Designing Explainable Text Classification Pipelines: Insights from IT Ticket Complexity Prediction Case Study

Aleksandra Revina, Krisztian Buza, and Vera G. Meister

Abstract Nowadays, enterprises need to handle a continually growing amount of text data generated internally by their employees and externally by current or potential customers. Accordingly, the attention of managers shifts to an efficient usage of this data to address related business challenges. However, it is usually hard to extract the meaning out of unstructured text data in an automatic way. There are multiple discussions and no general opinion in the research and practitioners' community on the design of text classification tasks, specifically the choice of text representation techniques and classification algorithms. One essential point in this discussion is about building solutions that are both accurate and understandable for humans. Being able to evaluate the classification decision is a critical success factor of a text classification task in an enterprise setting, be it legal documents, medical records, or IT tickets. Hence, our study aims to investigate the core design elements of a typical text classification pipeline and their contribution to the overall performance of the system. In particular, we consider text representation techniques and classification algorithms, in the context of their explainability, providing ultimate insights from our IT ticket complexity prediction case study. We compare the performance of a highly explainable text representation technique based on the case study tailored linguistic features with a common TF-IDF approach. We apply interpretable machine learning algorithms such as kNN, its enhanced versions, decision trees, naïve Bayes, logistic regression, as well as semi-supervised techniques to predict the ticket class

A. Revina (✉)
Chair of Information and Communication Management, Faculty of Economics and Management, Technical University of Berlin, 10623 Berlin, Germany
e-mail: revina@tu-berlin.de

K. Buza
Faculty of Informatics, Eötvös Loránd University, 1117 Budapest, Hungary
e-mail: buza@biointelligence.hu

A. Revina · V. G. Meister
Faculty of Economics, Brandenburg University of Applied Sciences, 14770 Brandenburg an der Havel, Germany
e-mail: vera.meister@th-brandenburg.de

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2021

W. Pedrycz and S. Chen (eds.), *Interpretable Artificial Intelligence: A Perspective of Granular Computing*, Studies in Computational Intelligence 937,
https://doi.org/10.1007/978-3-030-64949-4_10

label of low, medium, or high complexity. As our study shows, simple, explainable algorithms, such as decision trees and naïve Bayes, demonstrate remarkable performance results when applied with our linguistic features-based text representation. Furthermore, we note that text classification is inherently related to Granular Computing.

Keywords Text classification · Explainability · Linguistics · Machine learning · TF-IDF · IT tickets

1 Introduction

One of the most critical issues facing enterprises today is how to structure, manage, and convert large amounts of unstructured text data coming from inside or outside of the company into valuable information. Getting knowledge from unstructured text, also referred to as text analytics, has always been restricted by the ability of machines to grasp the real semantics of human language [1]. However, especially with the current technological advancements, many organizations are putting considerable efforts into developing successful text analytics strategies. From the industry branch perspective [2], these are pharmaceutical, medical and insurance companies mainly known for using text analytics to improve product development processes [3], make medical diagnoses [4], analyze claims, or predict frauds [5] correspondingly. From the organizational perspective [2], these are marketing and customer service departments that typically use text analytics techniques to extract actionable insights from customer-related data such as customer call notes, service requests, or social media. The goal is to understand customer problems better, react to service issues on time, or plan process improvement strategies [6].

In the text analytics tasks, such as classification, machines process the input from humans and produce the output for humans. In this regard, one key question remains the quality of the output delivered by machines. This information should build trust to ensure that humans accept the decisions made or suggested by the machine. Hence, understanding the way *how* the information was obtained plays a crucial role in creating confidence in all human–machine interaction tasks [7–11]. Lately, such a term as Explainable Artificial Intelligence (XAI) has been established to refer to those systems which goal is to provide transparency on how an AI system makes its decisions and predictions and performs actions [12]. The research on XAI has attracted a lot of attention in recent years. A number of concepts and definitions were elaborated. For a comprehensive overview, we refer to [11]. As the title suggests, in this chapter, we aim at building explainable text classification pipelines. We show that such pipelines can be advantageous not only in the sense of their explainability but also in prediction quality.

In our previous research [13–15], we performed an in-depth linguistic analysis of the IT ticket text descriptions originating from the IT ticket processing department of a big enterprise with more than 200,000 employees worldwide. Using the

elaborated linguistic representations of the ticket texts, we implemented a rule-based approach to predict the process complexity [16]. The precision of this approach reached approximately 62%. Apart from such a precision quality, the process of rule set establishment demanded a lot of analysis and testing effort on our and process experts' side. Another substantial disadvantage of such an approach, particularly in complex scenarios, is difficulty in maintenance, inability to learn and scale, as adding new rules requires revising the existing ones. With the motivation to address this challenge and especially develop an understanding of which factors—text representation techniques or text classification algorithms—play an essential role in the prediction quality, we suggest a new analysis in this study. Hereby, we put special emphasis on explainability and consider it using the Granular Computing paradigm [17].

Specifically, in the representation of the ticket text data, we compare the performance of a commonly accepted and widely used standard TF-IDF (Term Frequency-Inverse Document Frequency) [18] with the explainable linguistic approach described in our previous work [13, 14, 15]. Despite massive research efforts on text classification, the representation based on the linguistic features considered in this study has not been systematically compared with TF-IDF representation in the ticket classification context.

In the classification methods, we focus on various classifiers known for their explainability and widely used for text [19 and ticket [20] classification, including kNN, its enhanced versions, so-called hubness-aware classifiers [21, 22], decision trees [23], naïve Bayes [24], logistic regression [25], as well as semi-supervised techniques. The latter includes kNN with self-training [26], Semi-sUpervised ClassifiCation of TimE SerieS algorithm (SUCCESS) [27], and its improved version QuickSUCCESS [28]. Although state-of-the-art technology allows us to capture a considerable amount of instances in many applications (e.g., millions of images may be observed), in our case, it is costly and difficult to obtain class labels for a large number of tickets. On the one hand, to label the tickets, expert knowledge is required. On the other hand, as various aspects have to be taken into account, even experts need much time for this task. Therefore, due to the limited amount of labeled tickets, we also experimented with semi-supervised approaches for ticket classification.

Hence, our work makes some key methodological contributions. Our extensive analysis of linguistic features, TF-IDF, and various machine learning (ML) algorithms confirms the positive influence of linguistic style predictors on the prediction quality in general. Furthermore, simple, explainable algorithms in combination with linguistic features-based text representation demonstrate excellent performance results. The managerial and practical contributions of the research are related to the following points: (i) decision support for managers and ML experts in the design of text classification pipelines for diverse enterprise applications, (ii) addressing the limitations of our rule-based approach with ML.

In the following sections, we give an overview of the acknowledged related work in the field and present in detail the research methodology followed by the experiments on the case study datasets. Afterward, we discuss the implications of the findings and conclude with a summary and future research directions.

2 Related Work

In general, natural language in business processes has been largely studied. Different problems, such as extracting performance indicators [29] or checking process compliance [30], have been addressed. In the present chapter, we focus on the problem of designing explainable real-world text classification tasks related to an IT ticket processing case study. We structure the related work section as follows: (i) explainability and granularity, (ii) text representation, (iii) text classification, (iv) ticket classification research.

2.1 Explainability and Granularity

The interest towards explanation in AI has emerged in the press due to the prominent failures. For example, in 2018, an Uber's self-driving car caused the death of a pedestrian in the US [31], or in the same year, IBM Watson recommended unsafe and incorrect cancer treatments [32]. Hence, the ability to understand how such AI systems work is crucial for reliance and trust. According to [33], machines are advantageous only to that degree that their actions can be relied upon to achieve the objectives set by humans. XAI is supposed to clarify the rationale behind a decision-making process of an AI system, highlight the strengths and weaknesses of the process, and give an understanding of how the system will act in the future [12].

On the one hand, the term XAI can be considered straightforward and self-explainable. On the other hand, the confusion is brought by such terms as interpretability, transparency, fairness, explicitness, and faithfulness [11]. As a rule, they are used synonymously. However, some researchers imply a different meaning [7, 8, 34, 35], such as quasi-mathematical [35]. In this study, we refer to a commonly accepted definition from the Oxford English dictionary. The dictionary does not provide definitions for terms "explainable" or "explainability", but for "explanation": *a statement, fact or situation that tells why something happened*; *a statement or piece of writing that clarifies how something works or makes something easier to understand* [36].

To make this definition more feasible, we use the Granular Computing paradigm. In Granular Computing, one operates on the level of so-called "information granules" [17]. The granules of similar size are usually grouped in one layer. The more detailed and computationally intensive processing is required, the smaller information granules are created. Afterward, these granules are arranged in another layer, which results in the information processing pyramid [17].

Granular Computing has become popular in different domains. One can point out multiple research projects using Granular Computing concepts on various data types such as time series [37–39] and image data [40] as well as different application areas such as medicine [41, 42], finance [43], manufacturing [44].

2.2 Text Representation

Text representation is one of the essential building blocks of approaches for text mining and information retrieval. It aims to numerically represent the unstructured text documents to make them mathematically computable while transforming into a feature vector [45, 46]. We studied different techniques of text representation and feature extraction and structured them into three main categories: weighted word techniques, word embedding [18], and linguistic features.

2.2.1 Weighted Words

Weighted words approaches are based on counting the words in the document and computing the document similarity directly from the word-count space [19]. Due to their simplicity, Bag-of-Words (BoW) [47] and TF-IDF [48] can be considered as the most common weighted words approaches. The employed techniques usually rely on simple BoW text representations based on vector spaces rather than in-depth linguistic analysis or parsing [49]. Nonetheless, while these models represent every word in the corpus as a one-hot-encoded vector, they are incapable of capturing the word semantic similarity and can become very large and technically challenging [19]. To address these limitations, we use linguistic features in the proposed approach. As will be described in Sect. 3, the number of our linguistic features is independent of the size of the corpus, and they are capable of capturing relevant aspects of semantics.

2.2.2 Word Embedding

With the progress of research, new methods, such as word embedding, have come up to deal with limitations of weighted words approaches. Word embedding techniques learn from sequences of words by considering their occurrence and co-occurrence information. Each word or phrase from the corpus is mapped to a high dimensional vector of real numbers and trained based on the surrounding words over a huge corpus. Various methods have been proposed, Word2Vec [50], Doc2Vec [51], GloVe [52], FastText [53, 54], contextualized word representations [55, 56] being the most significant ones. These methods do consider the semantic similarity of the words. However, they need a large corpus of text datasets for training. To solve this issue, pre-trained word embedding models have been developed [57]. Unfortunately, the models do not work for the words outside of the corpus text data. Another limitation of the word embedding models is related to the fact that they are trained based on the words appearing in a selected window size parameter. For example, window size three means three words behind and three words ahead, making up six in total. This is an inherent limitation for considering different meanings of the same word occurring in different contexts. While addressing this limitation, contextualized word representations techniques based on the context of the word in a document [55, 56]

were developed. Nonetheless, in real-world applications, new words may appear (in the description of a new ticket, the customer may use phrases and specific words that have not been used before). These new words are not included in the corpus at training time. Therefore, these word embedding techniques will fail to produce a correct mapping for these new words. While this problem may be alleviated by retraining the model, this requires a lot of data and computational resources (CPU, RAM). In contrast, as it is discussed next, it is straightforward to use our approach in case of new words as it doesn't need training.

2.2.3 Linguistic Features

To address the aforementioned limitations of weighted words and words embeddings, text representations based on linguistic features have been introduced. Below, we list some examples.

Lexicon-based sentiment analysis is a well-established text classification technique [58]. Synonymy and hypernymy are known approaches to increase prediction quality [45]. Extensive research on the linguistic analysis of ticket texts has been performed in [59, 60]. The authors use parts-of-speech (PoS) count and specific terms extractions to define the severity of the reported problem. Coussement and Van den Poel extract the following linguistic features: word count, question marks, unique words, the ratio of words longer than six letters, pronouns, negations, assents, articles, prepositions, numbers, time indication [61]. The results of the study indicated a profoundly beneficial impact of combining traditional features, like TF-IDF and singular value decomposition, with linguistic style into one text classification model. However, the authors declare a demand for more experiments and research. This demand is addressed in our work.

