



Article

Performance Comparison of Machine Learning Algorithms in Classifying Information Technologies Incident Tickets

Domingos F. Oliveira ^{1,*} , Afonso S. Nogueira ² and Miguel A. Brito ^{3,*} 

¹ Department of Informatics and Computing, University Mandume Ya Ndemufayo, Lubango 3FJP+27X, Angola

² Department of Informatics, University of Minho, 4710-057 Braga, Portugal; afonsosalazar969@hotmail.com

³ Centro Algoritmi, Department of Information Systems, University of Minho, 4800-058 Guimaraes, Portugal

* Correspondence: dfilipe@umn.ed.ao (D.F.O.); mab@dsi.uminho.pt (M.A.B.)

Abstract: Technological problems related to everyday work elements are real, and IT professionals can solve them. However, when they encounter a problem, they must go to a platform where they can detail the category and textual description of the incident so that the support agent understands. However, not all employees are rigorous and accurate in describing an incident, and there is often a category that is totally out of line with the textual description of the ticket, making the deduction of the solution by the professional more time-consuming. In this project, a solution is proposed that aims to assign a category to new incident tickets through their classification, using Text Mining, PLN and ML techniques, to try to reduce human intervention in the classification of tickets as much as possible, reducing the time spent in their perception and resolution. The results were entirely satisfactory and allowed us to determine which are the best textual processing procedures to be carried out, subsequently achieving, in most of the classification models, an accuracy higher than 90%, making its implementation legitimate.



Citation: Oliveira, D.F.; Nogueira, A.S.; Brito, M.A. Performance Comparison of Machine Learning Algorithms in Classifying Information Technologies Incident Tickets. *AI* **2022**, *3*, 601–622. <https://doi.org/10.3390/ai3030035>

Academic Editors: Phivos Mylonas, Katia Lida Kermanidis and Manolis Maragoudakis

Received: 28 June 2022

Accepted: 18 July 2022

Published: 22 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: text mining; natural language processing; machine learning

1. Introduction

With the evolution of information technology (IT), it is necessary to decide if information systems (IS) departments should transpose this growth into the service desk, ticket management and maintenance fields, as these act as intermediaries between customers, employees and the IT department. Management and maintenance are based on a system where any employee of a company can report a problem by a textual description and selection of a category, which can be something related to a piece of equipment around their work scenario subject to a computer-intervention, with the possibility of assigning the responsibility for a resolution to one of the professionals in the IT department who specializes in problems of this nature, so that the employee can see their problem resolved as quickly as possible.

All tickets include information that characterizes a problem that has arisen during the services. In addition, the organization uses specific software for this purpose, where all tickets can be organized to allow appropriate management.

1.1. Problem Definition

To manage the tickets, the employee goes to the help desk platform, requests the creation of a key, details the subject and the problem category, assigns a priority and defines the incident category. The next step is to give the resolution to a support agent. The employee still has the option to describe the outlines of the occurrence to be resolved textually, and this process is completed with the ticket submission. However, not all users describe and categorize the problem they want to see solved correctly, often listing a problem category

that is totally in disagreement with the textual description they insert. They also do not assign the ticket to an IS professional; when they do, it is purely random and erroneous.

Thus, it becomes complicated for a technician to assess the core of the reported problem because they often deduce a solution according to the ticket category presented on the platform. When they check the ticket description, it denotes an inadequate characterization of the problem. Supposing the resolution is not assigned to a support agent, professionals must reassign the resolution to the correct technician, resulting in a delay in ticket resolution. Therefore, it would be interesting to implement a system that allows the submitted ticket to be recognized and forwarded to a specific IT professional with a history of solving problems of this nature.

1.2. Objectives

Considering the illustrated problem, we intend to propose a system that allows the analysis of the text entered by an employee in the description of a ticket, performing its classification using Machine Learning (ML) techniques and techniques of natural language processing so that it is automatically directed to a professional that is suitable to solve the problem to reduce the time lost by these professionals in the initial analysis of the ticket.

1.3. Methodologies

For the development of the project, we used CRISP-DM (Cross Industry Process for Data Mining) [1], used as a script in the practical component of this project, and DSR (Design Science Research) [2], which provides a set of metrics ideas for the research process.

1.4. Document Structure

This article is divided into five sections. The first is an introduction, presenting the framework, motivation, objectives and methodologies. The second section concerns the state of the art of the themes that are addressed, namely, the incident management systems, what they are for, how they work and the automation of their processes. In addition, concepts such as Text Mining and Text Categorization, Machine Learning and Natural Language are described. Then, all the steps performed in each phase of the CRISP-DM methodology are presented in detail. The data selection, processing, modeling and evaluation results are obtained. We also present a discussion and comparison of the text classification models in the fourth section. Finally, we present the conclusions of this project, difficulties and adversities encountered throughout this work, the results obtained together with the best techniques used and a point related to any future work.

2. Literature Review

We engaged in a search regarding the research process carried out in this work, as well as the enunciation, clearly and objectively, of all the information collected and studied on the topic of this article, including the details of numerous concepts related to the theme of the project, as well as work performed by other researchers and the analysis of the conclusions obtained by them, contributing to a more profound knowledge of the subject under study.

In this way, the literature review process for this work was carried out using a range of platforms and search engines with minimally recognized quality, which allowed access to a reasonable quantity of scientific articles, books, publications and relevant information for the bibliographical study. Of the platforms used, the ones that contributed the most information were Google, Google Scholar, IEEE Xplore, Research Gate, Science Direct and Scopus.

Studies related to the topic under study were searched. As a result, they had terms and words associated with the essential keywords of the document, namely Ticket Management, Incident Management Process, Text Mining, Text Mining in Incident Management and Text Categorization, Machine Learning, Automated Text Classification, Automated Incident Categorization, Natural Language Processing and Natural Language in Incident Management.

The criteria for the selection of an article were based on the title and abstract of the article/document and some keywords related to the topic under study. Since this stimulus did not occur very often, it was necessary to read beyond the article's abstract, validating the document's usefulness. In other cases, some exciting articles were found following the logic of identifying keywords. However, a specific subscription or license was required to obtain them. Here, registrations were made on those pages, and emails were sent to the authors of the desired documents, with relatively few responses. Next, research was conducted on the authors referenced in the found documents to combine knowledge, giving consistency to the acquired theory.

2.1. Information Technology Service Management

Information Technology Service Management (ITSM) is an approach to IT-related activities, with particular emphasis on its customers and employees and the proper maintenance of the quality levels of its services, allowing an organization's IT department to have a direct follow-up on their status [3].

In [4], the authors state that the main objective of ITSM is to maintain a climate of satisfaction regarding the services provided to customers, revealing the importance of maintaining service quality levels and ensuring sound management of IS department activities through transparent processes. Furthermore, as demonstrated by [5], regarding the state of available research on this topic, it was found that among the 21 ITSM sub-topics, one of the most popular was incident management, at 7.1% of works.

2.2. Incident Management Process

As defined in ITIL v3, "an incident is an unplanned outage of an IT service or a reduction in the quality of an IT service. The failure of a configuration item that has not yet affected the service is also an incident". Therefore, the main objective of Incident Management (IM) is to detect all types of incidents (which in the context of the problem can be called a ticket) and, in case they exist, to restore the operations affected by them as soon as possible, mitigating the effects that can be felt in the business progress of the organization and the employee [6].

A support agent detects an incident through Event Management, which is a process that monitors all events that occur with IT equipment [6]. At this stage, the Incident Management Process (IMP) begins by creating a ticket for an employee, technician or administrative staff, and IT professionals become aware of this through the monitoring platform. The most relevant practices of this process, according to [3,7], are incident detection (conducted through notices and triggers) and Incident Logging (reported via email, phone or through the help desk platform).

2.3. Text Mining

Text Mining consists of analyzing a vast, unstructured amount of text from various documents and extracting new information to answer specific questions. Thus, it provides basic pre-processing methods such as identification, the extraction of representative characteristics and mechanisms that allow the identification of complex patterns [8].

Regarding the organizational level, numerous manual tasks can be reduced by adopting its mechanisms, saving time and resources that support agents can focus on other activities. As one of these mechanisms, NLP is understood as a range of computational techniques that are used to analyze and represent textual descriptions at one or more levels of linguistic analysis to achieve a linguistic processing similar to what humans have for a set of tasks or applications [9].

With the application of Text Mining, it is possible to categorize, group, summarize and visualize texts. This requires going through a set of text selection, analysis and processing activities. For example, algorithms can convert much unstructured information into perfectly consumable data. This includes word pre-processing, which provides for Text Cleaning, Tokenization, Stop Word Elimination, Part of Speech Marking, Disambiguous

Sense (WSD) and Stemming. Feature Selection is another activity that aims to reduce the size of a dataset by removing features considered irrelevant for classification [10].

