Integrative Network Analysis: Unveiling Symptom-Disease Interactions and Enhancing Predictive Models

Andreoli C. • Ligari D. • Alberti A. • Scardovi M. [1]

[1] *Department of Computer Engineering, Data Science, University of Pavia, Italy*
Course of Financial Data Science

Github page: https://github.com/DavideLigari01/financial-project

Date: December 15, 2023

Abstract — We will write it once we have the results.

Keywords —Graph theory • Features Engineering • Community detection • Null models • Random forest • MLP

CONTENTS

1. INTRODUCTION

In the dynamic healthcare landscape, a profound understanding of symptom-disease interactions is crucial for accurate diagnosis and prediction. This report delves into this complex interplay using network analysis, integrating theoretical foundations and empirical data. Our dual-fold objective is to provide a nuanced descriptive analysis and identify key features for predictive models.

The foundation of our endeavor lies in an extensive literature review on network theory and disease prediction, ensuring a robust baseline for deeper insights. Guided by literature, we meticulously curate and analyze datasets, preparing the groundwork for constructing meaningful networks that encapsulate symptom-disease relationships.

Our analysis centers on intricate network structures, utilizing bipartite models and non-weighted links to distill meaningful patterns. We explore a spectrum of network metrics, ensuring statistical significance through null models. Community detection algorithms reveal hidden structures and relationships between diseases, enriching our understanding.

As we traverse the terrain of network analysis, we introduce novel metrics inspired by the Hidalgo-Hausmann framework, stratifying symptoms and diseases based on predictive importance. These, coupled with traditional measures like betweenness centrality, contribute to defining features for our predictive models.

With a robust foundation, we transition to predictive modeling, promising enhanced performance. Logistic regression, random forest, and multi-layer perceptron models are trained, tested, and validated with a focus on feature importance and model improvement strategies.

2. DATASET

The dataset used for this project is obtained from Kaggle and is available at the following link. It comprises disease names along with the symptoms reported by the respective patients.

Overview: The dataset encompasses 773 unique diseases and 377 symptoms, resulting in approximately 246,000 rows. It was artificially generated while preserving Symptom Severity and Disease Occurrence Possibility.

Data Encoding: To facilitate model training, the dataset utilizes one-hot encoding for each symptom, transforming categorical symptom data into a binary format.

Class Imbalance: The original dataset exhibited significant class imbalance, with some classes having only one sample and others containing thousands. We addressed this problem using Oversampling and Undersampling techniques, and the details are further elaborated in Section a.

Data Cleaning: To allow consistent Oversampling, classes (diseases) with fewer than three symptoms were excluded from the dataset, resulting in the removal of 25 classes. Additionally, diseases with no symptoms and symptoms with no associated diseases were deleted as well.

3. GOALS

The final objective of this project is to develop a robust **machine learning model** capable of predicting diseases based on reported symptoms. In pursuit of this goal, *two specific objectives* are outlined, both centered around **leveraging the network** structure:

1. **Feature Extraction:** Exploiting the network characteristics and metrics, aiming to extract novel features that can enhance the predictive capabilities of the model.

2. **Complexity Reduction:** Leveraging network information, we aim to reduce the number of symptoms, retaining only the most relevant ones. This strategic reduction aims to decrease training time while preserving the accuracy of the model.

By integrating these two objectives, the project aspires to not only advance the predictive capabilities of the machine learning model but also optimize its efficiency in handling the complexities inherent in disease prediction based on symptoms.

4. NETWORK METHODOLOGY

In this section there is a technical description of the methodology used to create and analyze the network.

a. Network Creation

In this project, a bipartite graph was created where nodes represent symptoms (lightcoral) and diseases (lightblue), and edges denote the presence of symptoms in diseases. For simplicity, the graph is unweighted, and all edges have a weight of 1. Prior to graph creation, a preliminary data analysis was conducted to identify isolated nodes. Additionally, an analysis of the node degree distribution was performed to assess whether the distribution follows a power law (see Section 5.1).

The analysis revealed the presence of several symptoms, precisely 52 out of 377, not associated with any disease. Consequently, these isolated symptoms were removed from the graph as they do not contribute informative content. Diseases without associated symptoms were also removed for the same reason.

Figure 1 illustrates the resulting bipartite graph. As observed, diseases tend to be peripheral, while symptoms tend to be central. This observation arises from the fact that symptoms are shared by multiple diseases, whereas diseases exhibit distinct symptoms. Furthermore, symptoms are fewer in number than diseases, making it more likely for a symptom to be shared among multiple diseases.

Figure 2 presents the unipartite graphs of symptoms and diseases.

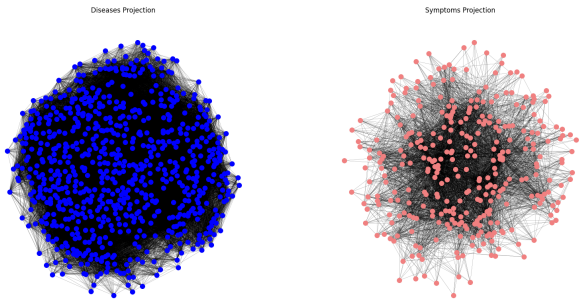

Fig. 2: Visual representation of symptom and disease unipartite graphs.

b. Method of Reflections

To identify influential nodes in the symptom-disease network, we introduce two indices that capture the relative importance of each actor. The first index, referred to as the Symptom Influence (*SI*) index, not only ranks symptom nodes based on their frequency (level-1) but also considers whether a symptom is present in diseases affected by numerous other symptoms (level-2) or in diseases affected by only a few symptoms.

Conversely, the second index, known as the Disease Influence (*DI*) index, assesses the distinct symptoms related to a disease (level-1) and whether a disease exhibits symptoms that affect many other diseases (level-2).

In other words, level-1 of these indices quantifies the number of symptoms associated with a disease, while level-2 measures the interconnectedness and broader impact of symptoms or diseases within the network. For the Symptom Influence (*SI*) index, level-2 takes into account the presence of a symptom across diseases, shedding light on whether a particular symptom tends to co-occur with a wide range of other symptoms or is more specific to a subset of diseases. This dual-level analysis provides a nuanced understanding of the significance of symptoms based not only on their individual prevalence (level-1) but also on

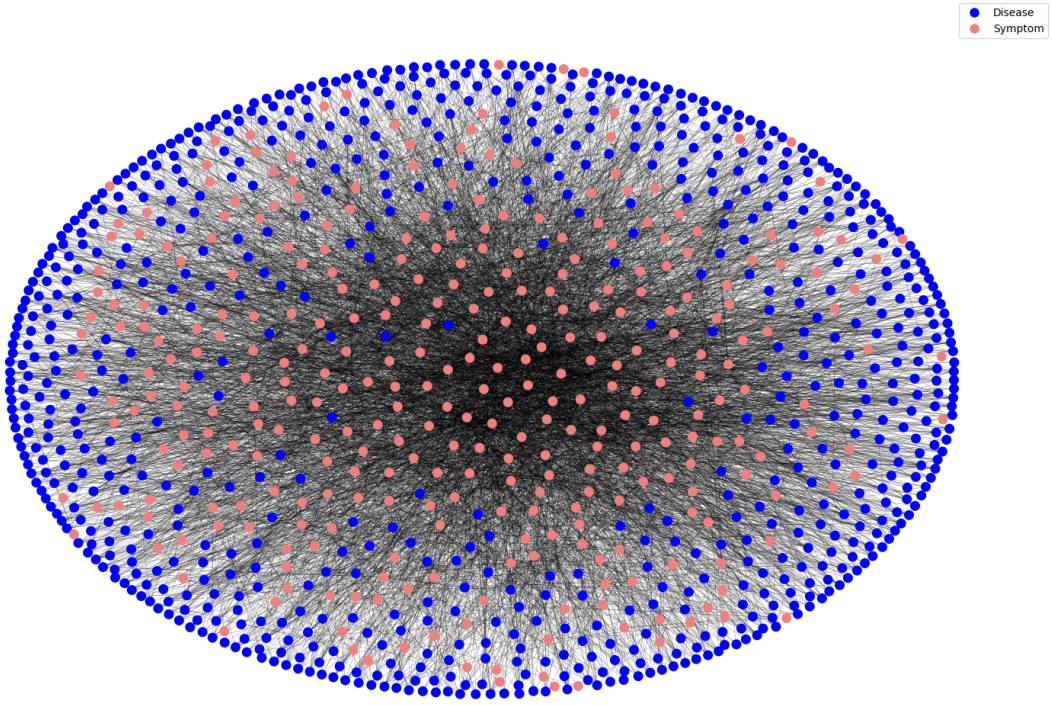

Fig. 1: Visual representation of the symptom-disease bipartite graph.

their associations with other symptoms across different diseases (level-2). Similarly, for the Disease Influence (*DI*) index, level-2 assesses the extent to which a disease's symptoms have ripple effects on other diseases, indicating the potential for cascading impacts within the network.

