

# Topic Modeling

By,

**Divesh R. Kubal**

Data Scientist, eClerx Services  
Center of Excellence – Machine Learning



# Table of Contents

1. Topic Modeling
2. Latent Semantic Analysis
3. Latent Dirichlet Allocation
4. Sentiment Analysis
5. Implementation



# Topic Modeling Introduction

- Topic Modeling is a technique to extract the hidden topics from large volumes of text.
- Topic Modeling, in the context of Natural Language Processing, is described as a method of uncovering hidden structure in a collection of texts.
- Definitions:
  - C: collection of documents containing N texts.
  - V: vocabulary (the set of unique words in the collection)



# Dimensionality Reduction

- Topic modeling is a form of dimensionality reduction. Rather than representing a text  $T$  in its feature space as  $\{\text{Word}_i: \text{count}(\text{Word}_i, T) \text{ for } \text{Word}_i \text{ in } V\}$ , we can represent the text in its topic space as  $\{\text{Topic}_i: \text{weight}(\text{Topic}_i, T) \text{ for } \text{Topic}_i \text{ in Topics}\}$ . Notice that we're using Topics to represent the set of all topics.



# Unsupervised Learning

- Topic modeling can be easily compared to clustering. As in the case of clustering, the number of topics, like the number of clusters, is a hyperparameter. By doing topic modeling we build clusters of words rather than clusters of texts. A text is thus a mixture of all the topics, each having a certain weight.



# A Form of Tagging

- If document classification is assigning a single category to a text, topic modeling is assigning multiple tags to a text. A human expert can label the resulting topics with human-readable labels and use different heuristics to convert the weighted topics to a set of tags.



# Why is Topic Modeling useful?

- **Text classification** – Topic modeling can improve classification by grouping similar words together in topics rather than using each word as a feature
- **Recommender Systems** – Using a similarity measure we can build recommender systems. If our system would recommend articles for readers, it will recommend articles with a topic structure similar to the articles the user has already read.
- **Uncovering Themes in Texts** – Useful for detecting trends in online publications for example



# Topic Modeling Algorithms

- **LSA or LSI** – Latent Semantic Analysis or Latent Semantic Indexing – Uses Singular Value Decomposition (SVD) on the Document-Term Matrix. Based on Linear Algebra
- **LDA** – Latent Dirichlet Allocation – The one we'll be focusing in this tutorial. Its foundations are Probabilistic Graphical Models
- **NMF** – Non-Negative Matrix Factorization – Based on Linear Algebra



- Latent Semantic Analysis, or LSA, is one of the foundational techniques in topic modeling. The core idea is to take a matrix of what we have — documents and terms — and decompose it into a separate document-topic matrix and a topic-term matrix.

- The first step is generating our document-term matrix.
- Given  $m$  documents and  $n$  words in our vocabulary, we can construct an  $m \times n$  matrix  $A$  in which each row represents a document and each column represents a word
- In the simplest version of LSA, each entry can simply be a raw count of the number of times the  $j$ -th word appeared in the  $i$ -th document.



# Working of LSA

- Once we have our document-term matrix  $A$ , we can start thinking about our latent *topics*.
- $A$  is very sparse, very noisy, and very redundant across its many dimensions. As a result, to find the few latent topics that capture the relationships among the words and documents, we want to perform dimensionality reduction on  $A$ .



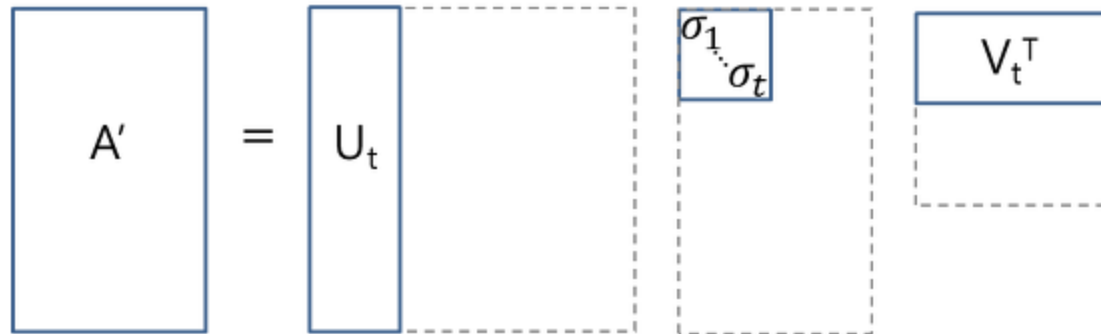
# Working of LSA

- This dimensionality reduction can be performed using truncated SVD.
- SVD, or singular value decomposition, is a technique in linear algebra that factorizes any matrix  $M$  into the product of 3 separate matrices:  $M=U*S*V$ , where  $S$  is a diagonal matrix of the singular values of  $M$ .
- Critically, truncated SVD reduces dimensionality by selecting only the  $t$  largest singular values, and only keeping the first  $t$  columns of  $U$  and  $V$ . In this case,  $t$  is a hyperparameter we can select and adjust to reflect the number of topics we want to find.