There are numerous techniques of Feature Selection, and some of the most used in projects of this nature are addressed here. For example, information gain, described by [11], calculates the number of bits of information guaranteed for the prediction of a category through the presence or absence of a term x in a given text document. It also allows researchers to select the terms that obtain the best results for information and can be defined as in [12,13].

The *Chi – Square*(x^2) statistical method is another technique that allows researchers to assess the independence between a term and its respective class [12]. Another technique is the Term Frequency–Inverse Document Frequency (*TF – IDF*), which is calculated by multiplying the term frequency (*TF*) by the inverse document frequency (*IDF*). Below is the *TF – IDF* calculation formula [11]:

$$TF - IDF = TF * IDF$$

We still have Text Transformation, in which, at this stage, a document is represented by the words they contain and their occurrences. This can follow two approaches, the Bag-of-Words or the Vector Space, as a process of selecting a subset of features to use in the creation of the models [14].

Thus, words with their respective weights can be seen as a model of Bag-of-Words (BOW) [15]. In this representation, of all the words that occur in a document, all their relations, whether semantic or syntactic, are removed [16]. For example, suppose there are two documents, where in the first it is written “The ticket was closed”, and in the second “The ticket is waiting for resolution”; then, Table 1 shows the occurrences of each of these words in the two documents.

Table 1. Illustrative representation of Bag-of-Words.

| | O “The” | Ticket | Foi “Was” | Fechado “Closed” | Está “It Is” | À “the” | Espera “Wait” | De “in” | Resolução “Resolution” |
|--------|---------|--------|-----------|------------------|--------------|---------|---------------|---------|------------------------|
| Text 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Text 2 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

Finally, each text is represented by a vector where each index refers to the occurrence count represented in Table 1.

- Vector text 1 = [111100000]
- Vector text 2 = [110011111].

Finally, we have an *N – Gram*, a probabilistic text model that assesses the dependency between words, with n referring to the number of words that aim to analyze this dependency relationship. An *N – Gram* is a sequence of n words *n – gram*. It can be a *2 – gram* (or bigram), which is a sequence of two words, such as “change toner”, “printer toner”, or “printer failed”. It can also be considered to be a sequence of three comments, which is a *3 – gram* (or trigram), such as “changing printer toner” or “printer toner failed” [17].

2.4. Machine Learning

The concept of ML is an idea that is significantly associated with Artificial Intelligence (AI) and can be considered one of its resources, justifying the media coverage that it has been obtaining and the possible competitive advantage that it can bring to organizations and their business.

ML is a form of AI that allows a system to learn from a set of data rather than by explicit programming. Doing so requires defining a complete and correct algorithm for that same task and then programming it into the equipment [18]. ML algorithms can be categorized in two ways: supervised or unsupervised.

Some supervised learning algorithms [19] applied in this project are Multinomial Naive Bayes (MNB) [20], Support Vector Machines (SVM) [21], Logistic Regression (LR) [22], Random Forest [22,23] and K-Nearest Neighbors (KNN) [24]. The last of the algorithms presented here is Stochastic Gradient Descent (SGD) [25]. In the data to be studied, the target variable must be categorized (labeled), usually correctly, to make an accurate “learning” so that the predictions can be as accurate as possible.

Evaluation Metrics

Another essential detail is the evaluation metrics used for classification problems. The first evaluation metric presented is Cross-Validation, a resampling technique used to evaluate ML models according to [25].

The confusion matrix is another evaluation metric. Each cell described in the confusion matrix totals the number of predictions made by a specific algorithm. A confusion matrix is a table of size $n \times n$ where it is possible to check the actual and predicted forecast, where n is equal to the number of classes [26].

Consider that a given algorithm, where n is 2, needs to predict 0 or 1. Assuming 1 is deemed to be positive and 0 is negative, there will be four different types of metrics:

- True Positives (VP)—the algorithm predicts 1 where it should predict 1.
- False Positives (FP)—the algorithm predicts 1 where it should predict 0.
- True Negatives (VN)—the algorithm predicts 0 where it should predict 0.
- False negatives (FN)—the algorithm predict 0 where it should predict 1.

Table 2, where the confusion matrix is represented, is based on these metrics [26]:

- Acuity/Accuracy—used to measure the proportion of correct predictions over the total number of instances assessed.
- Precision/Precision—used to measure positive patterns predicted correctly from the total of patterns predicted in a positive class.
- Sensitivity/Recall—used to measure the fraction of positive patterns that are correctly classified.

Table 2. Exemplary confusion matrix.

| Actual/Foreseen | 1 | 0 |
|-----------------|--------------------|--------------------|
| 1 | True Positives TP | False Positives FP |
| 2 | False Negatives FN | True Negatives TN |

We also have the F1-score, a metric used when the best accuracy and recall are desired simultaneously and this helps to understand the accuracy of the classifier. The higher the value, the better the performance of the model. In other words, it symbolizes the balance between the two metrics listed above [27].

- Averaged Precision—this metric calculates the average precision per class.
- Averaged Recall—this metric calculates the average recall per class.
- Averaged F-Measure—this metric calculates the average F-Measure by class.

Another metric is support, which represents the current number of occurrences for a given class in the data set. An imbalance reported by this metric in the training data set may indicate structural gaps in the scores obtained by the algorithm. It may require new data manipulation to avoid the stated imbalance. Its main objective is to diagnose the evaluation process.

The project has a data mining variant, where the solution involves a text classification mechanism using Natural Language Processing (NLP) and ML techniques. It was decided to adopt CRISP-DM for the practical part and DSR as the Information Systems (IS) research methodology. There is a connection between these two methodologies, as illustrated in Table 3.

In Table 3, we see some interconnected points. However, for the practical side of this project, CRISP-DM is much more efficient and valuable.

Table 3. CRISP-DM vs. DSR.

| CRISP-DM [1] | DSR [2] |
|---------------------------|--|
| 1. Business Understanding | 1. Problem Identification and Motivation |
| 2. Understanding the Data | 2. Definition of objectives |
| 3. Data Preparation | 3. Design and Development |
| 4. Modeling | 4. Demonstration |
| 5. Evaluation | 5. Evaluation |
| 6. Implementation | 6. Communication |

3. Text Classification Using CRISP-DM

The project arose from the need to include information that characterizes a problem during an employee’s services to the company in the context of support tickets.

3.1. Infrastructure

In the realization of this project, the tools provided by the organization were used as it is part of the oil industry, where it produces, sells and provides technical assistance for specific equipment, with thousands of employees in several sectors and departments, thus increasing the probability of the occurrence of a problem reported through the help desk portal. Therefore, all phases of CRISP-DM are detailed in a text classification project.

3.2. Business Understanding

Currently, the organization has a platform where its employees can report various technological factor-related incidents, often miscategorizing them. However, the textual description is sometimes clear about the real problem. Therefore, when the technicians deal with the ticket, they select the incidents to be resolved by the category defined by the employee. However, when analyzing the description, they realize that the class does not match the detailed description, making the problem difficult to solve.

To this end, after a set of processes, it is necessary to identify which are the best word processing techniques to be carried out and determine which ML models contribute best to the classification of the text, which is the ultimate purpose of the project. These steps are summarized in Figure 1.

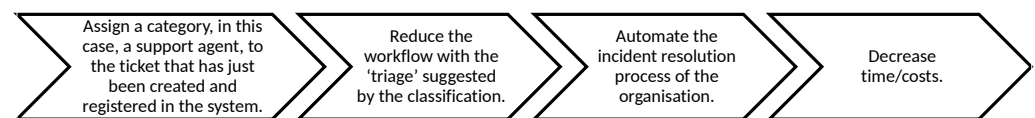


Figure 1. Ideal Ticket Classification System.

3.3. Data Understanding and Preparation

To carry out this project, the IT team of the company in which the project was carried out provided a dataset with 8837 rows and 12 columns, with each row referring to a ticket, amounting to a total volume of 5.3 MB and containing a significant amount of information of tickets registered from March 2018 to February 2020, as shown in Table 4.

Table 4. Distribution of tickets by dataset, taking into account the period.

| Dataset by Language | Record Period | Number of Tickets Registered |
|---------------------|--------------------------------|------------------------------|
| Portuguese Dataset | 12 March 2018–12 February 2020 | 4881 |
| Spanish Dataset | 13 March 2018–12 February 2020 | 1620 |
| English Dataset | 4 May 2018–11 February 2020 | 930 |
| Total | | 7431 |

As the organization has departments in various parts of the globe, three languages were identified in the ticket descriptions. Three different text detection methods were applied to divide the datasets by language to make the segregation as accurate as possible. As a result, there are three datasets: one in Portuguese, one in Castilian and one in English.

