Quantifying News Narratives to Predict Movements in Market Risk

Thomas Dierckx, Jesse Davis, and Wim Schoutens

Abstract The theory of *Narrative Economics* suggests that narratives present in media influence market participants and drive economic events. In this chapter, we investigate how financial news narratives relate to movements in the CBOE Volatility Index. To this end, we first introduce an uncharted dataset where news articles are described by a set of financial keywords. We then perform topic modeling to extract news themes, comparing the canonical latent Dirichlet analysis to a technique combining *doc2vec* and Gaussian mixture models. Finally, using the state-of-the-art XGBoost (Extreme Gradient Boosted Trees) machine learning algorithm, we show that the obtained news features outperform a simple baseline when predicting CBOE Volatility Index movements on different time horizons.

1 Introduction

Nowadays market participants must cope with new sources of information that yield large amounts of unstructured data on a daily basis. These include sources such as online new articles and social media. Typically, this kind of information comes in the form of text catered to human consumption. However, humans struggle to identify relevant complex patterns that are hidden in enormous collections of data. Therefore, investors, regulators, and institutions would benefit from more sophisticated automated approaches that are able to extract meaningful insights from such information. This need has become increasingly relevant since the inception of

T. Dierckx
Department of Statistics, KU Leuven, Leuven, Belgium
e-mail: thomas.dierckx@kuleuven.be

J. Davis
Department of Computer Science, KU Leuven, Leuven, Belgium
e-mail: jesse.davis@kuleuven.be

W. Schoutens (✉)
Department of Mathematics, KU Leuven, Leuven, Belgium
e-mail: wim.schoutens@kuleuven.be

S. Consoli et al. (eds.), *Data Science for Economics and Finance*,
https://doi.org/10.1007/978-3-030-66891-4_12

Narrative Economics [23]. This theory proposes that the presence of narratives in media influence the belief systems of market participants and even directly affect future economic performance. Consequently, it would be useful to apply advanced data science techniques to discern possible narratives in these information sources and assess how they influence the market.

Currently, two distinct paradigms exist that show potential for this task. First, *topic modeling* algorithms analyze the text corpora in order to automatically discover hidden themes, or topics, present in the data. At a high level, topic models identify a set of topics in a document collection by exploiting the statistical properties of language to group together similar words. They then describe a document by assessing the mixture of topics present in the document. That is, they determine the proportion of each topic present in the given document. Second, *Text Embedding* techniques infer vector representations for the semantic meaning of text. While extremely popular in artificial intelligence, their use is less prevalent in economics. One potential reason is that topic models tend to produce human-interpretable models as they associate probabilities with (groups of) words. In contrast, humans have more difficulties capturing the meaning of the vectors of real values produced by embedding methods.

In the context of narratives, preceding work in the domain of *topic modeling* has already shown that certain latent themes extracted from press releases and news articles can be predictive for future abnormal stock returns [10, 9] and volatility [3]. Similarly, researchers have explored this using *Text Embedding* on news articles to predict bankruptcy [16] and abnormal returns [25, 1].

The contribution of this chapter is multifaceted. First, we noticed that most research involving *topic modeling* is constrained by the intricate nature of natural language. Aspects such as rich vocabularies, ambiguous phrasing, and complex morphological and syntactical structures make it difficult to capture information present in a text article. Consequently, various imperfect preprocessing steps such as stopword removal, stemming, and phrase detection have to be utilized. This study therefore refrains from applying quantification techniques on raw news articles. Instead, we introduce an unprecedented corpus of historical news metadata using the *Financial Times* news API, where each news article is represented by the set of financial sub-topics it covers. Second, at the time of writing, this study offers the first attempt to investigate the interplay between narratives and implied volatility. We hypothesize that the presence of financial news narratives can instill fear in market participants, altering their perception of market risk and consequently causing movements in the CBOE Volatility Index, also known as the *fear index*. In order to test this hypothesis, we first extract latent themes from the news corpus using two different topic modeling approaches. We employ the canonical latent Dirichlet analysis but also an alternative methodology using the modern *doc2vec* and Gaussian mixture models. Finally, using the state-of-the-art XGBoost (Extreme Gradient Boosted Trees) machine learning algorithm, we model the interplay between the obtained news features and the CBOE Volatility Index. We show that we can predict movements for different time horizons, providing empirical evidence for the validity of our hypothesis.

The remainder of this chapter is structured as follows: Section 2 outlines the preliminary material necessary to understand the applied methodology in our study, which in turn is detailed in Sect. 3. Section 4 then presents the experimental results together with a discussion, and finally Sect. 5 offers a conclusion for our conducted research.

2 Preliminaries

Our approach for extracting news narratives from our news dataset builds on several techniques, and this section provides the necessary background to understand our methodology. Section 2.1 describes existing topic modeling methodologies. Section 2.2 presents the Gradient Boosted Trees machine learning model. Lastly, Sect. 2.3 defines the notion of market risk and its relation to the CBOE Volatility Index.

2.1 Topic Modeling

Topic models are machine learning algorithms that are able to discover and extract latent themes, or *topics*, from large and otherwise unstructured collections of documents. The algorithms exploit statistical relationships among words in documents in order to group them into topics. In turn, the obtained topic models can be used to automatically categorize or summarize documents up to scale that would be unfeasible to do manually.

This study considers two different approaches of topic modeling. Section 2.1.1 details the popular latent Dirichlet analysis (LDA). Sections 2.1.2 and 2.1.3 describe the paragraph vector technique and Gaussian mixture models, respectively. Note that only the former is an actual topic modeling algorithm. However, the Methodology section (Sect. 3) will introduce a topic modeling procedure by combining paragraph vector and Gaussian mixture models.

2.1.1 Latent Dirichlet Analysis

Latent Dirichlet analysis (LDA) [4] belongs to the family of generative probabilistic processes. It defines topics to be random distributions over the finite vocabulary present in a corpus. The method hinges on the assumption that every document exhibits a random mixture of such topics and that the entire corpus was generated by the following imaginary two-step process:

1. For every document d in corpus D, there's a random distribution θ_d over K topics where each entry $\theta_{d,k}$ represents the proportion of topic k in document d.