We adapt the level-N indices following the approach of Hidalgo et al.,[4] and Hidalgo and Hausmann,[5]. The level-N indices are defined as:

$$SI_{v,N} = \frac{1}{SI_{v,1}} \sum_{u} W(v,u) DI_{u,N-1} \tag{1}$$

$$DI_{u,N} = \frac{1}{DI_{u,1}} \sum_{v} W(v,u) SI_{v,N-1} \tag{2}$$

Here, $SI_{v,1}$ and $DI_{u,1}$ represent the level-1 indices, and $W(v,u)$ denotes the edge weight between symptom v and disease u. The level-1 indices are defined as follows:

$$SI_{v,1} = \sum_{u} W(v,u) \tag{3}$$

$$DI_{u,1} = \sum_{v} W(v,u) \tag{4}$$

Since our network is not weighted ($W(v,u) = 1$ if symptom v is associated with disease u and $W(v,u) = 0$ otherwise), $SI_{v,1}$ and $DI_{u,1}$ are equal to the degree of symptom v and disease u, respectively.

Statistical Validation of SI and DI

In our effort to identify significant nodes within the symptom-disease network, we focus on discerning topological properties that hold statistical significance. Our goal is to differentiate higher-order properties that are directly associated with local node features from those that emerge from the intricate interactions among nodes.

Relevant studies by Squartini, Fagiolo, and Garlaschelli [9, 8] and Spelta, Pecora, and Pagnottoni [7] highlight how higher-order network properties naturally capture structured group interactions. Here, a 'group' is defined as all players connected by a 'hyperlink,' representing the higher-order analog of a link.

The sampling of random graphs with specified properties plays a pivotal role in network analysis, serving as fundamental null models for identifying patterns, including communities and motifs.

To statistically assess the significance of SI and DI, we adopt a hypothesis testing approach based on a null model. Specifically, we posit **H0** as the hypothesis that SI and DI level-2 do not offer additional information compared to level-1, and **H1** as the opposite. To test these hypotheses, we generate 5000 random networks using a null model with the same level-1 properties as the original network.

With this ample set of null models, we assume that the distribution of SI and DI is Gaussian, leveraging the Central Limit Theorem (CLT). For each null model, we calculate the mean (μ) and standard deviation (σ) of SI and DI and compute the z-score for each SI and DI level-2, as expressed in the following equation:

$$z_{SI_{v,2}} = \frac{SI_{v,2} - \mu_{SI_{v,2}}}{\sigma_{SI_{v,2}}} \tag{5}$$

If **H0** holds true,the z-scores of SI and DI should be normally distributed with a mean of 0 and a standard deviation of 1. Conversely, if **H1** is true, the z-scores of SI and DI

should be normally distributed with a mean and standard deviation different from 0 and 1, respectively.

c. Betweenness Centrality

The betweenness centrality of a node v, as defined by Brandes [2], is calculated as the sum of the fraction of all-pairs shortest paths that pass through v:

$$c_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)} \tag{6}$$

where:

- V: The set of nodes.
- $\sigma(s,t)$: The number of shortest paths from node s to node t.
- $\sigma(s,t|v)$: The number of those shortest paths from node s to node t that pass through some node v other than s and t.
- If $s = t$, then $\sigma(s,t) = 1$.
- If $v \in \{s,t\}$, then $\sigma(s,t|v) = 0$.

To compute the betweenness centrality, the NetworkX function *nx.bipartite.betweenness_centrality* was utilized. This function implements the algorithm proposed by Brandes [1], specifically designed for bipartite graphs, and includes proper normalization for accurate results.

d. Communities Detection

Prior to applying any community detection algorithm, two crucial steps must be performed:

- **Graph Projections:** The bipartite graph needs to be projected into two separate graphs, one for each set of nodes. In our case, the two sets represent symptoms and diseases. To achieve this, the NetworkX function *nx.bipartite.projected_graph* is employed, returning the projection of the bipartite graph onto the specified nodes.

- **Compute Similarity:** The similarity between nodes needs to be computed. For our purposes, a co-occurrence matrix is created for each set of nodes. Taking the example of the co-occurrence matrix for symptoms, each entry s_{ij} represents the number of times the symptom i and the symptom j co-occur in the same disease.

Once the two graphs, with links weighted by node similarity, are obtained, the community detection algorithm can be applied. We utilized the Clauset-Newman-Moore greedy modularity maximization algorithm [3], implemented in the NetworkX function *nx.algorithms.community.greedy_modularity_communities*. This algorithm aims to find the partition of the graph that maximizes modularity, defined by Newman [6] as:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{7}$$

where:

- Q: Modularity of the network.
- A_{ij}: Element of the adjacency matrix representing the connection between nodes i and j.
- k_i and k_j: Degrees of nodes i and j, respectively.
- m: Total number of edges in the network.
- $\delta(c_i, c_j)$: Kronecker delta function, which is 1 if c_i is equal to c_j (i.e., nodes i and j belong to the same community) and 0 otherwise.
- The sum is taken over all pairs of nodes i and j.

5. Network Results

a. Method of reflection

We conducted an in-depth analysis of the Symptom Influence (*SI*) and Disease Influence (*DI*) indices, considering both level-1 and level-2 metrics in our symptom-disease network.

Level-1 Metrics

For the level-1 metrics, which quantify the individual prevalence of symptoms and diseases, we observed distinctive patterns, as illustrated in Figure 3. The Symptom Influence (*SI*) at level-1, representing the number of diseases associated with a symptom, demonstrated a wide distribution, ranging predominantly between 0 and 60. This suggests that certain symptoms exhibit a broad association with various diseases, showcasing the diverse nature of symptom-disease relationships.
Conversely, the Disease Influence (*DI*) at level-1, reflecting the number of symptoms related to a disease, exhibited a narrower range, typically falling between 2 and 12. This narrower range indicates that diseases tend to have a more focused set of associated symptoms.

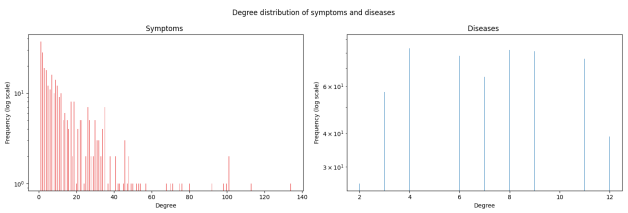

Fig. 3: Degree Distribution (level 1)

Level-2 Metrics

Moving to the level-2 metrics, which capture the interconnectedness and broader impact of symptoms or diseases within the network, we uncovered intriguing patterns (see

Figure 4). The Symptom Influence (*SI*) at level-2, reflecting the presence of symptoms across diseases and their associations with other symptoms, exhibited values ranging from 4 to 12. This suggests that certain symptoms not only co-occur within diseases but also form meaningful connections with a diverse set of other symptoms. A higher level-2 Symptom Influence (*SI*) implies a symptom's propensity to be associated with a wide range of symptoms, indicating its potential impact on various disease pathways.

On the other hand, the Disease Influence (*DI*) at level-2, quantifying the ripple effects of a disease's symptoms on other diseases, demonstrated a wider range, typically spanning from 10 to 80. This broader range signifies that certain diseases have a more extensive influence on the network by affecting a multitude of other diseases. A higher level-2 Disease Influence (*DI*) implies that a disease's symptoms not only contribute to its immediate associations but also have far-reaching consequences, affecting a network of interconnected diseases.

These results highlight the diverse roles played by symptoms and diseases in influencing the network when considering higher-order interactions.

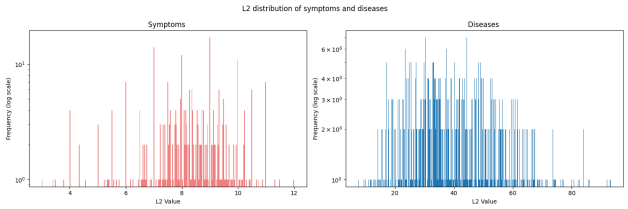

Fig. 4: L2 Distribution for both symptoms and diseases

Power Law CCDF

In our statistical validation of the Symptom Influence (*SI*) and Disease Influence (*DI*) indices at both level-1 and level-2, the analysis of the complementary cumulative probability distribution (CCDF) reveals distinctive patterns.

For level-1 diseases, the CCDF power-law behavior exhibits a rapid decrease, indicating that a small number of diseases have a disproportionately high influence. This steep decline suggests the presence of disease hubs that significantly impact the network dynamics. Conversely, for level-1 symptoms, the power-law CCDF demonstrates a slow decrease, starting at 0 and extending until 100. This suggests a more distributed influence of symptoms across diseases, with a considerable number of symptoms exhibiting varying degrees of prevalence. The gradual decline in the CCDF emphasizes the diverse roles played by symptoms in the network.

