CONTENTS

1 INTRODUCTION

The purpose of this final work is to describe and explore a well known method in topic modeling: Latent Dirichlet Allocation. It is part of a series of methods in the field of Natural Language Processing the have been developed and that have brought innovation on the way we train computers to understand and process the human language. Even though it has been around for some years (the original paper is from Blei, Ng, and Jordan, 2003), it is still one of the most used methods in the field of topic modeling.

A topic model is a probabilistic model which contains information about topics in the text. But what is exactly a topic? We can understand it as a theme, or underlying ideas represented in text. For example, if one takes a collection of articles from a newspaper, example of topics would be *politics*, *economy*, *sport*, etc. However, topics as intended here are not just a group of words from the same context, but a topic is meant as a probabilistic distribution of words. Put together, many topic distributions can help us describe documents as a probabilistic distribution of topics. Based on the word-count of each document we can learn about these topics and infer about the underlying distribution. The idea of seeing words in documents as the result of a generative process from topic distributions is the concept behind the model which is Latent Dirichlet Allocation: sets of words observed are explained by unobserved topics that also explain why some parts of the text are similar, both in the same

text and across documents. When interrogating a corpus of documents, in order to retrieve information or to scan through the text, instead of representing them using just the words they contain we can reduce the dimensionality by representing them as the topics that they were generated from. Furthermore, what is the result of topic modeling is not a topic with a label, but rather a probabilistic distribution of words that can only be described by the words it contains. By ordering the words in terms of probability, we can find the most relevant words, and we can associate them with a similar subject, but this step is not a result of the model.

Topic modeling, and in particular LDA, has been used in many fields from engineering to medicine and bioinformatics, but lately it has also been applied to the field of law. When preparing for a trial, lawyers and experts have to go through thousands of documents such as pieces of law, text evidence, or previous cases sentences. Having a tool such as topic models can help them navigate through increasingly large archives of documents and improve their information retrieval task, which in the case of electronic texts is called *electronic discovery*.

In the next pages, I'm going to unveil what is the theory and functioning of Latent Dirichlet Allocation in Part I, starting from topic models (Chapter 2) and moving on to LDA (Chapter 3). After that, in Part II, I will describe an application that I've developed using topic models in Python: it will train the LDA model on a corpus of sentences from the European Court of Justice, and I will show how the results can be used to summarize the documents and explore the collection. The code that I've written can be found on this GitHub repository at github.com/PietroDomi/lda-eucj.

Part I

THEORY

2

TOPIC MODELING

2.1 WHAT IS TOPIC MODELING?

Topic modeling is a research branch of the field of natural language processing and understanding (NLP), itself an area of application of machine learning. The aim of the subject is to develop topic models, which are a group of algorithms with the purpose of discovering the main themes (i.e. topics) that are present in a large unstructured collection of documents, called *corpus*. Their main field of application is the classification and analysis of text documents, but recently the techniques have evolved and can be adapted to other types of input. Regarding written content specifically, these algorithms are statistical methods that analyze the words of the original texts to discover the themes that are mixed together in each of them, how these are connected to each other and how they evolve over time. The result of topic modeling algorithms can be used to summarize, visualize, explore, and theorize about a corpus [3]. The key feature of such algorithms, is that they do not require any prior annotations or labeling of the documents: the topics emerge from the statistical analysis of the words. The power of these set of tools is well understood when we think about how they can be applied to electronic archives, which have now reached a size and a level of complexity beyond human management alone [18]. With the help of topic models, we can organize and summarize document collections at a scale that was impossible

before [2].

A topic model takes a collection of texts (the corpus) as input. It discovers a set of "topics" and the degree to which each document exhibits those topics. By the term *topic* we mean recurring themes that are discussed in the collection, and are identified by a set of recurring words across the documents. The model gives us a framework in which to explore and analyze the texts, but we did not need to decide on the topics in advance or manually code each document according to them. It is the model itself which finds a way of representing documents that is useful for navigating and understanding the collection. In Chapter 4 I will expose these features of topic models by applying Latent Dirichlet Allocation (LDA) to a corpus of sentences from the European Court of Justice, in order to uncover connections and links to better navigate and explore the documents. This is a process done frequently in legal proceedings and takes the name of *electronic discovery*.

ELECTRONIC-DISCOVERY In the law of common law jurisdictions, discovery is a pre-trial procedure in a lawsuit in which each party, through the law of civil procedure, can obtain evidence from the other party or parties by means of discovery such as interrogatories, requests for production of documents, requests for admissions and depositions. In this case, electronic-discovery (e-discovery) refers to the process of collecting and reviewing electronic documents (often referred to as electronically stored information or ESI) to identify their relevance to a legal case. It may be in plain text or converted into plain text using methods such as Optical Character Recognition (OCR). The amount of data to be dealt with in any single case can be

enormous, making the manual reviewing process cumbersome and expensive. Litigation costs are increasing and as a result, are removing the public dispute resolution process from reach of the average citizen and medium-sized company[1]. Thus, legal professionals have sought to employ information retrieval and machine learning methods to reduce manual labor and increase accuracy.

2.2 WHAT IS A (PROBABILISTIC) GENERATIVE MODEL?

Topic modeling resides in the larger field of *probabilistic modeling*, a field that has great potential for the analysis and study of textual data. Traditionally, statistics and machine learning provides a list of methods, and users of these tools are required to match their specific problems to general solutions. In probabilistic modeling, you are provided with a language for expressing assumptions about data and generic methods for computing with those assumptions [3]. In particular, LDA is a type of probabilistic model with hidden variables. In the context of textual analysis, LDA specifies a *generative process*, an imaginary probabilistic recipe that reproduces both the hidden topic structure and the observed words of the texts. Then, we rely on topic modeling algorithms to perform what is called *probabilistic inference*: given a corpus of documents, they try to guess what is the most likely hidden topical structure that generated the documents we observe. In generative probabilistic modeling, we treat the data as if they were arising from a generative process that includes hidden variables, unobserved. The algorithms for

1 "Litigation costs continue to rise and are consuming an increasing percentage of corporate revenue" (source uscourts.gov).

these models, such as topic models, try to go back to the generative probability distribution from the data that is observed. This generative process defines a joint probability distribution over both the observed and hidden random variables, then we perform data analysis by using that joint distribution to compute the conditional distribution of the hidden variables given the observed ones. This conditional distribution is what we call *posterior distribution* [2]. Latent Dirichlet Allocation, which we are going to elaborate on in Chapter 3, is a model built exactly upon this framework.

