DiagnoseMe: Predictive Symptom2Disease Analysis

Carla Ardiaca Romero

Abstract— The purpose of this project is to diagnose a disease based on a symptom's text, done it by the patient. The methodology adopted for the project involves several stages, including data cleaning, preparation for model training, model selection, and analysis of results. The chosen dataset, Symptom2Disease, encompasses a wide range of symptoms and 24 different medical diagnoses.

The results section details the effectiveness of the selected model (Random Forest) in diagnosing diseases from textual symptoms. The analysis includes an examination of the TOP1 and TOP3 score and the use of LIME (Local Interpretable Model-agnostic Explanations) for deeper insights into the model's predictions. A graphical user interface developed with Tkinter is also mentioned, which allows doctors to input symptoms and receive the most likely diagnoses.

In conclusion, the study demonstrates the potential of NLP and machine learning in medical diagnostics, suggesting the possibility of extending this approach to a broader spectrum of diseases. It also highlights the need for further improvements, such as incorporating word embeddings and expanding the training dataset, to enhance the accuracy and generalization capability of the model. The use of LIME visualizations is emphasized as a valuable tool for understanding the contribution of each symptom to the model's predictions, aiding medical professionals in the diagnostic process.

Keywords— Predictive Analysis, NLP, Symptom Classification, Medical Diagnostics, LIME, Tkinter

1 INTRODUCTION

Nowadays, predictive analysis in the field of medicine has become a very important tool for the prevention, identification, and treatment of diseases. This article describes a technique based on natural language processing (NLP) and machine learning to predict diseases from texts detailing patients symptoms.

The motivation for this project has been to predict the three most likely diseases based on the patient's symptoms, thus helping the doctor diagnose the illness and make medical diagnoses faster and more accurate. The result? A system that takes descriptions of symptoms and transforms them into data that our models can understand and process.
For this project, a dataset called Symptom2Disease has been used, which includes a wide range of symptoms and 24 different medical diagnoses. This database has been chosen and prepared with the goal of making our model's predictions as accurate as possible.

2 METHODOLOGY

In this study, a methodical and detailed approach has been adopted for the creation of a disease prediction system. The process has been divided into several stages, each of which has been important to achieve the desired results. The stages of this project have been:

2.1 Data Cleaning

The initial data processing aims to homogenize the text and remove words that are meaningless for diagnosing diseases. It is important to detect and eliminate them to later train the model correctly. This process starts with transforming all the text to lowercase to eliminate any inconsistency due to the differentiation between uppercase and lowercase.

The next step has been the elimination of contractions. By simplifying the form of words to their full version, the text is homogenized. The decision to suppress punctuation marks has also been made, as the analysis focuses strictly on the semantic content of the terms and not on their grammatical structure.

Finally, the step of eliminating *stopwords* has been vital. These words, although common in language use, do not provide significant value to the analytics of symptoms and, therefore, have been excluded from the dataset. With these actions, it is ensured that the data is optimally prepared for the training of predictive models.

2.2 Data Preparation for Model Training

Once the data has been cleaned, the next step is to prepare it for model training. The dataset is divided into two subsets: one for training (*train*) and another for testing (*test*).

To train a model, the text must be converted into a format that machine learning algorithms can process, for this reason, the text needs to be transformed into a number vector. This is why, a dictionary of the 100 most common words is created, assigning each a unique identifier (*word:id*).

Next, *padding* is performed on the text sequences, standardizing their length to ensure that all have the same dimension. Finally, *one-hot encoding* is used, a technique that converts this number vector representing the text into a binary representation that facilitates the training of models.

Regarding the target variable, *label encoding* is applied to the disease labels (dependent variable y), transforming the nomenclatures of diseases into numerical values. This conversion is crucial for use in classification models, as they operate on numerical values and not on texts.