Concerning the IT agents, the main ones are located in Portugal, corresponding to a group of six professionals: “Agent 1”, “Agent 2”, “Agent 3”, “Agent 4”, “Agent 5” and “Agent 6”. All of these statements support the entire organization, regardless of language. However, some technicians only help a specific language, such as in the case of English, in which technicians “Agent 7” and “Agent 8” only work.

Some attributes are not relevant for the classification and ticket routing process. However, from Table 5, the columns we consider and what each one represents are presented in more detail.

Table 5. Description of columns present in the dataset.

| Column | Type | Description |
|-----------|--------|--|
| AgentName | string | Indicates the Information Agent to which the ticket was assigned |
| Text | string | The column derived from the junction of the subject and description columns. |

The most important columns for the intended analysis are text and AgentName, given that, in the scenarios envisaged by the author, the remaining columns now do not translate anything of extreme importance for the intended classification.

3.3.1. Portuguese Language Dataset

Since most of the tickets were registered in Portugal and the central IT department is in that same country, the dataset whose language was detected in the tickets was mainly Portuguese. This dataset has a size of 2.6 MB, with 4881 rows and 12 columns.

However, the fact that there are numerous occurrences for this type of ticket does not mean that the time spent resolving incidents of this nature is high. For example, the technicians “Agent 2” and “Agent 3” hold most of the project type tickets; although their occurrence is much shorter, the resolution time is much longer. It is also important to mention that the technician “Agent 1” holds by far the largest number of tickets, as one of the technicians with the most years in the information technology department—a factor that accentuates the difference in the occurrence of tickets for the other technicians, namely the younger ones, “Agent 5” and “Agent 6”.

3.3.2. Spanish Language Dataset

This dataset has a total volume of 1 MB, with 1620 rows and 12 columns. This data set represents all tickets registered in the period stated in departments in Spain or Mexico. Regarding support agents, only the same nine technicians were removed as in the Portuguese-language dataset for the same reasons.

There is a connection between the technician “Agent 1” and the SAP ticket type since he is responsible for resolving most tickets of this nature. Once again, this technician rose above the others, presenting a large distribution. Given the reduced number of incidences for this dataset and the English language dataset, oversampling was not as significant as

with the Portuguese language dataset. Nevertheless, it is interesting to analyze the behavior of the majority (agent) classes.

3.3.3. English Language Dataset

This dataset has a total volume of about 0.6 MB, with 930 rows and 12 columns. This dataset represents all tickets registered in the period stated in departments in the United Kingdom or India. The same two technicians were removed again as in the previous datasets regarding the support agents. Still, two more technicians were not included in the Portuguese and Castilian languages datasets because they are employees who provide support only in the United Kingdom and India.

3.4. Data Processing

For manipulation, the languages R and Python were used. Over an early and archaic analysis of the data, the language detection process for the eventual splitting of the dataset by language and some additional transformations such as removing non-relevant technicals, with R, it became very intuitive to perform these initial tasks. A dedicated data modeling library, scikit-learn, was used for the modeling phase, which provides supervised and unsupervised machine learning algorithms, contributing many auxiliary methods such as cross-validation and feature selection.

The following are the set of steps taken regarding data processing:

- Converge columns—concerns the union of the subject and note columns, transforming them into a new column called text.
- Removal of non-relevant technicians—the purpose of this is to remove the records attributed to agents not in the organization.

Language detection and dataset division—three packages were imported in R, namely `cld2`, `cld3` and `textcat`, so the eventual language detection could be as accurate as possible. The first package, based on a Naive Bayesian classifier, uses a neural network model for language identification. Finally, the `textcat` package uses n -gram text categorization to identify the language. After this process, it is concluded that the `cld2` detection algorithm obtained better results. Finally, the dataset was divided into three new datasets per language.

Remove null values—when analyzing the dataset, some fields with Na (null) values were found in the text column that would undoubtedly have to be removed since it does not contain terms to be processed.

Select essential columns for the classification—as mentioned previously, only a few columns were considered necessary for the type to be carried out, with these being the category—in this case, the support agent—and the textual description of the ticket incident.

Random oversampling—considering that some machine learning algorithms do not handle unbalanced data well, the decision was made to multiply the records of agents with few occurrences in the dataset to obtain a balanced distribution.

Categorical coding—a numeric label was assigned to each support agent, as a categorical text variable (string), as this is necessary for the statistical method called Chi-square, as already noted. This is also mentioned in another phase of this processing as one of its parameters.

Word Processing

Most ML algorithms, including those implemented in this project, need textual data to be converted into a fixed-size vector format. That is, the classifiers require vectors of numerical features because that is the only way that it is possible that they can consume them; hence, the specific step of manipulating the text is so important that the implementation of the Machine Learning algorithms is conceived.

There are therefore several approaches to consider to extract the best features from the text. One of the solutions would be to choose the Bag-of-Words (BOW) model. However, the selected approaches were Term Frequency (TF) and Inverse Document Frequency (IDF). For this purpose, as previously mentioned, a library indicated for machine learning tasks,

scikit-learn, was used with the `sklearn.feature_extraction.text.TfidfVectorizer` component to estimate, as the name suggests, the exact vector representation of the $TF - IDF$ for each textual description of tickets to perform the general cleaning of the data:

- Remove HTML tags—After analyzing the data and the specific ticket field, some HTML tags were removed.
- Convert to lowercase—All words were converted to lowercase.
- Remove special characters and punctuation—The removal was performed by replacing the unique characters and punctuation with blank spaces because, after removing these symbols, some words were combined.
- Removing blanks—This step is essential as the words are analyzed one by one; since a blank space can be considered a word and since it is not of significant importance, it is imperative to remove them.
- Remove stopwords—Stopwords are words that have no relevance to the intended analysis. These can be words like “de” (“of”), “o” (“the”) and “a” (“the”), among others (prepositions, articles), which occur very often in the descriptions of tickets. Therefore, they are removed.
- Stemming—This refers to reducing a word to its root. In the case of the word “amigo” (male friend), it would be reduced to “amig”. So, it fits “amigo” (male friend), “amiga” (female friend) and other words of the same family.
- Tokenization—This refers to the phase of dividing a text into a set of tokens; in this case, each word in that text would represent a token.
- TfidfVectorizer—All readers are converted to an array of $TF - IDF$ features. What differentiates this process from CountVectorizer is that this last considers the number of times a word occurs in a document. In contrast, TfidfVectorizer considers the available weight in a document. By counting the occurrences of a word in a document, the importance of each word is calculated. Thus, it is convenient to detail the parameters defined in the specified method, `sklearn.feature_extraction.text.TfidfVectorizer`:
 - `underline_tf`: This causes a logarithmic increase in the $TF - IDF$ score compared to the frequency of a specific term. It arises in the problem in which, for example, there are 15 occurrences of a word in a text, but that does not mean that they are more critical than 1 occurrence of that term. In this case, this parameter is set to True.
 - `min_df`: When building a vocabulary, terms that have a TF strictly below the given limit are ignored. In this case, this parameter is provided a value of 5. This is because a word needs to be considered in at least 5 textual descriptions of a ticket.
 - `norm`: To reduce bias in the length of the document, this standardization parameter is used. At the same time, the $TF - IDF$ score of each term scales proportionately, based on the total score of that document.
 - `ngram_range`: This parameter represents the lower and upper limits of the value range n for different n -grams extracted from the text. The values of n to be used are between $min_n \leq n \leq max_n$.
 - `stopwords`: Defined as Portuguese, Spanish, or English, depending on the dataset being processed.

Table 6 shows the number of words for each computer agent in each dataset.

Table 6. Number of words per document for each computer agent.

