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 ScienSickness predictionce, University of Pavia, Italy*
Course of Financial Data Science

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

Date: December 9, 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 landscape of healthcare, understanding the intricate interplay between symptoms and diseases is paramount for effective diagnosis and prediction. This report embarks on a comprehensive journey through the realms of network analysis, leveraging both theoretical foundations and empirical data to unravel the complexities of symptom-disease interactions. Our dual-fold objective is to provide a nuanced descriptive analysis of these interactions while identifying key features to bolster predictive models.

The foundation of this endeavor lies in an extensive review of existing literature, drawing insights from seminal works on network theory and disease prediction. By establishing a baseline through prior research, we pave the way for a deeper understanding of the subject matter and ensure the relevance of our findings in the broader context of scientific inquiry.

Guided by insights gleaned from the literature, our exploration extends to the realm of data, where we meticulously curate and analyze datasets of varying sizes. Through a systematic process of exploratory data analysis and cleaning, we prepare the groundwork for constructing meaningful networks that encapsulate the relationships between symptoms and diseases.

The heart of our analysis lies in the creation of intricate network structures, employing bipartite models and non-weighted links to distill meaningful patterns. We delve into a spectrum of network metrics, from fundamental measures like degree distribution and clustering coefficients to more nuanced assessments of node importance and between-

ness centrality. Statistical significance is rigorously assessed through the lens of a null model, ensuring that our observations transcend mere chance.
Community detection algorithms further dissect the network, revealing hidden structures and relationships between diseases. This not only enriches our understanding but also lays the groundwork for subsequent analyses. As we traverse the terrain of network analysis, we introduce novel metrics inspired by the Hidalgo-Hausmann framework, stratifying symptoms and diseases based on their predictive importance. These metrics, coupled with traditional measures like betweenness centrality, contribute to the definition of features that fuel our predictive models.
With a robust foundation established, we transition to the realm of predictive modeling, where our feature-rich approach promises to enhance the performance of established models. Logistic regression, random forest, and multi-layer perceptron models are trained, tested, and validated, with a keen eye on feature importance and model improvement strategies.
This report unfolds as a holistic exploration, weaving together theoretical frameworks, empirical analyses, and predictive modeling into a cohesive narrative. As we traverse the intricate web of symptom-disease interactions, our aim is not only to elucidate the underlying dynamics but also to pave the way for more accurate and insightful predictive models in the realm of healthcare.

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 primary 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. **Computational Efficiency Improvement:** 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.
«««< HEAD

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 3 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 4 presents the unipartite graphs of symptoms and diseases.

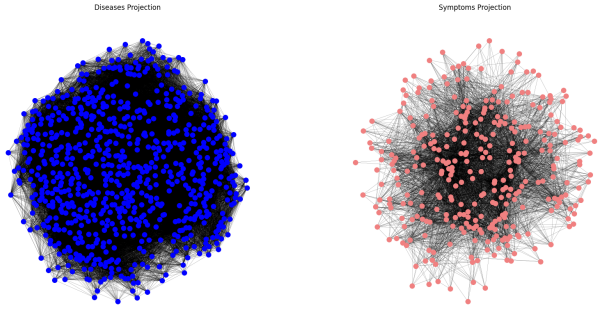


Fig. 2: Visual representation of symptom and disease unipartite graphs.