2.3 Selection and Training of Models

Several classification models have been trained to compare their performance. These models include Random Forest, Logistic Regression, Gradient Boosting, Gaussian Naive Bayes, and Bernoulli Naive Bayes. For each model, GridSearchCV is used for the selection of the best hyperparameters and to perform

Cross-Validation. The model with the best performance will be chosen to make predictions on the test subset.

2.4 Analysis of Results

Once the chosen model is trained, the testing phase is initiated. The test data is prepared following the same cleaning and transformation process as the training set. The metric chosen for the model classification has been the f1_weighted score. With these data, predictions are made to evaluate the f1-score accuracy of each disease in the model. The TOP 3 accuracy of the f1-score of each disease is also measured. In this project, a good result in TOP 3 accuracy is prioritized since in the medical field, several diseases can present similar symptoms, and it is important that the model has the ability to correctly identify the real disease among the main candidates so that a professional can then choose the true disease.

A Confusion Matrix has been made, and LIME Explainability applied to delve into the misclassified cases and better understand the characteristics of these misclassified cases. From the visualization of the misclassified diseases, words in the dictionary are manually replaced, and its size is expanded or reduced to achieve new predictions and try to obtain better results.

2.5 Development of the User Interface

Finally, a graphical user interface with Tkinter is created. Through this interface, users can enter their symptoms and receive the three most likely diagnoses based on the model's predictions. This interface is designed so that doctors receive the patient's symptoms and the three most probable diagnoses and with their knowledge choose the appropriate one.

3 RESULTS AND ANALYSIS

3.1 Selection of the most suitable model

In the results section, we explore the key findings obtained from our predictive model. This part of the work illustrates the effectiveness of the chosen model in identifying diseases from textual descriptions of symptoms. We will analyse the quantitative results and examine specific cases where the model correctly or incorrectly predicted the diagnosis, thereby providing insight into the advantages and limitations of this model.

The first step was to analyse and compare the different models used to train our training set and decide which one best fits our project:

Models	Best Score
Random Forest	0.8730
Logistic Regression	0.8474
Gradient Boosting	0.8251
Naive Bayes Gaussian	0.8112
Naive Bayes Bernoulli	0.85376

Table 1: Comparison of models using GridSearchCV in training.

The model that provided the best performance in the analysis has been Random Forest, with optimal hyperparameters of maximum depth (*max_depth*) of 60, maximum number of features (*max_features*) using *log2*, and 200 trees (*n_estimators*), achieving a score of 0.8730 with an initial dictionary of 100 words for training the model.

Words	Top1 score	Top3 score
100	0.8900	0.9700
150	0.9400	0.9916
200	0.9500	0.9916
250	0.9600	0.9875
350	0.9600	0.9916

Table 2: Comparison of F1-scores according to the size of the word dictionary.

With the Random Forest model selected, we investigated the impact of the dictionary size on the f1-scores of the Top1 and Top3 predictions. The results, shown in Table 2, indicate that increasing the dictionary size does not significantly improve the f1-score of the Top1 predictions. Prioritizing the Top3 metric due to its importance in the medical field, we opted for a 200-word dictionary. This offers a good variety of terms with good results without being too large. Although a larger dictionary like the 350-word one might seem better, in reality, a smaller dictionary also works very well and can even be faster and more efficient for real-life use.

Array content	Top1 score	Top3 score
[ ]	0.9500	0.9916
[also, get, causing]	0.9500	0.9916
[also, get, causing, lot, really, feeling]	0.9500	0.9875

Table 3: Comparison of F1-scores upon removing specific words from the dictionary.

In Table 3, it is analysed the impact of removing specific words from the 200-word dictionary on the performance of our Random Forest model. This table shows three of the options tested during the project's development. Finally, it has been decided to remove the words *also*, *get*, and *causing* from the dictionary. These words, according to the analysis, were contributing negatively to the prediction process, leading to incorrect classifications. Although their removal did not significantly increase the Top3 score, it does improve the quality of the top three predictions (increasing their probabilities).

The final model was Random Forest with the aforementioned hyperparameters and a 200-word dictionary (excluding the words *also*, *get*, and *causing*).