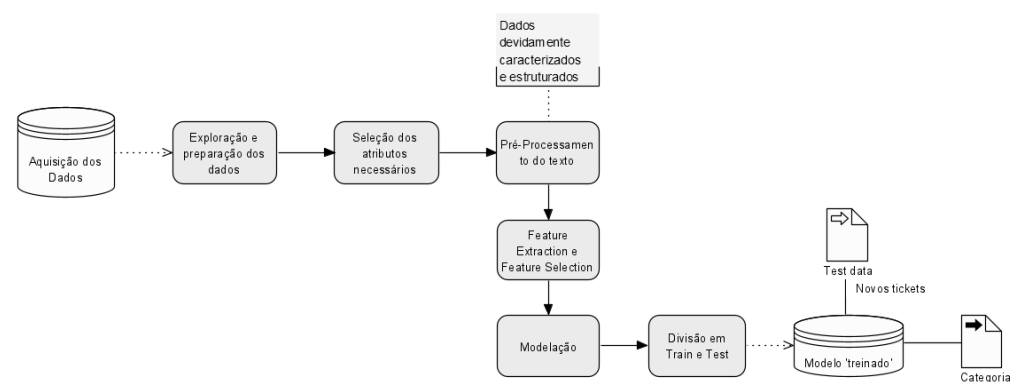
| | Portuguese | Spanish | English |
|-----------|------------|---------|---------|
| "Agent 6" | 46,436 | 6012 | 0 |
| "Agent 3" | 107,396 | 21,360 | 5160 |
| "Agent 1" | 98,748 | 48,510 | 5160 |
| "Agent 7" | X | X | 21,845 |
| "Agent 5" | 48,114 | 11,360 | 492 |
| "Agent 2" | 63,357 | 20,330 | 4911 |
| "Agent 8" | X | X | 4625 |
| "Agent 2" | 67,898 | 29,997 | 7467 |
| Total | 431,969 | 137,569 | 50,539 |

According to Table 6 above, the set of documents (corpus) in the Portuguese language totals 431,969 words, Castilian totals 137,569 and English totals 50,539. The text preprocessing steps are central to the modeling phase. The more relevant words are selected, the more data is provided to the algorithms that can facilitate and adjust this modeling. For each set of documents for each existing language, a specific linguistic tool for programming in Python called Natural Language Toolkit (NLTK) was used, which presents a predetermined set of stopwords in several languages, selected in Portuguese, Castilian and English, for the removal of irrelevant words. Then, as stipulated, the minimum number of documents present to remove the terms that did not meet this standard to perform $TF - IDF$ using TfidfVectorizer was selected.

3.5. Modeling

In the most critical phase of the project, several modeling techniques were applied, divided into train and test sets, and the models were built and all parameters configured and calibrated.

As shown in Figure 2, a set of critical processes was required to achieve the desired objective. First, a collection of data was analyzed to understand the key attributes for the scenarios in question, with their respective selection. The text was then processed in a set of steps to be detailed. This was followed by character extraction and selection, where the predictive elements were selected to build the model. Finally, in the modeling stage, the data were split, using the test data as an input to predict the expected categories.

**Figure 2.** Illustration of the set of processes required for modeling.

3.5.1. Feature Extraction and Feature Selection

It is possible to state that when implementing the TfidfVectorizer method, the feature extraction is pre-started, where the n -gram model is put into practice. This step converts all texts into an array of $TF - IDF$ features.

Imagine that the application of this method results in a table where the lines represent each analyzed sample and the lines show the words found. If a word appears once in an

example, a 1 will appear; otherwise, a 0. What differentiates this process from CountVectorizer is that the latter considers the number of times a word occurs in a document. In contrast, TfidfVectorizer is regarded as a word's general weight in a document. Counting the occurrences of a word in a document calculates each term's importance, facilitating the analysis of words with similar frequencies.

At this stage, it is essential to reduce the dimensionality of the feature. It is crucial to reduce the dimensionality of the feature space of each text document, which translates into the cluster of unique words that occur in the text documents. In Table 7, as an example, the unigrams and bigrams that most correlate with each category are represented in the Portuguese dataset.

Initially, a numerical vector representation of the textual description of the ticket was achieved using the weighted vectors of the Term Frequency–Inverse Document Frequency technique. With the exact representations, it is possible to train some classification methods to predict the category of some tickets that are not included in the dataset with a string that is merely exemplary of a description of an incident ticket and it is possible to predict the computer agent indicated for the same ticket.

Table 7. Most frequent unigrams and bigrams by computer agent.

| Category | Unigrams | Bigrams |
|-----------|--------------------------|--------------------------------------|
| "Agent 6" | "install" "project" | "replace PC" "I need to install" |
| "Agent 3" | "bi" "sgcp" | "him him" "power bi" |
| "Agent 1" | "crm" "error" | "error, error" "error crm" |
| "Agent 5" | "blocked" "system" | "blocked system" "system account" |
| "Agent 2" | "to set up" "ti" | "times ti" "collaborative portal" |
| "Agent 4" | "polycom" "checklist" | "confirm given" "form checklist" |

After all the data processing, the features and labels are extracted. Thus, numerous classification methods are available for the problem in question because of this scenario. As seen in the XSEDE Ticket System example listed in the Related Works, the Multinomial NB method is used for classification with discrete features, taking into account the frequency of the words in the documents and their sequences that improve the type; that is, in terms of the evaluation of the frequency of words, this is probably the best method. Thus, some predictions can be made with random strings representing an eventual incident ticket.

First, the dataset was divided into train and test sets using the *train_test_split* method, where the parameters were the data to be consumed and the respective labels. The *random_state* parameter was set to 0 to ensure that the generated divisions were reproducible. The scikit-learn library uses random permutations to generate the divisions. The given state, in this case, 0, is seen as a seed for the random number generator, ensuring that these numbers are generated in the same order.

Then, the train data were transformed into *TF – IDF* vectors, where for each line, there was an impressive list of integers associated with the weight calculated by *TF – IDF*.

The CountVectorizer method from the sci-kit-learn library counts the number of times a word occurs in a document, prioritizing those that occur more often and ignoring unusual words that could even provide some importance in processing the data with more efficiency.

Thus, a model was built using the train set in the classification method MultinomialNB, as already referenced, to make some predictions.

3.5.2. Selecting the Ideal Model

Considering some of the analyses performed, some ML algorithms showed good performance. Therefore, the goal of testing different categories of algorithms is to understand the difference in their respective implementations, deduce the core of potential problems related to unexpected results in their evaluation metrics and conclude the best solution for this case study.

We consider five scenarios to evaluate the influence that each one of them may have on the algorithms' performance. For the Portuguese dataset, with a significant difference in sample size compared to the other two datasets, six scenarios were considered, while for the other two sets, only one scenario was considered.

Regarding the experiments for the Portuguese dataset, in the first scenario (C1), stopwords were removed, stemming was applied to each word considered a unigram or bigram, and the oversampling in the dataset was discarded, taking into account the imbalances. In the second scenario (C2), only the oversampling process was applied. In the third scenario (C3), only the oversampling process was removed. In the fourth scenario (C4), the stopword removal and stem application tasks were not applied, maintaining the use of unigrams and bigrams. In the fifth scenario (C5), stemming was not involved, but only unigrams were considered. Finally, the sixth scenario (C6) was the same as in the fourth, but instead of unigrams, only bigrams were considered. Only Scenario 1 (C1) was considered for the remaining datasets since the amount of data was smaller. It was interesting to discard the oversampling process to evaluate the behavior of the majority classes at the level of occurrences, as shown in Table 8.

Table 8. Scenarios considered.

| Scenario (C) | N-Gram Variation |
|--------------|--|
| C1 | Unigrams + Bigrams + Stopwords + Stemming–Oversampling |
| C2 | Oversampling + Unigrams + Bigrams + Stopwords + Stemming |
| C3 | Oversampling + Unigrams + Bigrams + Stopwords |
| C4 | Oversampling + Unigrams + Bigrams |
| C5 | Oversampling + Stopwords + Unigrams |
| C6 | Oversampling + Stopwords + Bigrams |

First, the k -fold cross-validation method explained in the literary review was implemented. This allowed the dataset to be divided into k parts, with $k-1$ parts used for training and the rest for testing, repeating until k times were completed. The *sci-kit-learn* library was used for cross-validation, where it was defined that the parts (folds) to be divided would be $k = 5$.

Considering the standard algorithms, it was decided to perform a performance comparison between the following classification methods: Random Forest Classifier, Linear Support Vector Machine, MultinomialNB, Logistic Regression, K-Neighbors Classifier and Stochastic Descending Gradient Classifier. Regarding the categories of SVM methods that the *sci-kit-learn* library provides, two different types were used: LinearSVC and SGDClassifier.

Each ML model needed to be parameterized to adjust its behavior to a particular problem. Table 9 illustrates this.

Table 9. Parameterization carried out.

| Algorithm | Settings | Value |
|-------------------------|--------------|---------------|
| Random Forest | N_estimators | 200 |
| | Criterion | gini |
| | max_depth | 3 |
| | random_state | 0 |
| LinearSVC | loss | squared_hinge |
| | multi_class | ovr |
| | max_iter | 1000 |
| | Kernel | rbf |
| Multinomial Naive Bayes | Alpha | 1.0 |
| | Fit_prior | true |
| | Class_prior | None |
| | random_state | 0 |
| Logistic Regression | penalty | l2 |
| | max_iter | 200 |
| | n_neighbors | 5 |
| KNN | weights | Uniform |
| | max_iter | 100 |
| SGDC | penalty | l2 |
| | Loss | squared_hinge |

To be considered feasible models, some criteria were stipulated, taking into account the existing knowledge on the topic of text mining and text classification:

- Acuity/accuracy must be greater than 80%.
- Precision/precision must be greater than 75%.
- Sensitivity/recall must be greater than 80%.
- The error rate should not exceed the value of 20%.

