic to see which word and we get: **guitar**

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar theatre

Now we roll a dice to decide which topic we start with:

Movies

Now we roll a new dice within the technology topic to see which word and we get: **theatre**

Example from Intel AI Developer Program

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar theatre headphones. Actress sings best mouse performance.

We can continue this process over and over until we've generated essentially an entire document. This document will be made up of the topics we selected, and the words that are associated with that topic, all based on the probabilities.

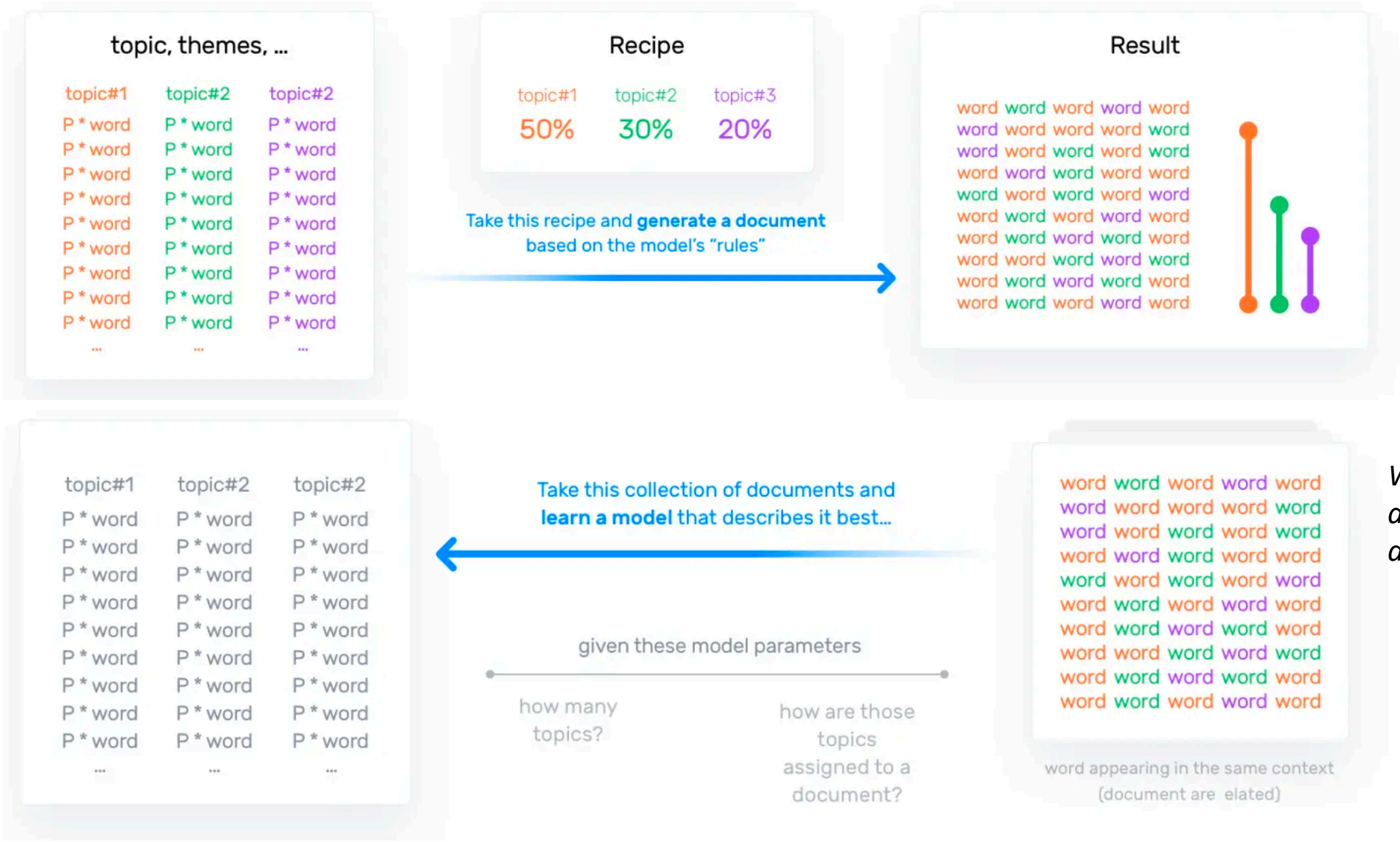
Example from Intel AI Developer Program

LDA in Practice

On the modelling side,

- Assume some topic distribution in the document
- Assume some word distribution for each topic
- Look at the corpus and try to find what topic and word distributions would be most likely to generate that corpus
- For this to converge, we need to tell it how many topics to look for