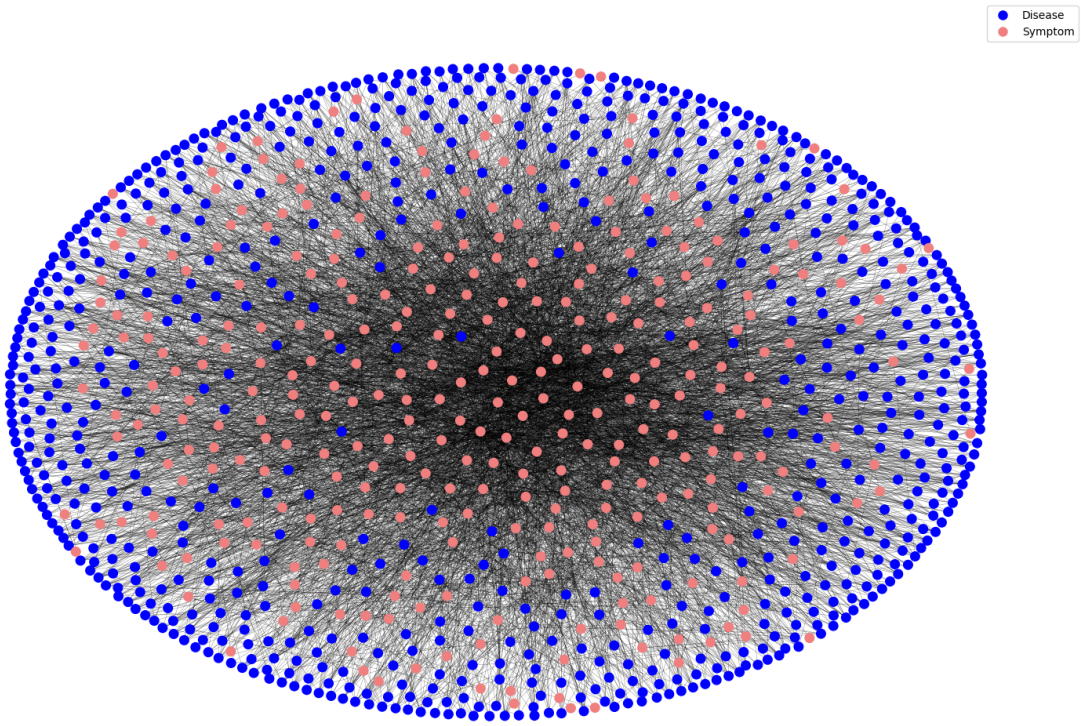


Fig. 1: Visual representation of the symptom-disease bipartite graph.

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 symptom commonality. This commonality measures 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 more detail, level-1 of these indices quantifies the number of symptoms associated with a disease, while level-2 measures the commonality of these symptoms. In network theory, level-2 measures are recognized as the average nearest neighbor strength.

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 distri-

bution 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 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 3 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 4 presents the unipartite graphs of symptoms and diseases.

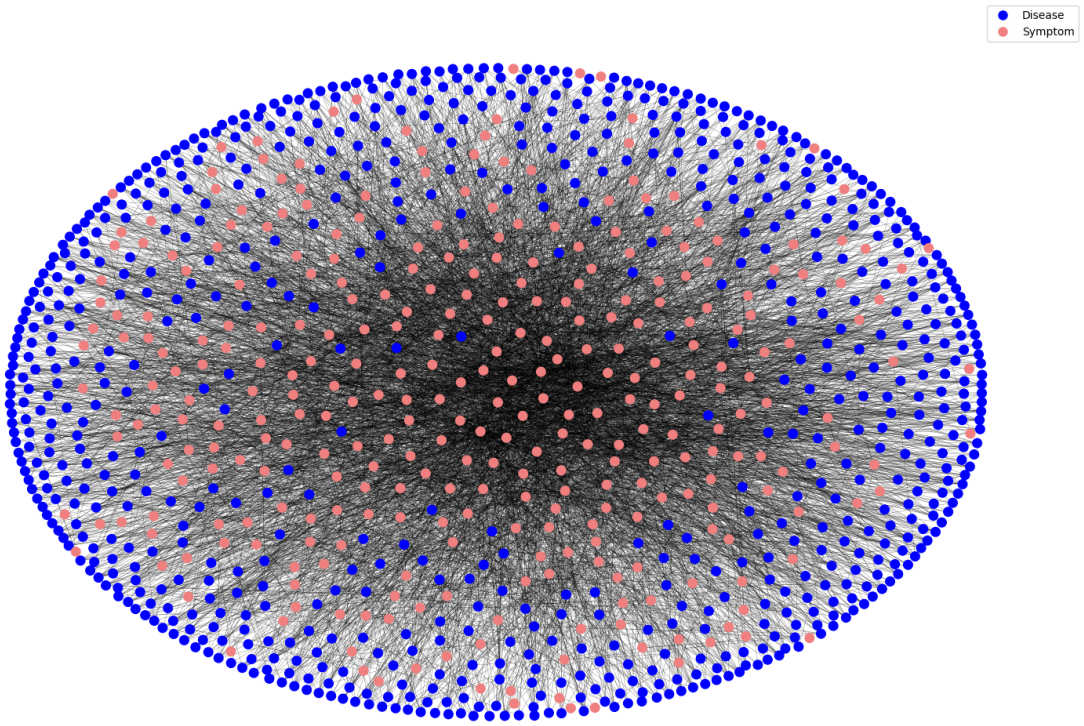


Fig. 3: Visual representation of the symptom-disease bipartite graph.