It is noted that the higher the accuracy, the better, since we are in a project that aims to compare the performance of the different models used.

3.6. Evaluation Methods

The results of the tests specified in the previous phase are presented and discussed in this section. The Portuguese language dataset shows the results obtained in all scenarios. In the first part, the performance of each algorithm in each scenario is represented after cross-validation at the time of model selection. Then, all the results of the predictions of the selected model or models are evaluated (those that present better results) through the analysis of confusion matrices and the previously mentioned metrics (precision, sensitivity, F-score). Finally, the results of the dataset in Spanish and English are presented, for which only one scenario was used.

3.6.1. English Dataset

Table 10 shows the insight obtained from each model selected in the cross-validation.

Table 10. Acuity and standard deviation obtained in each scenario.

| Algorithm Scenario | Average Acuity | | | | | | Standard Deviation | | | | | |
|--------------------|----------------|--------|--------|--------|--------|--------|--------------------|------|------|------|------|------|
| | C1 | C2 | C3 | C4 | C5 | C6 | C1 | C2 | C3 | C4 | C5 | C6 |
| KNN | 58.21% | 81.5% | 81.26% | 81.16% | 80.85% | 83.11% | 4.67 | 6.92 | 6.99 | 7.3 | 5.99 | 2.98 |
| LinearSVC | 65.09% | 97.35% | 97.51% | 97.6% | 94.17% | 90.57% | 4.01 | 0.41 | 0.98 | 0.35 | 0.65 | 0.91 |
| LR | 65.73% | 91.73% | 92.21% | 92.55% | 88.95% | 87.48% | 4.64 | 0.49 | 0.52 | 0.45 | 0.55 | 0.98 |
| MNB | 63.35% | 86.48% | 87.35% | 87.63% | 82.44% | 83.21% | 3.7 | 0.36 | 0.39 | 0.38 | 0.78 | 0.52 |
| RF | 40.51% | 29.36% | 29.94% | 29.96% | 30.51% | 27.81% | 2.8 | 1.05 | 1.47 | 1.28 | 1.23 | 0.8 |
| SGDC | 65.32% | 92.74% | 93.14% | 93.46% | 89.18% | 87.21% | 4.56 | 0.61 | 0.46 | 0.36 | 0.59 | 1.04 |

As shown in Table 10, the Linear Vector Support Classifier performed better than the others, with the Logistic Regression and Stochastic Gradient Descent Classifier models obtaining exciting results. This analysis, in general, is the same for all scenarios except the first. To maintain a balance between the number of standard processes referring to word processing and the accuracy obtained, it was decided to consider scenario 3 (C3), which corresponds to this middle ground.

The average accuracy of the performance obtained by the models selected in the cross-validation, taking into account $k = 5$, on the data destined for the train set, can be seen in Figure 3.

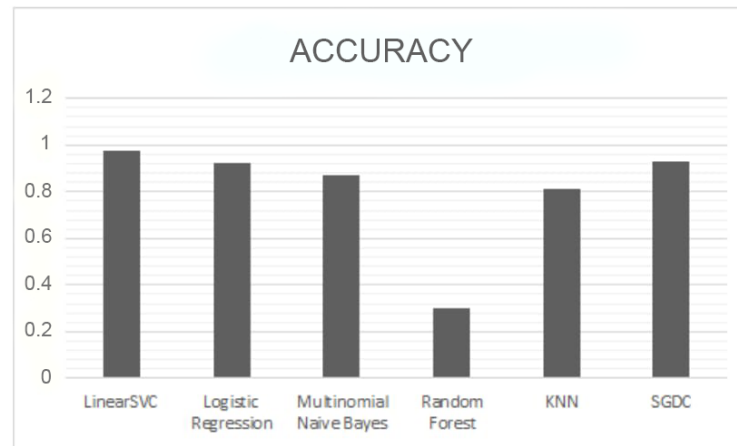


Figure 3. Accuracy of each model after cross-validation.

Analyzing Figure 3, the difference between the Random Forest algorithm and the others that obtained relatively similar results is clear. As an example, in Figure 4, the representation of the diagrams of extremes and quartiles alludes to the performance obtained by the models in the cross-validation in Scenario 3 (C3).

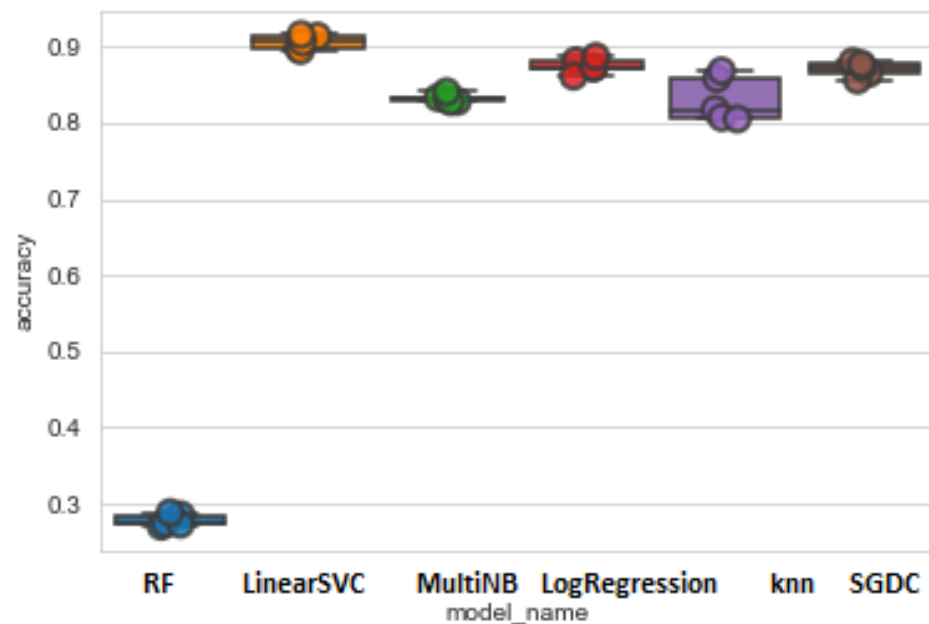


Figure 4. Diagram of extremes and quartiles (C3).

In Figure 4, as well as the diagram it represents, it is possible to observe the best performance obtained by LinearSVC.

3.6.2. Making Predictions

The analysis only took into account scenario 3 (C3) to maintain a balance between the number of standard processes alluding to the text processing used and the acuity obtained. In this way, for each model in scenario C3, 67% of the dataset was used for the training and the remaining 33% was used for testing. This was used for five-fold cross-validation for parameter estimation.

Following the standardized norms of data-mining projects, the dataset destined for training was, by chance, worked and "trained", and the data destined for the test were used to measure the performance of the models. The next step concerns the data fit used to train the models. From this moment, it is possible to obtain several metrics, the first being the confusion matrix, where it is possible to observe the number of categories that are correctly and incorrectly predicted, trying to understand their differences.

Importantly, metrics such as precision, sensitivity and F1-score, as previously referenced and detailed, can also be obtained after this process. Next, the results of the modeling precession are represented for each model.

Regarding the results obtained in each model after the modeling process, it is clear from Table 11 that the two similar models, LinearSVC and SGDC, presented the best results, with 93.12% and 90.01% accuracy, respectively.

However, the SGDC's modeling time was considerably longer, thus clearly winning over the other approach, LinearSVC. This is because LinearSVC tries to find the hyperplane, which can be considered a line that better separates the classes, maximizing the margin or distance between the closest points of the different courses.

Looking at the LR and MNB models, the two do not differ much in the results, obtaining accuracies of 88.65% and 85.03% , respectively. Still, the MNB model consumes the least time of all models, which is a very positive factor when it comes to saving time and resources.

Table 11. Model evaluation metrics.

| | Precision | Sensitivity | F1-Score | Support | Error | Acuity | Time(s) |
|-----------|-----------|-------------|----------|---------|--------|--------|---------|
| LinearSVC | 93.06% | 93.12% | 93.08% | 5137 | 6.87% | 93.12% | 1.1702 |
| LR | 88.63% | 88.65% | 88.57% | 5137 | 11.34% | 88.65% | 51.54 |
| SGDC | 89.87% | 90.01% | 89.93% | 5137 | 9.9% | 90.01% | 21.3412 |
| MNB | 85.36% | 85.03% | 84.81% | 5137 | 14.96% | 85.03% | 0.4999 |
| KNN | 81.58% | 63.42% | 64.32% | 5137 | 36.57% | 63.42% | 46.16 |
| RF | 68.95% | 29.21% | 20.38% | 5137 | 70.78% | 29.21% | 13.04 |

This situation arises because the MNB considers that each feature/word is conditionally independent, separating the occurrence of each word itself from the event of the other terms in a given ticket, which becomes computationally faster.