Generative models differ in concept from discriminative models, which instead of the joint distribution aim to learn directly the conditional distribution, without the support of hidden variables. In general, discriminative classifiers model the posterior probability distribution $p(y|x)$ of the input data x and the label y directly, or learn a direct map from inputs x to the class labels. Instead, generative classifiers learn a model of the joint probability $p(x, y)$ and make their predictions by using the Bayes rule[2] to calculate $p(y|x)$ and then picking the most likely label y [15].

2 Bayes rule: $p(y|x)p(x) = p(x|y)p(y) = p(x, y)$

2.3 INTRODUCTION TO LDA

The simplest topic model is Latent Dirichlet Allocation (LDA), which is a probabilistic model of texts. Loosely speaking, the model makes two assumptions:

1. There are a fixed number of patterns of word use, groups of terms that tend to occur together in documents. We call them *topics*.

2. Each document in the corpus exhibits the topics to varying degree.

We can then use the topic representations of the documents to analyze the collection in many ways. For example, we can isolate a subset of the texts based on which combination of topics they exhibit. Or, we can examine the words of the texts themselves and restrict attention to one set of words, finding similarities between them or trends in the language. Both of these analyses require that we know in advance the topics and which topics each document is about, then topic modeling uncovers this structure. The algorithm analyzes the texts to find a set of topics as patterns of tightly occurring terms, and how each document combines them [3].

For LDA, the generative process happens as follows:

1. First it chooses the topics drawn from a distribution over distributions.

2. Next, for each document, it chooses the topic weights to describe which topics each document is about.

3. Finally, choose a topic assignment from those topic weights for each word in the document, and then choose an observed word from the corresponding topic.

While the topic weights are chosen every time the model generates a new document, the topics themselves are chosen once for the whole collection. It deserves emphasis to state that this is a conceptual process: it defines the mathematical model where a set of topics characterizes the corpus, and each document shows them to different degree. Under these assumptions, the inference algorithm finds the topics that best describe the collection.

I used the word "topic" in many sentences so far, but what exactly is a *topic*? Formally, a topic is a probability distribution over terms. In each topic, different sets of terms have high probability, and we typically visualize the topics by listing those sets. Topic models find the set of terms that tend to occur together in the documents. They look like topics because terms that frequently occur together tend to be about the same subject [3].

2.4 ORIGINS AND EVOLUTION OF TOPIC MODELS

The origin of a topic model is latent semantic indexing (LSI), developed by Deerwester et al. in 1988 [6]; it has served as the basis for the development of a topic model. LSI is a well-known method in the field of information retrieval: it can group together words and phrases that have similar meaning. One can use these groups or concepts to represent the documents in a collection and keyword queries, and perform a "concept search" to retrieve relevant documents by defining a similarity score on the new representative domain. LSI typically performs matrix

factorization over a *term-frequency inverse document-frequency* (*tfidf*) matrix, using the concept of eigenvalue decomposition, and identifies patterns in the relationships between the document terms and concepts or topics [8].

Nevertheless, LSI is not a probabilistic model, and thus not an authentic topic model. Based on LSI, probabilistic latent semantic analysis (pLSA) was proposed by Hofmann [10] and was adapted by him to be a genuine topic model. Compared to standard LSA which stems from linear algebra and downsizes the occurrence tables (usually via a singular value decomposition), pLSA is based on a mixture decomposition derived from a latent class model. Considering observations in the form of co-occurrences (w, d) of words and documents (taking the name of *term-document* matrix, pLSA models the probability of each co-occurrence as a mixture of conditionally independent multinomial distributions[3]:

$$P(w, d) = \sum_c P(c)P(d|c)P(w|c) = P(d) \sum_c P(c|d)P(w|c) \ ,$$

where c is the words' topic. A note to be made is that the number of topics is a hyperparameter of this model that must be chosen in advance and is not estimated from the data, a similarity we'll encounter also in LDA.

Then, published after pLSA, Blei, Ng, and Jordan proposed Latent Dirichlet Allocation, a new enhanced model built on top of pLSA and that I am going to explore later (Chapter 3): this represented an even more complete probabilistic generative model. Nowadays, there is a growing number of probabilistic models that are based on LDA via combination with particular

3 The multinomial defined on this support $x_i \in \{0, \dots, n\}$, $i \in \{1, \dots, k\}$, $\sum x_i = n$, has the following probability mass function: $P(x_1, \dots, x_k) = \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}$.

tasks. There exist models that contain more complex hidden structures and generative processes of the texts. For example, researchers have developed models that include syntax, topic hierarchies, document networks, topics drifting through time, reader's libraries, and the influence of past articles on future articles. For each of these advancements, the project involved the assumption of a new kind of topical structure, the embodiment of itself in a generative process of documents, and the derivation of the related inference algorithm to discover that structure in observed collections [3].

2.5 THE PURPOSE OF TOPIC MODELS

The statistical topic models are meant to help interpret and understand texts, but it is still the user's job to do the actual interpreting and understanding. A model of text is built with a particular theory in mind and therefore it cannot provide evidence for the theory. In fact, the theory is built into the assumptions of the model. Rather than this, one hopes that the model helps the questioner in pointing to such evidence.

Therefore researchers in probabilistic modeling separate the essential activities of designing models and deriving their corresponding inference algorithms. The goal is to creatively design models with an intuitive language of components, and then for computer programs to derive and execute the corresponding inference algorithms on real data. Probabilistic models promise to give scholars a powerful language to articulate assumptions about their data and fast algorithms to compute with those assumptions on large archives [3].

3

LATENT DIRICHLET ALLOCATION

In this chapter I will examine and describe one of the most used probabilistic topic models for modeling text corpora and other collections of discrete data: Latent Dirichlet Allocation. LDA was first proposed by Blei, Ng, and Jordan in 2003 and is seen as the evolution of models such as Latent Semantic Indexing and pLSI [9].

The pLSI approach models each word in a document as a sample from a mixture model, where the mixture components are multinomial random variables that can be viewed as representation of topics. Thus, each word is generated from a single topic, and different words in a document may be generated from different topics. Each document is represented as a list of mixing proportions for these mixture components and thereby reduced to a probability distribution on a fixed set of topics. This approach was useful but incomplete with respect to a probabilistic model of text, since each document is represented as a list of numbers (the proportions for topics) and there is no generative probabilistic model for these numbers at the level of documents. LDA takes the model to a step further. Let's consider the fundamental probabilistic assumptions underlying the class of methods that includes LSI and pLSI: all of them are based on the assumption that the order of words in a document can be neglected - which is called "bag-of-words" - and also the assumption that the specific ordering of documents in a corpus can be neglected. These are assumptions of

exchangeability for the words in a document and the documents in a corpus. The step that LDA adds to this interpretation is the following: a representation theorem by de Finetti (1937) establishes that any collection of exchangeable random variables has a representation as a mixture distribution; thus, if we want to consider exchangeable representations for documents and words, we need to consider mixture models that capture this exchangeability [4].