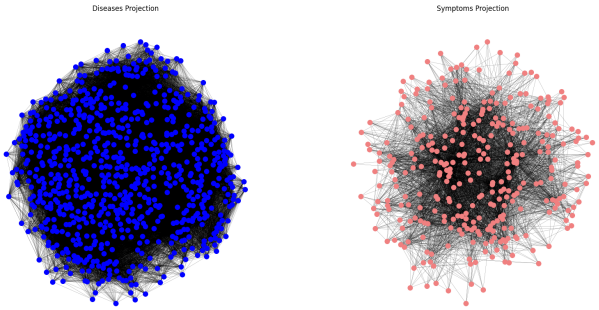


Fig. 4: Visual representation of symptom and disease unipartite graphs.

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 symptom commonality. This commonality measures 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 more detail, level-1 of these indices quantifies the number of symptoms associated with a disease, while level-2 measures the commonality of these symptoms. In network theory, level-2 measures are recognized as the average nearest neighbor strength.

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:

$$SI_{v,N} = \frac{1}{SI_{v,1}} \sum_{u} W(v,u) DI_{u,N-1} \tag{8}$$

$$DI_{u,N} = \frac{1}{DI_{u,1}} \sum_{v} W(v,u) SI_{v,N-1} \tag{9}$$

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:

$$SI_{v,1} = \sum_{u} W(v,u) \tag{10}$$

$$DI_{u,1} = \sum_{v} W(v,u) \tag{11}$$

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:

$$z_{SI_{v,2}} = \frac{SI_{v,2} - \mu_{SI_{v,2}}}{\sigma_{SI_{v,2}}} \tag{12}$$

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:

$$c_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)} \tag{13}$$

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:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{14}$$

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.

»»»> c1f1783e81797716f18a59a4644bbabaf1375634

6. NETWORK RESULTS

a. L1 and L2 Metrics

b. Betweenness Centrality

The examination of betweenness centrality in our bipartite network, as depicted in Figure 5, 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.

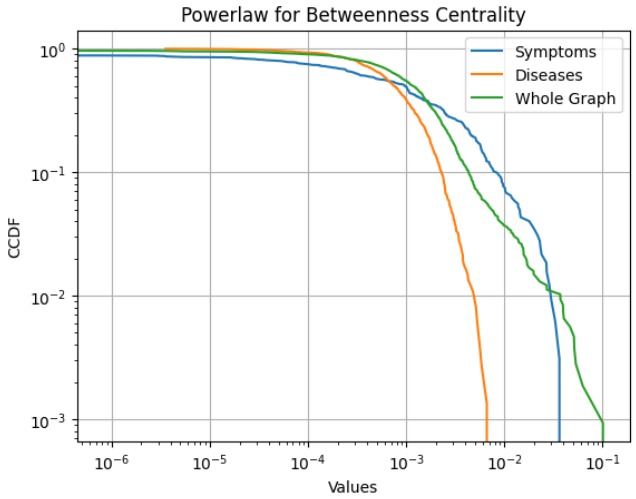


Fig. 5: Betweenness Centrality CDFs

Upon dissecting the centrality values into symptoms and diseases (see Figures 6 and 7), 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.

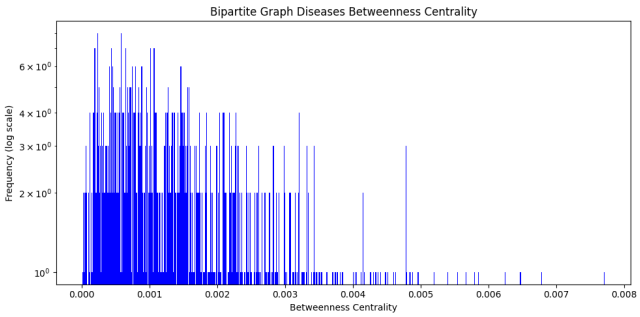


Fig. 6: Betweenness Centrality of the diseases

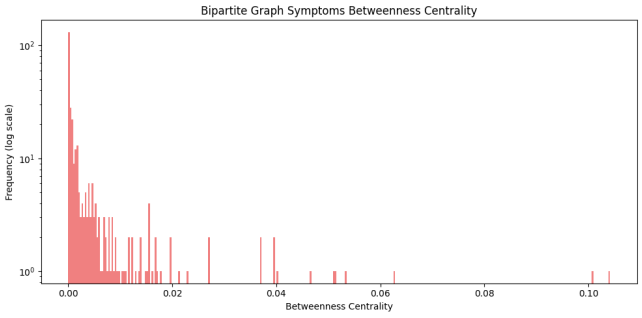


Fig. 7: Betweenness Centrality of the symptoms

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 8 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.