There is no unified opinion in the research community whether the linguistic approaches are good enough to substitute or enhance the traditional text representations as they are more complex to implement and do not compensate this complexity with the expected performance increase. Both proponents [62–67] and opponents [68] of the linguistic approach provide convincing experimental results. In our work, we amend these research discussions with case study-based findings while comparing the performance of linguistic features with TF-IDF. Word embedding techniques are not applicable in our study due to the following reasons: (i) pre-trained models would not perform well considering the domain-specific vocabulary which contains many new words compared to the training corpus, (ii) at the same time, a limited text corpus would not be enough for training a new model (or retraining an existing one), (iii) industrial applications prevailingly demand explainable models to be able to understand and correct classification mistakes.

Table 1 summarizes the strengths and weaknesses of the discussed techniques.

Table 1 Text representation techniques

Technique	Strengths	Weaknesses
Weighted words		
BoW, TF-IDF	• Simple to implement • Established approaches to extract the most descriptive terms in a document • Do not need data to train the mapping	• Do not consider syntax and semantics
Word embedding		
Word2Vec, Doc2Vec, GloVe, contextualized word representations	• Consider syntax and semantics • Contextualized word representations: consider polysemy	• Need much data for training • Consider only words that appear in the training data • Computationally expensive to train (CPU, RAM)
Linguistic features		
Word count, special characters, parts of speech, unique words, long words, context-specific taxonomies, lexicons, sentiment, etc	• Highly explainable and understandable • large choice of features • Do not necessarily depend on capturing semantics and context • Depending on the selected features, can capture both syntax and semantics • Do not need data to train the mapping	• Expert knowledge is required to define an appropriate set of features

2.3 Text Classification

Text classification, also referred to as text categorization or text tagging, is the task of assigning a text to a set of predefined categories [69]. Traditionally, this task has been done by human experts. Expert text classification, for example, remains widely used in qualitative research in the form of such tasks as coding or indexing [70, 71]. Nevertheless, with the growing amount of text data, the attention of the research community and practitioners shifted to automatic methods. One can differentiate three main groups of automatic text classification approaches: rule-based, ML-based, and a combination of both in a hybrid system. Below, the main two approaches will be shortly discussed.

2.3.1 Rule-Based Text Classification

As the name suggests, this kind of text classification system is based on a set of rules determining classification into a set of predefined categories. In general, rule-based classifiers are a popular data mining method applicable to diverse data types. It

became popular due to its transparency and relative simplicity [72]. The researchers distinguish various types of rule development: ML-based such as decision trees [23, 73], association rules [74, 75], handcrafted rules created by the experts, or hybrid approaches [76, 77]. It is essential to mention that rule-based systems work well with small rule sets and become challenging to build, manage, and change with the growing number of rules.

In our previous work (shortly mentioned in the introduction part), we conceptualized a recommender system for IT ticket complexity prediction using rule-based classification, i.e., handcrafted rules and rules based on decision trees [16]. Nonetheless, due to the complexity of the domain, the process of rule development consumed much time and effort. The system was difficult to manage and change.

2.3.2 Machine Learning-Based Text Classification

In the context of ML, text classification is a task which can be defined as follows: given a set of classification labels C and a set of training examples E, each of which has been assigned to one of the class labels in C, the system must use E to form a hypothesis to predict the class labels of previously unseen examples of the same type [78]. Hereby, in text classification, E is a set of labeled documents from a corpus. The labels can be extracted topics, writing styles, judgments of the documents' relevance [45]. Regarding the prediction itself, various techniques such as kNN, its enhanced versions (hubness-aware classifiers), decision trees, naïve Bayes, logistic regression, support vector machine, neural networks have been introduced [19]. ML approaches have been shown to be more accurate and easier to maintain compared to rule-based systems [79]. At the same time, it is challenging to select the best ML technique for a particular application [19]. Table 2 summarizes the strengths and weaknesses of the main approaches for text classification.

Most ML techniques, including all the aforementioned approaches, require a large amount of training data. However, as described previously, in our application, it is difficult to obtain labeled training data. In contrast, semi-supervised learning (SSL) allows inducing a model from a large amount of unlabeled data combined with a small set of labeled data. For an overview of major SSL methods, their advantages and disadvantages, we refer to [81]. For the above reasons, we also include semi-supervised ML classifiers in our study, in particular, kNN with self-training and SUCCESS [27]. Although SUCCESS showed promising results, it doesn't scale well to large datasets. In this chapter, we address this limitation by developing a scaling technique for SUCCESS, and we called the resulting approach QuickSUCCESS [28].

2.4 Ticket Classification Research

Ticket classification in the context of software development and maintenance has been studied to tackle such challenges as correct ticket assignment and prioritization,

Table 2 Text classification techniques

Technique	Strengths	Weaknesses
Rule-based classification		
ML-based rule development with decision trees, association rules, handcrafted rules	• Able to handle a variety of input data • Explainable	• With the growing number of rules—difficult to build, manage, maintain, and change
ML-based classification		
kNN, hubness-aware classifiers	• Non-parametric • Adapts easily to various feature spaces	• Finding an appropriate distance function for text data is challenging • prediction might become computationally expensive
SUCCESS	• Learns with few labeled instances	• Finding an appropriate distance function for text data is challenging • Computationally expensive training
Decision trees	• Fast in learning and prediction • Explainable	• Overfitting • Instability even to small variations in the data
Naïve Bayes	• Showed promising results on text data [80]	• Strong assumption about the conditional independence of features
Logistic regression	• Computationally inexpensive	• Does not model the interdependence between features • is not appropriate for non-linear problems
Support vector machines	• Robust against overfitting • Able to solve non-linear problems	• Lack of transparency in results • Choosing an appropriate kernel function may be challenging
Neural networks	• Can achieve rather accurate predictions	• Requires a large amount of data for training • May be extremely computationally expensive • Finding an efficient architecture and structure is difficult • Not explainable

prediction of time and number of potential tickets, avoiding duplicates, and poorly described tickets [20, 82].

One can observe various ticket text classification [83, 84] and clustering approaches [85, 86]. The most common classification algorithms are traditional support vector machines, naïve Bayes, decision trees, logistic regression, as well as algebraic and probability theory-based algorithms. The most popular text representation methods are weighted words techniques, such as TF-IDF [20, 87, 88]. The same as in the general text classification studies [61, 68], one can find strong proponents of the linguistic features also in the IT ticket domain [59, 60].

[59, 60] performed an extensive research on the linguistic analysis of ticket text descriptions. The authors used parts-of-speech (PoS) count and specific terms extractions to define the severity of the reported problem. Their findings evidence high classification accuracy in the range of 85–91% and the characteristic PoS distributions for specific categories.

2.5 Summary

This section provided an overview of acknowledged work featuring those subject areas necessary to understand the selected research set-up and methods as well as its contributions. While many text classification techniques exist, their implementation effort and performance are data and case study dependent, which makes the choice of the best fit solution even more difficult. This holds true for both general text classification tasks, like email classification [61], and, particularly, IT tickets [20]. At the same time, in the text representation subject area, there is no consensus among scientists if the linguistic representation is more beneficial than a traditional one, for example, the most widespread TF-IDF or BoW [68]. Whereas the researchers report a significant performance improvement of linguistic approach [62–67], the need for further experiments on the case study projects to be able to learn about the origin of the data [60], exploration of more linguistic features [45] or the necessity to experiment with other text classification tasks and algorithms [61] are declared. Hence, the present study aims at throwing more light on the discussion of selected elements in the design of a text classification pipeline putting emphasis on the importance of their explainability. Accordingly, the main purpose of the study is to improve our understanding of text classification applications in the context of the discussed text representation techniques and classification algorithms.

3 Methods

3.1 Feature Extraction

Overall, texts belong to an unstructured type of data. However, if one would like to use any kind of mathematical modeling and computer-enabled analysis, the unstructured text must be transformed into a structured feature space. The data needs to be pre-processed to remove unnecessary characters and words. Afterward, diverse feature extraction techniques can be applied to get a machine-understandable text representation. In our study, we compare two feature extraction techniques: common TF-IDF and the linguistic features-based approach elaborated in our previous work.

Term frequency (TF) weights measuring the frequency of occurrence of the terms in the document or text have been applied for many years for automatic indexing in the IR community [89]. TF alone produces low-quality search precision when high-frequency terms are equally spread over the whole text corpus. Hence, a new factor, inverse document frequency (IDF), was introduced to reflect the number of documents n to which a term is assigned in a collection of N documents and can be computed as log of N/n [48]. Taking into account both factors, a reasonable measure of text representation can be received using the product of TF and IDF [90]. TF-IDF is considered to be one of the most widely used text representation techniques [20] due to its simplicity and good quality of search precision, which can be achieved. Following the popularity of this technique, especially in the IT ticket domain, as well as for the reasons listed in Sect. 2, we perform the feature extraction procedure correspondingly and represent each ticket text in our case study corpus as a TF-IDF numerical vector.

To compare TF-IDF with a more transparent and human-understandable technique and as encouraged by the promising results of other researchers [61, 59, 60] and the growing importance of XAI research [91, 92], especially in the high risk and high consequence decision-making domains [93], we implement an explainable linguistic approach, based on our own case study specific set of linguistic features (see Table 3). These were designed for the classification task of IT ticket complexity prediction.

It is commonly distinguished among three levels of text understanding: (1) objective (answering the questions who, what, where, when) measured by semantic technologies, (2) subjective (an emotional component of a text)—by sentiment analysis, and (3) meta-knowledge (information about the author outside of the text) measured by stylometry or stylistic analysis [94]. Accordingly, we develop a set of features which are aggregated by the respective measures indicating the IT ticket complexity. We proposed these features in our initial works [13, 14, 15].[1] A detailed explanation of the linguistic features is provided below in text using an anonymized IT ticket example and summarized in Table 3.

[1] https://github.com/IT-Tickets-Text-Analytics.

Table 3 Overview of linguistic features illustrated by a ticket example

Aspects	Description	Linguistic feature
Ticket example: "Refresh service registry on the XYZ-ZZ YYY server. See attachment for details."		
Objective knowledge aspect [15]	Relative occurrence of words according to the taxonomy of Routine, semi-cognitive and cognitive terms	Routine = 0.8
		Semi-cognitive = 0.2
		Cognitive = 0
Subjective knowledge aspect [14]	Relative occurrence of words with positive, neutral and negative sentiment	Negative = 0
		Neutral = 1
		Positive = 0
Meta-knowledge aspect [13]	Word count	12
	Occurrence of nouns in all words	0.5
	Occurrence of unique nouns in all nouns	1
	Occurrence of verbs in all words	0.17
	Occurrence of unique verbs in all verbs	1
	Occurrence of adjectives in all words	0.07
	Occurrence of unique adjectives in all adjectives	1
	Occurrence of adverbs in all words	0
	Occurrence of unique adverbs in all adverbs	0
	Wording style	0 (no repeating words)

3.1.1 Objective Knowledge Aspect

Core research in Natural Language Processing (NLP) addresses the extraction of objective knowledge from text, i.e., which concepts, attributes, and relations between concepts can be extracted from text, including specific relations such as causal, spatial, and temporal ones [94]. Among diverse approaches, specifically, taxonomies and ontologies, are widely used in the business context [95, 96]. Thus, we suggest a specific approach of objective knowledge extraction using the Decision-Making Logic (DML) taxonomy [15] illustratively presented in Appendix I. Herewith, it is aimed to discover the decision-making nature of activities, called DML level. We use the following DML levels: *routine, semi-cognitive,* and *cognitive* (corresponding to the columns of the table in Appendix I). Using a Latent Dirichlet Allocation Algorithm (LDA) [97], we identify the most important keywords, see [15] for details. Each of the keywords is associated with a DML level. For example, the keywords

user, *test, request,* etc. are associated with *routine*, whereas the keyword *management* belongs to *cognitive*. We detect these keywords in IT ticket texts. Based on the total number of detected keywords, we calculate the relative occurrence of the words of each category in the ticket text. In the example shown in Table 3, the total number of detected words equals five, out of which four words belong to *routine* (*server, attach, detail, see*), one to *semi-cognitive* (*service*), and no word to *cognitive*. Thus, the corresponding features are calculated as follows: *routine:* $4/5 = 0.8$, *semi-cognitive:* $1/5 = 0.2$, and *cognitive:* $0/5 = 0$.

