HUB PROTEIN IDENTIFICATION OF CORONAVIRUS DISEASE : A MACHINE LEARNING PRACTICE

This project report is submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology in Information Technology at the Department of Engineering and Technological Studies, University of Kalyani.

Submitted by

Debasis Maji

Under Supervision of

Dr. Nabin Ghoshal

DEPARTMENT OF ENGINEERING AND TECHNOLOGICAL STUDIES

UNIVERSITY OF KALYANI

KALYANI, NADIA, WEST BENGAL-741235

CERTIFICATE

DEPARTMENT OF ENGINEERING AND TECHNOLOGICAL STUDIES
UNIVERSITY OF KALYANI
KALYANI, NADIA
WEST BENGAL - 741235

This is to certify that the project report entitled as“ **Hub Protein identification of Coronavirus Disease: A Machine Learning Practice** ” has completed by **Debasis Maji (Registration No: 100218 of 2020-2021)** in partial fulfilment of the requirement for the degree of Bachelor of Technology (B.Tech) in Information Technology at the Department of Engineering and Technological Studies, University of Kalyani, under the supervision of **Dr. Nabin Ghoshal** and **Mr. Arup Mallick.**

During the project, Debasis Maji has gained hands-on experience in using machine learning algorithms and graph analysis tools to identify hub proteins and pathways involved in COVID-19 pathogenesis. The participant has demonstrated proficiency in data preprocessing, feature selection, model training and validation, and visualization of the results.

________________	__________________	__________________
Mr. Arup Mallick	Dr. Nabin Ghoshal	Dr. Srinka Basu
Project Guide	Project Guide	Head of The Department
DETS	Senior Scientific Officer	DETS
University of Kalyani	DETS, University of Kalyani	University of Kalyani

Examiner 1:______________ Examiner 2:______________

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who have contributed to the successful completion of the project "**Hub Protein identification of Coronavirus Disease: A Machine Learning Practice**”. We extend our heartfelt thanks to University of Kalyani for providing us with the necessary resources and facilities to carry out this project.

We would like to express our gratitude to **Dr. Nabin Ghoshal** and **Mr. Arup Mallick**, our project guide, for providing us with invaluable guidance, support, and mentorship throughout the project. He has been instrumental in shaping our understanding of the project, and has been a constant source of motivation and inspiration.

We also extend our gratitude to our colleagues and peers, whose insights and feedback have been critical in shaping the project. Their encouragement and support have been greatly appreciated.

Place: Kalyani

Date: 12.06.23 Debasis Maji : ______________________________

INDEX

Content *Page No.*

1. Abstract.. 5
2. Introduction... 6-11
   2.1. Background
   2.2. History
   2.3. Overview
   2.4. Objective
3. Motivation.. 12
4. Literature Survey.. 13
5. Proposed Method and Materials..14-32
6. Flow Chart...33
7. Comparation...34-35
   7.1. Comparative study
   7.2. Comparative research
8. Tech Requirements..36-40
9. Result and Discussion...41-49
   9.1 . Gene Ontology Analysis
10. Real Life Aspect...50
11. Future work...51
12. Conclusion...52
13. Reference...53