3.1 THE MODEL

3.1.1 *Terminology*

In the following sections, I will use terms from NLP such as "words", "documents" and "corpora", but underneath each of them there is a formal statistical definition. Even if the model can be applicable to other problems involving collections of data that are not text, these terms are used to characterize precise entities. In particular:

- A *word* is the basic unit of discrete data, defined to be an item from a vocabulary indexed by $1, \dots, V$. We represent words using unit-basis vectors that have a single component equal to one and all other components equal to zero (one-hot vector). Thus, the v-th word in the vocabulary is represented by a V-vector w such that $w^v = 1$ and $w^u = 0$ $\forall\, u \neq v$.

- A *document* is a sequence of N words denoted by $\mathbf{w} = (w_1, w_2, \dots, w_N)$, where w_N is the n-th word in the sequence.

- A *corpus* is a collection of M documents denoted by $D = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M\}$.

3.1.2 *Statistical representation*

Latent Dirichlet Allocation is a probabilistic generative model of a corpus. The core idea is that documents are represented as random mixtures over latent topics, whereas each topic is characterized by a distribution over words. The generative process that is assumed by this model is the following. For each document $\mathbf{w}$ in a corpus D:

1. Choose $N \sim \mathrm{Poisson}(\xi)$.
2. Choose $\theta \sim \mathrm{Dir}(\alpha)$.
3. For each of the N words w_n:
   a) Choose a topic $z_n \sim \mathrm{Multinomial}(\theta)$.
   b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

The generative model shown above makes several simplifying assumptions. First, the dimensionality k of the Dirichlet distribution is assumed to be known and fixed. Second, the parameter β of the word probabilities is a $k \times V$ matrix, where $\beta_{ij} = p(w^j = 1|z^i = 1)$, which is a quantity to be estimated from the corpus. The distribution for N, the length of the document, is assumed to be Poisson, but other distributions can be used as alternatives. Also, being N independent of all the other data generating variables, it is an auxiliary variable, and it won't be studied in deep.

As per the name LDA, the distribution of the variable θ is a Dirichlet. A k-dimensional Dirichlet random variable θ can

take values in the $(k-1)$-simplex[1] and has the following probability density on this simplex:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1} \tag{1}$$

where the parameter α is a $k-$vector with components $\alpha_i > 0$, and where $\Gamma(x)$ is the Gamma function.

Given the parameters α and β, the joint distribution of a topic mixture θ, a set of N topics $\mathbf{z}$, and a set of N words $\mathbf{w}$ is given by:

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta) \tag{2}$$

where $p(z_n|\theta)$ is simply θ_i for the unique i such that $z_n^i = 1$. Integrating over θ and summing over z we obtain the marginal distribution of a document:

$$p(\mathbf{w}|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{n=1}^{N} \sum_{z_n} p\left(z_n|\theta\right) p\left(w_n|z_n, \beta\right) \right) d\theta. \tag{3}$$

Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(D|\alpha, \beta) = \prod_{d=1}^{M} \int p\left(\theta_d|\alpha\right) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p\left(z_{dn}|\theta_d\right) p\left(w_{dn}|z_{dn}, \beta\right) \right) d\theta_d$$

The distribution above summarizes the main idea of Latent Dirichlet Allocation, and its generative process. In order to understand it better, one can show it as a probabilistic graphical model (Figure 1). There are three levels to the LDA representation. The parameters α and β are corpus-level parameters, assumed to be sampled once in the process of generating a corpus. The variables θ_d are document-level variables, sampled once per document. Finally, the variables x_{dn} and w_{dn}

1 A k-vector θ lies in the $(k-1)$-simplex if $\theta_i \geqslant 0, \sum_{i=1}^{k} \theta_i = 1$.

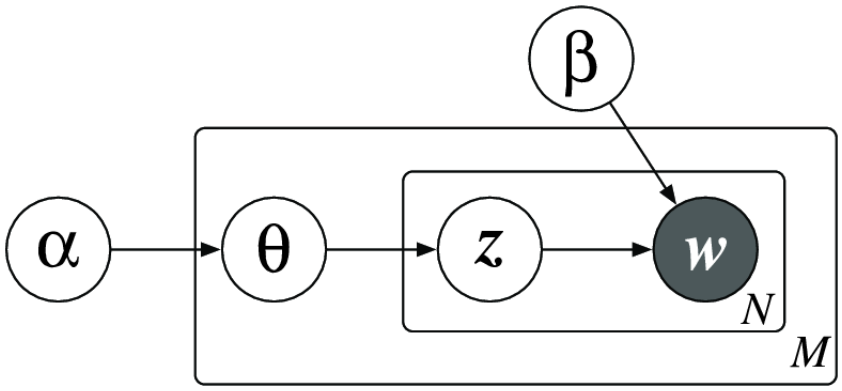


Figure 1: Graphical model representation of LDA. The outer box represents documents, the inner box represents topics and words within a document.

are word-level variables and are sampled once for each word in each document. A structure such as this one is often studied in Bayesian statistical modeling, and is known as *(conditionally independent) hierarchical models*.

3.1.3 *Exchangeability and bag-of-words*

In LDA and other topic models we often use the *bag-of-words* (BOW) assumption. A BOW model is a simplifying representation used in natural language processing according to which a text is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. AN alternative representation for a document, derived from the bag of words, is *tf-idf*, i.e. term frequency-inverse document frequency. It is a numerical statistic that is intended to reflect how important a word is to a document in a corpus. The value increases proportionally to the number of times a word appears in the document and decreases when the word is frequent in a higher number of documents in general. It was introduced by Jones in 1972, and it provides a secondary approach to BOW

that leads to different results of LDA, since the corpus representation has changed.