3.1.2 Subjective Knowledge Aspect

To extract the subjective knowledge aspect, we perform sentiment analysis [98]. In [14], we suggest a specific approach of business sentiment as an instrument for measuring the emotional component of an IT ticket. This latent information is extracted from the unstructured IT ticket texts with the help of a lexicon-based approach. As standard lexicons do not work well in our context of IT ticket classification, using the state-of-the-art VADER [99] and LDA, we developed a domain-specific lexicon, see [14] for details. Our lexicon can also be found in Appendix II.

Each of the words in the lexicon is associated with a positive, negative, or neutral sentiment. In particular, words with valence scores greater than 0 are considered to be positive, whereas words with valence score less than 0 are considered to be negative. All other words with 0 valence score are considered to have a neutral sentiment. For each IT ticket text, we determine the proportion of words with negative, neutral, and positive sentiment and use these values as features. In our example, there are no words with positive or negative sentiment. Therefore, the ticket is assigned to be entirely neutral.

3.1.3 Meta-Knowledge Aspect

In our case, meta-knowledge is the knowledge about the author of an IT ticket. The quality of the written text will likely depend on such factors as the author's professionalism and expertise, level of stress, and various psychological and sociological properties [94]. To extract the meta knowledge aspect, we suggest the following features [13]: (1) IT ticket text length, (2) PoS features, (3) wording style [13] calculated with the Zipf's law of word frequencies [100].

By length, we mean the number of words in the IT ticket text. This feature is motivated by the following observation: in most cases, IT tickets texts containing a few words, such as *update firewalls,* refer to simple daily tasks. Therefore, short ticket texts may be an indication of the low complexity of the ticket. In the example shown in Table 3, the length of the ticket is 12 words.

As for PoS features, we consider the following PoS tags: nouns, verbs, adjectives, and adverbs. For each of them, we calculate their absolute occurrence, i.e., the *total*

number of words having that PoS tag (for example, the total number of nouns). Subsequently, we calculate their relative occurrence, called *occurrence* for simplicity, i.e., the ratio of nouns, verbs, adjectives, and adverbs relative to the length of the text. We use these occurrences as features. In the example shown in Table 3, the occurrence of nouns (*registry, XYZ-ZZ, YYY, server, attachment, details*) in all words is $6/12 = 0.5$.

We also calculate the number of *unique* words having the considered PoS tags (for example, number of unique nouns). Then, we calculate the occurrence of *unique* nouns, verbs, adjectives, and adverbs relative to the number of *all* nouns, verbs, adjectives, and adverbs, respectively. We use these occurrences as features as well. In Table 3, the uniqueness of nouns is $6/6 = 1$ (no repeating words).

According to Zipf's word frequency law, the distribution of word occurrences is not uniform, i.e., some words occur very frequently. In contrast, others appear with a low frequency, such as only once. Our wording style feature describes how extreme is this phenomenon in the IT ticket text, i.e., whether the distribution of occurrences is close to a uniform or not. For details, we refer to [13].

3.2 Machine Learning Classifiers

After the features have been extracted and the texts are represented in the form of numerical vectors, they can be fed into ML classifiers. Choosing the best classifier is fairly acknowledged to be the essential step of any text classification pipeline.

In our study design, we put a high value on the explainability of the classification results. As fairly stated by [101], ML has demonstrated its high potential in improving products, services, and businesses as a whole. However, machines do not explain their predictions, which is a barrier to ML adoption. Hence, from the existing classifiers, we focus on simple algorithms known for their explainability, such as kNN, decision trees, naïve Bayes, and logistic regression [101]. In the attempt to improve the classification quality, we experiment with enhanced versions of kNN (hubness-aware classifiers) as well as its semi-supervised variations (kNN with self-training, SUCCESS, and QuickSUCCESS).

3.2.1 Standard kNN

Standard kNN is a non-parametric algorithm widely used both for classification in general and for text classification in particular [102]. It is simple to implement, works well with multi-class labels, and adapts easily to various feature spaces [19], which also determines the choice of the algorithm in this study.

3.2.2 Hubness-Aware Classifiers

One problem of kNN is related to the presence of so-called bad hubs, instances that are similar to a high amount of other instances, and this way may mislead the classification [103]. A hub is considered to be bad if its class label differs from the class labels of many of those instances that have this hub as their nearest neighbor. Thus, bad hubs have proven to be responsible for a surprisingly high fraction of the total classification error. To address this challenge, several hubness-aware classifiers have been proposed recently [21, 104]: kNN with error correction (ECkNN) [103, 105], hubness-weighted kNN (HWkNN) [106], hubness-fuzzy kNN (HFNN) [107], naive hubness-Bayesian kNN (NHBNN) [108]. We test these approaches to compare their performance with the standard kNN. In text data, document length can be correlated with hubness, so that shorter or longer documents might tend towards getting hubs [109]. The length of the tickets in our case study could vary from 5–10 words to 120–150. This is another reason for the choice of hubness-aware classifiers.

Moreover, the datasets contain prevailingly English and a small portion of German language texts. According to [110], document relevance is preserved across languages and it is possible to approximate document hubness if given access to data translations. Additionally, based on the results of previous research [111], we believe that hubness-aware classifiers could successfully address the problem of the imbalanced dataset which is also our case. The mentioned class imbalance in our dataset can be explained by the following reasons: (i) nature of the case study and dataset itself, i.e., many tickets of low complexity and considerably less of high; (ii) the tickets for a labeling task were selected randomly.

3.2.3 Semi-supervised Approaches—kNN with Self-Training and SUCCESS

One more challenge we encounter is a very small quantity of labeled data. This may be addressed by semi-supervised ML (SSML). In the study, we implement two SSML approaches, which are discussed below.

First, we use a simple and effective technique of self-training [112] that has been successfully applied in many real-world scenarios. A classifier, in our case kNN, is trained with a small number of labeled data and iteratively retrained with its own most confident predictions [26].

Second, we implement SUCCESS [27], SSML approach that shows promising results on time series datasets. We believe that its potential has not been exploited in the context of text classification. Therefore, we adapt it to ticket classification and evaluate its performance. Below, we will shortly review the SUCCESS approach.

We define the semi-supervised classification problem as follows: given a set of labeled instances $L = \{(x_i, y_i)\}_{i=1}^{l}$ and a set of unlabeled instances $U = \{x_i\}_{i=l+1}^{n}$, the task is to train a classifier using both L and U. We use the phrase *set of labeled training instances* to refer to L, x_i is the i-th instance, y_i is its label, whereas we say that U is the *set of unlabeled training instances*. The labeled instances (elements of

L) are called seeds. We wish to construct a classifier that can accurately classify any instance, i.e., not only elements of U. For this problem, we proposed the SUCCESS approach that has the following phases:

1. The labeled and unlabeled training instances (i.e., instances of U and L) are clustered with constrained single-linkage hierarchical agglomerative clustering. While doing so, we include cannot-link constraints for each pair of labeled seeds, even if both seeds have the same class labels. In our previous work [27], we measured the distance of two instances (time series) as their Dynamic-Time-Warping distance.
2. The resulting top-level clusters are labeled by their corresponding seeds.
3. The final classifier is 1-nearest neighbor trained on the labeled data resulting at the end of the 2nd phase. This classifier can be applied to unseen test data.

3.2.4 Resampling Approaches—kNN with Resampling and QuickSUCCESS

As we noticed rather slow training speed of semi-supervised classifiers, we implemented resampling [113], a technique similar to bagging, to accelerate the classification and possibly improve classification results [114]. In particular, we select a random subset of the data and train the model on the selected instances. This process is repeated several times. When making predictions for new instances, the predictions of all the aforementioned models are aggregated by majority voting. As the sample size is much smaller than the size of the original dataset and the training has a super-linear complexity, the training of several models on small samples is computationally cheaper than the training of the model on the entire dataset. While kNN with resampling has been established in the research community since a long time ago [115], to the best of our knowledge, ours was the first attempt to speed up the SUCCESS algorithm using resampling. We call the resulting approach QuickSUCCESS [28]. Next, we explain QuickSUCCESS in detail.

The core component of SUCCESS is constrained hierarchical agglomerative clustering. Although various implementations are possible, we may assume that it is necessary to calculate the distance between each pair of unlabeled instances as well as between each unlabeled instance and each labeled instance. This results in a theoretical complexity of $O(l(n-l) + (n-l)^2) = O(n^2 - l^2 - nl)$ at least, where l denotes the number of labeled instances ($l = |L|$), while n denotes the number of all instances (labeled and unlabeled) that are available at training time ($n = |L| + |U|$). Under the assumption that l is a small constant, the complexity of distance computations is $O(n^2)$.

Considering only the aforementioned distance computations required for SUCCESS, the computational costs in case of a dataset containing n/c instances is c^2-times lower than in case of a dataset containing n instances. Therefore, even if the computations have to be performed on several "small" datasets, the overall computational costs may be an order of magnitude lower. In particular, computing SUCCESS m-times on a dataset containing r instances has an overall computational

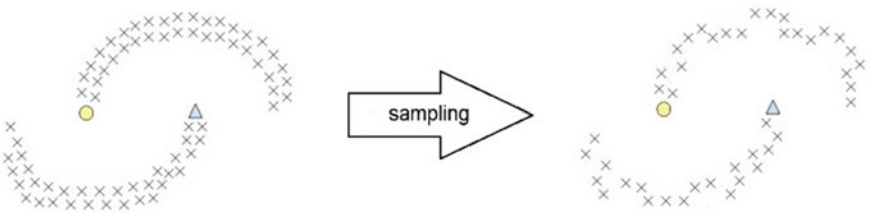

Fig. 1 When sampling the data, the size r of the sample should be chosen carefully so that the sampled data is representative

cost of $O(m \times r^2)$. Under the assumption that $r = n/c$ and $m \approx O(c)$, the resulting complexity is $O(n^2/c)$. Based on the analysis, we propose to speed up SUCCESS by repeated sampling. In particular, we sample the set of unlabeled instances m-times. From now on, we will use $U^{(j)}$ to denote the j-th sample, $1 \leq j \leq m$.

Each $U^{(j)}$ is a random subset of U containing r instances ($|U^{(j)}| = r$). For simplicity, each instance has the same probability of being selected. To train the j-th classifier, we use all the labeled instances in L together with $U^{(j)}$. When sampling the data, the size r of the sample should be chosen carefully so that the sampled data is representative in the sense that the structure of the classes can be learned from the sampled data. This is illustrated in Fig. 1.

Algorithm 1: Training QuickSUCCESS

Require: labeled training data L, unlabeled training data U, sample size r, number of classifiers m
Ensure: trained classifiers $\{C^{(i)}\}_{i=1}^{m}$, predicted labels for U
1: **for** i in 1 … m **do**
2: $U^{(i)} \leftarrow$ random_sample (U, r)
3: $C^{(i)}, \widehat{y^{(i)}} \leftarrow$ train SUCCESS $(L, U^{(i)})$
4: **for** x in $U^{(i)}$ **do**
5: vote for the label $\widehat{y^{(i)}}[x]$ for unlabeled instance x
6: **end for**
7: **end for**
8: return $\{C^{(i)}\}_{i=1}^{m}$, class labels predicted based on the majority vote

As the sampling is repeated m-times, we induce m classifiers denoted as $C^{(1)},\ldots,C^{(m)}$. Each classifier $C^{(j)}$ predicts the label for the unlabeled instances in the corresponding sample of the data, i.e., for the instances of $U^{(j)}$. More importantly, each of these classifiers can be used to predict the label of new instances, i.e., instances that are neither in L nor in U. Labels for the instances in U are predicted as follows. For each $x_i \in U$, we consider all those sets $U^{(j)}$ for which $x_i \in U^j$ and the classifiers that were trained using these datasets. The majority vote of these classifiers is the predicted label of x_i. Our approach is summarized in Algorithm 1.