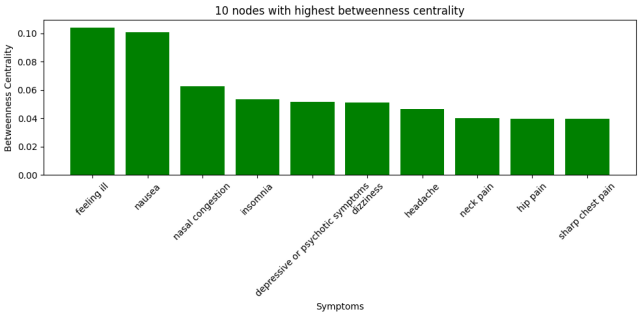


Fig. 8: Top 10 nodes with the highest betweenness centrality

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 9.

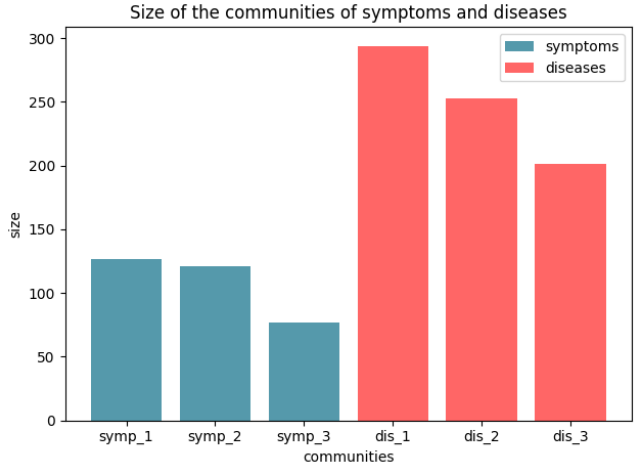


Fig. 9: Sizes of the communities of symptoms and diseases

For clinical relevance, examining symptoms communities provides valuable information about diseases associated with these symptoms. This is exemplified in Figures 10, 14, and 15. As an illustration, in the community 1 of symptoms (Figure 10), 'herniated disk' is the third most pointed disease by the symptoms of the community, with each symptom pointing, on average, to three diseases.

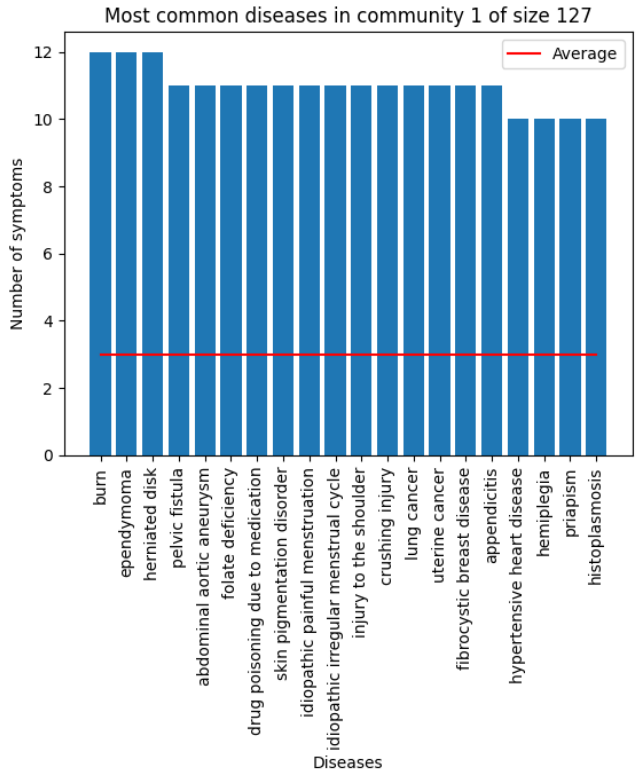


Fig. 10: Community 1 of symptoms

A similar study can be conducted for communities of diseases, as depicted in Figures 11, 16, and 17. This information aids in profiling diseases and understanding the significance of each symptom. For instance, in community 1 of diseases (Figure 11), the symptom 'sharp abdominal pain' is present in almost half of the diseases in the community, indicating its generic nature and limited discriminatory value.