Looking at the performance of the RF model, we realize that it presents a significant difference between the precision obtained and the other metrics, which means that the probability of assigning a new incident ticket to a class that presents itself as the majority, even if by a substantial difference because there was an oversampling in the minority classes, is enormous, which shows that the RF proved to be very poor for this type of textual classification.

Following the best model in this scenario, we have represented in Figure 5 the normalized confusion matrix of LinearSVC:

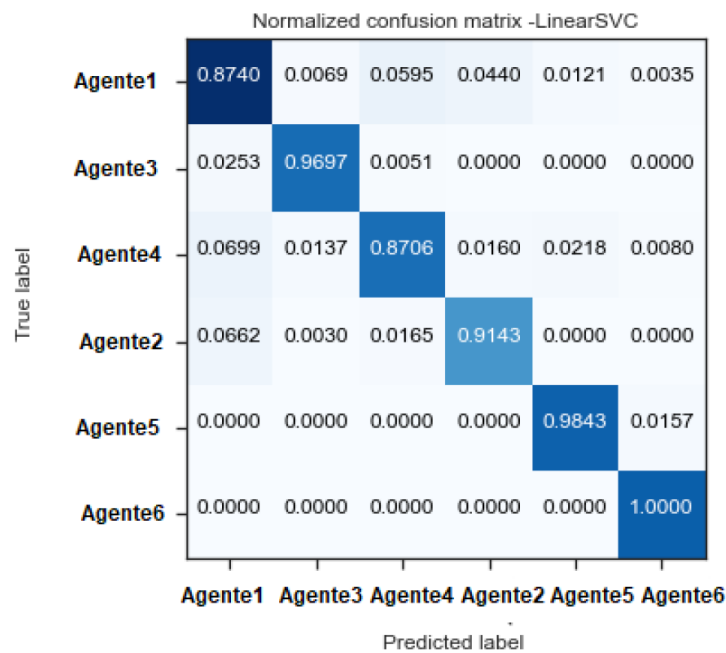


Figure 5. LinearSVC confusion matrix.

As we can see, the diagonal lines represent the percentage of times that the fundamental categories were correctly predicted. As we can see, the correct predictions exceed 90% in most cases. Not least, classes such as “Agente6”, “Agent3” and “Agent5”, which were initially minority categories, suffered a higher oversampling than the others, which means that the feature space is very consistent, which is reflected in the high percentage in these classes in this model.

3.6.3. Spanish and English Dataset

At this stage, it is essential to emphasize that only one scenario, scenario 1 (C1), was considered for these two datasets.

Analyzing Table 12, it is possible to verify that LinearSVC obtained better results. In general, compared with the Portuguese dataset, it can be quickly perceived that the sample size to be analyzed significantly influenced the understanding obtained, presenting a slight decrease.

Table 12. Acuity and standard deviation in the Spanish and English datasets.

| | Metric | LinearSVC | MNB | LR | RF | KNN | SGDC |
|---------|--------------------|-----------|--------|--------|--------|--------|--------|
| Spanish | Average Acuity | 82.67% | 81.04% | 82.1% | 78.97% | 80.1% | 81.35% |
| | Standard deviation | 1 | 1.4 | 1.4 | 0.03 | 1.1 | 1.9 |
| English | Average Acuity | 80.16% | 78.39% | 78.51% | 57.56% | 77.09% | 77.19% |
| | Standard deviation | 4.65 | 2.68 | 3.25 | 0.51 | 77.19 | 3.5 |

Following the logic of the presentation of results made in the Portuguese dataset, the results obtained in these two datasets are presented in Table 13.

Table 13. Spanish and English Dataset Metrics.

| | Algorithm | Precision | Sensitivity | F1-score | Acuity | Error Rate | Time(s) |
|---------|-----------|-----------|-------------|----------|--------|------------|---------|
| Spanish | LinearSVC | 79.17% | 81.93% | 78.77% | 81.93% | 18.06% | 2.19 |
| | MNB | 71.20% | 80.03% | 75.26% | 80.03% | 19.96% | 0.69 |
| | LR | 71.30% | 80.6% | 74.71% | 80.6% | 19.39% | 29.98 |
| | RF | 61.35% | 78.32% | 68.8% | 78.32% | 21.67% | 53.36 |
| | KNN | 78.99% | 81.17% | 76.81% | 81.17% | 18.82% | 28.68 |
| | SGDC | 77.75% | 80.41% | 78.76% | 80.41% | 19.58% | 24.88 |
| English | LinearSVC | 82.12 | 83.72 | 82.26 | 83.72 | 16.27 | 1.19 |
| | MNB | 78.29 | 77.4 | 74.17 | 77.4 | 22.59 | 0.79 |
| | LR | 79.10 | 78.4 | 75.52 | 78.4 | 21.59 | 17.68 |
| | RF | 41.77 | 57.14 | 42.76 | 57.14 | 42.85 | 46.47 |
| | KNN | 77.60 | 79.06 | 78.17 | 79.06 | 20.93 | 8.7 |
| | SGDC | 80.46 | 81.72 | 81.03 | 81.72 | 18.27 | 10.2 |

4. Discussion

At this point, the obtained results are discussed, examining the potential causes of the good or bad performance recorded in the most diverse methods studied, explaining the results in each scenario in the three different datasets and presenting the respective conclusions regarding the best strategy.

4.1. English Dataset

Comparing the times that a category was predicted correctly and incorrectly with confusion matrices is possible. As seen in the matrix, the y -axis represents the actual value, with the x -axis representing the predicted value. The diagonal line reveals the number of instances of a specific category; in this case, the IT agent was correctly predicted. On the other hand, the remaining cells represent the number of times a given category was predicted as another category.

This reveals some ambiguity in these categories. For example, in the confusion matrix of the LinearSVC model, the technician “Agent 1” was predicted as “Agent 4” 5.95% of the time, with the latter being predicted as “Agent 1” 6.99% of the time (with the highest percentages of cells referring to erroneous classifications), which demonstrates that both resolve tickets with very similar descriptions or, in other words, usually deal with incident tickets with the same type. This situation occurs numerous times for different categories.

Table 14 shows the performance metrics results of all the models listed above in the classification related to scenario 3 (C3).

Table 14. Metrics obtained, by class, in each model.

| Category | Metric | LinearSVC (%) | LR (%) | RF (%) | MNB (%) | KNN (%) | SGDC (%) |
|----------|-------------|---------------|--------|--------|---------|---------|----------|
| “Agent1” | Precision | 89.01 | 80.69 | 24.17 | 73.4 | 82.35 | 86.2 |
| | Sensitivity | 87.4 | 82.57 | 99.65 | 82.13 | 31.4 | 80.32 |
| | F1-Score | 88.2 | 81.62 | 38.9 | 77.52 | 45.47 | 83.16 |
| | Support | | | 1159 | | | |
| “Agent2” | Precision | 90.34 | 88.31 | 0 | 90.94 | 27.33 | 86.72 |
| | Sensitivity | 91.42 | 81.8 | 0 | 67.96 | 93.38 | 86.46 |
| | F1-Score | 90.88 | 84.93 | 0 | 77.7 | 42.28 | 86.59 |
| | Support | | | 665 | | | |
| “Agent3” | Precision | 97.21 | 95.21 | 93.56 | 91.07 | 92.2 | 93.25 |
| | Sensitivity | 96.96 | 90.53 | 23.86 | 87.62 | 50.75 | 95.95 |
| | F1-Score | 97.09 | 92.81 | 38.02 | 89.31 | 65.47 | 94.58 |
| | Support | | | 792 | | | |

Table 14. Cont.

| Category | Metric | LinearSVC (%) | LR (%) | RF (%) | MNB (%) | KNN (%) | SGDC (%) |
|----------|-------------|---------------|--------|--------|---------|---------|----------|
| "Agent4" | Precision | 90.04 | 84.8 | 100 | 81.71 | 85.66 | 84.89 |
| | Sensitivity | 87.05 | 80.52 | 5.15 | 73.19 | 29.43 | 84.3 |
| | F1-Score | 88.52 | 82.6 | 9.8 | 77.22 | 43.81 | 84.59 |
| | Support | | | 873 | | | |
| "Agent5" | Precision | 96.1 | 90.6 | 100 | 88.52 | 94.52 | 93.24 |
| | Sensitivity | 98.43 | 99.03 | 7.6 | 98.79 | 95.89 | 98.43 |
| | F1-Score | 97.25 | 94.63 | 14.14 | 93.37 | 95.2 | 95.76 |
| | Support | | | 828 | | | |
| "Agent6" | Precision | 97.15 | 95.87 | 100 | 92.91 | 96.81 | 97.01 |
| | Sensitivity | 100 | 99.14 | 5.9 | 99.14 | 100 | 99.14 |
| | F1-Score | 98.55 | 97.48 | 11.27 | 95.92 | 98.38 | 98.07 |
| | Support | | | 820 | | | |
| Acuidade | | 93.12 | 88.65 | 29.21 | 85.03 | 63.42 | 90.13 |

The model that obtained the best results in this scenario was LinearSVC. The LinearSVC model is based on the method of One vs Rest to find the hyperplane that best divides the classes, maximizing the distance between the closest points of different types.