The label of a new instance $x \notin L \cup U$ is predicted as the majority vote of all the classifiers $C^{(1)},\ldots,C^{(m)}$. We note that the computations related to the aforementioned classifiers $C^{(1)},\ldots,C^{(m)}$ can be executed in parallel, which may result in additional speed-up in case of systems where multiple CPUs are available, such as high-performance computing systems.

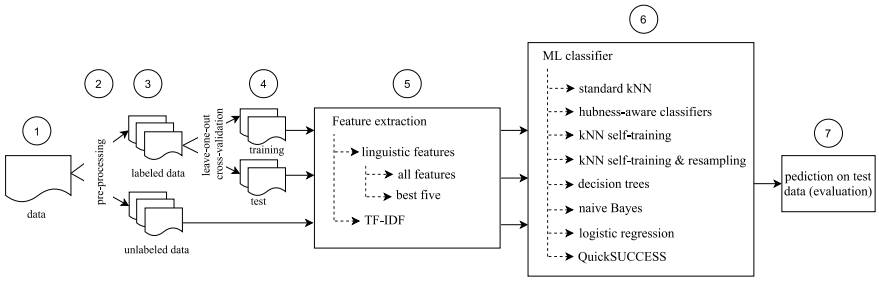

Fig. 2 Overview of text classification pipeline

3.2.5 Decision Trees, Naïve Bayes, Logistic Regression

We also experimented with decision trees, naïve Bayes, and logistic regression, a group of algorithms, commonly used both in general text mining [19] and ticket classification [20] context and recognized as explainable approaches. Decision trees perform effectively with qualitative features, which is the case of the ticket classification case study. They are known for their interpretability, also an important aspect for the case study, and fast in learning and prediction [19]. Naïve Bayes and logistic regression are chosen as traditional classifiers working well with text data. These classifiers can be easily and quickly implemented [18]. Thus, it may be relatively simple to integrate them into industrial systems.

4 Experimental Evaluation

Our experimental design is summarized in Fig. 2 and includes the following steps: (1) collect the case study data; (2) pre-process the data; (3) label part of the data; (4) split the data into training and test sets; (5) extract two sets of features: linguistic features and TF-IDF; (6) apply the classifier; (7) evaluate the results with common metrics. All the experiments were conducted using Python 3.6 and PyHubs[2] on an Intel® Core™ i7 machine with 16 GB RAM.

4.1 Case Study and Datasets

Being one of the fastest-growing sectors of the service economy, the enterprise IT service subject area has gained in importance both in research and practice. One popular stream of this research is IT service management (ITSM) [116, 117], which focuses on the servitization of the IT function, organization and management of IT

[2] https://www.biointelligence.hu/pyhubs/.

Table 4 Datasets

#	Time period	# of ticket texts	# of acquired labels
Data1	2015–2018	28,243	30
Data2	January–May 2019	4,684	60

service provision [118]. For a successful establishment of organizational infrastructure for IT services, IT practitioners implement IT service delivery with a reference process—the IT Infrastructure Library (ITIL) [119, 120]. Separate areas of ITIL, such as Incident, Problem, or Change management (CHM), deal with a large amount of unstructured text data, i.e., tickets issued in relation to the incidents, problems, or changes in the IT infrastructure products and services. These tickets serve as a case study in the present work.

We obtained two datasets (see Table 4) in the form of IT ticket texts originating from an ITIL CHM department of a big enterprise (step 1). They were received according to their availability and covered the whole period obtainable at the moment of this study. The first dataset (Data1) comprised 28,243 tickets created in 2015–2018. The data was pre-processed (step 2) by removing of stop words, punctuation, turning to lowercase, stemming and converted into a CSV-formatted corpus of ticket texts. The tickets texts contained prevailingly English texts (more than 80% of English words, a small portion of German words was present). The second dataset (Data2) comprised 4,684 entries in prevailingly English language from the period January to May 2019. The length of IT ticket texts varied from 5–10 words to 120–150.

As labeling of the data is a time-consuming and tedious process and our data comes from an operational unit fully overbooked in the daily work, which is often the case in the industry, we managed to acquire 30 and 60 labels for Data1 and Data2 respectively (step 3). To provide a correct label, the case study workers needed to analyze all the factors influencing the IT ticket complexity: number of tasks; number of Configuration Items (CIs), specifically applications; if the ticket had to be executed online or offline (planning of downtime and considering affected CIs); involvement of Change Advisory Board (CAB), etc. Therefore, labeling a single ticket could take up to 30 min. For complexity class labels, a qualitative scale of low, medium, and high has been selected to simplify the classification task and as a well-known scale of priority ratings [121]. Although two datasets are coming from the same case, a distinct period of time and the number of labels allowed us to test our approaches in two independent settings.

4.2 Experimental Settings

To evaluate classifiers, we use leave-one-out cross-validation [122]. We select one of the labeled instances as test instance and use all the remaining instances as training

data (step 4). The process of selecting a test instance and training the classifier using all the other instances is repeated as many times as the number of labeled instances so that in each round, a different instance is selected as the test instance.

As evaluation metrics, we use accuracy, average precision, average recall, and F-score (step 7). To compare the differences in the classifiers' performance, we use the binomial test by Salzberg [123] at the significance threshold of 0.05. In the case of semi-supervised classifiers, we consider all the unlabeled instances as available at training time.

When applying semi-supervised classifiers, we consider all the unlabeled instances as available at the training time. In the case of kNN with self-training, we set the number of self-training iterations to 100.

In the experiments, we tested various distance measures (such as Euclidean distance, Cosine distance and Manhattan distance) and different k values, i.e., 1, 3, 5, 7, and 9. Hereby, we could not detect major differences. Hence, we use kNN with $k = 1$ and Euclidian distance, which is also justified by theoretical and empirical studies [124, 125].

In the context of explainability of text representation techniques, specifically linguistic features, we include feature selection tests in our experiments to identify which features play an important role in prediction quality. Similarly to [126, 127], we use logistic regression to determine the most predictive features based on weights. Subsequently, we train our classifiers using the selected set of best-performing features to compare the changes in prediction results.

4.3 Comparison of SUCCESS and QuickSUCCESS

As an initial experiment, we measured the execution time of SUCCESS and QuickSUCCESS. We used the linguistic features representation. In the case of QuickSUCCESS approach, we selected $r = 100$ instances and trained $m = 100$ models. Table 5 shows our results: the execution time of one round of cross-validation (in seconds) and the accuracy. As one can see, the proposed technique leads to several orders of magnitude speed-up, both in case of Data1 and Data2, with negligible loss of prediction quality (the difference corresponds to the misclassification of one additional instance). Hence, in further experiments, we used the QuickSUCCESS algorithm.

Table 5 Execution time and accuracy of SUCCESS and QuickSUCCESS

	Time (s)	Accuracy
Data1 SUCCESS	75,831	0.600
QuickSUCCESS	2	0.523
Data2 SUCCESS	586	0.767
QuickSUCCESS	3	0.767

4.4 Results

In this section, we present our experimental results regarding text representation and text classification techniques on the two case study datasets. In Table 6, we provide obtained values of accuracy, an average of precision and recall calculated over the three prediction classes of low, medium, and high complexity. F-score is calculated based on the average precision and recall.

With the help of these values, we analyze how far the choice of text representation techniques and ML algorithms influences the quality of prediction. We compare the performance of classifiers using linguistic features with that of classifiers using TF-IDF on both Data1 and Data2.

As stated in the Experimental settings subsection, using logistic regression, we identified the most predictive features for Data1 and Data2. According to the weights of logistic regression, in the case of Data 1, five most important features are (1) relative occurrence of cognitive words, (2) occurrence of unique adjectives in all adjectives, (3) wording style, (4) occurrence of unique verbs in all verbs, (5) relative occurrence of words with positive sentiment. In the case of Data2, the five most important features are (1) relative occurrence of cognitive words, (2) occurrence of unique adjectives in all adjectives, (3) occurrence of unique verbs in all verbs, (4) relative occurrence of words with negative and (5) positive sentiments. As it can be concluded, the most essential feature appeared to be a relative occurrence of cognitive words, further referred to as *cognitive words feature*. After that, we trained the classifiers with the selected linguistic features. The results are summarized in Table 7.

It is to note that in both experimental settings on Data1 and Data2, we made prevailingly consistent observations, which are discussed below.

Text representation techniques

We compared linguistic features and TF-IDF while applying various classifiers in our text classification pipeline (Fig. 2). We point out a systematic improvement in the prediction quality of algorithms when using linguistic features. Whenever the difference between the two experiments was only text representation technique, we observed that the classifiers' performance using linguistic features was almost always higher than that in the case of using TF-IDF.

ML classifiers

Due to the higher number of labeled tickets, Data2 revealed an expected systematic classification quality increase compared to Data1, independently of applied algorithm or text representation technique.

As can be seen in Tables 6 and 7, the usage of our linguistic features delivers excellent performance with simple algorithms, such as decision trees and naïve Bayes. Both of them are statistically significantly better than other classifiers. Additionally, we observed that selecting the best set of features improves the performance of classifiers consistently. Hence, using a smaller set of features also simplifies the process of extraction and reduces computational costs. As mentioned above and according

Table 6 Accuracy, average precision, average recall and F-score of the examined classifiers using linguistic features and TF-IDF

Algorithm	Accuracy	Average precision	Average recall	F-score
Data1: linguistic features				
standard kNN	0.533	0.526	0.498	0.511
kNN self-training	0.533	0.526	0.498	0.511
kNN self-training & resampling	0.533	0.526	0.498	0.511
ECkNN	0.633	0.599	0.580	0.589
HFNN	0.600	0.411	0.490	0.447
HWkNN	0.533	0.526	0.498	0.511
NHBNN	0.433	0.144	0.333	0.202
decision trees	**1.000**	**1.000**	**1.000**	**1.000**
naïve Bayes	**0.967**	**0.972**	**0.944**	**0.958**
logistic regression	0.500	0.475	0.463	0.469
QuickSUCCESS	0.533	0.542	0.523	0.532
Data1: TF-IDF				
standard kNN	0.200	0.067	0.333	0.111
kNN self-training	0.200	0.067	0.333	0.111
kNN self-training & resampling	0.200	0.067	0.333	0.111
ECkNN	0.433	0.144	0.333	0.202
HFNN	0.433	0.144	0.333	0.202
HWkNN	0.200	0.067	0.333	0.111
NHBNN	0.433	0.144	0.333	0.202
decision trees	0.433	0.144	0.333	0.202
naïve Bayes	0.267	0.738	0.389	0.510
logistic regression	0.433	0.144	0.333	0.202
QuickSUCCESS	0.200	0.067	0.333	0.111
Data2: linguistic features				
standard kNN	0.750	0.593	0.576	0.584
kNN self-training	0.750	0.593	0.576	0.584
kNN self-training & resampling	0.750	0.593	0.576	0.584
ECkNN	0.733	0.586	0.568	0.577
HFNN	0.733	0.539	0.528	0.534
HWkNN	0.750	0.593	0.576	0.584
NHBNN	0.700	0.233	0.333	0.275
decision trees	**1.000**	**1.000**	**1.000**	**1.000**

(continued)

Table 6 (continued)

Algorithm	Accuracy	Average precision	Average recall	F-score
naïve Bayes	**1.000**	**1.000**	**1.000**	**1.000**
logistic regression	0.800	0.552	0.582	0.567
QuickSUCCESS	0.750	0.593	0.576	0.584
Data2: TF-IDF				
standard kNN	0.700	0.233	0.333	0.275
kNN self-training	0.700	0.233	0.333	0.275
kNN self-training & resampling	0.700	0.233	0.333	0.275
ECkNN	0.700	0.233	0.333	0.275
HFNN	0.700	0.233	0.333	0.275
HWkNN	0.700	0.233	0.333	0.275
NHBNN	0.700	0.233	0.333	0.275
decision trees	0.700	0.233	0.333	0.275
naïve Bayes	0.467	0.419	0.468	0.442
logistic regression	0.700	0.233	0.333	0.275
QuickSUCCESS	0.700	0.233	0.333	0.275

to the weights of logistic regression, the most important feature in the ticket text appears to be the *cognitive words feature.*

When enhancing kNN with self-learning, which is a semi-supervised technique and a known way to improve the learning process under the condition of a large number of unlabeled and a small number of labeled instances, we expected a noticeable increase in performance quality. Nonetheless, the evaluation results evidenced no improvement. Considering hubness-aware classifiers, in some cases, selected hubness-aware classifiers proved to be better or equally performing if compared to standard kNN.