In LDA, we assume that words are generated by topics (by fixed conditional distributions) and that those topics are infinitely exchangeable within a document. A finite set of random variables $\{z_1, \ldots, z_N\}$ is said to be *exchangeable* if the joint distribution is invariant to permutation. If π is a permutation of the integers from 1 to N :

$$p(z_1, \ldots, z_N) = p\left(z_{\pi(1)}, \ldots, z_{\pi(N)}\right)$$

An infinite sequence of random variables is *infinitely exchangeable* if every finite subsequence is exchangeable. De Finetti's representation theorem states that the joint distribution of an infinitely exchangeable sequence of random variables is as if a random parameter were drawn from some distribution and then the random variables in question were *independent* and *identically distributed*, conditioned on that parameter [5]. By de Finetti's theorem the probability of a sequence of topics and words must therefore have the form:

$$p(\mathbf{w}, \mathbf{z}) = \int p(\theta) \left(\prod_{n=1}^{N} p(z_n|\theta)\, p(w_n|z_n) \right) d\theta$$

where θ is the random parameter of a multinomial over topics. We obtain the LDA distribution over documents in Equation 3 by marginalizing out the topic variables and endowing θ with a Dirichlet distribution.

3.2 INFERENCE AND PARAMETER ESTIMATION

Approximate inference

In order to use LDA, on has to solve the key inferential problem, that is, the computation of the posterior distribution of the hidden variables given a document:

$$p(\theta, \mathbf{z} \mid \mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta)}{p(\mathbf{w} \mid \alpha, \beta)}$$

Unfortunately, this distribution is intractable to compute in general. Indeed, to normalize the distribution we marginalize over the hidden variables and write Equation 3 in terms of the model parameters:

$$p(\mathbf{w}|\alpha, \beta) = \frac{\Gamma\left(\sum_i \alpha_i\right)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta$$

which is intractable due to the coupling between θ and β in the summation over latent topics [7].

Although the posterior distribution is intractable for exact inference, a wide variety of approximate inference algorithms can be considered for LDA, such as Laplace approximation, variational approximation, Monte Carlo Markov Chain and online learning. One particular method, which is well known for inferring about intractable distributions is *variational inference*. The main idea of variational methods is to cast inference as an optimization problem. In the case one faces an intractable probability distribution p, variational techniques will try to solve an optimization problem over a class of tractable distributions $\mathcal{Q}$ in order to find $q \in \mathcal{Q}$ that is most similar to p. When talk-

ing about similarity between distribution oftentimes we use the *Kullback-Leibler divergence* which is defined as:

$$KL(q\|p) = \sum_x q(x) \log \frac{q(x)}{p(x)}$$

for distributions with discrete support. It is a measure of how two distributions are far from one another, being equal to 0 only if the distributions are the same. After finding an appropriate tractable distribution q, we will query this one rather than p in order to get an approximate solution. This method differs from the others since it doesn't involve sampling (such as MCMC) and it has some advantages and drawbacks: unlike sampling-based methods, variational approaches will almost never find the globally optimal solution; however, we will always know if they have converged. In practice, variational inference methods often scale better and are more amenable to techniques like stochastic gradient optimization, parallelization over multiple processors, and acceleration using GPUs.

Parameter Estimation

In Blei, Ng, and Jordan [4], it is described an empirical Bayes method for parameter estimation in the LDA model. Given a corpus of $D = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M\}$, one wishes to find parameters α and β that maximize the (marginal) log-likelihood of the data:

$$\ell(\alpha, \beta) = \sum_{d=1}^{M} \log p\left(\mathbf{w}_d|\alpha, \beta\right).$$

As mentioned before, the quantity $p\left(\mathbf{w}_d|\alpha, \beta\right)$ cannot be computed tractably. However, thanks to the variational inference method, we can get a tractable lower bound on the log likelihood, a bound which we can maximize with respect to α and

β. We can then find approximate empirical Bayes estimates for the LDA model via an alternating variational EM[2] procedure that maximizes a lower bound with respect to the model parameters α and β. The derivation returns the following iterative algorithm:

1. **E-step** For each document, find the optimizing values of the variational parameters $\{\gamma_d^*, \phi_d^* : d \in D\}$. This is done as described in the previous section.

2. **M-step** Maximize the resulting lower bound on the log likelihood with respect to the model parameters α and β. This corresponds to finding maximum likelihood estimates with expected sufficient statistics for each document under the approximate posterior which is computed in the E-step. Analytically,

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dni}^* w_{dn}^j$$

.

The two steps are repeated until the lower bound converges.

2 Expectation Maximization.

Part II

APPLICATION

4

AUTOMATIC ELECTRONIC DISCOVERY

In this chapter, I will elaborate on and develop an application of Latent Dirichlet Allocation. The application aims at helping lawyers and law experts in the process of electronic discovery, by applying this statistical tool to a dataset of court sentences. I will make use of Python as programming language, both for collecting the data and for running the statistical model. I collected a dataset of texts to make up the corpus from the publicly accessible database of the European Union. At eur-lex.europa.eu, one can find the official and comprehensive access to all of EU legal documents, run by the Publications Office of the European Union[1]. Among treaties, legal acts and international agreements signed by various limbs of the EU throughout its history, there are stored case-laws from the EU Court of Justice (EUCJ) including judgments and orders, opinions and views of Advocates General, opinions of the Court on draft international agreements. Via a research tool available on the website, I was able to obtain a list of all judgments of the EUCJ from 2000 until 2015, using the CELEX[2] reference number for EU legal documents. I will analyze the textual content of these sentences using LDA, and expose the possible patterns that may arise from the model.

1 For more info follow the link op.europa.eu.
2 To know more about the CELEX number see here eur-lex.europa.eu/content/help/faq/celex-number.html.

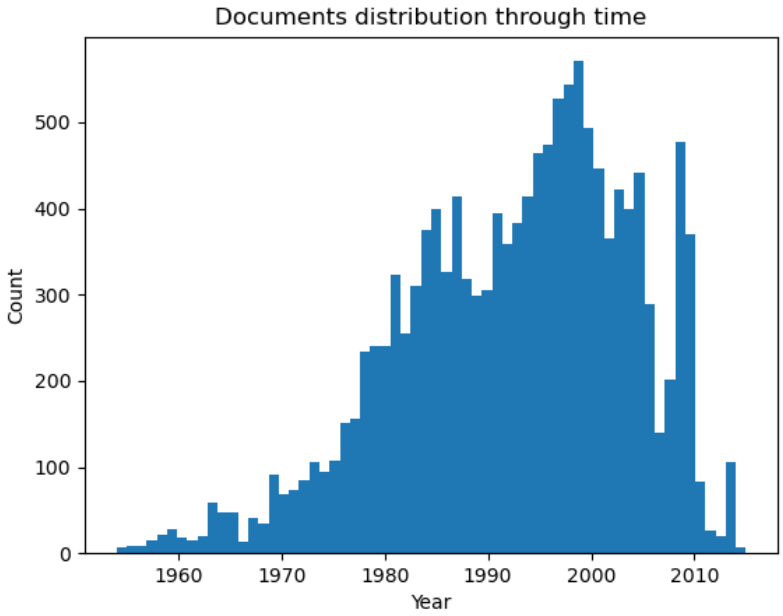


Figure 2: Distribution of judgments of the EUCJ available for download.