2. For each word w in document d, draw a topic z from θ_d and sample a term from its distribution over a fixed vocabulary given by β_z.

The goal of any topic modeling is to automatically discover hidden topic structures in the corpus. To this end, LDA inverts the previously outlined imaginary generative process and attempts to find the hidden topic structure that *likely* produced the given collection of documents. Mathematically, the following posterior distribution is to be inferred:

$$P(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{P(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}. \tag{1}$$

Unfortunately, Eq. 1 is generally deemed computationally intractable. Indeed, the denominator denotes the probability of seeing the observed corpus under any possible topic model. Since the number of possible topic models is exponentially large, it is computational intractable to compute this probability [4]. Consequently, practical implementations resort to approximate inference techniques such as online variational Bayes algorithms [13].

The inference process is mainly governed by the hyper-parameters K and Dirichlet priors α and η. The parameter K indicates the number of latent topics to be extracted from the corpus. The priors control the document-topic distribution θ and topic-word distribution β, respectively. Choosing the right values for these hyper-parameters poses intricate challenges due to the unsupervised nature of the training process. Indeed, there is no prior knowledge as to how many and what kind of hidden topic structures reside within a corpus. Most research assesses model quality based on manual and subjective inspection (e.g., [3, 9, 10]). They examine the most probable terms per inferred topic and subsequently gauge them for human interpretability. Because this is a very time-intensive procedure and requires domain expertise, an alternative approach is to use quantitative evaluation metrics. For instance, the popular perplexity metric [26] gauges the predictive likelihood of held-out data given the learned topic model. However, the metric has been shown to be negatively correlated with human interpretable topics [6]. Newer and better measures have been proposed in the domain of topic coherence. Here, topic quality is based on the idea that a topic is coherent if all or most of its words are related [2]. While multiple measures have been proposed to quantify this concept, the coherence method named C_v has been shown to achieve the highest correlation with human interpretability of the topics [20].

2.1.2 Paragraph Vector

Paragraph vector [15], commonly known as *doc2vec*, is an unsupervised framework that learns vector representations for semantics contained in chunks of text such as sentences, paragraphs, and documents. It is a simple extension to the popular

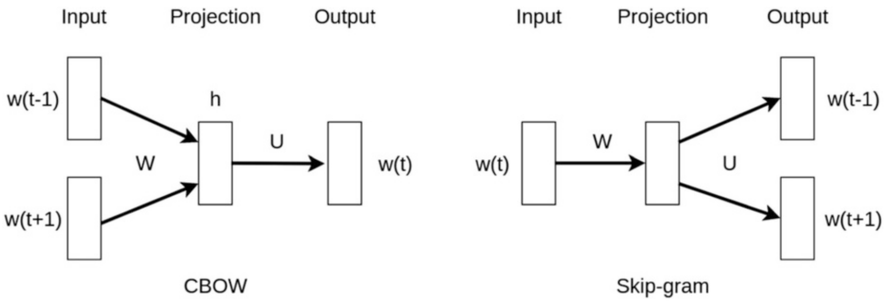

Fig. 1 The two *word2vec* approaches CBOW (left) and skip-gram (right) and their neural network architectures [17] for word predictions. The variables W and U represent matrices that respectively contain the input and output layer weights of the neural network. Function h is an aggregation function for the CBOW method to combine the multiple of input words w

word2vec model [17], which is a canonical approach for learning vector representations for individual words.

Word2vec builds on the distributional hypothesis in linguistics, which states that words occurring in the same context carry similar meaning [12]. There are two canonical approaches for learning a vector representation of a word: *continuous bag of words* (CBOW) and *skip-gram*. Both methods employ a shallow neural network but differ in input and output. CBOW attempts to predict which word is missing given its context, i.e., the surrounding words. In contrast, the *skip-gram* model inverts the prediction task and given a single word attempts to predict which words surround it. In the process of training a model for this prediction task, the network learns vector representations for words, mapping words with similar meaning to nearby points in a vector space. The architectures of both approaches are illustrated in Fig. 1. The remainder of this section continues to formally describe the CBOW method. The mathematical intuition of skip-gram is similar and can be inferred from the ensuing equations.

Formally, given a sequence of words $w_1, w_2, \ldots, w_N$, the objective of the *continuous bag of words* framework is to minimize the average log probability given by:

$$-\frac{1}{N}\sum_{n=k}^{N-k} \log p(w_n \mid w_{n-k}, \ldots, w_{n+k}) \tag{2}$$

where k denotes the number of context words to be considered on either side. Note that the value $2k + 1$ is often referred to as the window size. The prediction of the probability is typically computed using a softmax function, i.e.:

$$\log p(w_n \mid w_{n-k}, \ldots, w_{n+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \tag{3}$$

with y_i being the unnormalized log probability for each output word i, which in turn is specified by:

$$y = b + Uh(w_{n-k}, \ldots, w_{n+k}; W) \tag{4}$$

where matrix W contains the weights between the input and hidden layers, matrix U contains the weights between the hidden and output layers, b is an optional bias vector, and lastly h is a function that aggregates the multiple of input vectors into one, typically by concatenation or summation.

The word vectors are learned by performing predictions, as outlined by Eqs. 3 and 4, for each word in the corpus. Errors made while predicting words will then cause the weights W and U of the network to be updated by the backpropagation algorithm [21]. After this training process converges, the weights W between the input and hidden layer represent the learned word vectors, which span a vector space where words with similar meaning tend to cluster. The two key hyper-parameters that govern this learning process are the word sequence length n and the word vector dimension d. Currently no measures exist to quantify the quality of a learned embedding, so practitioners are limited to performing a manual, subjective inspection of the learned representation.