Additionally, under the condition of TF-IDF representation on Data2, the performance of almost all algorithmic approaches stayed on the same level (0.700, 0.233, 0.333, 0.275 for accuracy, precision, recall, and F-score respectively). Due to the small number of labeled instances and dominance of the low complexity tickets in the labeled set, our dataset can be described as an imbalanced one. This can be the reason that all classifiers (except for naïve Bayes) predicted the dominant class of low complexity.

5 Discussion

In this study, our focus was to gain a better understanding of those factors influencing the quality of prediction in text classification tasks under the lense of XAI and

Table 7 Accuracy, average precision, average recall and F-score of the examined classifiers using five best linguistic features

Algorithm	Accuracy	Average precision	Average recall	F-score
Data1: five best linguistic features				
standard kNN	0.667	0.589	0.585	0.587
kNN self-training	0.667	0.589	0.585	0.587
kNN self-training & resampling	0.667	0.589	0.585	0.587
ECkNN	0.667	0.578	0.580	0.579
HFNN	0.633	0.429	0.515	0.468
HWkNN	0.667	0.589	0.585	0.587
NHBNN	0.433	0.144	0.333	0.202
decision trees	**1.000**	**1.000**	**1.000**	**1.000**
naïve Bayes	**1.000**	**1.000**	**1.000**	**1.000**
logistic regression	0.633	0.611	0.580	0.595
QuickSUCCESS	0.733	0.814	0.636	0.714
Data2: five best linguistic features				
Standard kNN	0.817	0.711	0.670	0.690
kNN self-training	0.817	0.711	0.670	0.690
kNN self-training & resampling	0.700	0.233	0.333	0.275
ECkNN	0.750	0.523	0.558	0.539
HFNN	0.800	0.526	0.582	0.553
HWkNN	0.817	0.711	0.670	0.690
NHBNN	0.700	0.233	0.333	0.275
decision trees	**1.000**	**1.000**	**1.000**	**1.000**
naïve Bayes	**1.000**	**1.000**	**1.000**	**1.000**
logistic regression	0.850	0.554	0.606	0.579
QuickSUCCESS	0.867	0.791	0.693	0.739

granularity. At the same time, we addressed (i) the need for further experiments in industrial settings, especially with recent classifiers that have not been used for ticket classification before and (ii) limitations of our rule-based approach. Below we will discuss the explainability and granularity implications of the study results as well as methodological, managerial, and practical contributions.

5.1 Explainability and Granularity Implications

Since AI becomes an increasing part of our business and daily lives, we recognize the paramount importance of the explainability of AI-based solutions. The need to understand the decisions made by an AI system is crucial in the context of trust and primarily to efficiently address the errors made by such a system. In the present research, we closely studied the ITIL Change Management IT ticket processing of a big telecommunication company. Implementing the changes in the IT infrastructure means every time intervening in the organization's IT environment, its functions, and experience. Due to the criticality of this infrastructure, one can consider the IT ticket classification task as a high risk and high consequence decision-making process. Wrong decisions and actions based on those decisions can have severe consequences for the company, for example, a complete connection service outage for city districts or even country regions. Hence, the classification outcomes should be explainable and understandable for a human decision-maker. Therefore, we emphasize the explainability of the selected approaches, both in text representation and text classification.

In our study, we observed that simple, explainable algorithms, such as decision trees and naïve Bayes, can deliver excellent performance results when applied with our linguistic features-based text representation. Furthermore, we note that text classification is inherently related to Granular Computing [17, 128–134]. This can be accounted for the observation that in typical text classification pipelines, various representations of the data are considered (such as raw text, TF-IDF, domain-specific features) that correspond to various levels of abstraction.

5.2 Methodological Contributions

With the vast research on text classification in general, numerous limitations are reported, and demand for more experiments and case study-based findings is declared. The latter is one of the limitations we addressed in the chapter. Especially while designing linguistic features representation, it is essential to have some knowledge about the authors of the text and involve the experts in the design of features. This is also important with respect to the explainability of the approach. For example, in the given case study, we used the text length (word count) as one of the linguistic features. In the case study interviews, we found out that short IT tickets were prevailingly written by professionals to professionals. Thus, in case of simple, explicit, and already familiar requests, case study process workers usually received short texts composed in a very condensed telegraphic way, which is also supported by the heuristics of theory of the least effort [135]. Such information was particularly important due to the specificity of classification task—IT ticket complexity prediction.

However, based on our experiments with the weights of the linguistic features, we identified that the most predictive linguistic features are those related to the relative distribution of cognitive words, positive and negative business sentiments (the construction of which demanded the involvement of experts), and relative occurrences of unique verbs, adjectives and wording style feature (the development of which didn't demand the involvement of experts). The ticket length appeared to be unimportant in predicting the ticket label. Based on these finding as well as further discussions with case study process workers, we could explain this phenomenon by two reasons:

1. In some cases, we missed the original IT ticket texts in the provided datasets. Very often, in real-world settings, the rules are not followed. I.e., in our case, process workers found workarounds how to simplify and accelerate the IT ticket processing. The ticket templates were used, and the original ticket texts were not copied into the template.
2. In certain scenarios, the heuristics of the theory of least effort didn't work, meaning that also complex tickets were written in a condensed way. In this case, the complexity was defined by other factors, such as the number and criticality of affected configuration items.

As there is a general discussion on the advantages and disadvantages of various text representation techniques and classification algorithms, in our chapter, we offer a new comprehensive analysis of both topics. First, we systematically compared the efficiency of linguistic features and TF-IDF representation for IT ticket complexity prediction. To the best of our knowledge, it is the first attempt of such a research setting in the context of explainability. Second, using different datasets and text representation techniques, we consistently tested eleven ML classifiers and showed that simple algorithms, such as decision trees and naïve Bayes, achieved accurate prediction with the linguistic features representation. Hereby, the five best performing features delivered results of the same quality. Hence, building explainable text classification pipelines, i.e., using linguistic features with simple algorithms, can have great potential when applied in real-world tasks.

To sum up, systematic testing of the aforementioned representation techniques and classification algorithms and their application to the IT ticket classification task of complexity prediction embrace the significant potential for the text data researchers who seek to understand the role of specific factors influencing the prediction quality such as text representations and classifiers.

5.3 Managerial and Practical Contributions

From the managerial contribution perspective, our case study is related to the IT ticket processing area. With today's increased digitization, any enterprise maintains a broad application portfolio, often grown historically, which must be supported by large scale complex IT service environments [136]. This reveals a fundamental role

of IT support systems in any organization's support operations. Two essential steps of any IT ticket processing, which are their correct prioritization and assignment, attract the attention of managers, especially in the context of an ever-increasing amount of tickets, errors, long lead times, and lack of human resources [137–142]. While small companies still tend to perform these steps manually, large organizations dedicate high budgets to the implementation of various commercial text classification solutions. Usually, these approaches are complex monolithic software focused on accuracy at the cost of explainability and understandability for a process worker. As mentioned above, considering the IT infrastructure criticality, the IT helpdesk workers' ability to evaluate the automatic ticket classification decision allows to address efficiently the problem of wrongly classified tickets, this way reducing the errors, rework, and processing time.

An essential practical contribution of the present study is improving our own rule-based approach where we used the described linguistic features approach and both handcrafted and decision trees based rules to predict the IT ticket complexity of low, medium, or high [16]. Using the ML classification pipeline discussed in the chapter, we managed to achieve the best prediction quality without the need to define and constantly update the rules with the experts, which is a big hurdle in the management, maintenance, and scalability of such systems.

Thus, the managerial and practical contributions of the research can be summarized as follows: (i) providing insights into building explainable IT ticket classification applications for managers and ML experts designing text classification pipelines as well as for IT subject matter experts using them; (ii) addressing the limitations of our rule-based approach of IT ticket complexity prediction [16], namely inability to learn and scale, difficulty in analysis, testing, and maintenance.

6 Conclusion and Future Works

The goal of our work was to develop an understanding and provide a comparative analysis of text representation techniques and classifiers while focusing on the development of an explainable IT ticket classification pipeline. The obtained knowledge can be useful for decision support in the text classification task design of various enterprise applications, specifically in the IT ticket area. Additionally, we addressed the limitations of related work as well as our research. Below, we highlight the methodological, managerial, and practical contributions.

The methodological contributions can be summarized as follows: (i) case study based linguistic features extraction and the identification of the best set of features allow us to understand their predictive power; (ii) the comprehensive comparative analysis of linguistic features with TF-IDF and various ML algorithms confirms the positive influence of linguistic style predictors on the prediction quality already detected by [61]; (iii) our observation that simple algorithms work well if using appropriate linguistic features contributes to the general discussion on the advantages and disadvantages of various text classification algorithms [19, 20].

The following managerial and practical contributions of the work are presented and discussed: (i) our case study findings can provide decision support for ML experts in the design of text classification pipelines, (ii) improvement of our rule-based approach with ML.

As a part of future work, one can: (i) consider further information that may be related to IT ticket complexity, such as the number of tasks and configuration items per ticket; (ii) test other application cases in the IT ticket area and beyond, i.e., further explore the potential of linguistic features; (iii) as we showed that selecting an appropriate subset of linguistic features can considerably improve the performance of classifiers, one may conduct further experiments with more advanced feature selection techniques [143].

Acknowledgments A. Revina was supported by the Data Science and Engineering Department, Faculty of Informatics, Eötvös Loránd University. K. Buza was supported by the project ED_18-1-2019-0030. Project no. ED_18-1-2019-0030 (Application domain-specific highly reliable IT solutions subprogramme) has been implemented with the support provided from the National Research, Development, and Innovation Fund of Hungary, financed under the Thematic Excellence Programme funding scheme.

Appendix I: Taxonomy of Decision-Making Logic Levels

Following [15], we consider diverse semantic concepts: **R**esources, **T**echniques, **C**apacities, and **C**hoices, elements of RTCC framework. We designed contextual variables [144], based on which experts categorized words into one of the three DML levels and one of the four semantic concepts.[3]

Contextual variables	Decision-making logic levels		
	Routine	Semi-cognitive	Cognitive
	Conceptual aspects		
	RESOURCES		
Problem Processing Level	User, user request, task, test, check, target, release, contact role, access, interface, cluster, tool, client, file system, partner, node	Team, leader, project, colleague, property	Management, system, CAB, measure, approval

(continued)

[3] https://github.com/IT-Tickets-Text-Analytics.