4.1 BACKGROUND

As mentioned in Chapter 2, legal professionals have sought to employ information retrieval and machine learning methods to reduce manual labor and increase accuracy. In a typical computer assisted review (CAR) setting, one trains a computer to categorize documents based on relevancy to a legal case using a set of seed documents labeled by experts reviewers or lawyers. CAR has three main components: a domain expert, an analytics or categorization engine, and a method for validating results. The expert is a well trained human reviewer who can identify and label relevant and irrelevant documents; the categorization engine expands the expert's knowledge to the whole corpus via indexing, relevance-ranking, classification methods, and topic modeling; finally, a validation method such as statistical sampling helps lawyers validate whether the system's

results are desired by the review team. The critical task which is going to be affected by LDA is the intermediate one, i.e. the classification and automatic procedure for analyzing large samples of documents, which cannot physically be done even by the expert.

As thoroughly described in Chapter 2, probabilistic topic modeling allows us to represent the properties of a corpus with a small collection of topics or concepts, far fewer than the vocabulary size of a corpus. LDA enables us to infer the values of the latent topic variables and the topic structure of each document in the corpus. An added advantage of topic modeling, against other categorization methods, is that it allows for handling word polysemy and synonymy of words that causes poor precision and recall in keyword-based searches [6].

Electronic Discovery

In the field of legal proceedings, Electronic Discovery is usually practiced following a framework, known as the *Electronic Discovery Reference Model*, developed by an homonym agency. Such agency develops and publishes e-discovery and information governance frameworks[3]. Among other things, they created and perfectioned a common framework for e-discovery, i.e. the ERDM. The diagram represents a conceptual view of the e-discovery process, not a literal, linear or waterfall model. It also portrays an iterative process, where one might repeat the same step numerous times, aiming at more precise results, or circle back to earlier steps, to refine the approach as a better understanding of the data emerges from the preliminary find-

3 edrm.net/resources/frameworks-and-standards/.

ings. According to the EDRM, the process of e-discovery can be sum up into the following stages: identification, preservation, collection, processing, review, analysis, production, presentation. Besides the stage of processing, which later I'll show that is needed, the main focus of the LDA statistical model is on the analysis part, providing an enhanced tool for carrying out the task. Automated legal analysis is expected to greatly aid the process of case synthesis and to have an influence on all aspects of law [13]. Being able to automate analysis has the potential of greatly decreasing the manual work needed and allows for a larger portion of available case law to be used in research and case synthesis [17].

In my case, to investigate the possibility of automatically finding topics in court sentences, I'll make use of LDA to discover and annotate the dataset with thematic information. My objective is to show the functioning of the model on a new dataset, and for this purpose I'll make use of the EUCJ judgments collection.

4.2 DATA

As mentioned, the dataset used is composed of legal texts from the EUCJ. For the purpose of adding originality to the application, I chose to collect and analyze only sentences that were available in Italian, performing therefore natural language processing on this language. After performing data exploration, in Table 1 are shown some statistics of what the dataset looks like.

NUM. OF DOCUMENTS	2876	
AVERAGE DOCUMENT LENGTH	5623	words
AVERAGE LENGTH AFTER PROCESSING	2150	words
DICTIONARY SIZE (WITH K=5)	5640	words

Table 1: Dataset statistics for years 2000-2015.

The kind of document that I am going to work with is a plain text document, with a formal structure that is mainly respected in all examples, and follows this scheme in order:

A. Keywords (*Parole Chiave*)

B. Summary (*Massime*)

C. Parties (*Parti*)

D. Grounds (*Motivazione*)

E. Decision on costs (*Decisione relativa alle spese*)

F. Operative part (*Dispositivo*)

Apart from the summary, the main content which differs across all documents in terms of form of text and words used is "Grounds" and the following dispositions. Oftentimes, the "Grounds" section is the longest and thus richer of terms. The other sections include common text structures and formulas that are repeated throughout the corpus, as well as proper names of parties, judges, lawyers or EU countries' democratic bodies. Understanding the common structure of these documents allowed me to proceed to the following stage, the pre-processing.

Preprocessing

The original format of documents that I was able to download is `html`, and with the help of the Python library `html2text` I converted all of them into plain text documents. Once in this format, I managed to process the text and to keep only the sections that were significant to the uniqueness of the text, under my previous observation. In order to prepare the documents for the statistical model, and increase the effectiveness of the results, one has to first clean the text from any kind of impurity that might affect the result. The step that I undertook on the samples collected are the following:

1. First, as mentioned, I retrieved only the part of text corresponding to the Grounds, Decisions, and Operative parts;

2. Then, with the help of *regular expressions*, I eliminate from the content any kind of hyperlinks or layout structures that are in the text. I also take out all numbers and all law-numbering system references (such as CELEX or others), since when taken alone, these number don't make sense, nor they could represent some kind of topic.

3. I proceed to tokenization, that is, all sentences are broken into words, removing all trailing spaces and any type of punctuation.

4. The text is now a list of words, and from this list I remove all common and frequent words (such as "e", "per", "quindi", "dove") which in NLP are called *stopwords*. Furthermore, to simplify even more the model's job, I removed also the words with length less than 4 letters, which usually don't contain much meaning.

5. Finally, the words that remain are converted to lowercase, and then they are either stemmed or lemmatized.

Lemmatization and stemming

When processing natural language, there are a lot of words which are inflected into many forms, but which actually all come from the same term. This is the case with verb tenses, plurals, superlatives, and more. This phenomenon is present in any language, such as English, but even more in Italian, with an increased number of verb tenses, more articles and prepositions, and word altering (e.g. "bello" - "bellissimo", "casa" - "casetta"). From the point of view of computational linguistic, this gives an undesired effect, since the vocabulary of the corpus will contain many words with the same meaning but will be treated by the algorithm as if they were separate. For this reason, researchers in the field have come up with solutions, and the two most common are stemming and lemmatization:

- *Stemming* is the process of reducing inflected word to their word stem, or root form. This process is rough, since the last letters of the word are cut out in order to reach a common base form, which doesn't need to make sense in itself, but is a placeholder for the family of words that it represents. For example, the italian words "andare", "andato", "andò" are all mapped to the root "and".