# Working of LSA

- Intuitively, think of this as only keeping the  $t$  most significant dimensions in our transformed space.

$$A \approx U_t S_t V_t^T$$





# Features of LSA

- LSA can be used to find,
  - the similarity of different documents
  - the similarity of different words
  - the similarity of terms (or “queries”) and documents (which becomes useful in information retrieval, when we want to retrieve passages most relevant to our search query).



# Limitations of LSA

- Lack of interpretable embeddings (we don't know what the topics are, and the components may be arbitrarily positive/negative)
- Need for really large set of documents and vocabulary to get accurate results
- Less efficient representation


- Topic model that generates topics based on word frequency from a set of documents.
- Particularly useful for finding reasonably accurate mixtures of topics within a given document.



# Why do we need LDA?

- I want to find out the news highlights of France in 2018. I am given a dataset which contains all the news articles of the country from 2018.
- I make use of LDA to find out topics.
- Example: France won 2018 world cup



- 
- Consider the very relevant example of comparing probability distributions of topic mixtures. Let's say the corpus we are looking at has documents from 3 very different subject areas. If we want to model this, the type of distribution we want will be one that very heavily weights one specific topic, and doesn't give much weight to the rest at all. If we have 3 topics, then some specific probability distributions we'd likely see are:



- **Mixture X: 90% topic A, 5% topic B, 5% topic C**
- **Mixture Y: 5% topic A, 90% topic B, 5% topic C**
- **Mixture Z: 5% topic A, 5% topic B, 90% topic C**



# Working of LDA

- Create a collection of documents from news articles
- Each document represents a news article.
- Data cleaning is the next step:
  - Tokenizing
  - Stop-words removal
  - Stemming

# Working of LDA

- I want to find out 3 major topics from the documents that I have collected.

<b>3</b>	<b>2</b>	<b>1</b>	<b>3</b>	<b>1</b>
Football	World Cup	2018	Winners	France

	<b>Topic 1</b>	<b>Topic 2</b>	<b>Topic 3</b>
Doc i	2	1	2

# Working of LDA

	Topic 1	Topic 2	Topic 3
Football	1	0	35
World Cup	10	8	1
2018	42	1	0
Winners	0	0	20
France	50	0	1



# Working of LDA

- Consider the word – World Cup
- Reassign the topic for the given word.
- Decrement counts after removing current assignments.

	Topic 1	Topic 2	Topic 3
Doc i	2	0	2

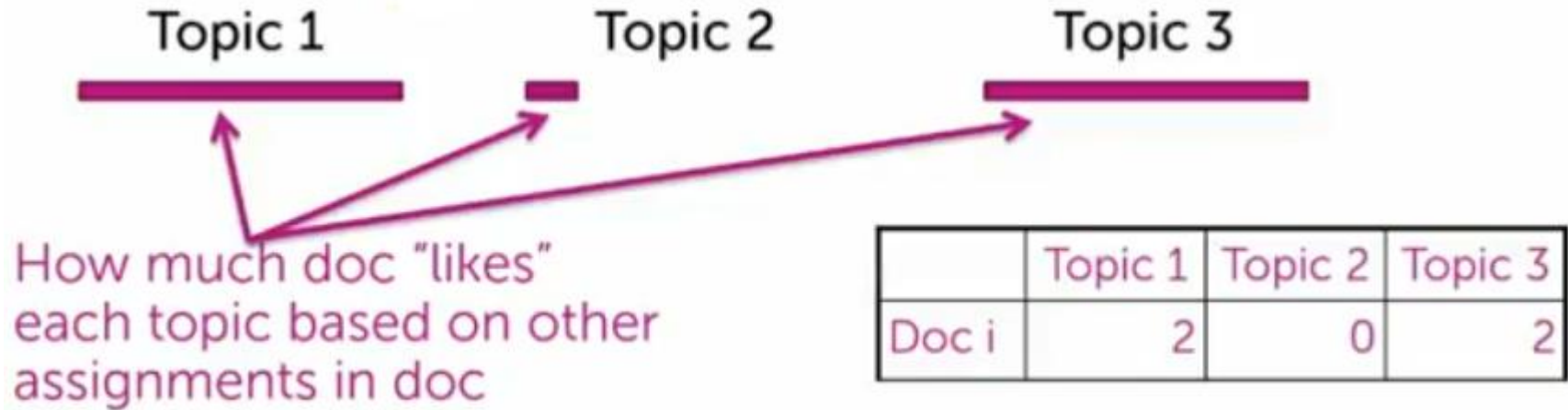
# Working of LDA

	Topic 1	Topic 2	Topic 3
Football	1	0	35
World Cup	10	7	1
2018	42	1	0
Winners	0	0	20
France	50	0	1