Allied to this factor, this approach attempts to minimize an objective function through the parameters referring to the loss and penalty functions, which can justify the excellent performance and the similarity of the results with the Logistic model. Regression contains these two loss and penalty functions. Therefore, it is possible to affirm that LinearSVC and SGDC present similar results since they share a standard parameter, the loss function, defaulting the *'squared_hinge'*, and both are linear classification models. The weighted average was used to calculate the accuracy, sensitivity and F1-score.

The metrics obtained in each model are presented in Table 14 below. For example, the LinearSVC, Multinomial Naive Bayes, Logistic Regression, Random Forest, K-Nearest Neighbors and Stochastic Gradient Descent Classifier models achieved accuracies of approximately 93.12%, 85.03%, 88.65%, 29.21%, 63.42% and 90.13% Figure 6.

Once again, it should be noted that the LinearSVC model performed better overall, with the Logistic Regression and Stochastic Gradient Descent Classifier model being very close to each other, the Multinomial Naive Bayes coming second and Random Forest being clearly last.

No less critically, the LinearSVC and Multinomial Naive Bayes models presented a relatively low execution time, at 1.17 and 0.49 s, respectively, which is always a good indication of the time/resource ratio. However, LinearSVC showed better inaccuracy and error rates, at 6.87% and 14.96%, respectively, which is more critical in the end. Regarding the Logistic Regression and Stochastic Gradient Descent Classifier models, they have very close acuity results (88.65% and 90.13%, respectively), which ends up determining the difference in time between them, at 51.54 and 21.34 s, respectively, and the error rate, at 11.34% and 9.9% respectively, with the SGDC model being the best option. Finally, regarding the KNN and Random Forest models, it is evident that the accuracy of the first, 63.42%, is greater than that of the second, 29.21%, with the choice being maintained in the KNN, despite the execution time, as the RF is less than that of the KNN.

A final point to note is that the error rate (the lower, the better) increases as there is a decrease in acuity when analyzing the models, with LinearSVC having the lowest error rate and Random Forest having the highest.

Regarding the analyzed classes, if the balance in the distribution of tickets for each were not taken into account, it would be verified that the technician "Agent 1" would have a good performance in all the defined metrics, given that it was the one that presented the most resolved tickets. However, as already stated, an oversampling of these same classes was carried out to prevent the less represented types from being considered.

Since the technicians “Agent 6”, “Agent 5” and “Agent 3” were the least represented classes, they suffered a significant increase in tickets after the oversampling process, which had an effect that was reflected in the metrics presented in Table 14 above. However, as this process did not involve the addition of synthetic tickets but a mere multiplication of existing tickets, the number of standard features was much higher than in the other classes that were previously most represented, resulting in an overall more excellent understanding than in the others.

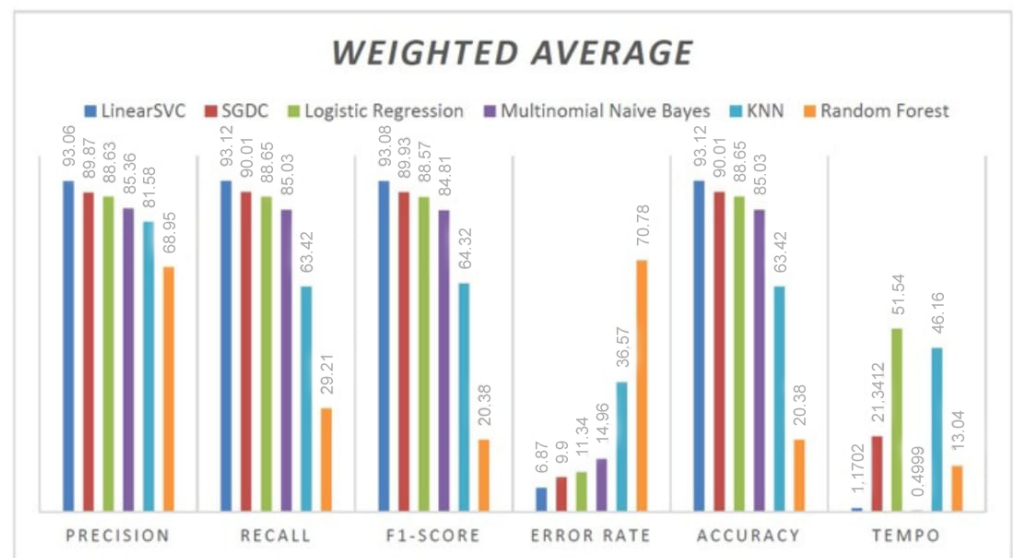


Figure 6. Weighted average final.

Using the LinearSVC model, it was possible to obtain some exciting predictions. To do so, the text to predict took into account the ticket category that the agents to be predicted usually solved, as well as the keywords used, as illustrated in Table 15.

Analyzing Table 15, it is concluded that the forecasts were practically all correct, excluding the last one, in which, despite “Agent 6” resolving tickets related to account blocking, technician “Agent 5” had a history of resolving tickets of this much more significant type.

Table 15. Some forecasts made.

| Possible Incident Ticket | Expected Computer Agent |
|--|-------------------------|
| “I need you to install a printer for me and replace the toner.” | “Agent 6” |
| “Power BI on my pc doesn’t work” | “Agent 3” |
| “I need you to cancel an order in SAP.” | “Agent 1” |
| “I need to recover a file that I ended up not saving to the PDM” | “Agent 4” |
| “What about the employee portal?” | “Agent 2” |
| “My account is blocked.” | “Agent 6” |

4.2. Portuguese Dataset Comparison (C3) vs. Castilian vs. English

As expected, the models obtained better results in the Portuguese dataset, given that it has a significantly larger size than the others, with LinearSVC obtaining a better result in the experiments carried out in all datasets and the Logistic Regression registering the second-best performance in the generality of experiences.

Another interesting point was the difference in the performance of the Random Forest model—despite being one of the models that had the worst performance, in the Castilian and English datasets, compared to the Portuguese dataset, the performance was not as different, because the datasets are more minor and because the imbalance of occurrences by each class is a positive factor for this model. There was a balance between models in the

datasets where oversampling was not applied. In Figure 7, the evaluation metrics for each dataset studied are represented.

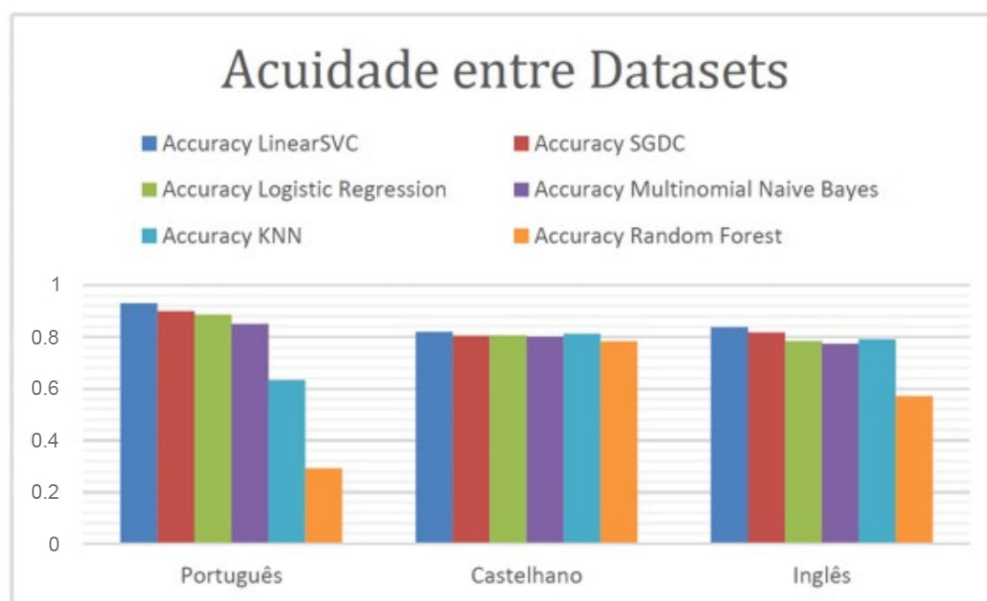


Figure 7. Comparison of test model accuracy between datasets analyzed.