Level-2 diseases, on the other hand, exhibit a slower power-law decrease, starting at 50 and reaching zero around 90. This gradual decline implies that a broader range of diseases contributes to the interconnectedness within the network, with a subset of diseases exerting influence across multiple others.

Finally, for level-2 symptoms, the power-law CCDF displays a rapid decrease around 10, emphasizing the existence of highly influential symptoms that play a pivotal role in connecting various diseases.

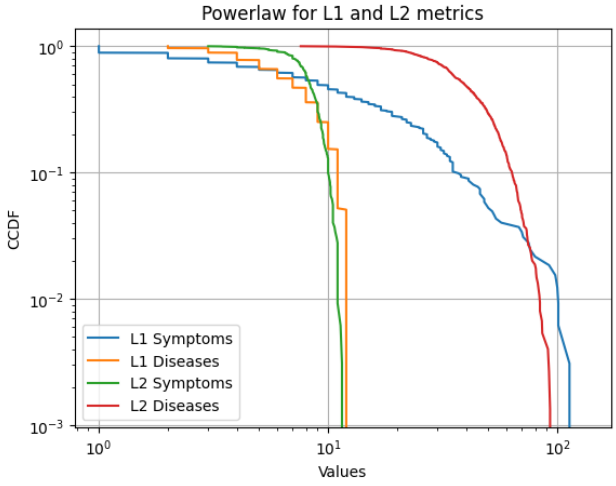

Fig. 5: Power Law Distribution of the level 1 and level 2 metrics

Statistical Validation of SI and DI

In our statistical validation of the Symptom Influence (*SI*) and Disease Influence (*DI*) indices at level-2, we aimed to discern whether these higher-order metrics provide additional information compared to level-1, and whether this information is statistically significant. Our null hypothesis (**H0**) posited that level-2 metrics do not offer additional insights beyond level-1, while the alternative hypothesis (**H1**) suggested the opposite.

Upon generating 5000 random networks with the same level-1 properties as the original network, we calculated z-scores for both Symptom Influence (*SI*) and Disease Influence (*DI*) at level-2.

Remarkably, the z-score distribution (Figure 6) for both symptoms and diseases exhibited a shape quite similar to a Gaussian distribution, with means close to zero.

For symptoms, the z-scores ranged between -4 and 3, indicating that the level-2 Symptom Influence (*SI*) values were generally lower than the mean but still within a reasonable range. This suggests that, on average, symptoms tend to exhibit a level-2 influence that aligns closely with the overall network structure.

Similarly, for diseases, the z-scores ranged from -3 to 4, signifying that the level-2 Disease Influence (*DI*) values were distributed around the mean. This implies that diseases, on average, have a level-2 influence that aligns with the overall network structure, showcasing a balance between localized effects and broader impacts.

The proximity of the mean to zero in both distributions suggests that, on average, the level-2 metrics for both symptoms and diseases do not significantly deviate from the null model. However, the broader range of z-scores signifies the presence of nodes with both positive and negative deviations, underscoring the heterogeneous nature of influences within the symptom-disease network.

In conclusion, our statistical validation reinforces that while the level-2 metrics follow a distribution akin to a Gaussian, the nuanced deviations reflected by the z-scores for both symptoms and diseases underscore the diverse and

context-dependent nature of their influences within the intricate network structure.

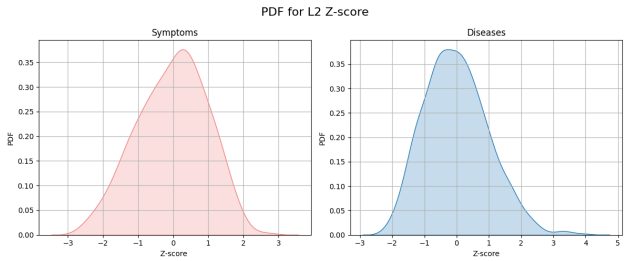

Fig. 6: Probability Density Function of the z-scores

b. Betweenness Centrality

The examination of betweenness centrality in our bipartite network, as depicted in Figure 7, reveals a Power Law Distribution, indicative of a scale-free structure. This implies the presence of a few central nodes that act as pivotal connectors, while the majority of nodes exhibit lower betweenness centrality.

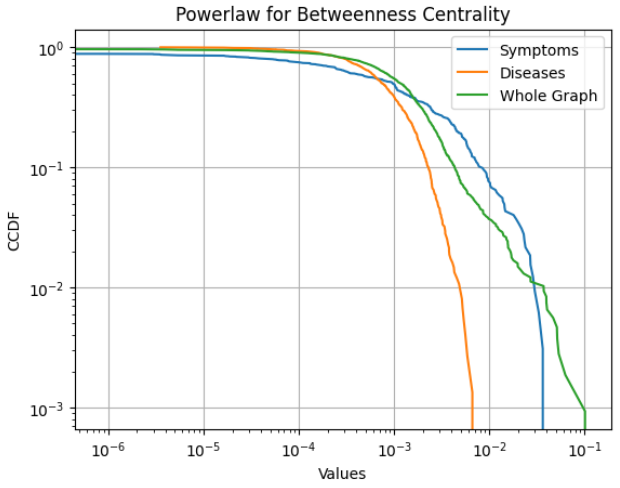

Fig. 7: Betweenness Centrality CDFs

Upon dissecting the centrality values into symptoms and diseases (see Figures 8 and 9), a notable observation emerges: symptoms tend to have higher betweenness centrality compared to diseases. To decipher the significance of this result, it's essential to delve into the interpretation of betweenness centrality.

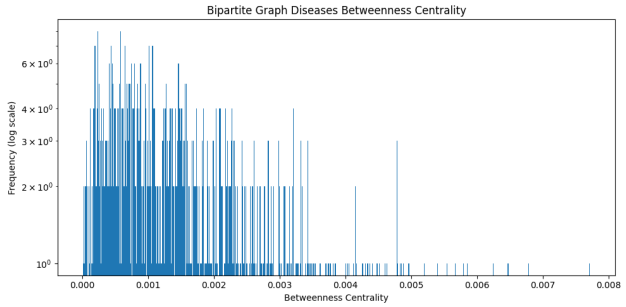

Fig. 8: Betweenness Centrality of the diseases

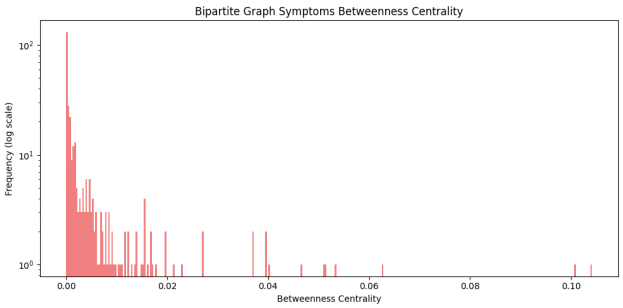

Fig. 9: Betweenness Centrality of the symptoms

In general, a symptom exhibits high betweenness centrality when it is linked to numerous diseases, and these diseases, in turn, are connected to a relatively limited set of symptoms. Conversely, a disease attains high betweenness centrality when it connects to numerous symptoms, and these symptoms are associated with relatively few diseases.
Analyzing our results (L1 and L2), it becomes evident that the higher betweenness centrality of symptoms is attributed to their connections with a multitude of diseases, while diseases, on the contrary, are linked to a relatively limited number of symptoms. From a predictive standpoint, this outcome presents a challenge as each symptom is not sufficiently specific, contributing to a broad array of disease classes.
Figure 10 highlights the top 10 nodes with the highest betweenness centrality, all of which are symptoms. As anticipated, these symptoms are more generic in nature, aligning with their central role in connecting various diseases.

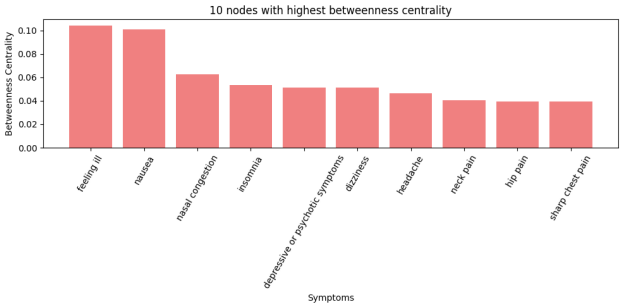

Fig. 10: Top 10 nodes with the highest betweenness centrality

c. Communities

The identification of communities within the network serves a dual purpose – facilitating network interpretation and enhancing the capabilities of our ML prediction model. From a network interpretation perspective, communities offer insights into disease-symptom relationships. A community of symptoms signifies a set of symptoms that frequently co-occur within the same diseases, while a community of diseases identifies a set of diseases often co-occurring within the same symptoms. The sizes of different communities are illustrated in Figure 11.