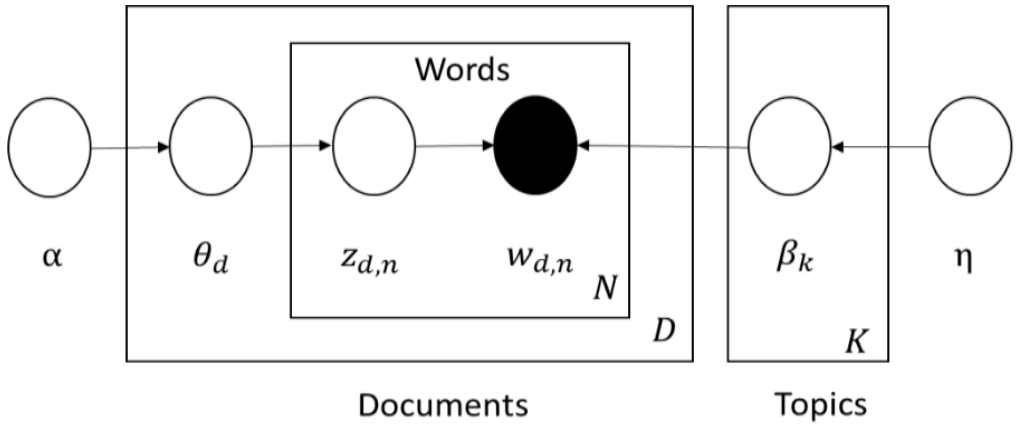
LDA - illustration

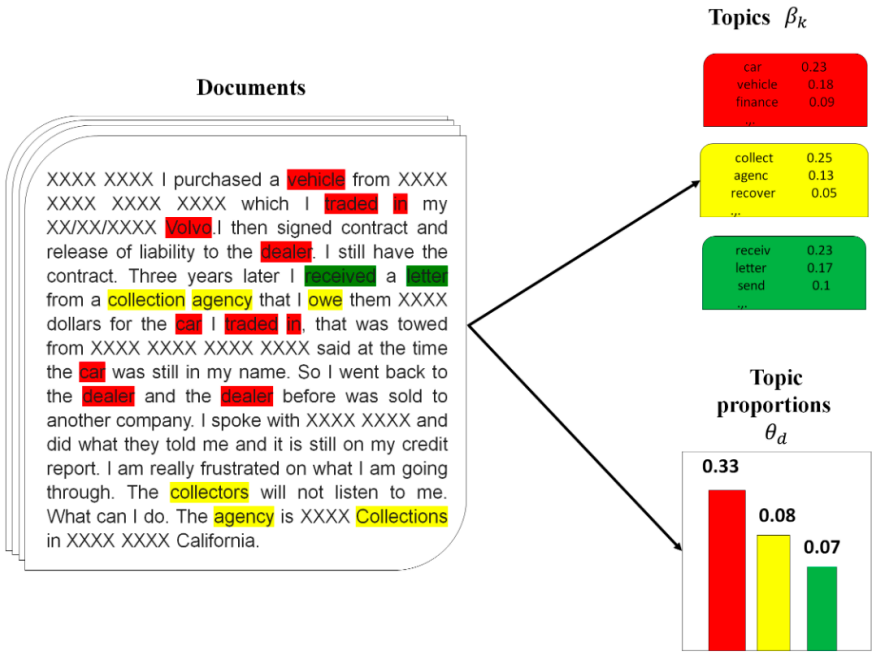


Words in the same document (context) are related

Source: monkeylearn.com/topic-analysis/

Formally ....





The corpus is a collection of D documents.
A document is a sequence of N words.
There areK topics in the corpus.
The boxes represent repeated sampling.