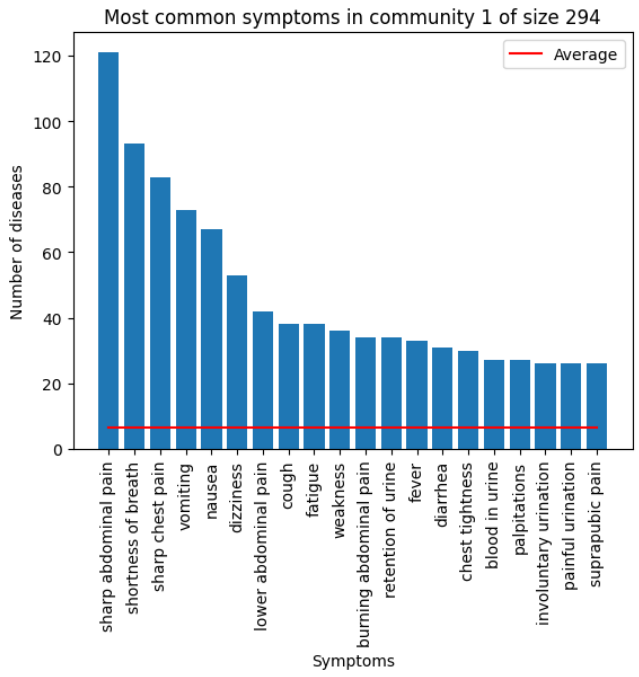


Fig. 11: Community 1 of diseases

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, injecting community information into the model beyond basic one-hot encoding of symptoms.

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. We performed a graphical analysis of their correlation (Figure 12), favoring the most uncorrelated features to provide the most complementary information.

Despite a non-negligible positive correlation in all cases, we opted for L1 and L2 due to their interpretability and the less pronounced positive correlation at lower L1 values. This allows for a substantial division of symptoms for the same value of L1 but different values of L2. Figure 13 illustrates the possibility of placing a proper threshold on L1 and L2 to obtain four classes of symptoms, each with a non-null number of symptoms.