Paragraph vector, or *doc2vec*, is a simple extension to *word2vec* which only differs in input. In addition to word vectors, this technique associates a vector with a chunk of text, or paragraph, to aid in predicting the target words. Note that *word2vec* builds word vectors by sampling word contexts from the entire corpus. In contrast, *doc2vec* only samples locally and restricts the contexts to be within the paragraph. Evidently, *doc2vec* not only learns corpus-wide word vectors but also vector representations for paragraphs. Note that the original frameworks depicted in Fig. 1 remain the same aside from some subtle modifications. The *continuous bag of words* extension now has an additional paragraph vector to predict the target word, whereas *skip-gram* now exclusively uses a paragraph vector instead of a word vector for predictions. These extensions are respectively called *distributed memory* (PV-DM) and *distributed bag of words* (PV-DBOW).

2.1.3 Gaussian Mixture Models

Cluster analysis attempts to identify groups of similar objects within the data. Often, clustering techniques make hard assignments where an object is assigned to exactly one cluster. However, this can be undesirable at times. For example, consider the scenario where the true clusters overlap, or the data points are spread out in such a way that they could belong to multiple clusters. Gaussian mixture models (GMM) that fit a mixture of Gaussian distributions on data overcome this problem by performing *soft* clustering where points are assigned a probability of belonging to each cluster.

A Gaussian mixture model [19] is a parametric probability density function that assumes data points are generated from a mixture of different multivariate

Gaussian distributions. Each distribution is completely determined by its mean μ and covariance matrix Σ, and therefore, a group of data points x with dimension D is modeled by the following Gaussian density function:

$$\mathcal{N}(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right). \tag{5}$$

The Gaussian mixture model, which is a weighted sum of Gaussian component densities, is consequently given by:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k) \tag{6}$$

$$\sum_{k=1}^{K} \pi_k = 1. \tag{7}$$

The training process is comprised of finding the optimal values for the weights π_k, means μ_k, and covariances Σ_k of each Gaussian component. Inferring these parameters is usually done using the expectation-maximization algorithm [14]. Note that Eqs. 6 and 7 require knowing k, which is the number of Gaussian components present in the data. However, in practice this is a hyper-parameter that must be tuned. A popular method to assess how well a Gaussian mixture model fits the data is by using the Bayesian Information Criterion [22], where the model with the lowest score is deemed best. This criterion is formally defined as:

$$BIC = \ln(n)k - 2\ln(\hat{L}). \tag{8}$$

where $\hat{L}$ is the maximized value of the likelihood function of the model, n the sample size, and k is the number of parameters estimated by the model. Increasing the number of components in the model will typically yield a higher likelihood of the used training data. However, this can also lead to overfitting. The Bayesian Information Criterion accounts for this phenomenon by introducing the term $\ln(n)k$ that penalizes a model based on the number of parameters it contains.

2.2 Gradient Boosted Trees

In the domain of machine learning, algorithms infer models on a given data in order to predict a supposed dependent variable. One of the most simple algorithms is CART [5], which builds a decision tree model. However, a single tree's prediction performance usually does not suffice in practice. Instead, ensembles of trees are built where the prediction is made by multiple trees together. To this end, the Gradient Boosted Trees algorithm [11] builds a sequence of small decision trees where each tree attempts to correct the mistake of the previous one. Mathematically, a Gradient Boosted Trees model can be specified as:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \ \ f_k \in F \tag{9}$$

where K is the number of trees and f is a function in the set F of all possible CARTs. As with any machine learning model, the training process involves finding the set of parameters θ that best fit the training data x_i and labels y_i. An objective function is therefore maximized containing both a measure for training loss and a regularization term. This can be formalized as:

$$\text{obj}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i) \tag{10}$$

where l is a loss function, such as the mean squared error, t the amount of learned trees at a given step in the building process, and Ω the regularization term that controls the complexity of the model to avoid overfitting. One way to define the complexity of a tree model is by:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{11}$$

with w the vector of scores on leaves, T the number of leaves, and hyper-parameters γ and λ.

2.3 Market Risk and the CBOE Volatility Index (VIX)

In the world of derivatives, options are one of the most prominent types of financial instruments available. A prime example is the European call option, giving the holder the right to buy stock for a pre-determined price K at time T. Options are exposed to risk for the duration of the contract. To quantify this risk, the expected price fluctuations of the underlying asset are considered over the course of the option contract. A measure that gauges this phenomenon is implied volatility and varies with the strike price and duration of the option contract. A famous example of such a measure in practice is the CBOE Volatility Index. This index, better known as VIX, is a measure of expected price fluctuations in the S&P 500 Index options over the next 30 days. It is therefore often referred to as the *fear index* and is considered to be a reflection of investor sentiment on the market.

3 Methodology

The main goal of this study is to explore the following question:

> Are narratives present in financial news articles predictive of future movements in the CBOE Volatility Index?

In order to investigate the interplay between narratives and implied volatility, we have collected a novel news dataset which has not yet been explored by existing research. Instead of containing the raw text of news articles, our dataset simply describes each article using a set of keywords denoting financial sub-topics. Our analysis of the collected news data involves multiple steps. First, because there are both many keywords and semantic overlaps among different ones, we use topic modeling to group together similar keywords. We do this using both the canonical Latent Dirichlet analysis and an alternative approach based on embedding methods, which have received less attention in the economics literature. Second, we train a machine-learned model using these narrative features to predict whether the CBOE Volatility Index will increase or decrease for different time steps into the future.

The next sections explain our applied methodology in more detail. Sect. 3.1 describes how we constructed an innovative news dataset for our study. Section 3.2 then rationalizes our choice for topic modeling algorithms and details both proposed approaches. Section 3.3 then elaborates on how we applied machine learning on the obtained narrative features to predict movements in the CBOE Volatility Index. Lastly, Sect. 3.4 describes the time series cross-validation method we used to evaluate our predictions.