Observable:
Dark circle - words within documents

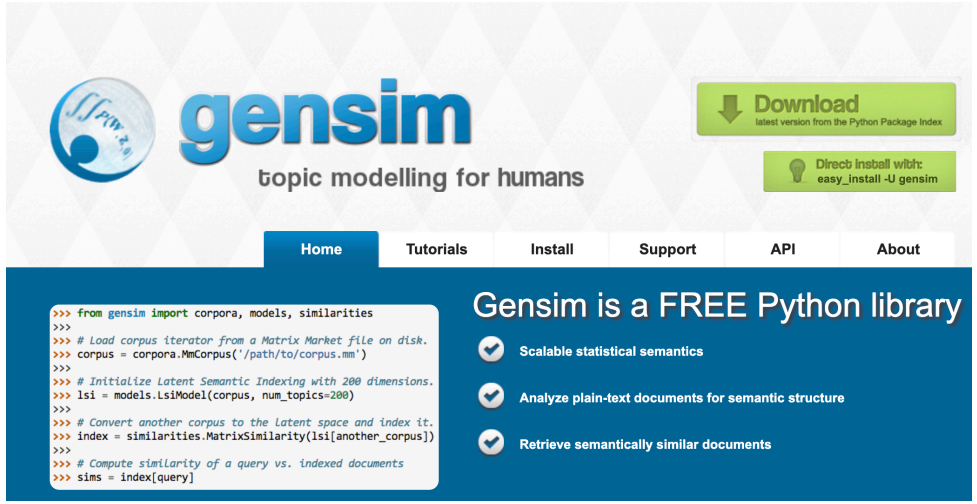
Must be inferred:
β – topics distribution over words
θ - topics distribution per document
Z - per-word topic assignment

Hyper-parameters
η is the hyperparameter for prior distributions of β
α is the hyperparameter for prior distributions of θ

$$p(z, \theta, \beta | w, \alpha, \eta) = \frac{p(z, \theta, \beta | \alpha, \eta)}{p(w | \alpha, \eta)}$$

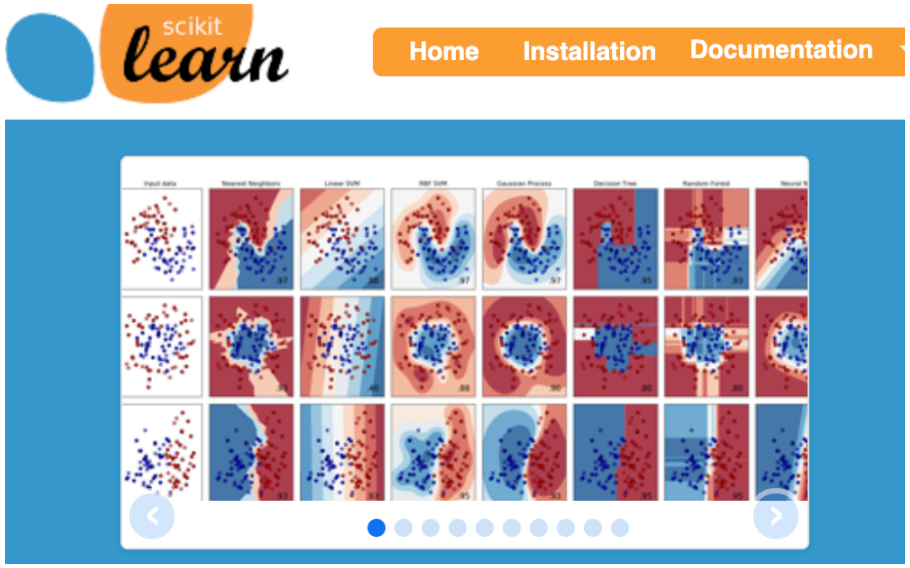
Image: https://arxiv.org/ftp/arxiv/papers/1807/1807.07468.pdf

Python Libraries for Topic Modelling



Gensim is a Python library for topic modelling, document indexing and similarity retrieval with large text corpora.

Target audience is the natural language processing (NLP) and information retrieval (IR) community.



scikit-learn is a Python libarary for machine learning for data mining and data analysis

Built on NumPy, SciPy, and matplotlib

Open-source, commercially usable - BSD license

Scikit Learn Latent Dirichlet Allocation


```
lda = LatentDirichletAllocation(n_components=10, max_iter=5,
learning_method='online', learning_offset=50., random_state=0)

document_topic = lda_model.fit_transform(document-term-vector)
topic_terms = lda_model.components_
```

n_components - number of topics
max_iter – number of iterations in the training
learning_method = Method used to update components_. Can be batch or online
learning_offset = A (positive) parameter that downweighs early iterations in online learning
random_state = random seed generator