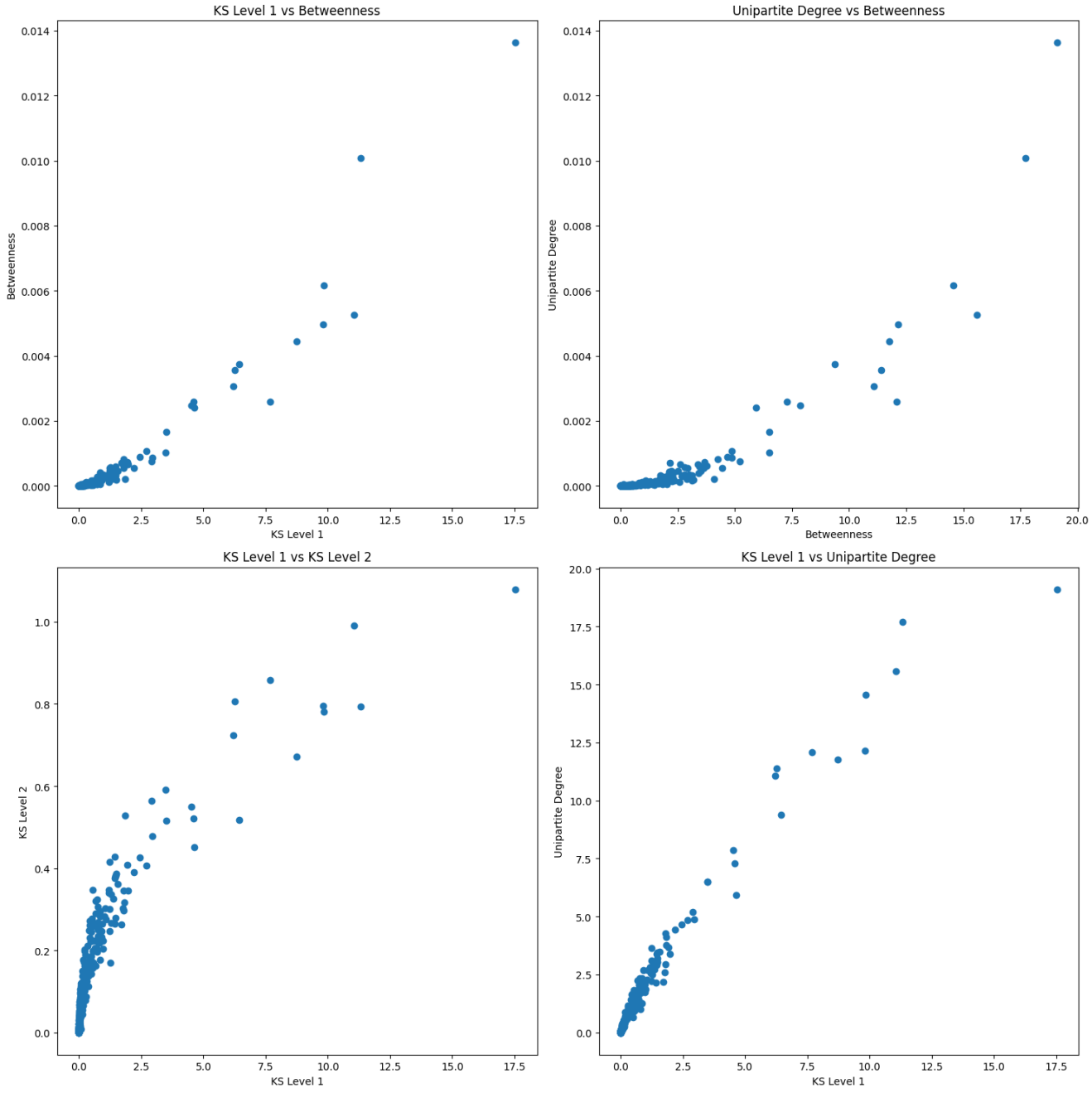
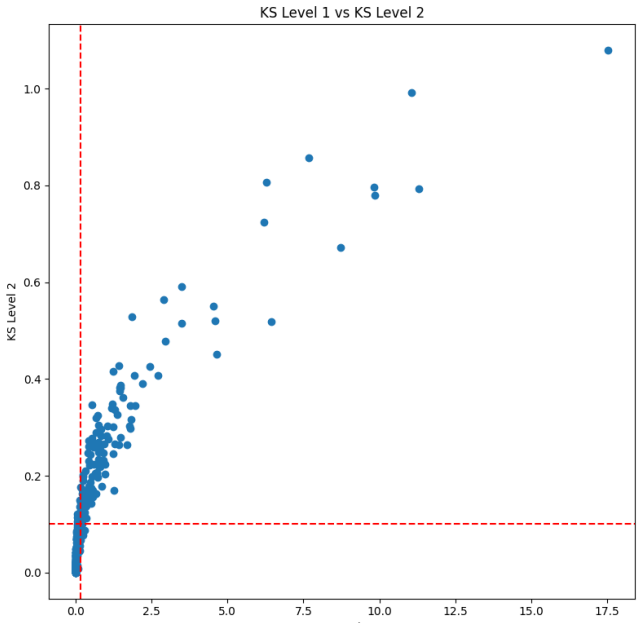


Fig. 12: Correlation between features



To interpret these classes, we briefly recall the meanings of L1 and L2. The former represents the number of diseases associated with a symptom, while the latter measures the number of symptoms associated with those diseases. Consequently, we define 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 8.
It is worth saying that all the other features (betweenness centrality, community count, and community size) represent a valuable alternative to L1 and L2. Their use in computational time reduction is not explored in this work, but it is a possible future development.

7. 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.

- **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 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. 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 parameters 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 parameters 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.

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 parameter combination using Algorithm 2. Each model was then trained with the best parameters and the best features combination, and the model with the highest accuracy was chosen. Finally, the best overall model was trained with the entire dataset.