3.1 News Data Acquisition and Preparation

We used the *Financial Times* news API to collect keyword metadata of news articles published on global economy spanning the years 2010 and 2019. Every article is accompanied by a set of keywords where each keyword denotes a financial sub-topic the article covers. Keywords include terms such as *Central Banks*, *Oil*, and *UK Politics*. In total, more than 39,000 articles were obtained covering a variety of news genres such as opinions, market reports, newsletters, and actual news. We discarded every article that was not of the news genre, which yielded a corpus of roughly 26,000 articles. An example of the constructed dataset can be seen in Fig. 2.

```
2013-08-14: ['Emerging markets', 'Global Economy']
2013-08-14: ['Central banks', 'Markets', 'UK business & economy', '...']
2013-08-14: ['Support services', 'French economy', 'Global Economy', '...']
2013-08-14: ['Central banks', 'US quantitative easing', 'US economy', '...']
2013-08-14: ['Central banks', 'Austerity Europe', 'Global Economy', '...']
```

Fig. 2 An example slice of the constructed temporally ordered dataset where a news article is represented by its set of keywords

We investigated the characteristics of the dataset and found 677 unique financial keywords. Not all keywords are as equivalently frequent as the average and median keyword frequency is respectively 114 and 12 articles. Infrequent keywords are probably less important and too specific. We therefore decided to remove the keywords that had occurred less than five times, which corresponds to the 32nd percentile. In addition, we found that keywords *Global Economy* and *World* are respectively present in 100 and 70% of all keywords sets. As their commonality implies weak differentiation power, we omitted both keywords from the entire dataset. Ultimately, 425 unique keywords remain in the dataset. The average keyword set is 6 terms long and more than 16,000 unique sets exist.

Note that in the following sections, terms like *article*, *keyword set*, and *document* will be used interchangeably and are therefore equivalent in meaning.

3.2 Narrative Extraction and Topic Modeling

There are several obvious approaches for extracting narratives and transforming the news corpus into a numerical feature matrix. The most straightforward way is to simply consider the provide keywords about financial sub-topics and represent each article as a binary vector of dimension 1×425, with 1 binary feature denoting the presence/absence of each of the 425 unique keywords. However, this approach yields a sparse feature space and more importantly neglects the semantics associated with each keyword. For example, consider the scenario where three sets are principally equal except for respectively those containing the terms *Federal Reserve*, *Inflation*, and *Climate*. Using the aforementioned approach, this scenario would yield three vectors that are equal in dissimilarity. In contrast, a human reader would use semantic information and consider the first two sets to be closely related. Naturally, incorporating semantic information is advantageous in the context of extracting narratives. We therefore employ topic modeling techniques that group keywords into abstract themes or *latent topics* based on co-occurrence statistics. This way, a keyword set can be represented as a vector of dimension $1 \times K$, denoting the present proportion of each latent topic k_i. In doing so, keyword sets become more comparable on a semantic level, solving the previously outlined problem. Figure 3 demonstrates the result of this approach, where an over-simplified scenario is depicted using the three keyword sets from the previous example. The keyword sets containing the keywords *Federal Reserve* and *Inflation* are now clearly mathematically more similar, suggesting the persistence of some narrative during that time.

To conclude formally, given a series of N news articles each represented by a keyword set, we first transform every article into a vector representing a mixture of K latent topics. This yields a temporally ordered feature matrix X of dimension $N \times K$ where each entry $x_{n,k}$ represents the proportion of topic k in article n. We then aggregate the feature vectors of articles published on the same day by summation,

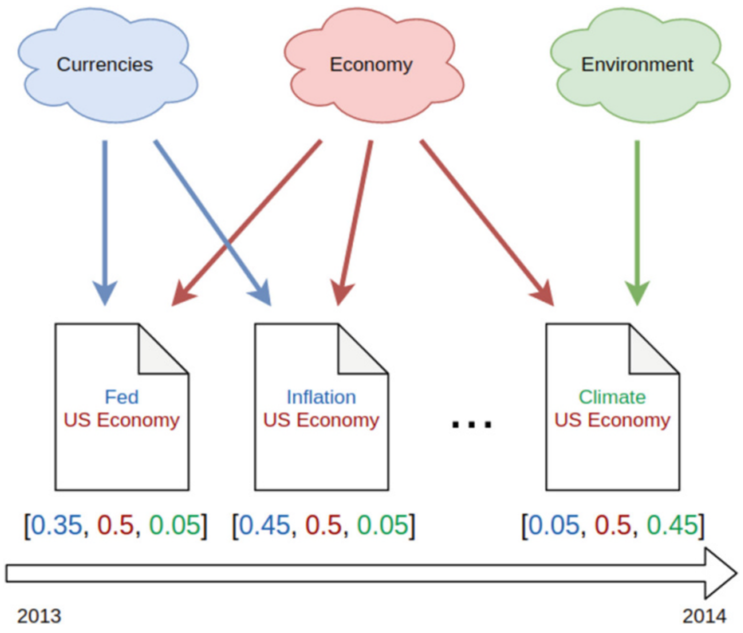

Fig. 3 An illustration of keyword sets being expressed as combinations of their latent themes. In this scenario, the three existing latent themes (clouds) make the documents directly comparable. As a consequence, more *similar* documents are closer to each other in a vector space

producing a new feature matrix X' of dimension $T \times K$, where each entry $x_{t,k}$ now represents the proportion of topic k on day t.

The following sections present how we employed two different approaches to achieve this transformation.

3.2.1 Approach 1: Narrative Extraction Using Latent Dirichlet Analysis

In our study, we utilized the Python library *Gensim* [18] to build LDA topic models. As explained in Sect. 2.1.1, the learning process is primarily controlled by three hyper-parameters K, α, and β. In the interest of finding the optimal hyper-parameter setting, we trained 50 different LDA models on all news articles published between the years 2010 and 2017 by varying the hyper-parameter K from 20 to 70. Prior distributions α and β were automatically inferred by the algorithm employed in *Gensim*. Subsequently, we evaluated the obtained models based on the proposed topic coherence measure C_v [20]. Figure 4 shows the coherence values for different values of K.