Exercise 1 - Create a LDA Topic Model

Refer to Jupyter Notebook: ex1_lda_basic.ipynb

```
1  from sklearn.feature_extraction.text import CountVectorizer
2  from sklearn.decomposition import LatentDirichletAllocation
3  documents = ["cat eat rice", "secret message", "today go shopping"]
4  tf_vectorizer = CountVectorizer()
5  tf_vectorized_documents = tf_vectorizer.fit_transform(documents)
6  tf_feature_names = tf_vectorizer.get_feature_names()
7
8  no_topics = 10
9  lda_model = LatentDirichletAllocation(n_components=no_topics)
10 lda_output = lda_model.fit_transform(tf_vectorized_documents)
11 lda_components = lda_model.components_
```

Term document matrix

topic- term distribution

Document-topic distribution

pyLDAvis

Python library for interactive topic model visualization.

It is designed to help users interpret the topics in a topic model that has been fit to a corpus of text data. The package extracts information from a fitted LDA topic model to inform an interactive web-based visualization.

It provides a global view of the topics (and how they differ from each other), while at the same time allowing for a deep inspection of the terms most highly associated with each individual topic.

Stable version:

```
pip install pyldavis
```

Explanation video: https://youtu.be/lksL96ls4o0

Original Paper: https://nlp.stanford.edu/events/illvi2014/papers/sievert-illvi2014.pdf

pyLDAvis with Sckit Learn

Stable version:

```
pip install pyldavis
```

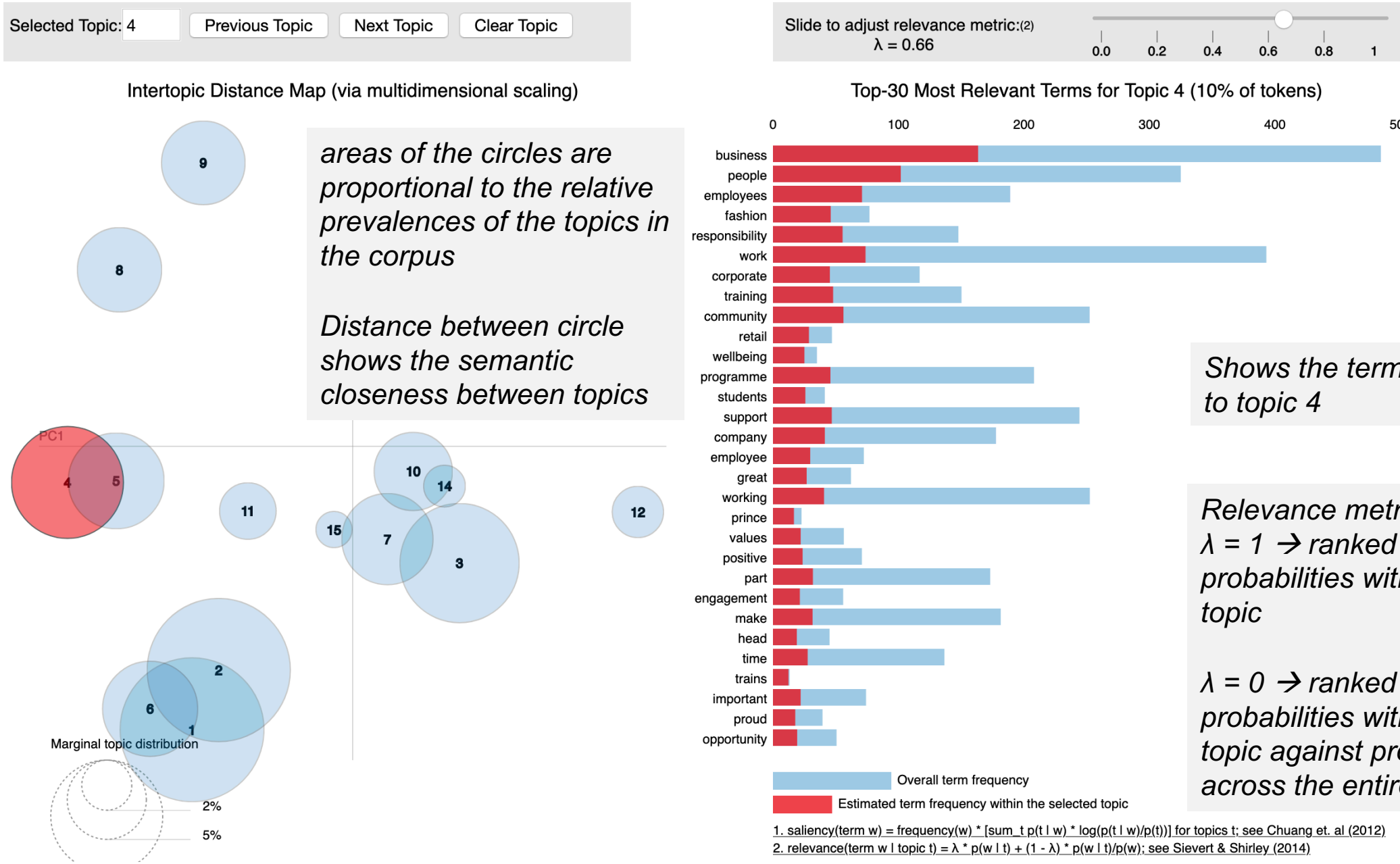
API documentation:
https://pyldavis.readthedocs.io/en/latest/modules/API.html

```
vectorizer = CountVectorizer()
doc_term = vectorizer.fit_transform(documents)

lda_model = LatentDirichletAllocation(n_components=3)
doc_topics = lda_model.fit_transform(tf_vectorized_documents)

pyLDAvis.enable_notebook()
panel = pyLDAvis.sklearn.prepare(lda_model, doc__term, vectorizer)
panel
```