Algorithm 1 Feature Selection Algorithm

```
BestFeatureComb ← EmptySet
for each model do
    CurrentModel ← EmptyModel
    BestModAccuracy ← 0
    Parameters ← InitializeRandomParameters
    for i = 1 to NumberOfFeatures do
        BestAccuracy ← -1
        for each feature do
            TrainModel(CurrentModel, Parameters)
            CurrentAccuracy ← GetAccuracy(CurrentModel)
            if CurrentAccuracy >= BestAccuracy then
                BestAccuracy ← CurrentAccuracy
                BestFeature ← feature
            end if
            UpdateModel(CurrentModel, BestFeature)
        end for
        FreezeFeatures(CurrentModel)
        ModelAccuracy[i] ← GetAccuracy(CurrentModel)
        FeaturesComb[i] ← GetFeat(CurrentModel)
    end for
    BestComb ← FeaturesComb[Max(ModelAccuracy)]
    BestFeatureComb ← BestComb ∪ BestFeatureComb
end for
return BestFeatureComb
```

Algorithm 2 Best Model Selection Algorithm

```
CurrentAccuracy ← 0
for each model do
    CreateGrid(Parameters)
    BestParameters ← GridSearchCV(model)
    TrainModel(model, BestParameters)
    CurrentAccuracy ← GetAccuracy(model)
    if CurrentAccuracy >= BestAccuracy then
        BestAccuracy ← CurrentAccuracy
        BestModel ← model
        BestParameters ← Parameters
    end if
end for
FullDataTrain(BestModel, BestParameters)
BestAccuracy ← GetAccuracy(BestModel)
ReduceFeatures(BestModel, BestParameters)
return BestParameters, BestModel, BestAccuracy
```

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

- **Symptoms Model**: The best model with the optimal parameters and the symptoms as features

- **Other Features Model**: The best model with the optimal parameters and the best features combination

With these models in hand, we could compare their prediction performance to evaluate whether the network features contributed to an accuracy improvement. Additionally, we applied the feature reduction technique discussed in Section d to both models, assessing whether network features could reduce computational complexity without compromising accuracy.