3.2 TOP1 Score Analysis

Using this model, a score of 0.95 is obtained, and a confusion matrix is generated to analyse the model's performance in classifying diseases. This matrix helps to visualize instances where the model has succeeded and where it has failed. For example, it can be seen that some diseases like psoriasis, varicose veins, and chickenpox have been classified with a high degree of accuracy, while others, like fungal infections or impetigo, have had a couple of misclassified cases. This information is crucial for improving the model, as it indicates that these diseases are more difficult to predict and require special attention by the doctors.

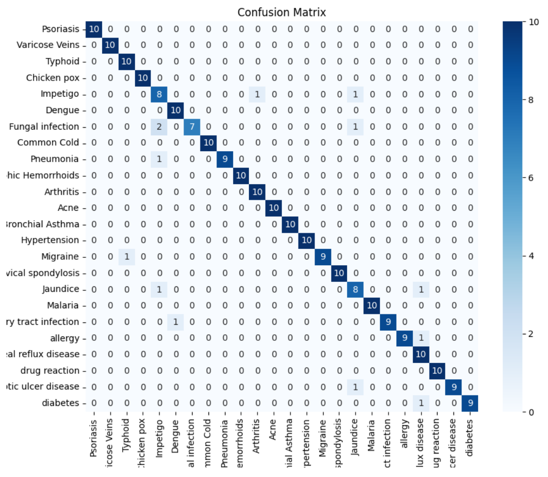


Fig. 1: Confusion matrix Top1 f1-score

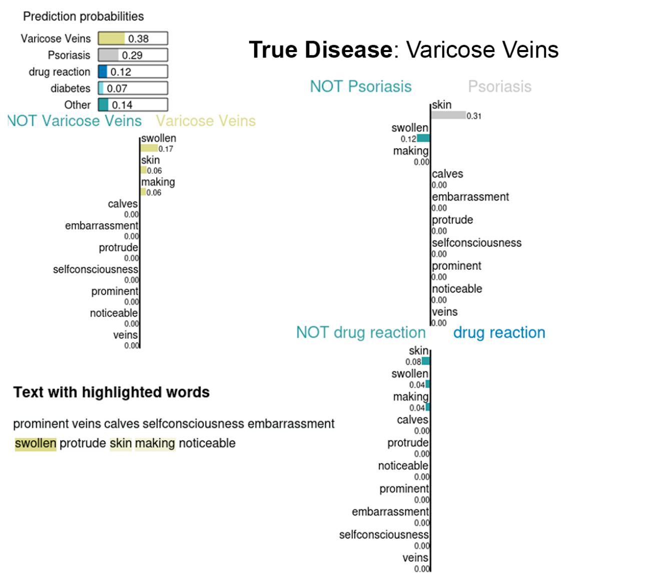


Fig. 3: Example of a well-classified LIME image as Top1 using the F1-score metric.

The use of LIME (Local Interpretable Model-agnostic Explanations) in this work has been very important, as it has allowed us to understand the internal workings of the Random Forest model for each symptom text.

Figure 4 analyses a LIME image, illustrating an incorrect classification by the model. The true disease is typhus, but the model has erroneously predicted it as a reaction to medication.

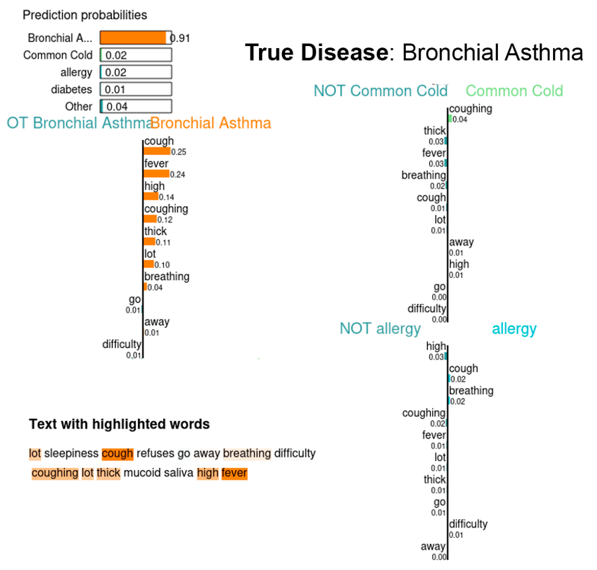


Fig. 2: Example of a well-classified LIME image as Top1 using the F1-score metric.