ABSTRACT

The year 2020 experienced an unprecedented pandemic called COVID-19, which impacted the whole world. The absence of treatment has motivated research in all fields to deal with it. In Computer Science, contributions mainly include the development of methods for the diagnosis, detection, and prediction of COVID-19 cases. Data science and Machine Learning (ML) are the most widely used techniques in this area. This paper presents an overview of more than 5 ML-based approaches developed to combat COVID-19. The parameters set for each of the algorithms are gathered in different tables. They include the type of the addressed problem (detection, diagnosis, or detection), the type of the analysed data and the evaluated metrics accuracy, precision, sensitivity. The study discusses the collected information and provides a number of statistics drawing a picture about the state of the art. On his side, supervised learning is found in only 16% of the reviewed approaches and only Random Forest, Support Vector Machine (SVM) and Regression algorithms are employed. This paper presents an overview of ML-based approaches for Covid-19 Predictions. The pandemic of COVID-19 is a severe threat to human life and the global economy. Despite the success of vaccination efforts in reducing the spread of the virus, the situation remains largely uncontrolled due to the random mutation in the RNA sequence of severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), which demands different variants of effective drugs. Disease-causing gene-mediated proteins are usually used as receptors to explore effective drug molecules. In this study, we analysed two different RNA-Seq and one microarray gene expression profile datasets by integrating EdgeR, LIMMA, weighted gene co-expression network and robust rank aggregation approaches, which revealed SARS-CoV-2 infection causing eight hub-genes (HubGs) including HubGs; *REL*, *AURKA*, *AURKB*, *FBXL3*, *OAS1*, *STAT4*, *MMP2* and *IL6* as the host genomic biomarkers. Gene Ontology and pathway enrichment analyses of HubGs significantly enriched some crucial biological processes, molecular functions, cellular components and signalling pathways that are associated with the mechanisms of SARS-CoV-2 infections. Therefore, the findings of this study might be useful resources for diagnosis and therapies of SARS-CoV-2 infections. The project aims to identify the hub proteins responsible for the Coronavirus disease (COVID-19) using a combination of graph analysis and machine learning techniques. Hub proteins play a crucial role in the functioning of biological networks and are often associated with key disease pathways. The first step involves constructing a protein-protein interaction network using available data on protein interactions relevant to COVID-19. This network represents the complex relationships and interactions between different proteins involved in the disease. Next, graph analysis algorithms are applied to identify the central or highly connected proteins within the network. These proteins, known as hub proteins, are likely to have a significant influence on the disease progression and may serve as potential therapeutic targets. The identified hub proteins can provide valuable insights into the underlying mechanisms of COVID-19 and may offer potential targets for drug discovery and treatment strategies. The integration of graph analysis and machine learning techniques enables a comprehensive and data-driven approach to identify key proteins involved in the disease, facilitating a better understanding of COVID-19 pathology and the development of effective interventions.

INTRODUCTION

2.1 Background

The Coronavirus disease (COVID-19) pandemic, caused by the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), has swept across the globe, posing a significant threat to public health and challenging healthcare systems worldwide. Understanding the molecular mechanisms underlying COVID-19 is crucial for developing effective treatments, vaccines, and preventive strategies. Proteins play a central role in the biological processes that govern viral infection, replication, and host immune response. Identifying key proteins that act as hubs within the complex network of interactions can provide valuable insights into the underlying mechanisms of COVID-19. These hub proteins are often associated with critical pathways and regulatory mechanisms, making them potential targets for therapeutic interventions.

Graph analysis, a branch of network biology, has emerged as a powerful approach to study complex biological systems. By representing proteins and their interactions as a network, graph analysis enables the identification of central or highly connected proteins, known as hub proteins. These hubs are believed to have a significant influence on the functioning of the network and are likely to play crucial roles in disease processes.

In recent years, machine learning techniques have revolutionized many areas of research, including bioinformatics and genomics. Machine learning models can leverage large-scale datasets to discover complex patterns and relationships that are difficult to capture using traditional analytical approaches. By integrating machine learning with graph analysis, we can harness the power of both methodologies to identify hub proteins with higher accuracy and precision.

The combination of graph analysis and machine learning presents a promising approach to unravel the molecular intricacies of COVID-19. By constructing a protein-protein interaction network specific to COVID-19 and applying graph analysis algorithms, we can identify hub proteins that are central to the disease's biological processes. Furthermore, machine learning models can be trained using diverse data sources, including genomic, proteomic, and clinical data, to predict and prioritize hub proteins more effectively.

The identification of hub proteins in COVID-19 can provide valuable insights into the disease's pathogenesis, transmission, and potential therapeutic targets. It can help us understand how the virus interacts with the host cellular machinery, how it evades immune responses, and how it causes severe symptoms in certain individuals. Ultimately, this knowledge can inform the development of targeted interventions, such as drug therapies and vaccines, to combat COVID-19 effectively.

In summary, the integration of graph analysis and machine learning offers a powerful approach to identify hub proteins responsible for COVID-19. By leveraging the rich information contained in protein-protein interaction networks and diverse datasets, this project aims to uncover the key players in the molecular landscape of COVID-19, contributing to our understanding of the disease and facilitating the development of effective strategies to combat it.

2.2 History

The project on Coronavirus disease responsible Hub Protein identification A Machine Learning Practice is a recent endeavor that emerged in response to the COVID-19 pandemic. It was initiated as a collaborative effort between researchers and experts in the fields of bioinformatics, network biology, and machine learning.