8. 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

b. New Features Effect

c. Best Model Analysis

d. Computational Complexity

9. Conclusion

10. Future Works

11. Appendix

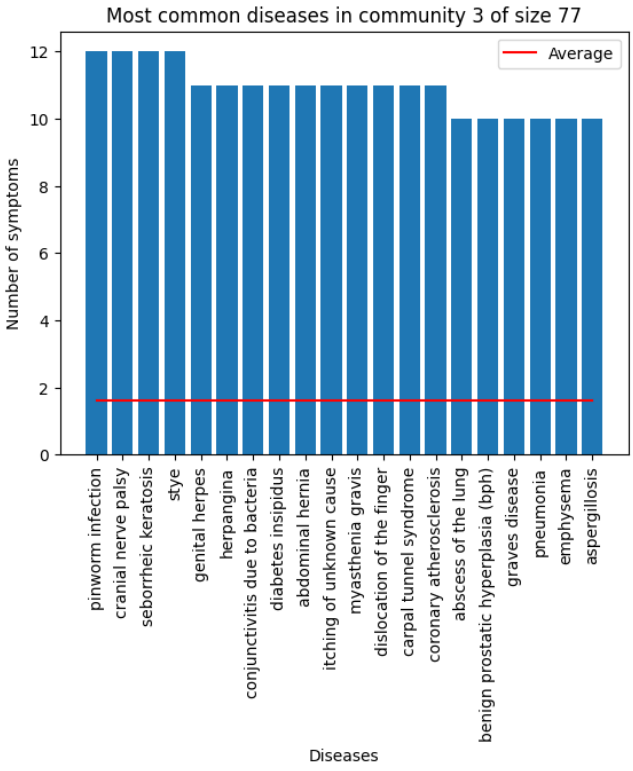


Fig. 15: Community 3 of symptoms

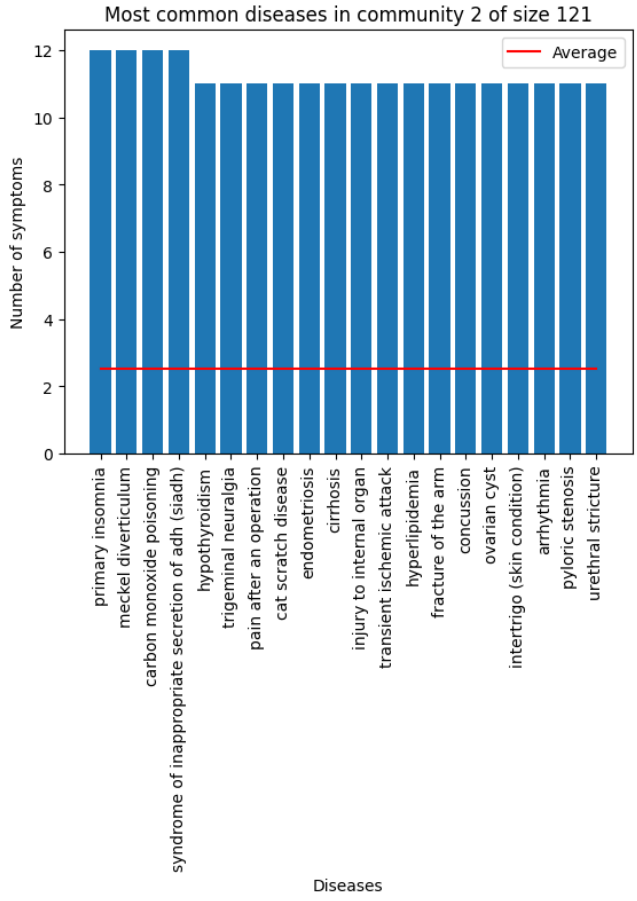


Fig. 14: Community 2 of symptoms

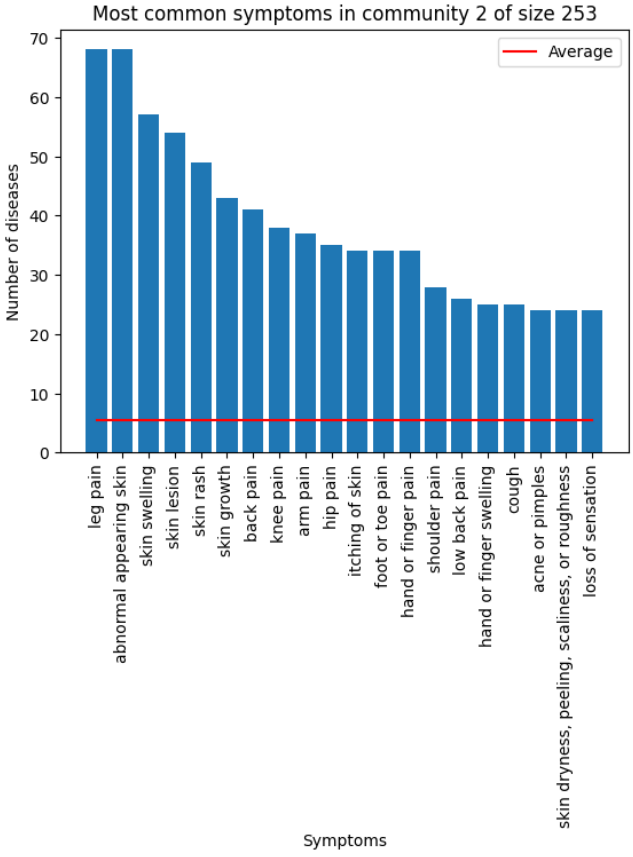


Fig. 16: Community 2 of diseases

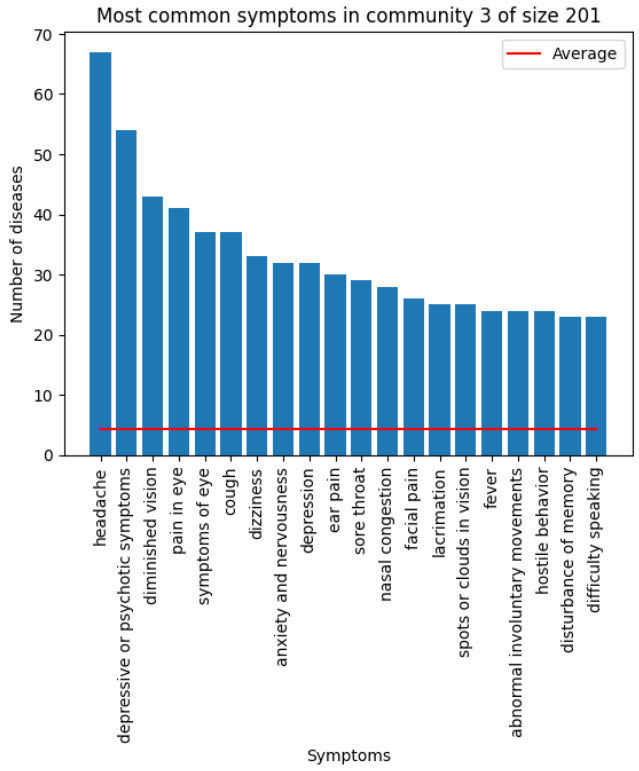


Fig. 17: Community 3 of diseases

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.