Note that the model achieving the highest score is not necessarily the best. Indeed, as the number of parameters in a model increases, so does the risk of overfitting. To alleviate this, we employ the elbow method [24] and identify the

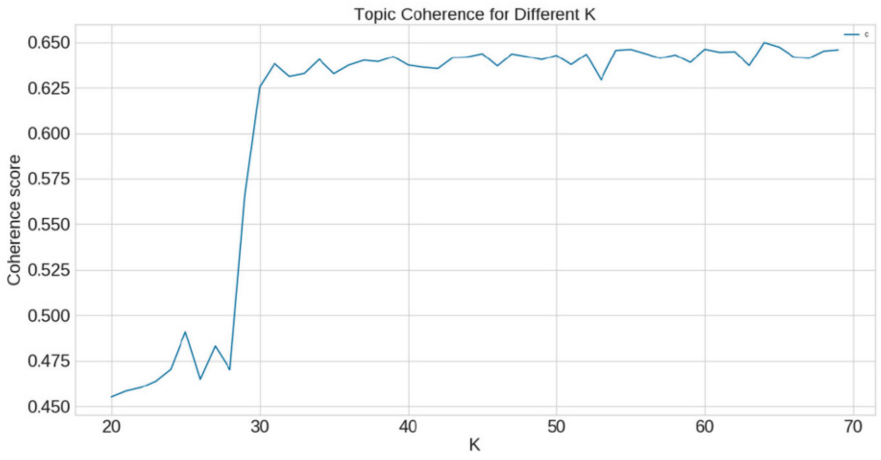

Fig. 4 Topic coherence score achieved by different LDA models for varying values of k. Results were obtained by training on news articles published between the years 2010 and 2017

smallest number of k topics where the score begins to level off. We observed this phenomenon for $k = 31$, where the graph (Fig. 4) shows a clear angle or so-called elbow. Although a somewhat subjective method, this likely yields an appropriate value for K that captures enough information without overfitting on the given data.

Finally, we can transform N given news articles into a temporally ordered feature matrix X of dimension $N \times 31$ using the best performing topic model *LDA(31)*. In turn, we aggregate the feature vectors of articles published on the same day by summation, transforming matrix X into matrix X' of dimension $T \times 31$.

3.2.2 Approach 2: Narrative Extraction Using Vector Embedding and Gaussian Mixture Models

As LDA analyzes documents as bag of words, it does not incorporate word order information. This subtly implies that each keyword co-occurrence within a keyword set is of equal importance. In contrast, vector embedding approaches such as *word2vec* and *doc2vec* consider co-occurrence more locally by using the word's context (i.e., its neighborhood of surrounding words). In an attempt to leverage this mechanism, we introduced order in the originally unordered keyword sets. Keywords belonging to the same financial article are often related to a certain degree. Indeed, take, for example, an article about Brexit that contains the keywords *Economy*, *UK Politics*, and *Brexit*. Not only do the keywords seem related, they tend to represent financial concepts with varying degrees of granularity. In practice, because keyword sets are unordered, more specialized concepts can end up in the vicinity of more general concepts. Evidently, these concepts will be less related, which might introduce noise for vector embedding approaches looking at a word's

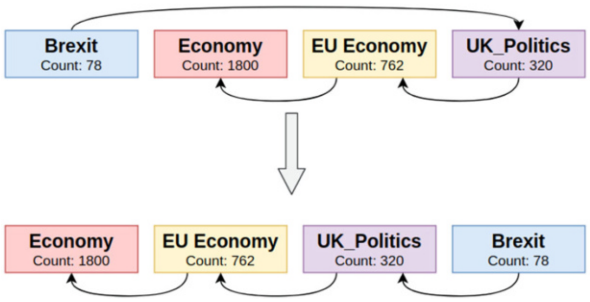

Fig. 5 An illustration of ordering a keyword set based on total corpus frequency. The arrow is an indicator of subsumption by a supposed parent keyword

context. We therefore argue that by ordering the keywords based on total frequency across the corpus, more specific terms will be placed closer to their subsuming keyword. This way, relevant terms are likely to be brought closer together. An example of this phenomenon is demonstrated in Fig. 5.

Note that the scenario depicted in Fig. 5 is ideal, and in practice the proposed ordering will also introduce noise by placing incoherent topics in each other's vicinity. The counts used for ordering were based on news articles published between 2010 and 2017.

For the purpose of topic modeling, we combined *doc2vec* with Gaussian mixture models. First, *doc2vec* is trained on a collection of ordered keyword sets, generating a vector space where similar sets are typically projected in each other's vicinity. Next, a Gaussian mixture model is fitted on this vector space to find k clusters or *latent topics*. In doing so, each document can then be expressed as a mixture of different clusters. *doc2vec* allows retrieving the original document associated with a certain vector. This way, we can compute word frequencies for each cluster, which in turn allows us to interpret them.

In practice, we built *doc2vec* models using the Python library *Gensim*. Recall that sliding window size w and vector dimension d are both important hyper-parameters to the training process. Unlike LDA, there is no quantifiable way to assess the effectiveness of an obtained vector space. We therefore built six *doc2vec* models using both *PV-DBOW* and *PV-DM*, choosing different sliding window sizes $w \in \{2, 5, 8\}$ for a constant $d = 25$. Most research utilizing these techniques tends to use arbitrary vector dimensions without experimental validation (i.e., [17, 15, 8]), suggesting that performance isn't very sensitive to this hyper-parameter. Our decision for the dimension hyper-parameter was ultimately also arbitrary, but chosen to be on the low end given that we are analyzing a relatively small corpus with a limited vocabulary. Each of the obtained vector spaces is then fitted with a Gaussian mixture model to cluster the vector space into k different topics. For each vector space, we found the optimal value for k by fitting 50 different Gaussian mixture models with $k \in \{20, 70\}$. We then applied the elbow technique, introduced

Table 1 The optimal number of Gaussian mixture components for each vector space obtained by using *doc2vec* with vector dimension $d = 25$ and window size $w \in \{2, 5, 8\}$. The results were found by applying the elbow method on the BIC of the Gaussian mixture models

	2	5	8
PV-DBOW	32	38	40
PV-DM	34	36	30

in Sect. 3.2.1, on the graphs of the obtained Bayesian Information Criterion scores. Table 1 presents the optimal values for k found for each vector space.

For each configuration, we can now transform the N given news articles into a temporally ordered feature matrix X of dimension $N \times K$ by first obtaining the vector representation for each article using *doc2vec* and subsequently classifying it with the associated Gaussian mixture model. Again, feature vectors of articles published on the same day are aggregated by summation, transforming matrix X into matrix X' of dimension $T \times K$.