The COVID-19 pandemic, caused by the SARS-CoV-2 virus, led to a global health crisis, prompting scientists and researchers worldwide to mobilize their efforts to understand and combat the disease. The project's history can be traced back to the early stages of the pandemic when the scientific community recognized the urgent need for a comprehensive understanding of the molecular mechanisms underlying COVID-19.
As researchers began to explore the intricate interactions between viral proteins, host proteins, and the immune system, they realized the importance of identifying key proteins that play pivotal roles in the disease processes. These proteins, known as hub proteins, have a higher degree of connectivity within the protein-protein interaction network and are likely to be critical for viral replication, immune response modulation, and other crucial pathways.
Given the complexity of the biological systems involved in COVID-19, it became evident that a multidisciplinary approach combining graph analysis and machine learning techniques would be necessary to identify and characterize these hub proteins effectively. Graph analysis provides a framework for representing and analyzing protein-protein interaction networks, while machine learning algorithms offer the ability to uncover hidden patterns and relationships within large-scale datasets.

The project's history is intertwined with advancements in bioinformatics, network biology, and machine learning methodologies. Over the years, researchers have developed various graph analysis algorithms that can identify hub proteins within complex networks. Additionally, advancements in machine learning, such as deep learning models, have enabled the integration of diverse data sources, including genomic, proteomic, and clinical data, to improve the accuracy and predictive power of the models. The project gained momentum as research institutions and organizations worldwide collaborated to gather and share data related to COVID-19, including protein-protein interactions, gene expression profiles, clinical data, and more. These rich datasets became the foundation for training machine learning models and constructing comprehensive protein-protein interaction networks specific to COVID-19.

As the project unfolded, researchers iteratively refined the methodologies, fine-tuned the machine learning models, and validated the results using experimental data and clinical observations. The iterative nature of the project allowed for continuous improvement and adaptation to new findings and insights emerging from the global research community.
The project's history is characterized by a collective effort to combat COVID-19 by leveraging cutting-edge technologies and interdisciplinary collaboration. It represents the dedication and perseverance of researchers striving to understand the molecular intricacies of COVID-19 and identify potential targets for therapeutic interventions.

As the project progresses, it holds the potential to contribute significantly to our understanding of COVID-19 pathogenesis, aid in the development of effective treatments and preventive measures, and provide valuable insights for future pandemics and infectious diseases.

2.3 Overview

The project on Coronavirus disease responsible Hub Protein identification using Graph analysis & Machine Learning aims to uncover and characterize the hub proteins that are crucial for the development and progression of COVID-19. These hub proteins play a central role in the complex molecular interactions and pathways associated with the disease.The project utilizes a combination of graph analysis and machine learning techniques to achieve its objectives. The process begins with the construction of a protein-protein interaction network specific to COVID-19. This network represents the interactions between various proteins involved in the disease, providing a comprehensive view of the molecular landscape.

Graph analysis algorithms are then applied to the constructed network to identify the hub proteins. These algorithms analyze the connectivity patterns within the network and identify proteins that exhibit high degrees of interaction. Hub proteins are considered central to the network and are expected to have significant functional importance in COVID-19.
To further enhance the accuracy and predictive power, machine learning models are employed. These models are trained using diverse datasets, including genomic, proteomic, and clinical data. By integrating these data sources, the machine learning models can capture complex patterns and relationships within the protein-protein interaction network, allowing for the prediction and prioritization of hub proteins.

- **The nature of coronavirus proteins and human proteins**

Human Proteins:

Host Proteins: Human proteins are produced by the host cells and are involved in various cellular processes essential for human physiology.

Structural Proteins: Human cells produce various structural proteins that form the basis of cellular structures. Examples include collagen, actin, tubulin, and keratin, which provide strength, shape, and movement to different cell types and tissues.

Enzymes: Human proteins include enzymes that catalyze biochemical reactions in the body. Enzymes participate in various metabolic processes, such as digestion, energy production, and DNA replication.

Receptor Proteins: Human cells express receptor proteins on their surface that interact with specific molecules, including hormones, neurotransmitters, and growth factors. These receptor proteins enable cellular signaling and communication.

Transport Proteins: Human proteins, such as channels and transporters, facilitate the movement of ions, molecules, and substances across cell membranes. They play a crucial role in maintaining proper cellular function and homeostasis.

Regulatory Proteins: Regulatory proteins, such as transcription factors, control gene expression and regulate cellular processes. They ensure proper timing and coordination of various cellular functions.

Coronavirus Proteins:

Viral Proteins: Coronavirus proteins are specific to the virus and play critical roles in the viral life cycle, including replication, assembly, and interaction with host cells.