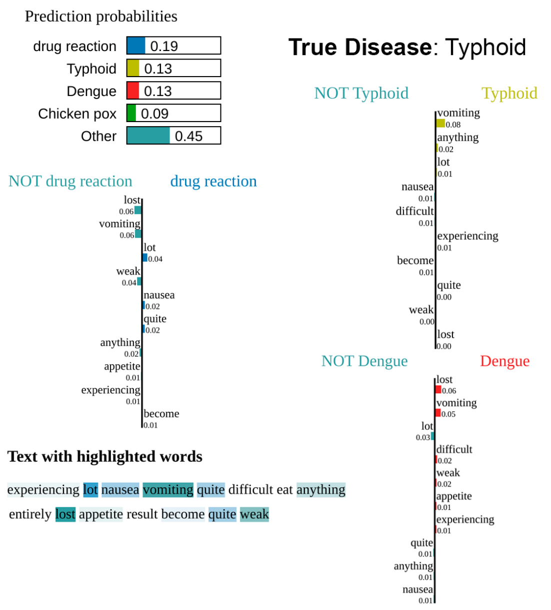


Fig. 4: Example of a misclassified LIME image as Top1 using the F1-score metric.

LIME images provide a detailed view of how our Random Forest model accurately identifies diseases.

In these examples of well-classified diseases, keywords such as *fever* and *cough* for bronchial asthma (fig 2) and *swollen* and *veins* for varicose veins (fig3) are highlighted, which are indicative of the symptoms. In addition, the presence of these keywords in correct predictions is often accompanied by high significant prediction probabilities, allowing us to trust the model's ability to make accurate and reliable diagnoses, an essential feature for its application in the field of medicine.

This mistake can be attributed to the fact that symptoms of *nausea* and *vomiting*, clearly associated with typhus, are also common in other illnesses, including adverse drug reactions. This case underscores the complexity of differentiating between diagnoses with similar clinical manifestations. It is also noted that the probabilities assigned to each disease in the incorrect prediction are lower than those observed in correct classifications. This indicates that, although the model has identified certain relevant symptoms, it did not have sufficient certainty to make a definitive prediction. In medical settings, this fact can be taken into account by medical personnel when relying more or less on the predictions made by the model.

To achieve better disease prediction, it would be necessary to have more training data and/or more specific symptom texts from the patient.

3.3 TOP3 score

The analysis of the confusion matrix and LIME predictions underline the importance of using a Top 3 metric to evaluate the model instead of limiting it to Top 1. This is especially relevant in cases where the probabilities of correct predictions are low, and there is a high symptomatic similarity between different conditions. In these cases, the model may not be sure of its main prediction, but it can identify the correct disease within the three most probable ones.

With a Top 3, this model achieves a score of 0.9916, so, the model shows a high degree of accuracy in most cases. However, there are two misclassified cases:

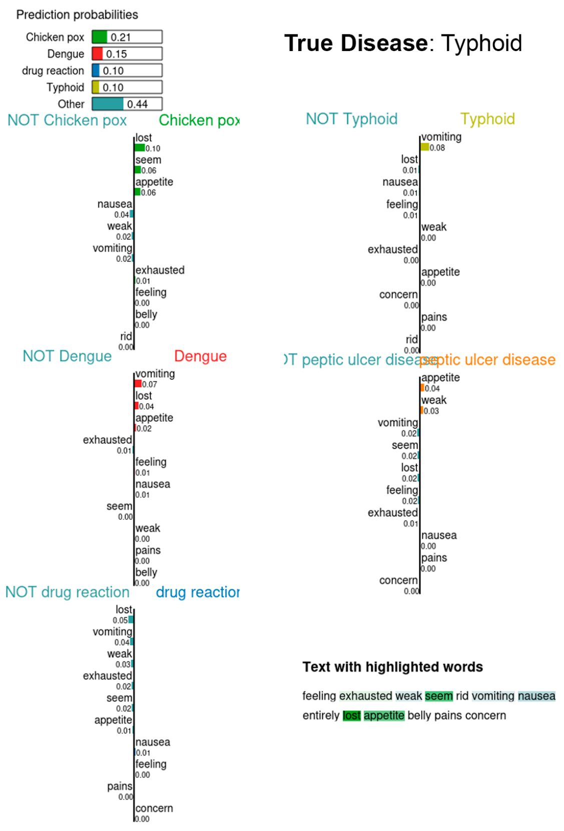


Fig. 5: Lime explainability of the first misclassified prediction using the TOP3 f1_weighted score metric.

This first case, of typhus, shared symptoms with other diseases such as chicken pox or reactions to medications have led to confusion. This highlights the model's difficulty in differentiating between diseases with similar clinical manifestations, especially when symptoms are common to several conditions. To accurately detect this case, we, perhaps, would need a richer database that includes more examples of each disease.

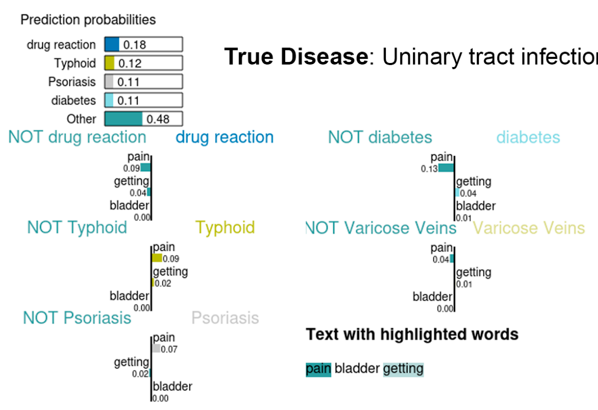


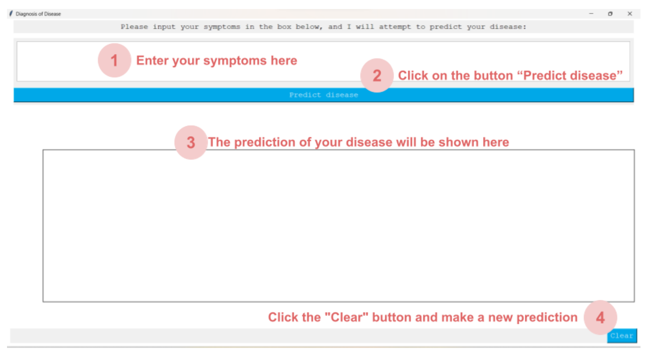
Fig. 6: Lime explainability for the second misclassified prediction using TOP3 f1_weighted score as the metric.

In this second case, the urinary tract infection was not correctly identified, possibly due to the presence of only a few clinically significant words in the symptom text. This misclassification was not considered a model error, it is considered an outlier, as the user entered insufficient symptomatic information to explain the disease.

3.4 Graphic interface with Tkinter

The graphical interface created for this project, developed with Tkinter, is an application aimed at diagnostic assistance. It is designed for doctors to enter symptoms in natural language and obtain a list of the three most likely diseases. This functionality can significantly help healthcare professionals in determining the correct diagnosis, providing them with a quick and efficient reference database based on predictive analysis of symptoms reported by users.