Once again, it should be noted that, in general, LinearSVC obtained first place in terms of performance values.

5. Conclusions

There are numerous studies of text classification methods using ML techniques but only a few concerning the classification of tickets and their automatic assignment, with some solutions. This study made it possible to meet the central objective, proposing a ticket classifier system that uses ML techniques to automatically categorize new IT incident tickets.

The system has as its foundation the textual description and the computer agent that solves the ticket. In addition, DSR and CRISP-DM were used, forming the basis of this project's success.

Analyzing the results, for the dataset with the Portuguese language, it was possible to verify that the model that obtained the best results was the LinearSVC model (93.12% of accuracy), outperforming the SGDC and Logistic Regression models (90.01% and 88.65% accuracy, respectively), leaving the Multinomial Naive Bayes, KNN and Random Forest models as the last ones. From this point of view, it can be stated that the results support the studies already carried out in the scientific community.

It should be noted that oversampling positively affected the results obtained. The applications of several text processing techniques were crucial for the significant increase in the models' results. In the case of the Spanish and English datasets, the smaller sample size and the non-implementation of an oversampling category hurt performance.

With the application of a system that automatically classifies tickets, the analysis and resolution time would be drastically reduce, along with the costs related to the resources involved in the manual process of assigning keys to the IT agent. Future work could use other oversampling methods such as SMOTE (Synthetic Minority Oversampling Technique) and ADASYN (Adaptive Synthetic Sampling Approach). Another situation to test would be to compare the influence of taking into account more n -grams, as well as to discard the joining of the subject column and the textual description, compare the metrics separately and add the type, priority and subcategory column to the textual description. Another interesting point to study in the future would be the use of neural network architecture,

comparing three specific algorithms: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Hierarchical Attention Network (HAN).

Author Contributions: All authors who contributed substantially to the study's conception and design were involved in preparing and revising the manuscript until the final version's approval. D.F.O. and A.S.N. were responsible for bibliographic research, manuscript, data collection, and statistical analysis. M.A.B. Actively contributed to all parts of the article, including interpreting the text's data, revision, and approval. In addition, all authors contributed to the development of the data collection instrument. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by FCT—Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

Informed Consent Statement: Ethics approval is not applicable in this study.

Data Availability Statement: Not applicable in this study. However, the study data sets used or analysed are available in the manuscript tables.

Conflicts of Interest: The authors declare that there are no conflict of interest regarding the publication of this paper.

References

- Oliveira, D.F.; Brito, M.A. Development of Deep Learning Systems: A Data Science Project Approach. In *Information Systems and Technologies*; WorldCIST 2022. Lecture Notes in Networks and Systems; Rocha, A., Adeli, H., Dzemyda, G., Moreira, F., Eds.; Springer: Cham, Switzerland, 2022; Volume 469. [\[CrossRef\]](#)
- Hevner, A.R.; March, S.T.; Park, J.; Ram, S. Two Paradigms on Research Essay Design Science in Information Systems Research. *MIS Q.* **2004**, *28*, 75–79. [\[CrossRef\]](#)
- Ferreira, M.C. Incident Routing: Text Classification, Feature Selection, Imbalanced Datasets, and Concept Drift In Incident Ticket Management. Master's Thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, 2017.
- Iden, J.; Eikebrokk, T.R. Implementing IT Service Management: A systematic literature review. *Int. J. Inf. Manag.* **2013**, *33*, 512–523. [\[CrossRef\]](#)
- Shahsavarani, N.; Ji, S. Research in Information Technology Service Management (ITSM): Theoretical Foundation and Research Topic Perspectives. *CONF-IRM Proc.* **2011**, *30*, 1–17.
- Tang, X.; Todo, Y. A Study of Service Desk Setup in Implementing IT Service Management in Enterprises. *Technol. Investig.* **2013**, *4*, 190–196. [\[CrossRef\]](#)
- Silva, S.A.T.D. Automatization of Incident Categorization. Ph.D. Thesis, University Institute of Lisbon, Lisbon, Portugal, 2018.
- Gulo, C.A.; Rúbio, T.R.; Tabassum, S.; Prado, S.G. Mining scientific articles powered by machine learning techniques. *Open Access Ser. Inform.* **2015**, *49*, 21–28. [\[CrossRef\]](#)
- Hass, N.; Hendrix, G.; Hobbs, J.; Moore, R.; Robinson, J.; Rosenschein, S.; Moore, R.; Robinson, J.; Rosenschein, S. DIALOGIC: A Core Natural-Language Processing System. In Proceedings of the 9th Conference on Computational Linguistics, Prague, Czech Republic, 5–10 July 1982. [\[CrossRef\]](#)
- Forman, G. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *J. Mach. Learn. Res.* **2002**, *1*, 1289–1305. [\[CrossRef\]](#)
- Bahassine, S.; Madani, A.; Al-Sarem, M.; Kissi, M. Feature selection using an improved Chi-square for Arabic text classification. *J. King Saud Univ. Comput. Inf. Sci.* **2020**, *32*, 225–231. [\[CrossRef\]](#)
- Hussein, E.; Aliwy, A. Improving Feature Selection Techniques for Text Classification Esraa Hussein Abdul Ameer Alzuabidi. Ph.D. Thesis, University of Kufa, Kufa, Iraq, 2018.
- Al-harbi, O. A Comparative Study of Feature Selection Methods for Dialectal Arabic Sentiment Classification Using Support Vector Machine. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **2019**, *19*, 167–176.
- Justicia De La Torre, C.; Martín-Bautista, M.J.; Sánchez, D.; Vila, M.A. Text mining: Intermediate forms for knowledge representation. In Proceedings of the 4th Conference of the European Society for Fuzzy Logic and Technology and 11th French Days on Fuzzy Logic and Applications, EUSFLAT-LFA 2005 Joint Conference, Warsaw, Poland, 1 December 2005; pp. 1082–1087.
- George, K.S.; Joseph, S. Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature. *IOSR J. Comput. Eng.* **2014**, *16*, 34–38. [\[CrossRef\]](#)
- Langley, P.; Carbonell, J.G. Approaches to machine learning. *J. Am. Soc. Inf. Sci.* **1984**, *35*, 306–316. [\[CrossRef\]](#)
- Jurafsky, D.; Martin, J.H. Speech and language processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. *Rev. Soc. Econ.* **2018**, *40*, 76–78. [\[CrossRef\]](#)
- Vedala, D. Building a Classification Engine for Ticket Routing in IT Support Systems. Master's Thesis, Aalto University, Espoo, Finland, 2018.
- Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning*, 2nd ed.; Springer: New York, NY, USA, 2009. [\[CrossRef\]](#)

20. Hartmann, J.; Huppertz, J.; Schamp, C.; Heitmann, M. Comparing automated text classification methods. *Int. J. Res. Mark.* **2019**, *36*, 20–38. [[CrossRef](#)]
21. Mao, W.; Wang, F.Y. Cultural Modeling for Behavior Analysis and Prediction. *Adv. Intell. Secur. Inform.* **2012**, 91–102. [[CrossRef](#)]
22. Misra, S.; Li, H. *Noninvasive Fracture Characterization Based on the Classification of Sonic Wave Travel Times*; Elsevier Inc.: Amsterdam, The Netherlands, 2020; pp. 243–287. [[CrossRef](#)]
23. Belgiu, M.; Drăgu, L. Random forest in remote sensing: A review of applications and future directions. *ISPRS J. Photogramm. Remote Sens.* **2016**, *114*, 24–31. [[CrossRef](#)]
24. Maertens, R.; Long, A.S.; A.White, P. Performance of the InVitro TransgeneMutation Assay in MutaMouse FE1Cells: Evaluation of NineMisleading (“False”) Positive Chemicals. *Environ. Mol. Mutagen.* **2010**, *405*, 391–405. [[CrossRef](#)]
25. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv*, **2016**, arXiv:1609.04747.
26. Hossin, M.; Sulaiman, M.N. A Review on Evaluation Metrics for Data Classification Evaluations. *Int. J. Data Min. Knowl. Manag. Process* **2015**, *5*, 1–11. [[CrossRef](#)]
27. Meesad, P.; Boonrawd, P.; Nui pian, V. A Chi-Square-Test for Word Importance Differentiation in Text Classification Natural Language Processing Techniques and Application. View project Text Classification View project A Chi-Square-Test for Word Importance Differentiation in Text Classification. In Proceedings of the International Conference on Information and Electronics Engineering, Singapore, 6 January 2011 .