3.3 Predicting Movements in Market Risk with Machine Learning

In our study, we took the CBOE Volatility Index as a proxy for market risk. Instead of solely studying 1-day-ahead predictions, we chose to predict longer-term trends in market risk as well. Consequently, we opted to predict whether the CBOE Volatility Index closes up or down in exactly 1, 2, 4, 6, and 8 trading days.

We downloaded historical price data of VIX through Yahoo Finance. Data points represent end-of-day close prices and have a daily granularity. To construct the actual target feature, we define the n-day-ahead difference in market implied volatility on day i as $y_i^* = (ivolatility_{i+n} - ivolatility_i)$ where $ivolatility_i$ denotes the end-of-day market-implied volatility on day i. We consider the movements to be upward whenever $y_i^* > 0$ and downward whenever $y_i^* \leq 0$. The final target feature is therefore a binary feature obtained by applying case equation 12.

$$y_i = \begin{cases} 1, & \text{if } y_i^* > 0. \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

In order to predict our target variable, we chose to employ XGBoost's implementation of Gradient Boosted Trees [7]. The implementation is fast and has been dominating Kaggle data science competitions since its inception. Moreover, because forest classifiers are robust to large feature spaces and scaling issues, we do not have to perform standardization or feature selection prior to utilization. Ultimately, we used eight distinctive XGBoost configurations in each experiment, with $max\_depth \in \{4, 5, 6, 7\}$, and $n\_estimators \in \{200, 400\}$. These models were trained on a temporally ordered feature matrix X^* of dimension $T \times (K + 1)$, obtained by concatenating the feature matrix X comprised of narrative features of dimension $T \times K$ together with the CBOE Volatility Index' close prices. Note that special care was taken to not introduce data leakage when using topic models to obtain the narrative feature matrix X. To this end, each prediction for given day t was made using feature vectors obtained by a topic model that was trained on news articles published strictly before day t.

3.4 Evaluation on Time Series

The Gradient Boosted Trees are evaluated using cross-validation, where data is repeatedly split into non-overlapping train and test sets. This way models are trained on one set and afterward evaluated on a test set comprised of unseen data to give a more robust estimate of the achieved generalization. However, special care needs to be taken when dealing with time series data. Classical cross-validation methods assume observations to be independent. This assumption does not hold for time series data, which inherently contains temporal dependencies among observations. We therefore split the data into training and test sets which take the temporal order into account to avoid data leakage. To be more concrete, we employ Walk Forward Validation (or Rolling Window Analysis) where a sliding window of t previous trading days is used to train the models and where trading day t_{t+1+m} is used for the out-of-sample test prediction. Note that special care needs to be taken when choosing a value for m. For example, if we want to perform an out-of-sample prediction for our target variable 2 days into the future given information on day t_i, we need to leave out day t_{i-1} from the train set in order to avoid data leakage. Indeed, the training data point t_{i-1} not only contains the information of narratives present on the said day but also whether the target variable has moved up or down by day t_{i+1}. It is evident that in reality we do not possess information on our target variable on day t_{i+1} at the time of our prediction on day t_i. Consequently, m has to be chosen so that $m \geq d - 1$ where d denotes how many time steps into the future the target variable is predicted.

Table 2 illustrates an example of this method where t_i denotes the feature vector corresponding to trading day i and predictions are made 2 days into the future. Note that in this scenario, when given a total of n observations and a sliding window of length t, you can construct a maximum of $n - (t + m)$ different train-test splits. Moreover, models need to be retrained during each iteration of the evaluation process, as is the case with any cross-validation method.

Table 2 Example of Walk Forward Validation where t_i represents the feature vector of trading day i. In this example, a sliding window of size three is taken to learn a model that predicts a target variable 2 days into the future. During the first iteration, we use the feature vectors of the first 3 consecutive trading days to train a model (underlined) and subsequently test the said model on the 5th day (bold), leaving out the 4th day to avoid data leakage as described in Sect. 3.4. This process is repeated j times where, after each iteration, the sliding window is shifted in time by 1 trading day

Iteration	Variable roles
1	$\underline{t_1\ t_2\ t_3}\ \cancel{t_4}\ \mathbf{t_5}\ t_6\ \cdots\ t_n$
2	$t_1\ \underline{t_2\ t_3\ t_4}\ \cancel{t_5}\ \mathbf{t_6}\ \cdots\ t_n$
⋮	⋮
j	$t_1\ \cdots\ \underline{t_{n-4}\ t_{n-3}\ t_{n-2}}\ \cancel{t_{n-1}}\ \mathbf{t_n}$

4 Experimental Results and Discussion

In this section, we present our experimental methodology and findings from our study. The study consists of two parts. First, we examined the soundness of our two proposed strategies for performing topic modeling on keyword sets. To this end, we contrasted the predictive performance of each strategy to a simple baseline for different prediction horizons. Second, we investigated the interplay between the prediction horizon and each feature setup on predictive performance.

4.1 Feature Setups and Predictive Performance

We examined whether feature matrices containing narrative features (obtained by the methodologies proposed in Sects. 3.2.1 and 3.2.2) achieve a better predictive accuracy compared to a simple baseline configuration that solely uses the daily CBOE Volatility Index' closing values as the predictive feature. To this end, we investigated the predictive performance for predicting CBOE Volatility Index movements for 1, 2, 4, 6, and 8 days ahead.

The Gradient Boosted Trees were trained on a sliding window of 504 trading days (2 years), where the out-of-sample test case was picked in function of the prediction horizon and according to the method outlined in Sect. 3.4. Because the optimal hyper-parameters for both our topic modeling approaches were found by utilizing news articles published between 01/01/2010 and 31/12/2017 (Sect. 3.2), we constrained our out-of-sample test set to the years 2018 and 2019 to avoid data leakage. Consequently, the trained Gradient Boosted Trees models were evaluated on 498 different out-of-sample movement predictions for the CBOE Volatility Index. Each proposed feature setup had a unique temporally ordered feature matrix of dimension $1002 \times C_i$, where C_i denotes the number of features for a particular setup i. We chose to quantify the performance of our predictions by measuring the predictive accuracy. Note that the target variable is fairly balanced with about 52% down movements and 48% up movements.