- On the other hand, *lemmatizing* is a more educated process. It consists of mapping the word using its meaning and part of speech, so that even altered forms of words can be mapped to the right base form. For example, the word "meglio" would be mapped with "bene", which is the base form, even if its root is different.

Being that the two processes are mutually exclusive, I have implemented in my code an option that contemplates the two methods, and proceeds afterwards with different processed documents leading to different results. For the lemmatization algorithm I used the `spacy` Python library by Honnibal and Montani, while for stemming I used the Snowball Stemmer from Bird, Loper, and Klein's `nltk` library.

4.3 MODEL

In order to be able to train the model of Latent Dirichlet Allocation, there's still one step that the documents have to undergo before the actual training. Every document has to be converted from a list of tokens into a bag of words (BOW) or a *tfidf* matrix. The former means that each word is considered separately in the document, and the order of terms is no longer relevant. What matters is the count of appearances of the word in the text, and how is it frequent in a document or rare in another. The latter one instead takes a step forward from BOW and converts the count into a ratio of the term frequency over the inverse document frequency. This is able to capture new information than BOW, such as the case when a word is not frequent in a document, but it is also used in a subset of other documents, making it a plausible indicator of a shared underlying topic.

Once the BOW is computed for each document, a shared dictionary is created, mapping for convenience every word in the corpus with a numerical id. Then, we can finally apply LDA to the dataset. The model I use and the algorithms to estimate the parameters are provided by Python's library `gensim` [16]. There are hyperparameters that are required by the model to be

chosen. Among these, we have: the number of topics k, which must be chosen by the user; the prior probabilities of each topic α; and the prior belief on word-probabilities η. In order to find the optimal combination of such parameters, I have run different times the model, and I will show the evaluation in the next section.

4.4 EVALUATION AND RESULTS

Even though probabilistic topic models are a popular tool for text analysis, providing both a predictive and latent topic representation of the corpus, one must not fall into the assumption that the latent space discovered is generally meaningful and useful, and that it would be hard to evaluate the assumption due to its unsupervised training process. However, it is important to identify the quality of a trained model, and to do so we require an objective measure. The traditional approach, and also the more subjective, consists in manually exploring the result: listing the more relevant words of each topic, classifying a known document and see if the model captures the subject. The alternative approach consists in computing an intrinsic evaluation metric, named topic coherence.

Topic coherence measures the score of a single topic by measuring the degree of semantic similarity between high scoring words in the topic. These measurements help distinguish between topics that have a semantic interpretation and topics that are just an artifact of the statistical inference. Even among coherence measures, there are different ways in which it can be calculated. The function that I will use is taken from `gensim`, and it implements the *UMass* measure. Proposed by Mimno et

al. (2011), it implements an asymmetrical confirmation measure between top words pairs. The summation of *UMass coherence* accounts for the ordering among the top word of a topic. The resulting formula is:

$$C_{UMass} = \frac{2}{N \cdot (N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} \log \frac{P\left(w_i, w_j\right) + \epsilon}{P\left(w_j\right)}$$

where N is the number of top words to be compared, ϵ is the smoothing parameter (to avoid the logarithm of zero), while $P\left(w_i, w_j\right)$ and $P\left(w_j\right)$ are probabilities. Word probabilities are estimated based on word co-occurrences or document frequencies of the original documents used for learning the topics.

I have used topic coherence with the UMass measure to perform tuning of the hyperparameters of the LDA model and what I've found is in Table 2. In order to speed up the process, I didn't use the whole corpus, but instead I restricted it to texts from 2010 onward, yielding 612 documents. The result that I've found is that the coherence measure is not very different among the top 5 combinations, and that 16 topics, with $\alpha = 0.31$ and $eta = symmetric$[4] is a the first place. For this reason, I will use these as parameters from now on.

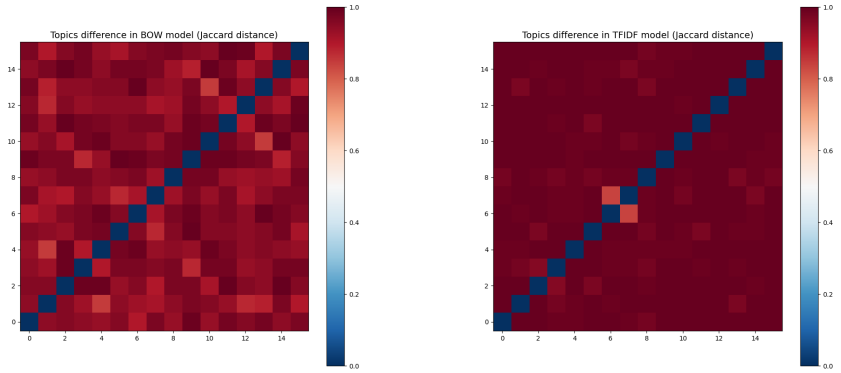
Furthermore, I also have experimented with different features outside of the LDA model. In particular, I had the choice of whether to stem or lemmatize the words during preprocessing, and whether to use the simple BOW or move the corpus representation to TFIDF. What I've discovered is that the topic results were more distinct, in terms of hidden probability distributions of the topics, when using the *tfidf* and the lemmatization techniques. As can be seen in Figure 3, the dissimilarity across topics for BOW is lower than for TFIDF. For this reason,

4 The prior is uniform on all the topics.

NUM. OF TOPICS	ALPHA PRIOR	ETA PRIOR	COHERENCE
16	0.31	symmetric	0.2698
4	symmetric	0.01	0.2687
2	0.31	0.01	0.2632
2	symmetric	0.61	0.2626
18	0.01	0.31	0.2603

Table 2: Results of hyperparameter tuning. The grid of tested parameters is {*k*: 2, ..., 20, 50; *alpha*: 0.01, 0.31, 0.61, 0.91, 1, symmetric, asymmetric[5]; *eta*: 0.01, 0.31, 0.61, 0.91, 1, symmetric, asymmetric}.

I have adopted this setting for the rest of the application. In Appendix A, you can find the complete list of the 16 topics with their top words and probabilities that were the result of training LDA on the corpus from 2000 to 2015.



(a) BOW topics similarity (b) TFIDF topics similarity

Figure 3: Topics similarity with different corpus representations: bag-of-words and *tfidf*. Red means two topics are very dissimilar, blue means they are similar.