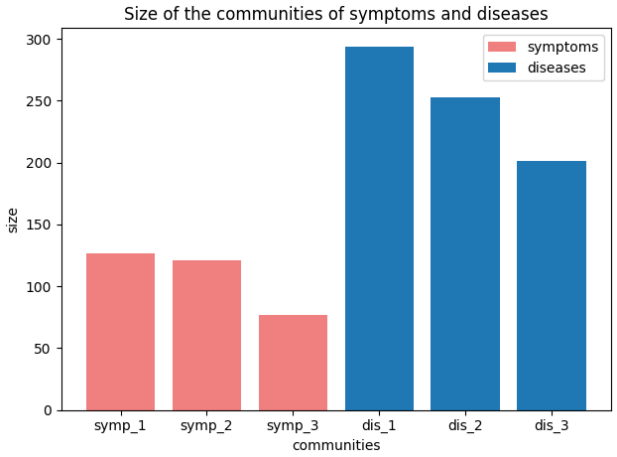

Fig. 11: Sizes of the communities of symptoms and diseases

For clinical relevance, examining symptoms communities provides valuable information about diseases associated with these symptoms. This is exemplified in Figures 12, 25, and 26. As an example, in the symptoms' community 1 (Figure 12), 'burn' has 12 symptoms, which makes it a specific disease for community 1, considering that on average each disease has only three symptoms from that community.

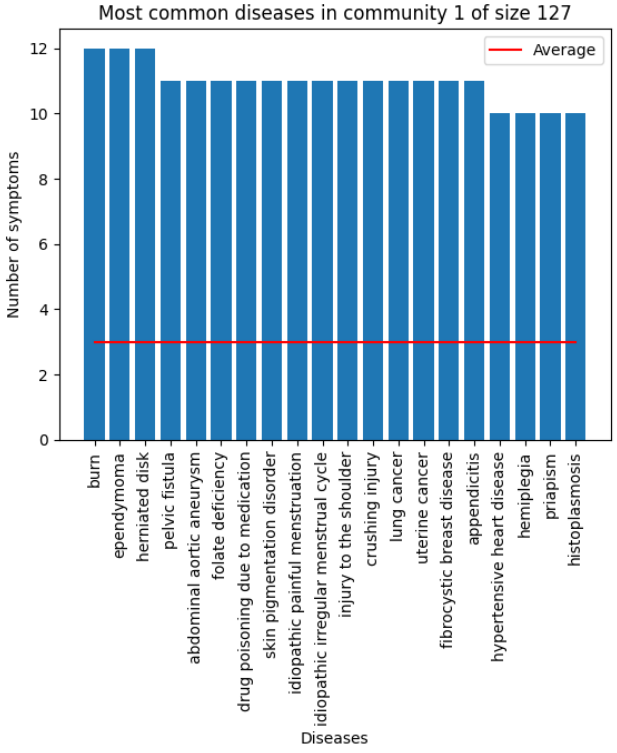

Fig. 12: Community 1 of symptoms

A similar study can be conducted for communities of diseases, as depicted in Figures 13, 27, 28. This information aids in profiling diseases and understanding the significance of each symptom. For instance, in community 1 of diseases (Figure 13), the symptom 'sharp abdominal pain' is present in almost half of the diseases in the community, indicating its generic nature and limited discriminatory value.

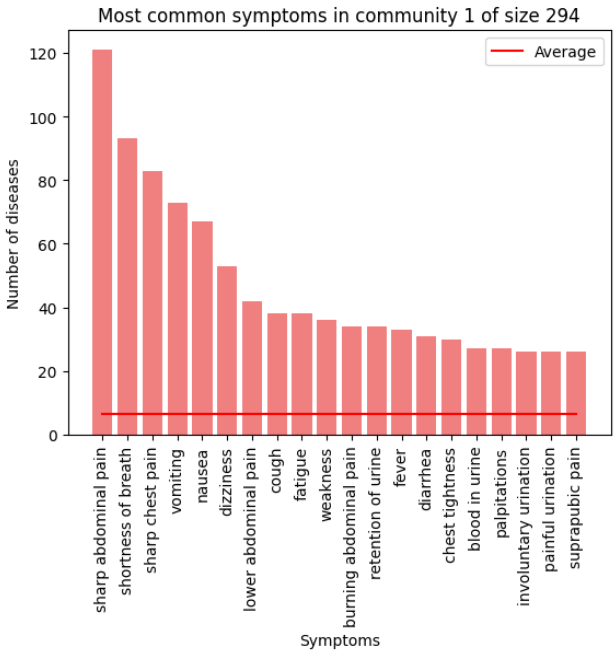

Fig. 13: Community 1 of diseases

Transitioning to the creation of features for the ML model, two types of features were developed:

- **Community Count:** This feature counts how many symptoms of the symptom vector belong to each community. Each symptom community is characterized by different pointed diseases. The model can learn to prioritize diseases associated with the community with the highest count.

- **Community Size:** This feature replaces each symptom in the symptom vector with the size of the community to which the symptom belongs. It enables the model to distinguish between symptoms belonging to small and large communities. If many symptoms from small community are present, the associated diseases may be more likely.

It is noteworthy that communities can also contribute to improving the computational efficiency of the model. For example, a symptom associated with many diseases may be less informative and could potentially be removed from the symptom vector. However, we opted for a comprehensive approach using a combination of L1 and L2 measures to address this issue.

d. Most Important Actors

As previously mentioned, our objective extends beyond feature extraction; we aim to leverage network information to enhance the computational efficiency of the model. The strategy involves reducing the number of symptoms, retaining only the most significant ones, to decrease training time while maintaining high accuracy. Various approaches were tested, including L1, L2, betweenness centrality, and the degree of the unipartite projection of symptoms. To se-

lect the most appropriate approach, we examined the correlation between these features (Figure 14). Indeed, a high correlation between features means that they provide similar information, and therefore retaining both features would be redundant. On the other hand, a low correlation between features indicates that they provide complementary information, and this enhances in a considerable way the quality of the choice.
For this reason, as clearly shown in Figure 14, we decided to use L1 and L2 to discriminate among the symptoms in a more effective manner.

To practically create the classes, we need to define thresholds for L1 and L2. We decided to consider a symptom as very important for our model if it is present in less than 0.5 times the average of L1 diseases, translating into a threshold of 8.21 for L1. Consequently, we adjusted the L2 threshold to maintain a proper balance between the classes, setting it to 8.

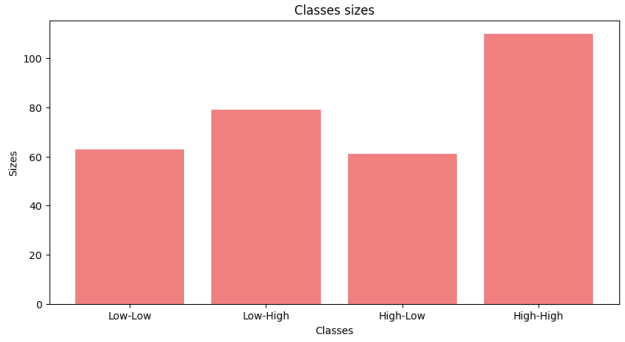

Fig. 15: Symptoms divided into the four classes

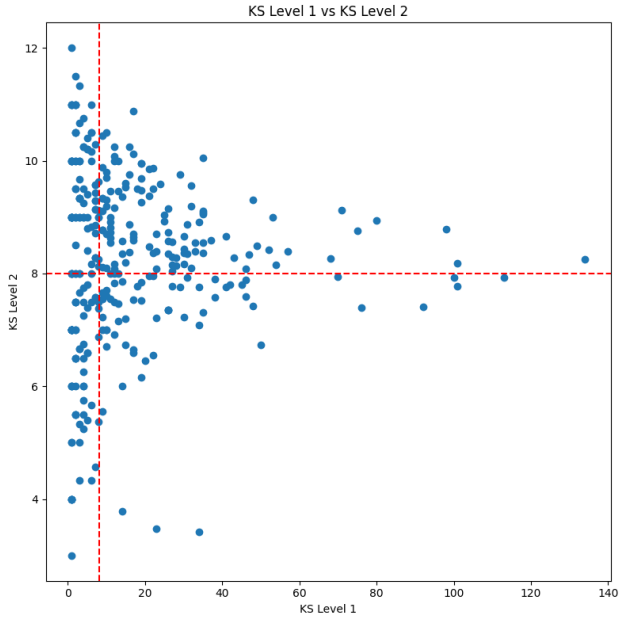

Fig. 16: Division based on L1 and L2 values

As demonstrated in Figure 15, the four classes have different sizes, and more importantly, they provide very different and valuable information.

To shed light on these classes, a brief reminder of the meanings of L1 and L2 is warranted. L1 denotes the number of diseases associated with a symptom, whereas L2 quantifies the number of symptoms linked to those diseases. As a result, we categorize the features into the following classes:

- **High L1 - High L2**: Symptoms with high degree and high L2. These symptoms are less crucial for prediction as they contribute to many classes (diseases), which are also connected to many other symptoms.