First, to examine the baseline configuration, predictions and evaluations were done using a temporally ordered feature matrix X_{vix} of dimension 1002×1 where each entry x_t represents the CBOE Volatility Index closing value for trading day t. Second, to study the performance of the feature matrix obtained by the latent Dirichlet analysis method outlined in Sect. 3.2.1, predictions and evaluations were done using a temporally ordered feature matrix X_{lda} of dimension $1002 \times (31 + 1)$. This feature matrix contains 31 topic features and an additional feature representing daily CBOE Volatility Index closing values. Lastly, to investigate the performance of the feature matrices obtained by using *doc2vec* and Gaussian mixture models outlined in Sect. 3.2.2, predictions and evaluations were done using six different temporally ordered features matrices X^i_{d2v} of dimension $1002 \times (K_i + 1)$ where K_i denotes the amount of topic features associated with one of the six proposed configurations. Note again that an additional feature representing daily CBOE Volatility Index closing values was added to the feature matrices.

Table 3 presents the best accuracy scores obtained by the Gradient Boosted Trees for different prediction horizons, following the methodology outlined in Sects. 3.3 and 3.4. First, Table 3 shows that for each prediction horizon except for the last one, there exists a feature setup that improves the predictive performance compared to the baseline. Second, for the scenario where movements are predicted 4 days into the future, all feature setups manage to outperform the baseline. In addition, all *doc2vec* feature setups manage to outperform the baseline and latent Dirichlet analysis feature setups for 6-day-ahead predictions. Third, the number of feature setups that outperform the baseline (bold numerals) increases as we predict further into the future. However, this trend does not hold when predicting 8 days into the future. Lastly, the *doc2vec* scenario, where PV-DM is used with a window size of two, seems to perform best overall except for the scenario where movements are predicted 2 days ahead.

Table 3 This table shows different feature setups and their best accuracy score obtained by Gradient Boosted Trees while predicting t-days ahead CBOE Volatility Index movements during 2018–2019 for $t \in \{1, 2, 4, 6, 8\}$. It demonstrates the contrast between simply using VIX closing values as a predictive feature (baseline) and feature matrices augmented with narrative features using respectively latent Dirichlet analysis (Sect. 3.2.1) and a combination of *doc2vec* and Gaussian mixture models (Sect. 3.2.2). Bold numerals indicate whether a particular setting outperforms the baseline, where underlined numerals indicate the best performing setting for the given prediction horizon

	$t = 1$	$t = 2$	$t = 4$	$t = 6$	$t = 8$
Baseline	54.0	51.5	53.4	54.4	<u>56.1</u>
LDA(31)	**55.7**	**52.4**	**54.7**	52.2	55.4
D2V(PV-DM, 2)	**<u>57.3</u>**	51.6	**<u>59.1</u>**	**<u>57.7</u>**	53.8
D2V(PV-DM, 5)	53.5	**53.7**	**57.8**	**57.3**	55.2
D2V(PV-DM, 8)	53.4	**53.8**	**57.5**	**57.0**	55.6
D2V(PV-DB, 2)	53.1	**<u>54.0</u>**	**55.0**	**55.5**	55.2
D2V(PV-DB, 5)	**55.0**	**52.3**	**57.3**	**56.2**	55.3
D2V(PV-DB, 8)	**54.2**	**52.5**	**57.0**	**55.6**	55.7

In conclusion, the narrative features contribute to an increased predictive performance compared to baseline. The *doc2vec* approach seems to yield the best performing models overall, consistently outperforming both the baseline and latent Dirichlet analysis feature setups for 4- and 6-day-ahead predictions. Lastly, the results suggest that the prediction horizon has an effect on predictive performance. The next section will investigate this further.

4.2 The Effect of Different Prediction Horizons

The results shown in Sect. 4.1 suggest that the prediction horizon influences the predictive performance for all different feature setups. In this part of the study, we investigated this phenomenon more in depth by examining to what degree feature setups outperform the baseline in function of different prediction horizons. The results are displayed in Fig. 6, where a bar chart is used to illustrate this interplay. Note that for both *doc2vec* scenarios using respectively PV-DM and PV-DBOW, the

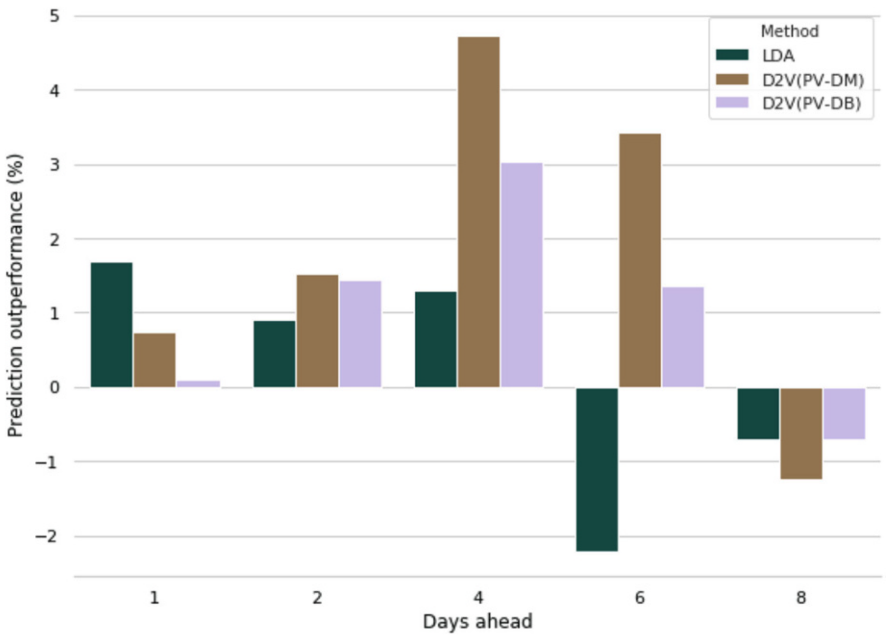

Fig. 6 This bar chart illustrates the effect of predictive performance when using different prediction horizons for different feature setups. The height of a bar denotes the outperformance of the given method compared to the baseline method of just using VIX closing values as the predictive feature. Note that for both D2V (PV-DM) and D2V (PV-DB), the accuracy scores were averaged across the different window size configurations prior to computing the prediction outperformance

accuracy scores were averaged across the different window size configurations prior to comparing the prediction performance compared to the baseline method.