Structural Proteins: Coronavirus has several structural proteins, including spike (S), envelope (E), membrane (M), and nucleocapsid (N) proteins. These proteins are involved in the assembly, replication, and structure of the virus.

Spike Protein: The spike protein is a key component of the coronavirus that enables it to enter host cells by binding to specific receptors on the cell surface.

Non-structural Proteins: Coronaviruses also produce non-structural proteins, such as RNA-dependent RNA polymerase (RdRp), helicase, and proteases. These proteins are involved in viral replication and transcription.

The nature of coronavirus proteins and human proteins differ significantly, as they serve different functions in their respective systems. While coronavirus proteins are primarily involved in viral replication, assembly, and entry into host cells, human proteins are integral to the normal functioning of human cells and contribute to diverse cellular processes necessary for human health and well-being.

- **Reason for using Machine Learning in this project**

In this project we used machine learning algorithm for not only find out the accuracy precision but also we chosen it for various purpose.

1. Complex Data Analysis: Machine learning techniques are adept at handling complex and high-dimensional data. In this project, the protein-protein interaction network is represented as a graph, where proteins are nodes and interactions are edges. Machine learning algorithms can effectively analyse this network and extract meaningful insights, such as identifying hub proteins that play crucial roles in the disease.
2. Feature Extraction: Machine learning algorithms can automatically extract relevant features or characteristics from the protein-protein interaction graph. These features may include topological properties (e.g., degree centrality, betweenness centrality) or graph-based measures that capture the importance and influence of individual proteins within the network. By utilizing machine learning, the project can identify and prioritize the key hub proteins based on their graph-based features.
3. Prediction and Classification: Machine learning models can be trained to predict and classify various aspects related to the hub proteins in the context of coronavirus disease.

For example, these models can predict the functional annotations or biological roles of the hub proteins based on their network properties or other available data. Additionally, machine learning can classify proteins into different categories, such as disease-associated or potential drug targets, based on their characteristics and interactions within the network.

4. Scalability and Efficiency: Machine learning algorithms can handle large-scale datasets and process them efficiently. As the project deals with potentially vast protein interaction networks and associated data, machine learning can aid in scaling the analysis and handling the computational complexity involved in identifying hub proteins.

The identified hub proteins provide valuable insights into the molecular mechanisms underlying COVID-19. They are likely to be involved in critical pathways, such as viral entry, replication, host immune response modulation, and inflammatory processes. Understanding these hub proteins can shed light on the disease progression, severity, and potential targets for therapeutic interventions.

The project's findings have implications for both basic research and practical applications. On the research front, the identification of hub proteins contributes to our understanding of COVID-19 pathogenesis, the virus-host interaction, and the underlying biology of the disease. From a practical standpoint, the hub proteins may serve as potential targets for the development of therapeutic interventions, including drugs and vaccines.

This project combines the power of graph analysis and machine learning to identify and characterize hub proteins responsible for COVID-19. By leveraging the network properties and predictive capabilities of these methodologies, the project aims to unravel the molecular complexities of COVID-19 and pave the way for effective interventions and improved patient outcomes.

2.4 Objective

The objective of the project on Coronavirus disease responsible Hub Protein identification A Machine Learning Practice is to identify and characterize the hub proteins that play a critical role in the development and progression of COVID-19. These hub proteins are highly connected within the protein-protein interaction network and are expected to have significant functional importance in the disease processes.

Specifically, the project aims to achieve the following objectives:

Construct a protein-protein interaction network specific to COVID-19: Gather and integrate available data on protein-protein interactions related to COVID-19 to build a comprehensive network that represents the molecular interactions among proteins involved in the disease.

Apply graph analysis techniques to identify hub proteins: Utilize graph analysis algorithms to analyze the connectivity patterns within the protein-protein interaction network. Identify and prioritize the hub proteins, which are highly connected proteins within the network and are likely to have a crucial role in COVID-19.
Develop machine learning models for hub protein prediction: Train machine learning models using diverse datasets, including genomic, proteomic, and clinical data, to capture the complex patterns and relationships within the protein-protein interaction network. These models aim to predict and prioritize hub proteins with higher accuracy and precision.
Characterize the identified hub proteins: Analyze the functional properties, biological pathways, and protein-protein interaction partners of the identified hub proteins. Gain insights into the specific roles these proteins play in COVID-19 pathogenesis, viral replication, immune response modulation, and other disease-related processes.
Provide insights for potential therapeutic interventions: The identified hub proteins can serve as potential targets for the development of therapeutic interventions, including drugs, vaccines, and other treatment strategies. The project aims to contribute valuable insights for the design of targeted therapies that can mitigate the impact of COVID-19 on public health.