- **High L1 - Low L2**: Symptoms with high degree and low L2. These symptoms should not be removed a priori since they can be useful. For instance, a symptom may be associated with many diseases, but those diseases may only be associated with that symptom. In this case, the symptom is very important for prediction.

- **Low L1 - High L2**: Symptoms with low degree and high L2. These symptoms may be important for prediction, as they contribute to few diseases.

- **Low L1 - Low L2**: Symptoms with low degree and low L2. These symptoms are the most important for prediction since they contribute to few classes (diseases), and those classes are also connected to few other symptoms.

According to the above considerations, we can start from the last class and iteratively add symptoms from the other classes, monitoring the impact on both accuracy and training time. The results are analyzed in Section 7.
Figures 15 and 16 illustrate the division of symptoms into the four classes. The same entire analysis was conducted for diseases, in this case focusing on the high-L1-high-L2 class, which contains the most complex diseases under a symptomatology perspective. The results are reported in Figures 29 and 30.

To provide a complete picture of the most important symptoms we report the composition of the low-L1-low-L2 class in Figure 17.

6. ML Model Methodology

This section provides a comprehensive overview of the methodologies employed in the construction of the machine learning model. The discussion encompasses various techniques designed to handle the intricacies of model building, coupled with a logical flow that guides the entire process.

a. Preliminary Data Preparation

Before delving into model development, a data preprocessing pipeline was employed to properly prepare them for the subsequent steps, facing the problems of class imbalance and training computational complexity.

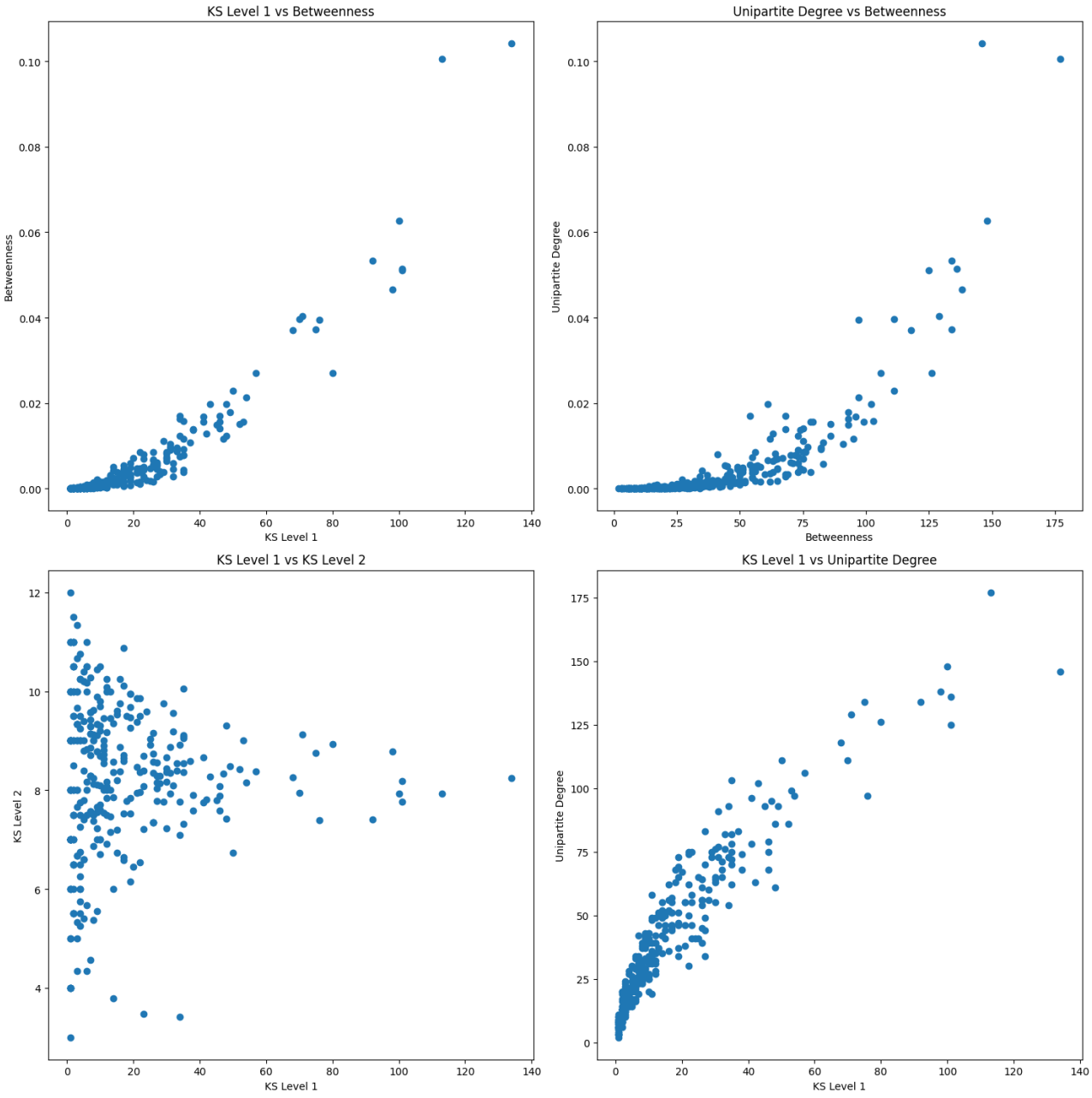

Fig. 14: Correlation between features

- **Random Sampling**: Given the extensive nature of hyperparameter tuning, we adopted a random sampling strategy, picking around 10% of the dataset. Instead of training the models on the entire dataset for each hyperparameter combination, the random subset was used to expedite the tuning phase without sacrificing model representativity.

- **Class Imbalance with Oversampling and Undersampling**: The dataset was highly unbalanced across its 700 disease classes. To mitigate this, a combination of oversampling and undersampling techniques was applied. The former was performed for minority classes, while the latter was applied to the majority classes, ensuring all the diseases were adequately represented during training, preventing dominance and biases in the model.

b. Feature Extraction

A pivotal phase in constructing a machine learning model is feature extraction. In addition to the one-hot vector representation of symptoms, the network analysis affords us the following features:

- **L1 and L2 Measures**: A vector with values representing the L1 and L2 measures for each symptom.

- **Betweenness Centrality**: A vector with values denoting the betweenness centrality of each symptom.

- **Community Count**: A vector indicating the number

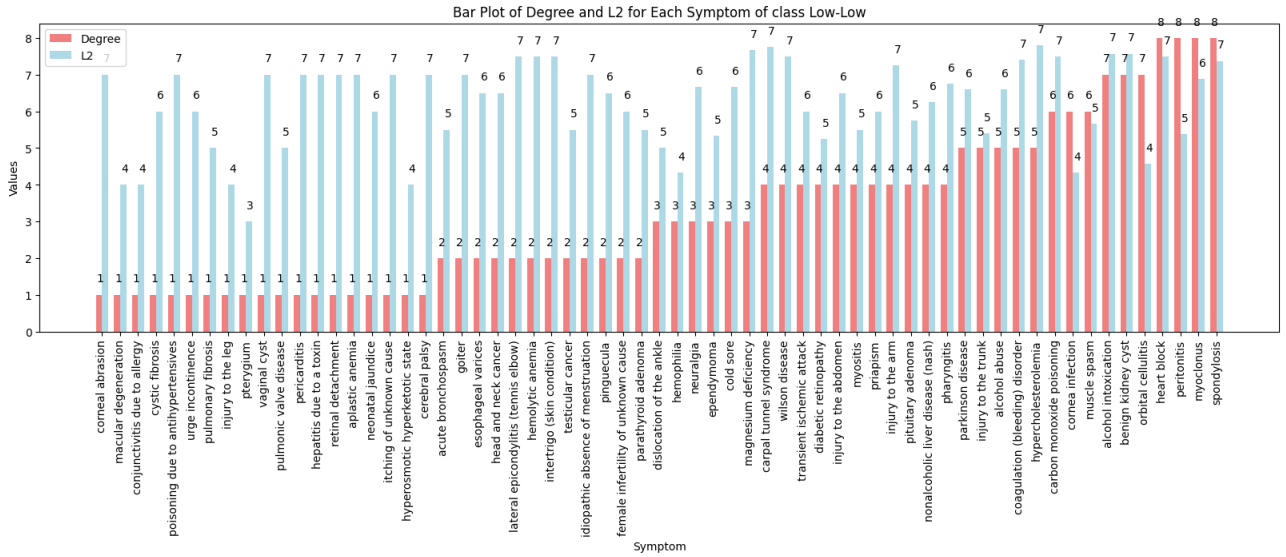

Fig. 17: Composition of the low-L1-low-L2 class for symptoms

of symptoms belonging to each community.

- **Community Size**: A vector replacing symptoms with the size of the community to which they belong.