Thinking Task: Which of the 3 parameters contain the actuals term / words ?

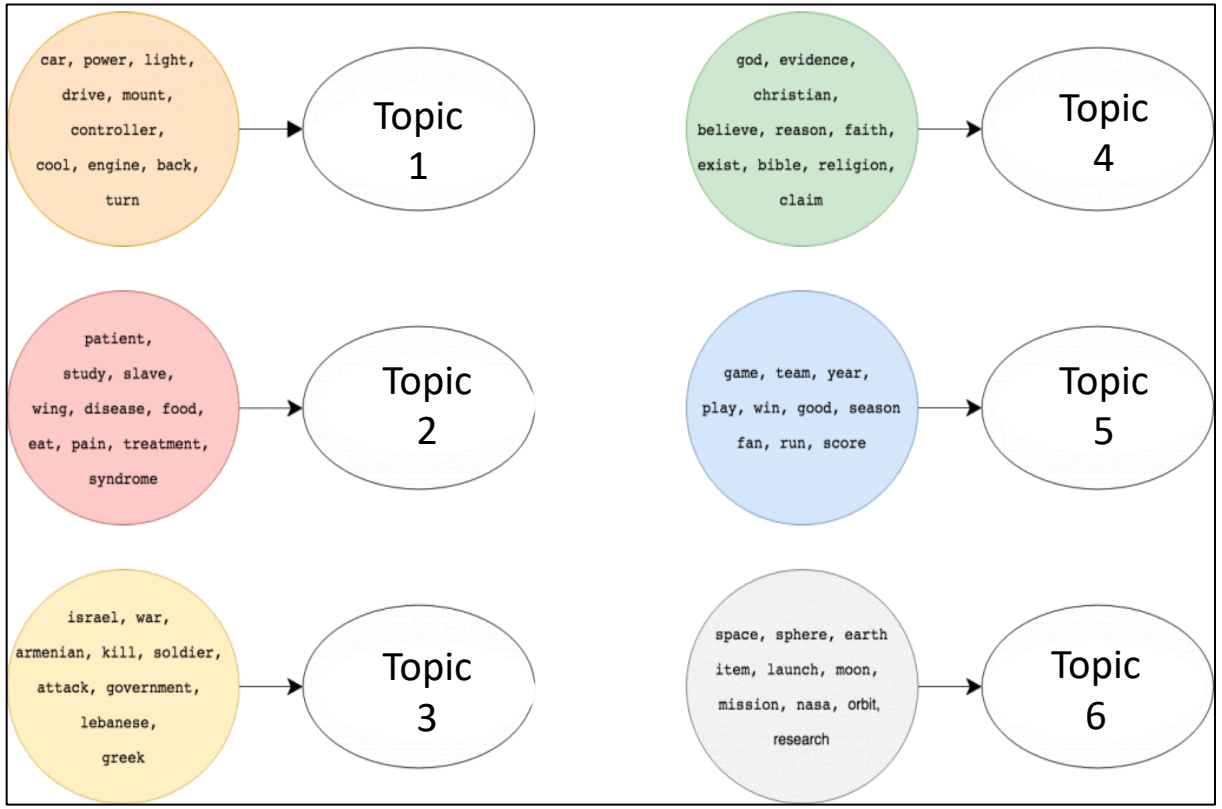


https://pyldavis.readthedocs.io/en/latest/

Exercise 2 - Derive and Visual Topic Model

Refer to Jupyter Notebook:
ex2-_lda_derive_and_visualise_topics.ipynb

Inferring Topics to Terms



Lda_model.n_components determines the number of topics

lda_model.components_ determines the terms/words associated with a topic

lda_model.transform-fit(...) contains the document-topic weightage

	Topic0	Topic1	Topic2	Topic3	Topic4	Topic5	Topic6	Topic7	Topic8	Topic9	dominant_topic
Doc0	0.53	0	0	0	0	0.24	0	0	0.2	0	0
Doc1	0	0	0	0	0	0.55	0	0	0.42	0	5
Doc2	0.3	0.28	0	0	0	0	0	0	0.39	0	8

Inferring Documents to Terms

	Topic0	Topic1	Topic2	Topic3	Topic4	Topic5	Topic6	Topic7	Topic8	Topic9	dominant_topic
Doc0	0.53	0	0	0	0	0.24	0	0	0.2	0	0
Doc1	0	0	0	0	0	0.55	0	0	0.42	0	5
Doc2	0.3	0.28	0	0	0	0	0	0	0.39	0	8

```
# column names
topicnames = ["Topic" + str(i) for i in range(lda_model.n_components)]

# index names
docnames = ["Doc" + str(i) for i in range(len(documents))]

# Make the pandas dataframe
df_document_topic = pd.DataFrame(np.round(lda_output, 2),
                                 columns=topicnames, index=docnames)

# Get dominant topic for each document
dominant_topic = np.argmax(df_document_topic.values, axis=1)
df_document_topic['dominant_topic'] = dominant_topic
```

Exercise 3 - Find Dominant Topics

Refer to Jupyter Notebook:
ex3-lda_find_dominant_topics_in_documents.ipynb

Quantitative Measures
What is the right number of topics

Perplexity measures how well a probability model **predicts** a sample.

A lower perplexity is generally considered as "better" in predicting the sample.

Positive number:
100 is better than 180