(continued)

Contextual variables	Decision-making logic levels		
	Routine	Semi-cognitive	Cognitive
	Conceptual aspects		
Accuracy	Time, application, product, configuration item, CI, right, instance, machine, minute, hour, day, week, detail, description	Description, environment, requirement, validity, reason, solution, method, problem, rule, modification	
Situational Awareness	Name, password, group, directory, number, email, package, phone, ID, IP, attachment	Request for change, RfC, customer, rollout, backout	Server farm
Information	Server, file, location, dataset, network, data, patch, port, information, type, root, certificate, account, device, cable, parameter, agent, folder, disk, fallback, database, db, backup, version, tool, firewall, system, hotfix, supervisor, reference, instruction, format	Requestor, software, downtime, production, power-supply, outage, service, case	Risk, freeze, impact
	TECHNIQUES		
Experience	Need, see, deploy, document, monitor, use, follow, note, provide, test, contain, accompany, inform, consist, describe	Implement, create, support, require, classify	Approve, delegate, propose
Action Choice	Start, finish, monitor, import, export, run, stop, step, end, put, send, switch, install, reject, update, upgrade, include, replace, remove, move, begin, make, get, migrate, open, initialize, revoke	Deploy, migrate, process, modify, forget, increase, miss	Freeze

(continued)

(continued)

Contextual variables	Decision-making logic levels		
	Routine	Semi-cognitive	Cognitive
	Conceptual aspects		
Effort	Cancel, rundown, decommission, restart, delete, set, add, activate, reboot, specify, agree upgrade, mount, execute, transfer, write, find	Perform, modify, assign, check, need, expect, verify	Define
	CAPACITIES		
Specificity	Additional, preapproved, affected, initial, attached, internal, external, reachable, regular, active, scheduled, next, whole, formal, virtual, wrong, individual, administrative, local	Secure, separate, specific, technical, urgent, separate, corrected, minor, normal	Related, multiple, multi-solution, major, high, small, big
Decisions Formulation	New, old, preinstalled, fixed, ready, following, current, valid, primary, necessary	Available, necessary, important, significant, successful, appropriate, relevant, main, further, responsible	Possible, many, desired, different, various
Predictability	Actual, full, online, standard, responsible, administrative, existing, minimum, same, visible	Strong, temporary, offline, previous, last, other, more, much, similar, standard	Random, strong randomized, encrypted, expected
	CHOICES		
Precision	Automatically, instead, manually, there, where, here, separately, additionally, internally	Normally, newly, shortly, urgently, temporarily	Maybe, randomly, likely

(continued)

(continued)

Contextual variables	Decision-making logic levels		
	Routine	Semi-cognitive	Cognitive
	Conceptual aspects		
Scale	Permanently, currently, still, now, often, never, already, just, always, yet, anymore, firstly, before, together, daily, meanwhile, really, furthermore, afterwards, therefore	Again, later, however, usually, previously, recently	Soon
Ambiguity	Correctly, therefore, accordingly, actually, consequently, completely, simultaneously, anyway, necessarily	Well, enough, immediately, easily, simply	Approximately, properly

Appendix II: Business Sentiment Lexicon with Assigned Valences[4]

Tickets	ITIL	Valence
Expressions		
No risk, no outage		+2
Be so kind, would be nice		0.5
Disaster recovery, set alarms warnings, poison attack vulnerability, critical security leaks, fan, outstanding windows updates, thank you, kind regards, would like, best regards	Request for change, RfC	0
Big measure	Projected service outage, change advisory board, high impact, major change	−0.5
Single key words		
Kind, success, correct, like, nice	Well, successful, happy	0.5

(continued)

[4] https://github.com/IT-Tickets-Text-Analytics.

(continued)

Tickets	ITIL	Valence
Disaster, recovery, affected, stop, disable, dump, alarm, warning, poison, attack, vulnerability, error, prevent, drop, cancel, delete, exclude, problem, problems, faulty, failed, destroy, defective, obsolete, lack, security, leak, crash, please, support, optimize, grant, privilege, create, dear, acceptance, clarity, restore, increase, danger, balance, right, deny, wrong, retire, missing, weak, invalid, see, follow, yes, allow, approve, approval, confirm	Problem, failed, information, operational, identify, order, include, adequately, procedure, necessary, assess, criteria, clear, provide, potentially, identification, adequate, initiate, value, KPI, standard, schedule, align, properly, release, accurate, report, organization, continuous, ensure, service, beneficial, stakeholder, requirement, correct, record, essential, clearly, RfC, support, tool, relevant, attempt, subsequently, configuration, different, follow, directly, CI, potential, request, individual, plan, work, evaluate, author, organizational, manage, number, financial, status, low, chronological, recommend, responsible, model, accountable, handle, timescale, business, normal, submit, update, create, manual, consider, backout, accept, item, project, deliver, formal, data, iterative, produce, local, describe, test, improve, result, deployment, deploy, technical, management, repeatable, determine, minimum, develop, appropriate, activate, implement, require, process, evaluation, customer, contractual, authorize, share, acceptable	0
Blocked, critical	Cost, PSO, CAB, important, unauthorized, major, significant, undesirable, incomplete, delegate, avoid, coordinate, immediately, significantly	−0.5
Offline, risk, outage, emergency, downtime	Impact, risk, emergency, incident, outage, downtime	−1
Rejected	Unacceptable	−2

References

1. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Pearson (2009)
2. Müller, O., Junglas, I., Debortoli, S., vom Brocke, J.: Using text analytics to derive customer service management benefits from unstructured data. MIS Q. Exec. **15**, 243–258 (2016)
3. Jiao, J. (Roger), Zhang, L. (Linda), Pokharel, S., He, Z.: Identifying generic routings for product families based on text mining and tree matching. Decis. Support Syst. **43**, 866–883 (2007). https://doi.org/10.1016/j.dss.2007.01.001
4. Luque, C., Luna, J.M., Luque, M., Ventura, S.: An advanced review on text mining in medicine. WIREs Data Min. Knowl. Disc. **9**(3). Wiley Online Library (2019). https://doi.org/10.1002/widm.1302

5. Wang, Y., Xu, W.: Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud. Decis. Support Syst. **105**, 87–95 (2018). https://doi.org/10.1016/j.dss.2017.11.001
6. Ibrahim, N.F., Wang, X.: A text analytics approach for online retailing service improvement: evidence from Twitter. Decis. Support Syst. **121**, 37–50 (2019). https://doi.org/10.1016/j.dss.2019.03.002
7. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining Explanations: An Overview of Interpretability of Machine Learning. In: Proc. IEEE 5th Int. Conf. Data Sci. Adv. Anal. DSAA 2018, pp. 80–89 (2018)
8. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Pedreschi, D., Giannotti, F.: A survey of methods for explaining black box models. ACM Comput. Surv. **51** (5), article no. 93 (2018)
9. Lee, J.D., See, K.A.: Trust in automation: designing for appropriate reliance. Human Factor Editor's Collection **46** (1), 50–80 (2004). https://doi.org/10.1518/hfes.46.1.50_30392
10. Jennings, N.R., Moreau, L., Nicholson, D., Ramchurn, S., Roberts, S., Rodden, T., Rogers, A.: Human-agent collectives. Commun. ACM **57** (12), 80–88 (2014). https://doi.org/10.1145/2629559
11. Rosenfeld, A., Richardson, A.: Explainability in human–agent systems. Auton. Agent. Multi. Agent. Syst. **33**, 673–705 (2019). https://doi.org/10.1007/s10458-019-09408-y
12. Rai, A.: Explainable AI: from black box to glass box. J. Acad. Mark. Sci. **48**, 137–141(2020). https://doi.org/10.1007/s11747-019-00710-5
13. Rizun, N., Revina, A., Meister, V.: Discovery of stylistic patterns in business process textual descriptions: IT Ticket Case. In: Proc. 33rd International Business Information Management Association Conference (IBIMA), pp. 2103–2113, Granada (2019)
14. Rizun, N., Revina, A.: Business sentiment analysis. Concept and method for perceived anticipated effort identification. In: Proc. Inf. Syst. Develop. Inf. Syst. Beyond (ISD), Toulon (2019)
15. Rizun, N., Revina, A., Meister, V.: Method of Decision-Making Logic Discovery in the Business Process Textual Data. In: Proc. Int. Conf. Bus. Inf. Syst. (BIS), pp. 70–84, Sevilla (2019). https://doi.org/10.1007/978-3-030-20485-3_6
16. Revina, A., Rizun, N.: Multi-criteria knowledge-based recommender system for decision support in complex business processes. In: Koolen, M., Bogers, T., Mobasher, B., and Tuzhilin, A. (eds.) Proceedings of the Workshop on Recommendation in Complex Scenarios co-located with 13th ACM Conference on Recommender Systems (RecSys 2019), pp. 16–22, CEUR, Copenhagen (2019)
17. Pedrycz, W.: Granular computing: an introduction. In: Annual Conference of the North American Fuzzy Information Processing Society (NAFIPS), pp. 1349–1354 (2001). https://doi.org/10.1109/nafips.2001.943745
18. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manag. **24**, 513–523 (1988). https://doi.org/10.1016/0306-4573(88)90021-0
19. Kowsari, K., Meimandi, K.J., Heidarysafa, M., Mendu, S., Barnes, L.E., Brown, D.E.: Text classification algorithms: a survey. Information **10** (4), 1–68, MDPI (2019). https://doi.org/10.3390/info10040150
20. Cavalcanti, Y.C., da Mota Silveira Neto, P.A., Machado, I. do C., Vale, T.F., de Almeida, E.S., Meira, S.R. de L.: Challenges and opportunities for software change request repositories: a systematic mapping study. J. Softw. Evol. Process. **26**, 620–653 (2014). https://doi.org/10.1002/smr.1639
21. Tomašev, N., Buza, K.: Hubness-aware kNN classification of high-dimensional data in presence of label noise. Neurocomputing. **160**, 157–172 (2015). https://doi.org/10.1016/J.NEUCOM.2014.10.084
22. Tomašev, N., Buza, K., Marussy, K., Kis, P.B.: Hubness-aware classification, instance selection and feature construction: survey and extensions to time-series. In: Stańczyk, U. and Jain, L.C. (eds.) Feature Selection for Data and Pattern Recognition. pp. 231–262. Springer, Berlin (2015). https://doi.org/10.1007/978-3-662-45620-0_11

23. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**, 81–106 (1986). https://doi.org/10.1007/BF00116251
24. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. UK Cambridge University Press, pp. 253–286 (2009)
25. Hosmer, D.W., Lemeshow, S., Sturdivant, R.X.: Applied Logistic Regression. Hoboken (2013)
26. Triguero, I., García, S., Herrera, F.: Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. Knowl. Inf. Syst. **42**, 245–284 (2015). https://doi.org/10.1007/s10115-013-0706-y
27. Marussy, K., Buza, K.: SUCCESS: A new approach for semi-supervised classification of time-series. In: International Conference on Artificial Intelligence and Soft Computing, pp. 437–447, Springer, Zakopane (2013). https://doi.org/10.1007/978-3-642-38658-9_39
28. Buza, K., Revina, A.: Speeding up the SUCCESS Approach for Massive Industrial Datasets. In: Proc. Int. Conf. Innov. Intell. Syst. Appl. (INISTA), pp. 1–6, IEEE Digital Library, Novi Sad (2020)
29. van der Aa, H., Leopold, H., del-Río-Ortega, A., Resinas, M., Reijers, H.A.: Transforming unstructured natural language descriptions into measurable process performance indicators using Hidden Markov Models. Inf. Syst. **71**, 27–39 (2017). https://doi.org/10.1016/J.IS.2017.06.005
30. van der Aa, H., Leopold, H., Reijers, H.A.: Checking process compliance against natural language specifications using behavioral spaces. Inf. Syst. **78**, 83–95 (2018). https://doi.org/10.1016/J.IS.2018.01.007
31. The Economist explains–Why Uber's self-driving car killed a pedestrian | The Economist explains | The Economist, https://www.economist.com/the-economist-explains/2018/05/29/why-ubers-self-driving-car-killed-a-pedestrian, last accessed 2020/08/21
32. IBM's Watson recommended "unsafe and incorrect" cancer treatments—STAT, https://www.statnews.com/2018/07/25/ibm-watson-recommended-unsafe-incorrect-treatments/, last accessed 2020/08/21
33. Russell, S.: Human Compatible: Artificial Intelligence and the Problem of Control. Penguin Publishing Group (2019)
34. Samek, W., Wiegand, T., Müller, K.-R.: Explainable artificial intelligence: understanding, visualizing and interpreting deep learning models. arXiv (2017)
35. Lipton, Z.C.: The mythos of model interpretability. Commun. ACM. **61**, 35–43 (2016)
36. Explanation noun–Definition, pictures, pronunciation and usage notes | Oxford Learner's Dictionary of Academic English at OxfordLearnersDictionaries.com, https://www.oxfordlearnersdictionaries.com/definition/academic/explanation, last accessed 2020/08/21
37. Singh, P., Dhiman, G.: A hybrid fuzzy time series forecasting model based on granular computing and bio-inspired optimization approaches. J. Comput. Sci. **27**, 370–385 (2018). https://doi.org/10.1016/j.jocs.2018.05.008
38. Hryniewicz, O., Kaczmarek, K.: Bayesian analysis of time series using granular computing approach. Appl. Soft Comput. J. **47**, 644–652 (2016). https://doi.org/10.1016/j.asoc.2014.11.024
39. Han, Z., Zhao, J., Wang, W., Liu, Y., Liu, Q.: Granular computing concept based long-term prediction of gas tank levels in steel industry. In: IFAC Proceedings Volumes (IFAC-PapersOnline), pp. 6105–6110, IFAC Secretariat (2014). https://doi.org/10.3182/20140824-6-za-1003.00842
40. Hu, H., Pang, L., Tian, D., Shi, Z.: Perception granular computing in visual haze-free task. Expert Syst. Appl. **41**, 2729–2741 (2014). https://doi.org/10.1016/j.eswa.2013.11.006
41. Ray, S.S., Ganivada, A., Pal, S.K.: A granular self-organizing map for clustering and gene selection in microarray data. IEEE Trans. Neural Networks Learn. Syst. **27**, 1890–1906 (2016). https://doi.org/10.1109/TNNLS.2015.2460994
42. Tang, Y., Zhang, Y.Q., Huang, Z., Hu, X., Zhao, Y.: Recursive fuzzy granulation for gene subsets extraction and cancer classification. IEEE Trans. Inf. Technol. Biomed. **12**, 723–730 (2008). https://doi.org/10.1109/TITB.2008.920787

43. Saberi, M., Mirtalaie, M.S., Hussain, F.K., Azadeh, A., Hussain, O.K., Ashjari, B.: A granular computing-based approach to credit scoring modeling. Neurocomputing. **122**, 100–115 (2013). https://doi.org/10.1016/j.neucom.2013.05.020
44. Leng, J., Chen, Q., Mao, N., Jiang, P.: Combining granular computing technique with deep learning for service planning under social manufacturing contexts. Knowl.-Based Syst. **143**, 295–306 (2018). https://doi.org/10.1016/j.knosys.2017.07.023
45. Scott, S., Matwin, S.: Text Classification Using WordNet Hypernyms. In: Usage of WordNet in Natural Language Processing Systems, pp. 45–51, ACL Anthology (1998)
46. Yan, J.: Text Representation. In: Encyclopedia of Database Systems, pp. 3069–3072, Springer US, Boston, MA (2009). https://doi.org/10.1007/978-0-387-39940-9_420
47. Zhang, Y., Jin, R., Zhou, Z.H.: Understanding bag-of-words model: a statistical framework. Int. J. Mach. Learn. Cybern. **1**, 43–52 (2010). https://doi.org/10.1007/s13042-010-0001-0
48. Sparck, K.J.: A statistical interpretation of term specificity and its application in retrieval. J. Doc. **28**, 11–21 (1972). https://doi.org/10.1108/eb026526
49. Radovanovic, M., Ivanovic, M.: Text Mining: Approaches and Applications. Novi Sad J. Math. **38**, 227–234 (2008)
50. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. arXiv (2013)
51. Le, Q. V., Mikolov, T.: Distributed Representations of Sentences and Documents. arXiv (2014)
52. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proc. Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, Association for Computational Linguistics, Stroudsburg, PA, USA (2014). https://doi.org/10.3115/v1/D14-1162
53. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: FastText: Compressing text classification models. arXiv (2016)
54. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. arXiv (2016)
55. Melamud, O., Goldberger, J., Dagan, I.: context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In: Proc. 20th SIGNLL Conference on Computational Natural Language Learning, pp. 51–61, Association for Computational Linguistics, Stroudsburg, PA, USA (2016). https://doi.org/10.18653/v1/K16-1006
56. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv (2018)
57. Rezaeinia, S.M., Ghodsi, A., Rahmani, R.: Improving the accuracy of pre-trained word embeddings for sentiment analysis. arXiv (2017)
58. Yin, F., Wang, Y., Liu, J., Lin, L.: The Construction of Sentiment Lexicon Based on Context-Dependent Part-of-Speech Chunks for Semantic Disambiguation. IEEE Access **8** (2020). https://doi.org/10.1109/access.2020.2984284
59. Sureka, A., Indukuri, K.V.: Linguistic analysis of bug report titles with respect to the dimension of bug importance. In: Proc. 3rd Annual ACM Bangalore Conference on COMPUTE '10, pp. 1–6, ACM Press, Bangalore (2010). https://doi.org/10.1145/1754288.1754297
60. Ko, A.J., Myers, B.A., Chau, D.H.: A Linguistic analysis of how people describe software problems. In: Visual Languages and Human-Centric Computing (VL/HCC'06), pp. 127–134, IEEE, Brighton (2006). https://doi.org/10.1109/VLHCC.2006.3
61. Coussement, K., Van den Poel, D.: Improving customer complaint management by automatic email classification using linguistic style features as predictors. Decis. Support Syst. **44**, 870–882 (2008). https://doi.org/10.1016/J.DSS.2007.10.010
62. Fürnkranz, J., Mitchell, T., Riloff, E.: Case study in using linguistic phrases for text categorization on the WWW. Workshop Series Technical Reports WS-98-05, Association for the Advancement of Artificial Intelligence (AAAI), Palo Alto, California (1998)
63. Mladenic, D., Grobelnik, M.: Word sequences as features in text-learning. In: Proc. 17th Electrotechnical and Computer Science Conference (ERK98), pp. 145–148, Ljubljana (1998)
64. Raskutti, B., Ferrá, H., Kowalczyk, A.: Second order features for maximising text classification performance. In: Proc. European Conference on Machine Learning, pp. 419–430, Springer (2001). https://doi.org/10.1007/3-540-44795-4_36

65. Bekkerman, R., El-Yaniv, R., Tishby, N., Winter, Y.: On feature distributional clustering for text categorization. In: Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 146–153, ACM Press, New York (2001). https://doi.org/10.1145/383952.383976
66. Tan, C.M., Wang, Y.F., Lee, C.D.: The use of bigrams to enhance text categorization. Inf. Process. Manag. **38**, 529–546 (2002). https://doi.org/10.1016/S0306-4573(01)00045-0
67. Scott, S., Matwin, S.: Feature engineering for text classification. In: Proc. ICML-99, 16th International Conference on Machine Learning, pp. 379–388, Morgan Kaufmann Publishers, Bled (1999)
68. Moschitti, A., Basili, R.: Complex linguistic features for text classification: a comprehensive study. In: European Conference on Information Retrieval, pp. 181–196, Springer, Sunderland (2004). https://doi.org/10.1007/978-3-540-24752-4_14
69. Sasaki, M., Kita, K.: Rule-based text categorization using hierarchical categories. In: Proc. IEEE International Conference on Systems, Man, and Cybernetics, pp. 2827–2830, IEEE, San Diego (1998). https://doi.org/10.1109/ICSMC.1998.725090
70. Mason, J.: Qualitative Researching. Sage Publications Ltd (2002)
71. Seidel, J., Kelle, U.: Computer-Aided Qualitative Data Analysis: Theory, Methods and Practice. SAGE Publications Ltd, London (1995)
72. Chua, S., Coenen, F., Malcolm, G.: Classification inductive rule learning with negated features. In: Proc. 6th International Conference Advanced Data Mining and Applications (ADMA), pp. 125–136, Springer, Chongqing (2010). https://doi.org/10.1007/978-3-642-17316-5_12
73. Rokach, L.: Decision forest: twenty years of research. Inf. Fusion. **27**, 111–125 (2016). https://doi.org/10.1016/j.inffus.2015.06.005
74. García, E., Romero, C., Ventura, S., Calders, T.: Drawbacks and solutions of applying association rule mining in learning management systems. In: Proc. International Workshop on Applying Data Mining in e-Learning, pp. 13–22, CEUR (2007)
75. Kaur, G.: Association rule mining: a survey. Int. J. Comput. Sci. Inf. Technol. **5**, 2320–2324 (2014)
76. Hu, H., Li, J.: Using association rules to make rule-based classifiers robust. In: 16th Australasian Database Conference, pp. 47–54, Australian Computer Society, Newcastle (2005)
77. Chakravarthy, V., Joshi, S., Ramakrishnan, G., Godbole, S., Balakrishnan, S.: Learning decision lists with known rules for text mining. IBM Research, 835–840 (2008)
78. Mitchell, T.M.: Machine Learning. McGraw Hill (1997)
79. Uzuner, Ö., Zhang, X., Sibanda, T.: Machine learning and rule-based approaches to assertion classification. J. Am. Med. Informatics Assoc. **16**, 109–115 (2009). https://doi.org/10.1197/jamia.M2950
80. Frank, E., Bouckaert, R.R.: Naive Bayes for text classification with unbalanced classes. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 503–510, Springer (2006). https://doi.org/10.1007/11871637_49
81. Zhu, X.: Semi-Supervised Learning Literature Survey. Technical report, University of Wisconsin-Madison (2008)
82. Lazarov, A., Shoval, P.: A rule-based system for automatic assignment of technicians to service faults. Decis. Support Syst. **32**, 343–360 (2002). https://doi.org/10.1016/S0167-9236(01)00122-1
83. Ahsan, S.N., Wotawa, F.: Impact analysis of SCRs using single and multi-label machine learning classification. In: Proc. IEEE International Symposium on Empirical Software Engineering and Measurement. ACM Press, Bolzano/Bozen (2010). https://doi.org/10.1145/1852786.1852851
84. Ahsan, S.N., Ferzund, J., Wotawa, F.: Automatic Classification of Software Change Request Using Multi-label Machine Learning Methods. In: Proc. 33rd Annual IEEE Software Engineering Workshop, pp. 79–86, IEEE (2009). https://doi.org/10.1109/SEW.2009.15