Given the diverse scales of these features, normalization becomes imperative for their cohesive integration into the model without introducing biases. To achieve this, we opted for *MaxAbs* normalization. This normalization scales each feature individually, ensuring that the maximal absolute value of each feature in the training set becomes 1.0, while preserving the sparsity of data.

c. Model Choice

The number of machine learning classification models available for disease prediction is vast. We decide to focus on three models that are widely used in the literature and that are known to perform well in a variety of contexts: Logistic Regression, Random Forest, and Multilayer Perceptron (MLP).

Logistic Regression

- **Strengths**: Logistic Regression's computational efficiency makes it an attractive choice for initial exploration and baseline performance assessment. Its simplicity facilitates interpretability, providing insights into the impact of individual symptoms on disease prediction.
- **Considerations**: While efficient, Logistic Regression assumes a linear relationship between features and the log-odds of the target, potentially limiting its ability to capture complex non-linear patterns.

Random Forest

- **Strengths**: Random Forest is renowned for its robustness in handling large and diverse datasets, making it well-suited for our expansive dataset with 700 disease classes. Moreover, Its ability to capture non-linear relationships ensures that complex patterns within the symptoms' one-hot encoded features are effectively modeled.
- **Considerations**: The ensemble nature of Random Forest provides resilience against overfitting, a crucial factor in the context of disease prediction.

Multilayer Perceptron (MLP)

- **Strengths**: MLPs are adept at capturing intricate relationships in high-dimensional datasets, aligning with the complexity inherent in our 300-feature symptom representation.
- **Considerations**: Their capacity for adapting to non-linear mappings positions MLPs as powerful tools in unraveling the nuanced interactions between symptoms and diseases.

d. Operative Flow

Once the features are ready, the core part of the model-building process can begin. The operational flow was quite complex, and it is summarized in Figure 18.
We trained three different models: a Logistic Regression, a Random Forest, and a Multi-Layer Perceptron (MLP).
For each model, we faced the challenge of selecting both the best hyperparameters and the most effective features. The interdependence between these two aspects makes the optimal approach to explore all the possible combination of features and for each combination trying all the hyperparameters combination. This approach is not feasible in terms of computational effort leading us to adopt a greedy approach. We firstly split the features into two groups: the symptoms' one-hot vector and the remaining features. The former is used to train a base model, while the latter is utilized to explore the potential improvement brought by the new features.

Fig. 18: Operative Flow of the ML Model

Algorithm 1 Feature Selection Algorithm

```
RemainingFeatures ← AllFeatures
FeatureSet ← EmptySet
BestAccuracies ← EmptySet
Parameters ← InitializeRandomParameters
i ← 0
while RemainingFeatures is not Empty do
    BestAccuracy ← 0
    for each feature in RemainingFeatures do
        CurrentFeatureSet ← feature ∪ FeatureSet
        Model ← EmptyModel
        TrainModel(odel, CurrentFeatureSet, Parameters)
        CurrentAccuracy ← GetAccuracy(Model)
        if CurrentAccuracy >= BestAccuracy then
            BestAccuracy ← CurrentAccuracy
            BestFeature ← feature
    BestAccuracies[i] ← BestAccuracy
    RemainingFeatures ← RemainingFeatures − BestFeature
    FeatureSet ← BestFeature ∪ FeatureSet
    BestFeatureCombinations[i] ← FeatureSet
    i ← i + 1
BFC ← BestFeatureCombinations[ArgMax(BestAccuracies)]
return BFC
```

Using Algorithm 1, we determined the best feature combination for each group (symptoms and other features). Subsequently, given the optimal feature combination, we identified the best Hyperparameters combination using Algorithm 2. Each model was then trained with the best hyperparameters and the best features combination. For each group of three models available at this point (3 models with symptoms and 3 models with other features), we selected the best one according to their accuracy value (Section a). Only at this point the two winning models were trained with the whole dataset to provide a more precise evaluation of their performance and were compared to assess the result of our first Goal.
As regard the last Goal, the best between the above two models undergone the feature reduction process discussed in Section d and its computational time was compared to the one of the full features model.

The quest for optimal hyperparameters (hparams) in machine learning models is often constrained by computational resources. In light of these limitations, we adopted a resource-efficient greedy search strategy to navigate the vast hyperparameter space. Our approach unfolds in several stages. Initially, we randomly initialize hyperparameters (hparams) to initiate the search. Subsequently, we employ a stepwise exploration, beginning with the first hyperparameter. For this, we perform an initial search over a small set of values (e.g., 0.001, 0.01, 1, 10, 100). If the optimum lies at one of the extremes, we extend the search to encompass values in the corresponding direction.; conversely, if it resides within an intermediate range, we conduct a more focused exploration in a narrower interval. This process is iteratively repeated for each hyperparameter, gradually refining our understanding of the optimal regions within the hyperparameter space. This iterative approach serves a dual purpose. First, it conserves computational resources by avoiding an exhaustive search over all potential combinations. Second, it capitalizes on the information gleaned from earlier iterations to guide subsequent searches efficiently. By strategically determining the next set of values based on the observed trends, we strike a balance between exploration and exploitation, ultimately converging to a set of hyperparameters (hparams) that maximizes model performance. This resource-conscious strategy is paramount when computational resources are limited, allowing us to derive meaningful results within practical constraints.

Algorithm 2 Greedy Hyperparameter Search

```
hparams ← randomInitialization
for each hparam in hparams do
    src_range ← initialSrcRange
    best_value ← curr_hparams[hparam]
    accuracy ← 0
    for value in src_range do
        curr_hparams[hparam] ← value
        performance ← eval(model, curr_hparams)
        if performance better than accuracy then
            best_value ← value
            accuracy ← performance
    if best_value is at the lower extreme then
        src_range ← extendSrcRangeLower(best_value)
    else if best_value is at the upper extreme then
        src_range ← extendSrcRangeUpper(best_value)
    else
        src_range ← narrowSrcRange(best_value)
    for value in src_range do
        curr_hparams[hparam] ← value
        performance ← eval(model, curr_hparams)
        if performance better than accuracy then
            best_value ← value
            accuracy ← performance
    curr_hparams[hparam] ← best_value
```

At the conclusion of these procedures, we obtained the following two models:

- **Symptoms Model**: The best model with the optimal hparams and the symptoms as features
- **Other Features Model**: The best model with the optimal hyperparameters and the best features combination

7. ML Model Results

In this section we present the results of the ML models we have trained. We then deeply inspect the best performing model, in order to understand its features and its performance.

a. Model Selection

As previously shown by the operative flow in Figure 18, there are several phases involved in the selection of the optimal prediction model. Given our limited resources we chose to take a greedy approach by performing the feature selection first, and then optimizing the hyperparameters at a later time for each of the three models considered.

Features Selection

To reduce the amount of time spent training the models to select the best hyperparameters, it is best to first limit the number of features considered. The selection of the most useful features was performed using a forward stepwise selection, following a greedy a approach that aims at maximizing the accuracy. The hyperparameters were initalized with the default values provided by the library scikit-learn.

As depicted in Figure 19, we can see that the best accuracy with the logistic regression model is reached after the third iteration, with little improvement with respect to the model using a single feature. This kind of model seems to favor information about communities and centrality, while the L1 and L2 measures in some cases actually worsen its performance.

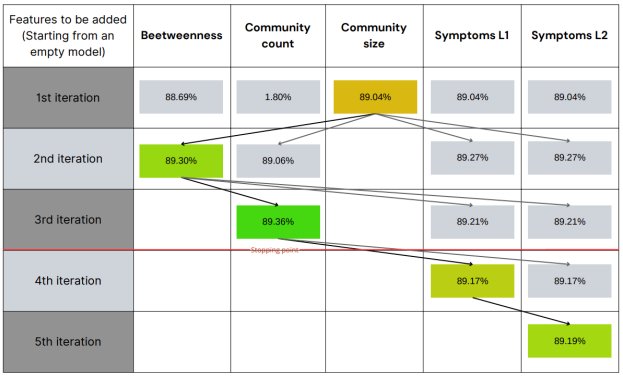

Fig. 19: Accuracy of the logistic regression models over the iterations of the forward stepwise feature selection

Figure 20 shows that the random forest models behave similarly to the logistic regression, as the best accuracy is reached after the same number of steps, employing the same features. These results start to reveal which features are the best when it comes to classification. It is also worth noting that the best random forest model has a slightly worse accuracy than logistic regression, but that might be due to the random choice of the model's parameters.