Log-likelihood measures how well the model **fits** the sample.

The higher the likelihood, the better.

Negative number:
-100 is better than -180

Steps

1: Define reasonable range of k (min and max) for the text corpus

2 : For each k, calculate the score and perplexity (*see codes below)*

```
lda_model = LatentDirichletAllocation(n_components=k          )
lda_output = lda_model.fit_transform(tf_vectorized_documents )
log_likelihood = lda_model.score(tf_vectorized_documents )
perplexity = lda_model.perplexity(tf_vectorized_documents)
```

3 : Find the k with the highest log_likelihood and lowest perplexity

Determines score and perplexity against the vectorised document

Exercise 4 - Finding the right number of topics

Refer to Jupyter Notebook: ex4-lda_parameter_selection.ipynb

References

LDA in Python – How to Grid Search Best Topic Model , www.machinelearningplus.com/nlp/topic-modeling-python-sklearn-examples/

Evaluation Methods For Topic Models, www.dirichlet.net/pdf/wallach09evaluation.pdf

Latent Dirichlet Allocation, www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Topic Analysis, www.monkeylearn.com/topic-analysis

LDA Topic Models, https://www.youtube.com/watch?feature=oembed&v=3mHy4OSyRf0

Visualizing Topic Models, www.liferay.de.dariah.eu/tatom/topic_model_visualization.html

An intro to topic models for text analysis
, https://medium.com/pew-research-center-decoded/an-intro-to-topic-models-for-text-analysis-de5aa3e72bdb

Applications of Topic Models, https://mimno.infosci.cornell.edu/papers/2017_fntir_tm_applications.pdf,
www.liferay.de.dariah.eu/tatom/topic_model_visualization.html

Latent Dirichlet Allocation (LDA) for Topic Modeling of the CFPB Consumer Complaints,
https://arxiv.org/ftp/arxiv/papers/1807/1807.07468.pdf