85. Rus, V., Nan, X., Shiva, S., Chen, Y.: Clustering of defect reports using graph partitioning algorithms. In: Proc. 21st International Conference on Software Engineering & Knowledge Engineering (SEKE'2009), pp. 442–445, Boston (2009)
86. Santana, A., Silva, J., Muniz, P., Araújo, F., de Souza, R.M.C.R.: Comparative analysis of clustering algorithms applied to the classification of bugs. In: Lecture Notes in Computer Science, pp. 592–598, Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34500-5_70
87. Kanwal, J., Maqbool, O.: Bug prioritization to facilitate bug report triage. J. Comput. Sci. Technol. **27**, 397–412 (2012). https://doi.org/10.1007/s11390-012-1230-3
88. Menzies, T., Marcus, A.: Automated severity assessment of software defect reports. In: Proc. IEEE International Conference on Software Maintenance, pp. 346–355, IEEE (2008). https://doi.org/10.1109/ICSM.2008.4658083
89. Luhn, H.P.: A statistical approach to mechanized encoding and searching of literary information. IBM J. Res. Dev. **1**, 309–317 (1957). https://doi.org/10.1147/rd.14.0309
90. Salton, G., Yang, C.: On the specification of term values in automatic indexing. J. Doc. **29**, 351–372 (1973). https://doi.org/10.1108/eb026562
91. Shin, D., Park, Y.J.: Role of fairness, accountability, and transparency in algorithmic affordance. Comput. Human Behav. **98**, 277–284 (2019). https://doi.org/10.1016/j.chb.2019.04.019
92. Diakopoulos, N., Koliska, M.: Algorithmic transparency in the news media. Digit. Journal. **5**, 809–828 (2017). https://doi.org/10.1080/21670811.2016.1208053
93. Hepenstal, S., McNeish, D.: Explainable artificial intelligence: what do you need to know? In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 266–275, Springer (2020). https://doi.org/10.1007/978-3-030-50353-6_20
94. Daelemans, W.: Explanation in computational stylometry. In: Proc. Int. Conf. on Intelligent Text Processing and Computational Linguistics, pp. 451–462, Springer, Samos (2013). https://doi.org/10.1007/978-3-642-37256-8_37
95. Taxonomy Strategies | Bibliography–Taxonomy Strategies, https://taxonomystrategies.com/library/bibliography/, last accessed 2019/09/02
96. Blumauer, A.: Taxonomies and Ontologies | LinkedIn Blog Article, https://www.linkedin.com/pulse/taxonomies-ontologies-andreas-blumauer/, last accessed 2019/09/02
97. Blei, D.: Probabilistic topic models. Commun. ACM. **55**, 77–84 (2012). https://doi.org/10.1145/2133806.2133826
98. Liu, B.: Sentiment analysis and opinion mining. Synth. Lect. Hum. Lang. Technol. **5**, 1–184 (2012). https://doi.org/10.2200/S00416ED1V01Y201204HLT016
99. Hutto, C.J., Gilbert, E.: VADER: A parsimonious rule-based model for sentiment analysis of social media text. In: Proc. 8th Int. Conf. on Weblogs and Social Media (ICWSM-14), Ann Arbor (2014)
100. Zipf, G.K.: Selected Studies of the Principle of Relative Frequency in Language. Harvard University Press (1932)
101. Molnar, C.: Interpretable Machine Learning. A Guide for Making Black Box Models Explainable. Creative Commons Attribution-Noncommercial-ShareAlike 4.0 International License (2020)
102. Jiang, S., Pang, G., Wu, M., Kuang, L.: An improved K-nearest-neighbor algorithm for text categorization. Expert Syst. Appl. **39**, 1503–1509 (2012). https://doi.org/10.1016/J.ESWA.2011.08.040
103. Buza, K., Nanopoulos, A., Nagy, G.: Nearest neighbor regression in the presence of bad hubs. Knowledge-Based Syst. **86**, 250–260 (2015). https://doi.org/10.1016/J.KNOSYS.2015.06.010
104. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: popular nearest neighbors in high-dimensional data. J. Mach. Learn. Res. **11**, 2487–2531 (2010)
105. Buza, K., Neubrandt, D.: A new proposal for person identification based on the dynamics of typing: preliminary results. Theor. Appl. Inf. **28**, 1–12 (2017). https://doi.org/10.20904/281-2001

106. Radovanović, M., Nanopoulos, A., Ivanović, M.: Nearest neighbors in high-dimensional data. In: Proc. 26th Annual International Conference on Machine Learning, pp. 1–8, ACM Press, New York (2009). https://doi.org/10.1145/1553374.1553485
107. Tomašev, N., Radovanović, M., Mladenić, D., Ivanović, M.: Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. Int. J. Mach. Learn. Cybern. **5**, 16–30 (2011). https://doi.org/10.1007/978-3-642-23199-5_2
108. Tomasev, N., Radovanović, M., Mladenić, D., Ivanović, M.: A probabilistic approach to nearest-neighbor classification. In: Proc. 20th ACM international conference on Information and knowledge management, ACM, Glasgow (2011). https://doi.org/10.1145/2063576.2063919
109. Radovanović, M.: High-Dimensional Data Representations and Metrics for Machine Learning and Data Mining (2011)
110. Tomašev, N., Rupnik, J., Mladenić, D.: The role of hubs in cross-lingual supervised document retrieval. In: Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 185–196, Springer, Gold Coast (2013). https://doi.org/10.1007/978-3-642-37456-2_16
111. Tomašev, N., Mladenić, D.: Class imbalance and the curse of minority hubs. Knowl.-Based Syst. **53**, 157–172 (2013). https://doi.org/10.1016/J.KNOSYS.2013.08.031
112. Ng, V., Cardie, C.: Weakly supervised natural language learning without redundant views. In: Proc. Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pp. 173–180 (2003)
113. Shao, J., Tu, D.: The Jackknife and Bootstrap. Springer, New York (1995)
114. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach. Learn. **36**, 105–139 (1999). https://doi.org/10.1023/A:1007515423169
115. Firte, L., Lemnaru, C., Potolea, R.: Spam detection filter using KNN algorithm and resampling. In: Proc. IEEE 6th International Conference on Intelligent Computer Communication and Processing, pp. 27–33 (2010). https://doi.org/10.1109/ICCP.2010.5606466
116. Orta, E., Ruiz, M., Hurtado, N., Gawn, D.: Decision-making in IT service management: a simulation based approach. Decis. Support Syst. **66**, 36–51 (2014). https://doi.org/10.1016/j.dss.2014.06.002
117. Ruiz, M., Moreno, J., Dorronsoro, B., Rodriguez, D.: Using simulation-based optimization in the context of IT service management change process. Decis. Support Syst. **112**, 35–47 (2018). https://doi.org/10.1016/j.dss.2018.06.004
118. Fielt, E., Böhmann, T., Korthaus, A., Conger, S., Gable, G.: Service management and engineering in information systems research. J. Strateg. Inf. Syst. **22**, 46–50 (2013). https://doi.org/10.1016/j.jsis.2013.01.001
119. Eikebrokk, T.R., Iden, J.: Strategising IT service management through ITIL implementation: model and empirical test. Total Qual. Manag. Bus. Excell. **28**, 238–265 (2015). https://doi.org/10.1080/14783363.2015.1075872
120. Galliers, R.D.: Towards a flexible information architecture: integrating business strategies, information systems strategies and business process redesign. Inf. Syst. J. **3**, 199–213 (1993). https://doi.org/10.1111/j.1365-2575.1993.tb00125.x
121. Saaty, T.L.: The analytic hierarchy and analytic network processes for the measurement of intangible criteria and for decision-making. In: Multiple Criteria Decision Analysis: State of the Art Surveys, pp. 345–405, Springer, New York (2005). https://doi.org/10.1007/0-387-23081-5_9
122. Webb, G.I., Sammut, C., Perlich, C., Horváth, T., Wrobel, S., Korb, K.B., Noble, W.S., Leslie, C., Lagoudakis, M.G., Quadrianto, N., Buntine, W.L., Quadrianto, N., Buntine, W.L., Getoor, L., Namata, G., Getoor, L., Han, Xin Jin, J., Ting, J.-A., Vijayakumar, S., Schaal, S., Raedt, L. De: Leave-One-Out Cross-Validation. In: Encyclopedia of Machine Learning, pp. 600–601, Springer US, Boston (2011). https://doi.org/10.1007/978-0-387-30164-8_469
123. Salzberg, S.L.: On comparing classifiers: pitfalls to avoid and a recommended approach. Data Min. Knowl. Discov. **1**, 317–328 (1997). https://doi.org/10.1023/A:1009752403260

124. Xi, X., Keogh, E., Shelton, C., Wei, L., Ratanamahatana, C.A.: Fast time series classification using numerosity reduction. In: Proc. 23rd Int. Conf. on Machine Learning (ICML), pp. 1033–1040, ACM Press, New York (2006). https://doi.org/10.1145/1143844.1143974
125. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory. **13**, 21–27 (1967). https://doi.org/10.1109/TIT.1967.1053964
126. Ng, A.Y.: Feature selection, L1 vs. L2 regularization, and rotational invariance. In: Proc. 21st Int. Conf. on Machine Learning (ICML), pp. 615–622, ACM Press, New York (2004). https://doi.org/10.1145/1015330.1015435
127. Cheng, Q., Varshney, P.K., Arora, M.K.: Logistic regression for feature selection and soft classification of remote sensing data. IEEE Geosci. Remote Sens. Lett. **3**, 491–494 (2006). https://doi.org/10.1109/LGRS.2006.877949
128. Pedrycz, W.: Granular Computing: Analysis and Design of Intelligent Systems. CRC Press (2018)
129. Pedrycz, W., Skowron, A., Kreinovich, V.: Handbook of Granular Computing. Wiley (2008)
130. Yao, J.T., Vasilakos, A.V., Pedrycz, W.: Granular computing: perspectives and challenges. IEEE Trans. Cybern. **43**, 1977–1989 (2013). https://doi.org/10.1109/TSMCC.2012.2236648
131. Bargiela, A., Pedrycz, W.: Toward a theory of granular computing for human-centered information processing. IEEE Trans. Fuzzy Syst. **16**, 320–330 (2008). https://doi.org/10.1109/TFUZZ.2007.905912
132. Pedrycz, W., Homenda, W.: Building the fundamentals of granular computing: a principle of justifiable granularity. Appl. Soft Comput. J. **13**, 4209–4218 (2013). https://doi.org/10.1016/j.asoc.2013.06.017
133. Pedrycz, W.: Allocation of information granularity in optimization and decision-making models: towards building the foundations of granular computing. Eur. J. Oper. Res. **232**, 137–145 (2014). https://doi.org/10.1016/j.ejor.2012.03.038
134. Pedrycz, W.: Granular computing for data analytics: a manifesto of human-centric computing. IEEE/CAA J. Autom. Sinica **5** (6), pp. 1025–1034 (2018). https://doi.org/10.1109/JAS.2018.7511213
135. Zipf, G.K.: Human behavior and the principle of least effort: an introduction to human ecology. Martino Pub (2012)
136. Diao, Y., Bhattacharya, K.: Estimating Business Value of IT Services through Process Complexity Analysis. In: Proc. IEEE Netw. Oper. Manage. Symp. (NOMS) (2008)
137. Paramesh, S.P., Shreedhara, K.S.: Automated IT service desk systems using machine learning techniques. In: Lecture Notes in Networks and Systems **43**, 331–346, Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-2514-4_28
138. Paramesh, S.P., Ramya, C., Shreedhara, K.S.: Classifying the unstructured IT service desk tickets using ensemble of classifiers. In: Proc. 3rd Int. Conf. on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), pp. 221–227 (2018). https://doi.org/10.1109/CSITSS.2018.8768734
139. Roy, S., Muni, D.P., Tack Yan, J.J.Y., Budhiraja, N., Ceiler, F.: Clustering and labeling IT maintenance tickets. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 829–845, Springer (2016). https://doi.org/10.1007/978-3-319-46295-0_58
140. Dasgupta, G.B., Nayak, T.K., Akula, A.R., Agarwal, S., Nadgowda, S.J.: Towards auto-remediation in services delivery: context-based classification of noisy and unstructured tickets. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 478–485, Springer (2014). https://doi.org/10.1007/978-3-662-45391-9_39
141. Agarwal, S., Sindhgatta, R., Sengupta, B.: SmartDispatch: Enabling efficient ticket dispatch in an IT service environment. In: Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining , pp. 1393–1401 (2012). https://doi.org/10.1145/2339530.2339744
142. Agarwal, S., Aggarwal, V., Akula, A.R., Dasgupta, G.B., Sridhara, G.: Automatic problem extraction and analysis from unstructured text in IT tickets. IBM J. Res. Dev. **61**, 41–52 (2017). https://doi.org/10.1147/JRD.2016.2629318

143. Szenkovits, A., Meszlényi, R., Buza, K., Gaskó, N., Ioana Lung, R., Suciu, M.: Feature Selection with a Genetic Algorithm for Classification of Brain Imaging Data. In: Stanczyk, U., Zielosko, B., Jain, L.C. (eds.) Advances in Feature Selection for Data and Pattern Recognition, pp. 185–202, Springer, Cham (2018)
144. Rizun, N., Taranenko, Y.: Simulation Models of Human Decision-Making Processes. Manag. Dyn. Knowl. Econ. **2** (2), pp. 241-264 (2014).