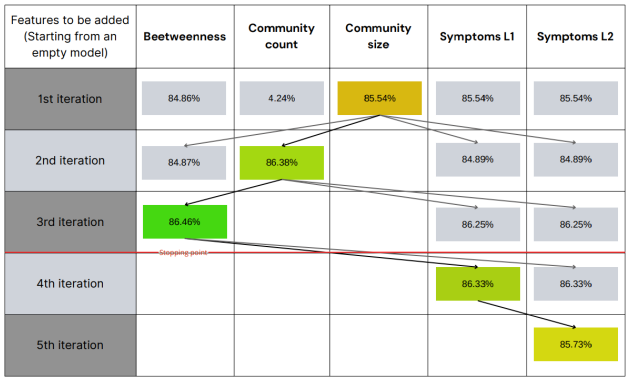

Fig. 20: Accuracy of the random forest models over the iterations of the stepwise feature selection

The multi-layer perceptron model exhibits a different performance from the other two, given by the fact that the accuracy starts to lower after adding the second feature. As underlined by Figure 21, with respect to the other models information about the betweenness is not needed to achieve the best possible results, which could be explained by the better ability of the MLP to adapt to non-linearities. This particular implementation of neural network has one hid-

den layer with 100 neurons, and in our case its performance is better than the random forest model, but still slightly worse than the logistic regression. We expect this to change after the optimization of the hyperparameters.

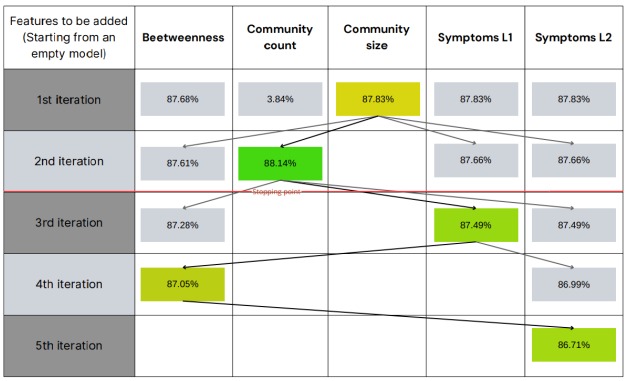

Fig. 21: Accuracy of the MLP models over the iterations of the stepwise feature selection

Hyperparameters Selection

Model Comparison

As depicted in Figure 18, we now have six different models: three with only symptoms and three with both symptoms and new features. The best model was selected from each group based on their accuracy values. Figures 22 and 23 indicate consistently low overfitting and similar performance across all models in both groups. Despite their varying complexity, the models achieve similar performance on the test set, meaning that a linear separation boundary is sufficient to properly classify the features vector. This founds justification in the highly discriminative nature of the features of this dataset.

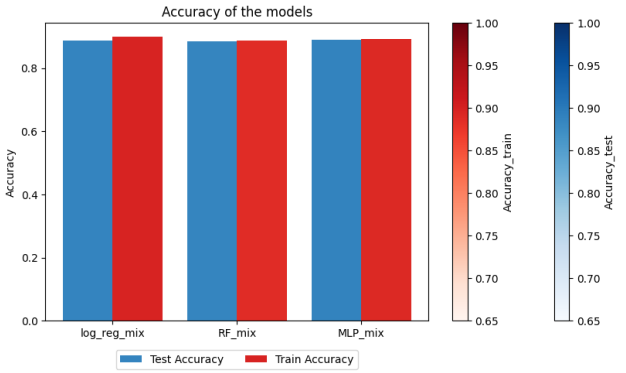

Fig. 23: Accuracy of the three models with new features

b. New Features Effect

The best model from each group was further trained on the full balanced dataset to ensure a more reliable performance evaluation. The results in Figure 24 reveal a minimal difference between the two groups. This addresses our **first goal**: the new features, while not improving the model, offer a comparable performance to using symptoms alone. However, It's essential to note that the new features are more numerous than the symptoms, contributing to a more complex model. In conclusion, the extracted network features are not a superior alternative to symptoms.

It is noteworthy that the 'simplicity' of the dataset, leads to a very high accuracy in all models, thus may also affect the performance evaluation of the new features, which have a small room to improve the model. Therefore, a possible avenue for future exploration could involve the use of more complex dataset, to better assess the performance of the new features.

Another viable option for future work is to use the new features as a complement to the symptoms.