First, Fig. 6 shows that for 1-day-ahead predictions, the narrative features obtained by using latent Dirichlet analysis perform better than *doc2vec* when performances are averaged across the different window sizes. However, note that the results from Sect. 4.1 show that the best performance for 1-day-ahead prediction is still achieved by an individual *doc2vec* feature setup. Nonetheless, this indicates that the performance of *doc2vec* feature setups is sensitive to the window size hyper-parameter. Second, a clear trend is noticeable looking at the outperformance achieved by both *doc2vec* PV-DM and PV-DBOW scenarios for different prediction horizons. Indeed, the performance for both scenarios increases by extending the prediction horizon. Moreover, the PV-DM method seems to consistently beat the PV-DBOW method. Third, the optimal prediction horizon for the *doc2vec* feature setups seems to be around 4 days, after which the performance starts to decline. Lastly, no feature setup is able to outperform the baseline model on a prediction horizon of 8 days.

In conclusion, we can state that the predictive performance of both latent Dirichlet analysis and *doc2vec* behaves differently. The best performance is achieved by *doc2vec* for a prediction horizon of 4 days, after which the performance starts to decline. This may suggest that the narrative features present in news only influence market participants for a short period of time, with market reaction peaking about 4 days into the future. Note that our study provides no evidence for causality.

5 Conclusion

Our study provides empirical evidence in favor of the theory of Narrative Economics by showing that quantified narratives extracted from news articles, described by sets of financial keywords, are predictive of future movements in the CBOE Volatility Index for different time horizons. We successfully demonstrate how both latent Dirichlet analysis and *doc2vec* combined with Gaussian mixture models can be used as effective topic modeling methods. However, overall we find that the *doc2vec* approach works better for this application. In addition, we show that the predictive power of extracted narrative features fluctuates in function of prediction horizon. Configurations using narrative features are able to outperform the baseline on 1-day, 2-day, 4-day, and 6-day-ahead predictions, but not on 8-day-ahead predictions. We believe this suggests that the narrative features present in news only influence market participants for a short period of time. Moreover, we show that the best predictive performance is achieved when predicting 4-day-ahead movements. This may suggest that market participants not always react instantaneously to narratives present in financial news, or that it takes time for this reaction to be reflected in the market.

References

1. Akita, R., Yoshihara, A., Matsubara, T., & Uehara, K. (2016, June). Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)* (pp. 1–6).
2. Aletras, N., & Stevenson, M. (2013). Evaluating topic coherence using distributional semantics. In *Evaluating Topic Coherence Using Distributional Semantics Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)* (pp. 13–22). Association for Computational Linguistics.
3. Atkins, A., Niranjan, M., & Gerding, E. (2018). Financial news predicts stock market volatility better than close price. *The Journal of Finance and Data Science, 4*(2), 120–137.
4. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res., 3*, 993–1022.
5. Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth and Brooks. ISBN 9780412048418
6. Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems, NIPS'09, Red Hook, NY* (pp. 288–296). Red Hook: Curran Associates Inc.
7. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16, 13–17-August-2016* (pp. 785–794).
8. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* (Vol. 1, pp. 4171–4186).
9. Feuerriegel, S., & Gordon, J. (2018). Long-term stock index forecasting based on text mining of regulatory disclosures. *Decision Support Systems, 112*, 88–97.
10. Feuerriegel, S., & Pröllochs, N. (2018). Investor reaction to financial disclosures across topics: An application of latent dirichlet allocation. *Decision Sciences*. Article in Press. https://doi.org/10.1111/deci.12346
11. Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics, 29*(5), 1189–1232.
12. Harris, Z. S. (1954). Distributional structure. *WORD, 10*(2–3), 146–162.
13. Hoffman, M., Bach, F. R., & Blei, D. M. (2010). Online learning for latent dirichlet allocation. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta (Eds.), *Advances in Neural Information Processing Systems 23* (pp. 856–864). Red Hook: Curran Associates, Inc.
14. Jin, X., & Han, J. (2011). Expectation-Maximization Algorithm. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of machine learning* (pp. 387–387). Boston: Springer. https://doi.org/10.1007/978-0-387-30164-8_291
15. Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *31st International Conference on Machine Learning, ICML 2014* (Vol. 4, pp. 2931–2939).
16. Mai, F., Tian, S., Lee, C., & Ma, L. (2019). Deep learning models for bankruptcy prediction using textual disclosures. *European Journal of Operational Research, 274*(2), 743–758.
17. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*. Available at: https://arxiv.org/abs/1301.3781
18. Řehůřek, R. (2021). *Gensim-topic modelling for humans*. Last accessed on 12 March, 2021. Available at: https://radimrehurek.com/gensim/
19. Reynolds, D. (2015). Gaussian mixture models. In S. Z. Li & A. K. Jain (Eds.), *Encyclopedia of Biometrics*. Boston: Springer. https://doi.org/10.1007/978-1-4899-7488-4_196

20. Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM '15, New York, NY* (pp. 399–408). New York: Association for Computing Machinery.
21. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning internal representations by error propagation* (pp. 318–362). Cambridge: MIT Press.
22. Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics, 6*(2), 461–464.
23. Shiller, R. J. (2019). *Narrative economics: How stories go viral and drive major economic events*. Princeton: Princeton University Press.
24. Thorndike, R. (1953). Who belongs in the family? *Psychometrika, 18*(4), 267–276.
25. Vargas, M., Lima, B., & Evsukoff, A. (2017). Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA 2017)* (pp. 60–65).
26. Wallach, H. M., Murray, I., Salakhutdinov, R., & Mimno, D. (2009). Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, New York, NY* (pp. 1105–1112). New York: Association for Computing Machinery.