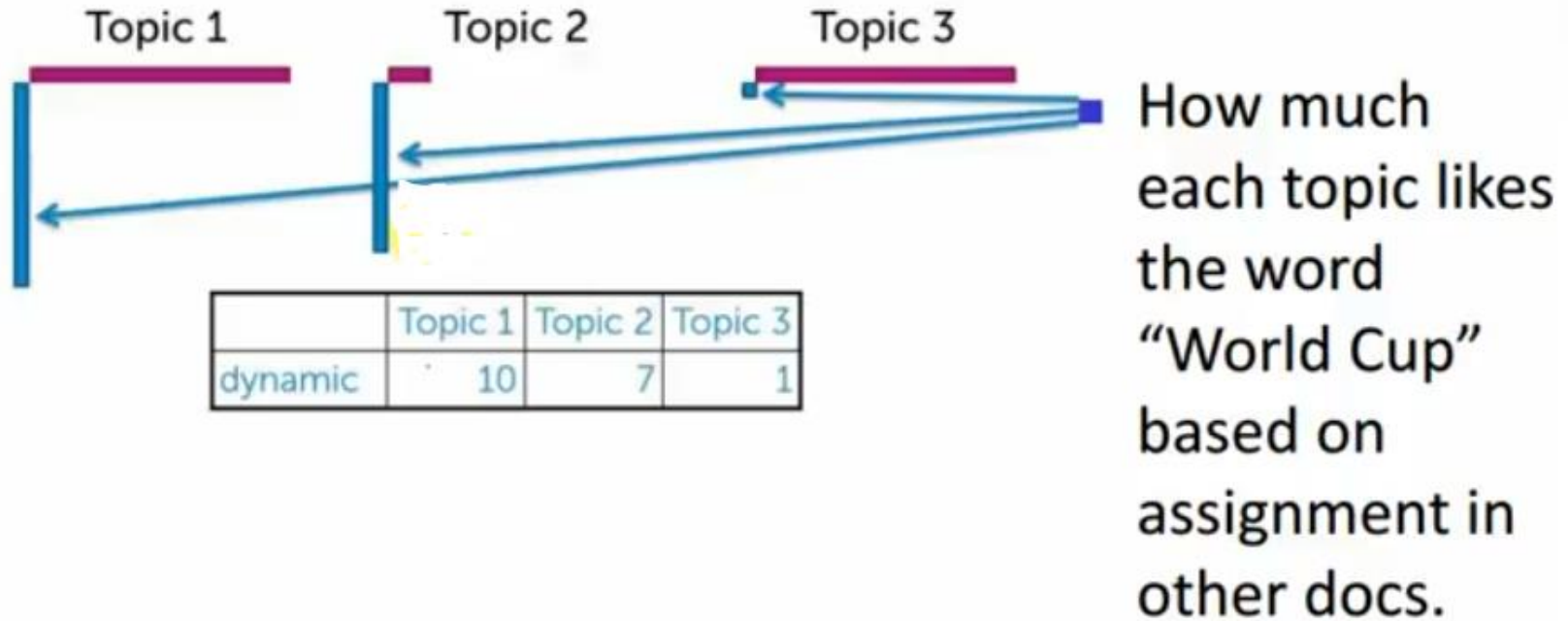


# Working of LDA

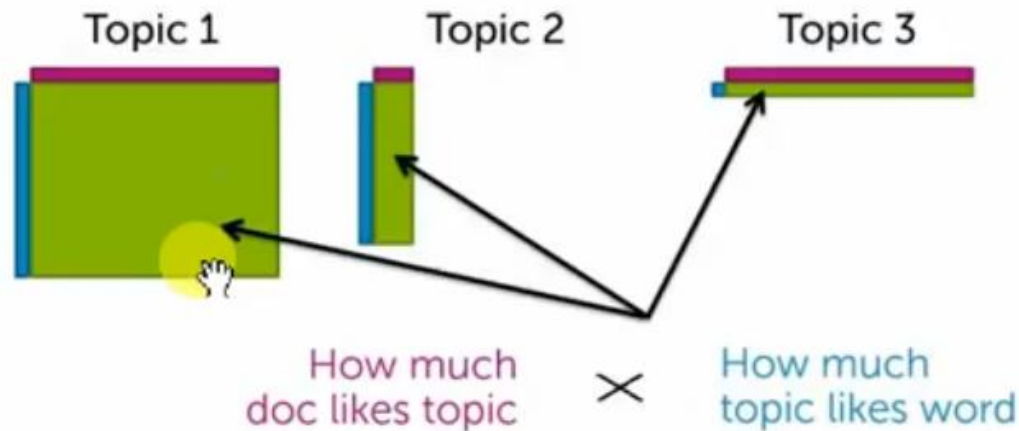
- Reassign the topic based on a probability calculation.



# Working of LSA



# Working of LDA



<b>3</b>	<b>1</b>	<b>1</b>	<b>3</b>	<b>1</b>
Football	World Cup	2018	Winners	France



# Working of LDA

```
ldamodel = gensim.models.ldamodel.LdaModel(corpus,  
num_topics=2, id2word = dictionary, passes=20)
```

```
>>> print(ldamodel.print_topics(num_topics=1,  
num_words=3))
```

```
0.070*football + 0.054*france + 0.050*world_cup
```