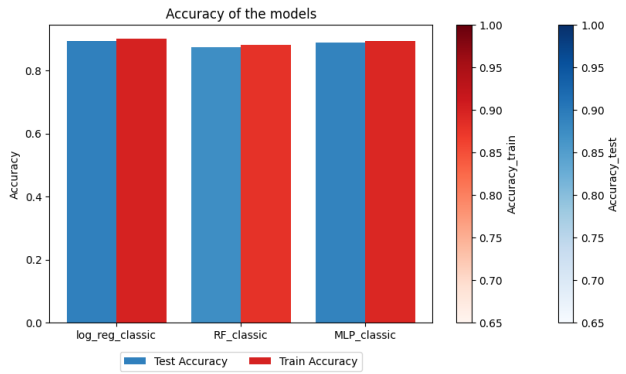

Fig. 22: Accuracy of the three models with only symptoms

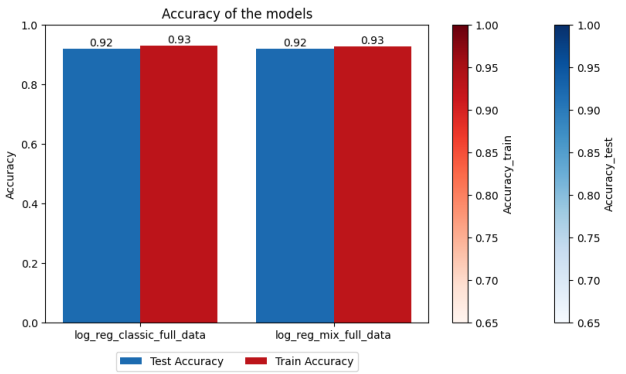

Fig. 24: Accuracy of the best models from both groups

c. Best Model Analysis

d. Computational Complexity

8. Conclusion

9. Future Works

10. Appendix

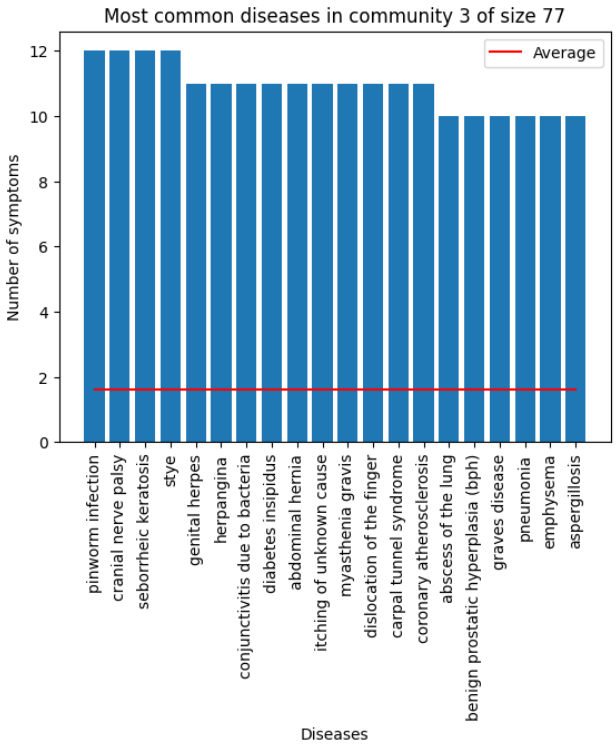

Fig. 26: Community 3 of symptoms

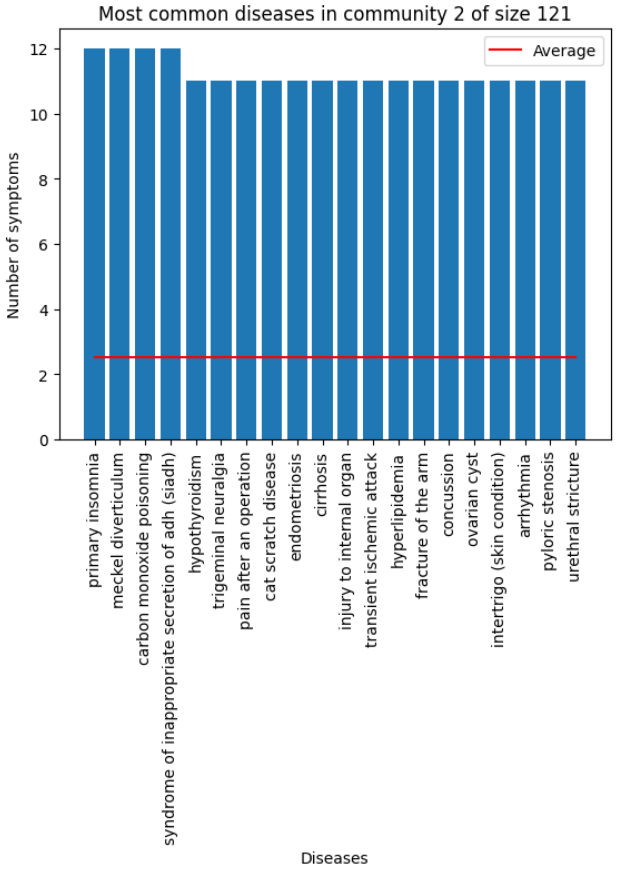

Fig. 25: Community 2 of symptoms

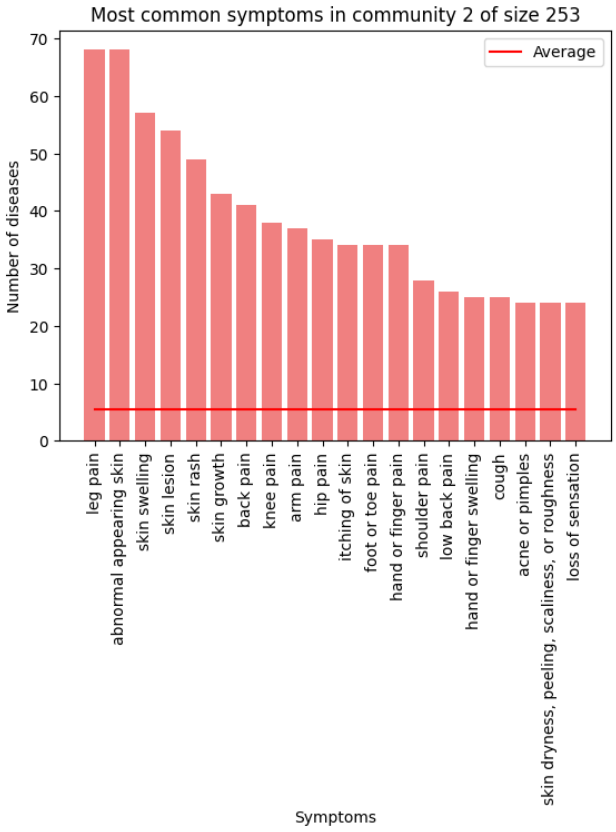

Fig. 27: Community 2 of diseases

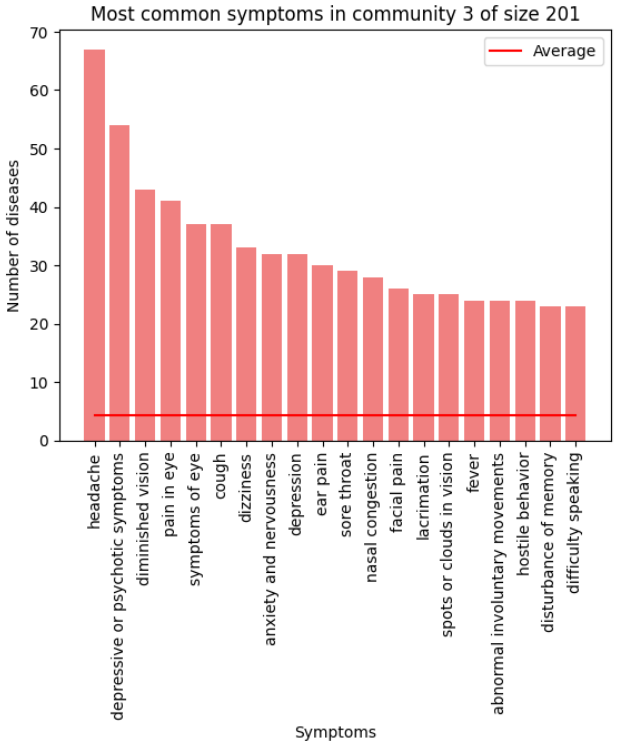

Fig. 28: Community 3 of diseases

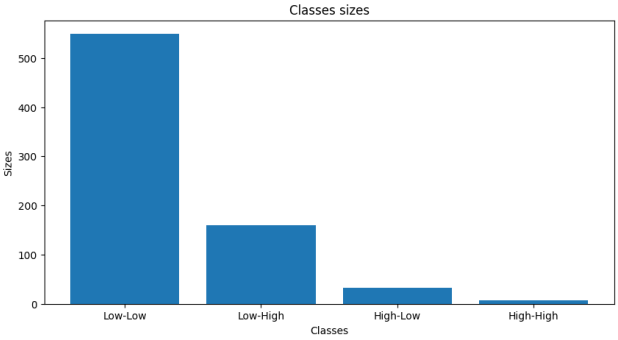

Fig. 29: Diseases divided into the four classes

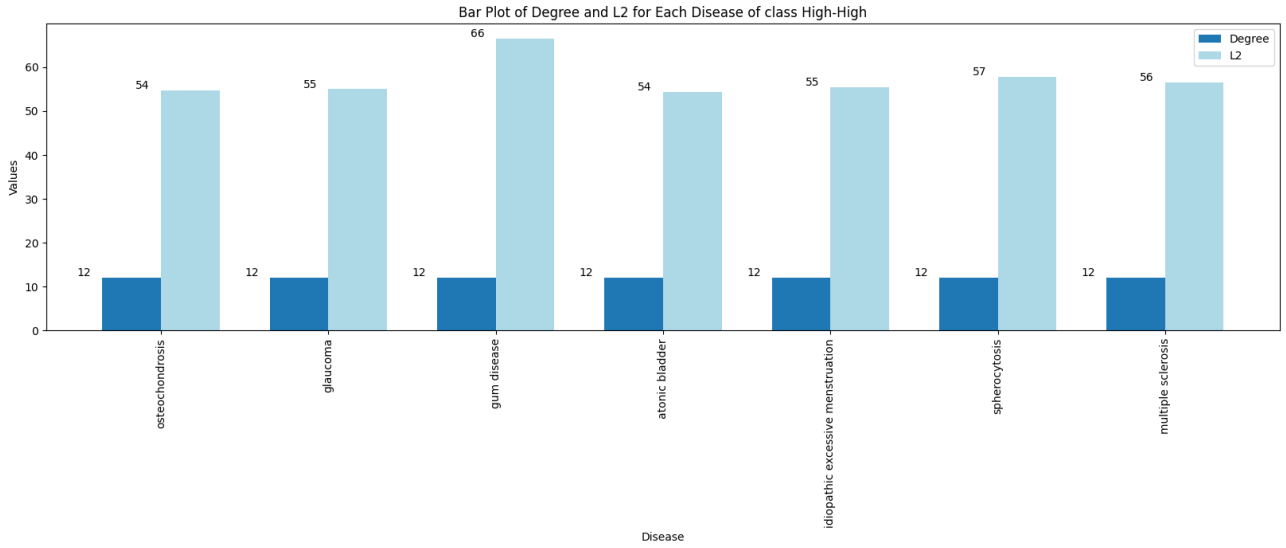

Fig. 30: Composition of the high-L1-high-L2 class for diseases

REFERENCES

[1] Ulrik Brandes. "A Faster Algorithm for Betweenness Centrality". In: *The Journal of Mathematical Sociology* 25 (Mar. 2004). DOI: 10.1080/0022250X.2001.9990249.

[2] Ulrik Brandes. "On variants of shortest-path betweenness centrality and their generic computation". In: *Social Networks* 30.2 (May 2008), pp. 136–145. ISSN: 0378-8733. DOI: 10.1016/j.socnet.2007.11.001.

[3] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. "Finding community structure in very large networks". In: *Physical Review E* 70.6 (Dec. 2004). arXiv:cond-mat/0408187, p. 066111. ISSN: 1539-3755, 1550-2376. DOI: 10.1103/PhysRevE.70.066111.

[4] C. A. Hidalgo et al. "The Product Space Conditions the Development of Nations". In: *Science* 317.5837 (July 2007), pp. 482–487. ISSN: 0036-8075. DOI: 10.1126/science.1144581.

[5] César A. Hidalgo and Ricardo Hausmann. "The building blocks of economic complexity". In: *Proc. Natl. Acad. Sci. U.S.A.* 106.26 (June 2009), pp. 10570–10575. DOI: 10.1073/pnas.0900943106.

[6] M. E. J. Newman. "Modularity and community structure in networks". In: *Proceedings of the National Academy of Sciences of the United States of America* 103.23 (June 2006), pp. 8577–8582. ISSN: 0027-8424. DOI: 10.1073/pnas.0601602103.

[7] Alessandro Spelta, Nicoló Pecora, and Paolo Pagnottoni. "Assessing harmfulness and vulnerability in global bipartite networks of terrorist-target relationships". In: *Social Networks* 72 (Jan. 2023), pp. 22–34. ISSN: 0378-8733. DOI: 10.1016/j.socnet.2022.08.003.

[8] Tiziano Squartini, Giorgio Fagiolo, and Diego Garlaschelli. "Randomizing world trade. I. A binary network analysis". In: *Phys. Rev. E* 84.4 (Oct. 2011), p. 046117. ISSN: 2470-0053. DOI: 10.1103/PhysRevE.84.046117.

[9] Tiziano Squartini, Giorgio Fagiolo, and Diego Garlaschelli. "Randomizing world trade. II. A weighted network analysis". In: *Phys. Rev. E* 84.4 (Oct. 2011), p. 046118. ISSN: 2470-0053. DOI: 10.1103/PhysRevE.84.046118.