Instead of using topic coherence, we can try and have a look to how the model behaves when trying to classify a known document. For this purpose I will keep the model with the best hyperparameters found above. I have chosen a document from the

more recent ones, dating back to 2015 (CELEX `62014CJ0407`)[6]. The title says: *"Rinvio pregiudiziale — Politica sociale — Direttiva 2006/54/CE — Parità di trattamento fra uomini e donne in materia di occupazione e impiego — Licenziamento a carattere discriminatorio — Articolo 18 — Risarcimento o riparazione del danno effettivamente subìto — Carattere dissuasivo — Articolo 25 — Sanzioni — Danni punitivi"*, which means that the judgment will talk about labor rights, discrimination, men and women. Now, let's see how the LDA model classifies this document:

1. With a score of 0.613 the first topic is:
   0.012*"medicinale" + 0.010*"unione" + 0.009*"informazione" + 0.007*"tutelare" + 0.007*"medico" + 0.007*"penale" + 0.006*"trattamento" + 0.006*"autorizzazione" + 0.005*"dare" + 0.005*"personale";

2. With a score of 0.354 the second topic is:
   Topic: 0.041*"lavorare" + 0.029*"lavoratore" + 0.015*"tempo" + 0.013*"accordare" + 0.011*"periodare" + 0.011*"anno" + 0.009*"discriminazione" + 0.009*"trattamento" + 0.008*"durare" + 0.008*"contrattare";

3. With a score of 0.027 the third topics is:
   Topic: 0.046*"lavorare" + 0.034*"lavoratore" + 0.018*"datore" + 0.016*"imprendere" + 0.013*"trasferimento" + 0.012*"collettivo" + 0.011*"contrattare" + 0.010*"impresa" + 0.010*"attività" + 0.009*"sociale".

What we can conclude is that the model has been able to capture that this document is about medicine and medical benefits at work (first topic), and about labor, contracts and companies (second and third topics). As we can observe, topic 2 and 3

6 The entire text is available at eur-lex.europa.eu.

are overlapped for some words, meaning that either the model wasn't able to discern between two similar but different arguments or that the number of topics might have been a little high, suggesting that some of them could have been merged.

A

APPENDIX: LDA MODEL - TOPIC RESULTS

TOPIC	WORD	PROBABILITY
1	medicinale	0.038946338
	assicurazione	0.02667459
	pensione	0.019182913
	medico	0.016719319
	malattia	0.013138699
	brevettare	0.011038435
	prestazione	0.010790659
	cura	0.010747596
	vecchiaia	0.010598718
	familiare	0.010369804
	disoccupazione	0.008485593
	previdenziale	0.00820897
	previdenza	0.008033576
	medicare	0.007895619
	umano	0.007761372
	laboratorio	0.00705408
	contributivo	0.0065615145
2	capitale	0.03189261
	società	0.030620446
	reddito	0.023830859
	residente	0.019766463
	fiscale	0.019235337
	dividendo	0.014780506
	conferimento	0.013850701
	impostare	0.012724949
	società	0.011953875
	azionista	0.010470096

	portoghese	0.009780883
	imposizione	0.009247488
	aliquota	0.008437535
	investimento	0.008348524
	fusione	0.008333725
	quota	0.007989702
	controllato	0.00753127
3	marchiare	0.055937897
	registrazione	0.02371548
	marco	0.021762313
	uami	0.020543775
	distintivo	0.0144584915
	segnare	0.013693064
	turco	0.012416199
	denominazione	0.011662561
	alimentario	0.011032935
	opposizione	0.0091193775
	segno	0.008662247
	consumatore	0.008372857
	animale	0.007808558
	etichettatura	0.0077828607
	anteriore	0.007219826
	alimento	0.0071772747
	registrare	0.0070802406
4	franchigia	0.012776719
	creditizio	0.010724443
	viii	0.0076314523
	association	0.005523818
	verwaltungsgerichtshof	0.00543515
	adeguatezza	0.004531005
	italiana	0.0039957874
	affari	0.0039357715
	schneider	0.0032587436
	dioikitiko	0.0032536262
	subordinazione	0.0031893142

	reciprocamente	0.003109886
	integrità	0.0027490624
	infanzia	0.00268283
	impact	0.0026784744
	ossigenare	0.0025956284
	quinquennale	0.0023964543
5	conto	0.02120499
	ceco	0.016636664
	direttive	0.014771643
	polonia	0.014771411
	atti	0.011781267
	umano	0.010636103
	energetico	0.01039814
	riciclaggio	0.009850537
	protocollo	0.00712806
	provento	0.0071069323
	requisiti	0.0070833117
	veterinario	0.0070550214
	elettricità	0.0066796043
	estonia	0.0065682
	annuale	0.0065553132
	quotato	0.006158385
	terrorismo	0.0058994787
6	guidare	0.019001318
	patire	0.01857122
	varieta	0.009656797
	proteina	0.0069880206
	periodico	0.0067376606
	motorio	0.006322839
	deposito	0.006013217
	sementare	0.0046179127
	eccesso	0.0041331826
	omega	0.0037508302
	transizione	0.0033966722
	rendiconto	0.003169299

	technology	0.0031580438
	raccomandato	0.0030739622
	dignita	0.0030355002
	verificatisi	0.00270875
	parare	0.0027025046
7	inadempimento	0.05092107
	direttiva	0.029711556
	impartire	0.028276555
	omessa	0.026948629
	parlamento	0.023734042
	mancata	0.021995815
	conformarsi	0.020992761
	ricorso	0.017560001
	trasposizione	0.016413063
	fondatezza	0.015966095
	dispositivo	0.015771367
	granducato	0.01386533
	lussemburgo	0.013344288
	scadenzare	0.013255314
	repubblica	0.011636479
	incombere	0.011397816
	settima	0.010521661
8	rifiuto	0.022503544
	ambiente	0.012196629
	ambientale	0.011548477
	francese	0.010698498
	italiano	0.009922414
	repubblica	0.009687228
	trasposizione	0.009655274
	irlanda	0.00926764
	conformarsi	0.0092409495
	progetto	0.009117553
	superficie	0.0086352
	impianto	0.008411073
	progettare	0.00827369