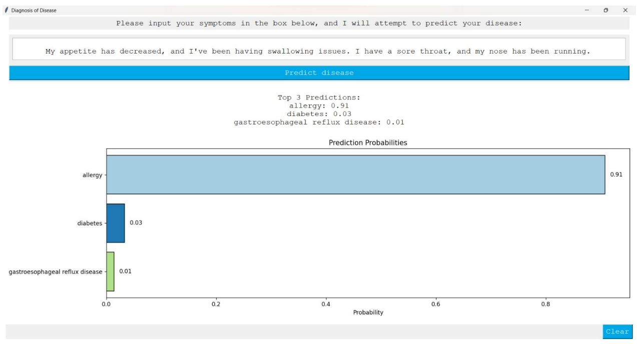
The graphical interface allows users to enter their symptoms, click on *Predict disease* to process the information, and view the disease prediction along with associated probabilities. The *Clear* button can also be used to start a new query.



Diagnosis of Disease
Please input your symptoms in the box below, and I will attempt to predict your disease:
1 Enter your symptoms here
2 Click on the button "Predict disease"
Predict disease
3 The prediction of your disease will be shown here
Click the "Clear" button and make a new prediction 4
Clear

Fig. 7: Instructions for operating the graphical interface

In the following example of the Tkinter graphical interface, the user has entered symptoms including reduced appetite, difficulty swallowing, sore throat, and nasal discharge. The system has processed this information and provided a prediction with a high probability of an allergy as the most likely diagnosis, followed by diabetes and gastroesophageal reflux disease as less probable.



```
Diagnosis of Disease
Please input your symptoms in the box below, and I will attempt to predict your disease:

My appetite has decreased, and I've been having swallowing issues. I have a sore throat, and my nose has been running.

Top 3 Predictions:
allergy: 0.91
diabetes: 0.03
gastroesophageal reflux disease: 0.01
```

Fig. 8: Example of the graphical interface in operation.

This tool helps prioritize diagnostic considerations in a clinical setting, allowing medical professionals to make a more informed assessment.

4 Conclusions

This work demonstrates the effectiveness of using natural language processing and machine learning techniques to diagnose a set of 24 diseases from textual symptoms. Using the Random Forest model, optimized through hyperparameter search, has provided promising results, as demonstrated by the high precision of the Top 3 metric. Moreover, this study suggests the possibility of extending its application to include a broader spectrum of diseases beyond the initial 24.

To further improve this score, the incorporation of word embeddings, which could better capture the semantic context of symptoms, could be considered. Expanding the training dataset can also provide a more comprehensive representation of the different symptomatic manifestations, enhancing the model's ability to generalize. However, it should be noted that low probabilities in predictions may indicate uncertain classifications, and might require a more detailed review by doctors. Additionally, exploring additional predictive models could offer more suitable alternatives for specific cases not tested in this project.

LIME visualizations (Local Interpretable Model-Agnostic Explanations) have proven to be a valuable tool for delving into the contribution of each symptom to the final model prediction. This methodology allows medical professionals to better understand how the model reaches its conclusions and which symptoms carry more weight in the final decision. This can be crucial for identifying key factors in the diagnostic process and enhancing confidence in the model's results.

Finally, the graphical interface developed with Tkinter represents a significant tool for doctors. This interface can facilitate preliminary diagnosis by visually presenting the model's results and interacting with patient data. However, it is important to emphasize that this tool should only be considered as a support and guidance in the diagnosis process of healthcare personnel.

5 Github project link

For seamless access to the comprehensive work and files associated with this study, a QR code is provided which directs to the GitHub repository. This enables readers to explore the methodology, data, and code used throughout the research in detail.

Fig. 9: Scan this QR code to access the GitHub repository associated with this work.

References

[1] [Sequencing - Turning sentences into data]. (2023, November 17). Retrieved from `https://www.youtube.com/watch?v=r9QjkdSJZ2g`

[2] [What does Keras Tokenizer method exactly do?]. (2023, November 17). Retrieved from `https://stackoverflow.com/questions/51956000/what-does-keras-tokenizer-method-exactly-do`

[3] [Introducción a Tcl/Tk (tkinter)]. (2023, December 8). Retrieved from `https://recursospython.com/guias-y-manuales/introduccion-a-tkinter/`