	impattare	0.00818843
	piano	0.007896673
	programmare	0.0077311085
	legislativo	0.007628837
9	lavorare	0.005191218
	lavoratore	0.005168551
	convenzione	0.0031047356
	contrattare	0.0029972557
	attività	0.0029665574
	prestazione	0.0027651708
	cittadino	0.0026025355
	soggiornare	0.0022941753
	attività	0.0021040435
	decretare	0.00208649
	unione	0.0019861702
	territorio	0.0019541983
	servizo	0.0018978204
	datore	0.0018627411
	accordare	0.0017982905
	autorita	0.0017791983
	germania	0.0017216415
10	latta	0.027017293
	quantitativo	0.012682486
	libera	0.012656854
	casa	0.012140723
	national	0.0096608065
	asilo	0.009630517
	emergenza	0.009100772
	pericolo	0.0085994415
	search	0.008467595
	lattiero	0.0083493395
	prelievo	0.007696815
	mark	0.0075570154
	double	0.006791874
	quotes	0.0067375246

	asterisk	0.006737485
	question	0.0067294445
	english	0.0065030605
11	acqua	0.041098427
	veicolo	0.025290007
	veicolare	0.01875517
	zuccherare	0.017986918
	autoveicolo	0.015429957
	refluo	0.014343414
	immatricolazione	0.014149236
	urbano	0.013108024
	nord	0.012442305
	bretagna	0.010472716
	nave	0.008723491
	vittima	0.008678163
	immatricolare	0.008099966
	motore	0.007986291
	autovettura	0.007934083
	gran	0.007168198
	portuale	0.0068605966
12	doganale	0.03934744
	merce	0.019591523
	importazione	0.015239799
	dazio	0.01477775
	esportazione	0.011966461
	voce	0.010424635
	codice	0.009846432
	restituzione	0.009398345
	classificazione	0.0087693175
	accisa	0.008199768
	tariffario	0.0077605266
	sottovoce	0.0076141646
	noto	0.0072596567
	carne	0.007225971
	obbligazione	0.0071607404

	transitare	0.007021806
	bevanda	0.0067867707
13	habitat	0.028499203
	sito	0.026630767
	zona	0.026506111
	conservazione	0.026358293
	uccello	0.022758322
	naturale	0.018961659
	selvatico	0.015267469
	cacciare	0.012454098
	popolazione	0.01023852
	area	0.008764983
	fauna	0.008400272
	flora	0.007261478
	protezione	0.0067961803
	stupefare	0.0064705187
	allevamento	0.006188494
	catturare	0.0061815865
	cabotaggio	0.006063358
14	sesto	0.017837955
	appaltare	0.017561164
	bene	0.016827552
	appalto	0.016532399
	servizo	0.016365435
	aggiudicazione	0.015204226
	passivo	0.013117141
	cessione	0.012711805
	rete	0.011562647
	fornitura	0.01060481
	gara	0.010277606
	operazione	0.009579483
	detrazione	0.008403622
	universale	0.00830547
	telecomunicazione	0.008170406
	locazione	0.008118416

	aggiudicatrice	0.007906391
15	tribunale	0.014141874
	aiuto	0.011267533
	aiutare	0.0109441895
	impugnare	0.010820735
	impugnazione	0.007880059
	recuperare	0.00563854
	controverso	0.0048256703
	pescare	0.0046397
	mercato	0.0046182317
	infrazione	0.004329106
	animale	0.004309066
	ammenda	0.0042832154
	annullamento	0.004248679
	sanzione	0.004074614
	violazione	0.00386304
	produttore	0.0038354397
	penale	0.003788304
16	volare	0.0117775835
	bulgaria	0.011565335
	romania	0.011390333
	rumore	0.011294781
	donazione	0.00795914
	ottico	0.0076153288
	mediazione	0.0075948513
	riservatezza	0.007181601
	cancellazione	0.0057392837
	forno	0.005429461
	periziare	0.0052639027
	nullità	0.0046216575
	adesione	0.0045590107
	technische	0.0037791592
	acustico	0.0036607706
	levare	0.0035567256
	concentrato	0.0035059622

BIBLIOGRAPHY

[1] Steven Bird, Edward Loper, and Ewan Klein. *Natural Language Processing with Python*. O'Reilly Media Inc, 2009.

[2] David M. Blei. "Probabilistic Topic Models." In: *Commun. ACM* 55.4 (Apr. 2012), 77–84. ISSN: 0001-0782.

[3] David M Blei. "Topic modeling and digital humanities." In: *Journal of Digital Humanities* 2.1 (2012), pp. 8–11.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation." In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[5] Bruno De Finetti. "La prévision: ses lois logiques, ses sources subjectives." In: *Annales de l'institut Henri Poincaré*. Vol. 7. 1. 1937, pp. 1–68.

[6] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. "Indexing by latent semantic analysis." In: *Journal of the American society for information science* 41.6 (1990), pp. 391–407.

[7] James M Dickey. "Multiple hypergeometric functions: Probabilistic interpretations and statistical uses." In: *Journal of the American Statistical Association* 78.383 (1983), pp. 628–637.

[8] Clint P. George, Sahil Puri, Daisy Zhe Wang, Joseph N. Wilson, and William F. Hamilton. "SMART Electronic Legal Discovery Via Topic Modeling." In: *FLAIRS Conference*. 2014.

[9] Thomas Hofmann. "Probabilistic latent semantic indexing." In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. 1999, pp. 50–57.

[10] Thomas Hofmann. "Unsupervised learning by probabilistic latent semantic analysis." In: *Machine learning* 42.1-2 (2001), pp. 177–196.

[11] Matthew Honnibal and Ines Montani. "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing." To appear. 2017.

[12] Karen Sparck Jones. "A statistical interpretation of term specificity and its application in retrieval." In: *Journal of documentation* (1972).

[13] John O McGinnis and Russell G Pearce. "The great disruption: How machine intelligence will transform the role of lawyers in the delivery of legal services." In: *Actual Probs. Econ. & L.* (2019), p. 1230.

[14] David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. "Optimizing semantic coherence in topic models." In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 2011, pp. 262–272.

[15] Andrew Y. Ng and Michael I. Jordan. "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes." In: *Advances in Neural Information Processing Systems 14*. Ed. by T. G. Dietterich, S. Becker, and Z. Ghahramani. MIT Press, 2002, pp. 841–848.

[16] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling with Large Corpora." English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[17] Ylja Remmits. "Finding the Topics of Case Law: Latent Dirichlet Allocation on Supreme Court Decisions." In: (2017).

[18] Ibrar Yaqoob, Ibrahim Hashem, Abdullah Gani, Salimah Mokhtar, Ejaz Ahmed, Nor Anuar, and Athanasios Vasilakos. "Big Data: From Beginning to Future." In: *International Journal of Information Management* 36 (Dec. 2016).

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

https://bitbucket.org/amiede/classicthesis/