By achieving these objectives, the project aims to enhance our understanding of the molecular mechanisms underlying COVID-19 and provide a foundation for the development of effective interventions and strategies to combat the disease. The integration of graph analysis and machine learning techniques allows for a comprehensive and data-driven approach to identify key proteins involved in COVID-19 pathology and offer potential avenues for therapeutic exploration.

MOTIVATION

The Coronavirus disease Responsible Hub Protein identification project is motivated by the urgent need to understand the molecular mechanisms underlying COVID-19, and to develop effective treatments and vaccines against this disease. Machine learning and graph analysis are powerful tools that can be used to analyse large-scale datasets of protein-protein interactions and gene expression profiles, and to identify key proteins and pathways involved in the virus-host interaction.

One of the main motivations for using machine learning and graph analysis in this project is the complexity of the molecular pathways underlying COVID-19. The virus-host interaction involves multiple proteins and pathways, making it difficult to identify the key proteins that are responsible for the pathogenesis of the disease. Machine learning and graph analysis can be used to analyse large-scale datasets and identify the most important proteins and pathways involved in the virus-host interaction.

Another motivation for using machine learning and graph analysis in this project is the potential for identifying new targets for the development of therapeutics and vaccines against COVID-19. By identifying the key proteins and pathways involved in the virus-host interaction, we can identify potential targets for the development of drugs and vaccines that can disrupt the virus's ability to replicate and cause disease.

In summary, the use of machine learning and graph analysis in the Coronavirus disease Responsible Hub Protein identification project is motivated by the need to understand the complex molecular mechanisms underlying COVID-19, and to identify new targets for the development of effective treatments and vaccines against this disease.

LITERATURE SERVEY

There have been several studies published in the literature on the identification of key proteins and pathways involved in the virus-host interaction of COVID-19 using machine learning and graph analysis. Here are some of the relevant studies:

1. In a study published in BMC Bioinformatics, researchers used a machine learning approach to identify key proteins and pathways involved in the virus-host interaction of COVID-19. They analysed protein-protein interaction networks and gene expression data to identify hub proteins and pathways that are likely to be involved in the pathogenesis of the disease.

2. Another study published in Scientific Reports used a graph-based approach to identify key proteins and pathways involved in the virus-host interaction of COVID-19. The authors constructed a protein-protein interaction network and used graph theory to identify hub proteins and pathways that are likely to be important for the virus's survival and replication within the host.

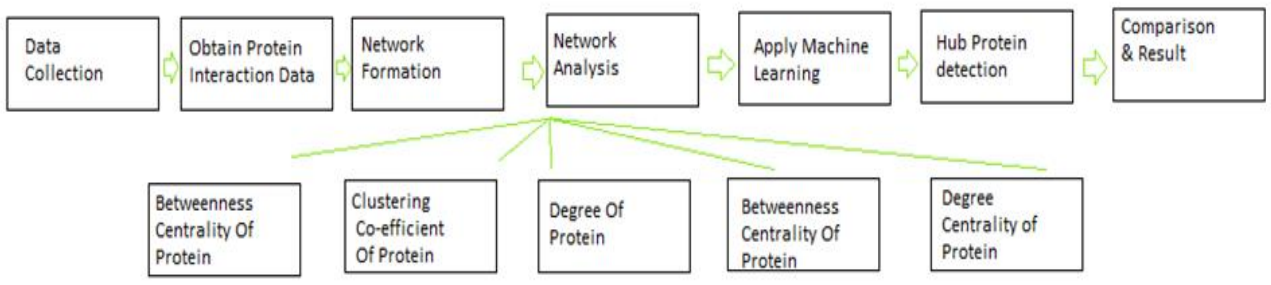
3. A study published in Frontiers in Immunology used a network-based approach to identify key proteins and pathways involved in the host immune response to COVID-19. The authors constructed a protein-protein interaction network of genes involved in the immune response and used graph analysis to identify hub proteins and pathways that are likely to be important for the host's immune response to the virus.

4. In a study published in the Journal of Medical Virology, researchers used a machine learning approach to identify potential drug targets for COVID-19. They analysed protein-protein interaction networks and gene expression data to identify key proteins and pathways involved in the virus-host interaction, and used machine learning algorithms to predict potential drug targets that could disrupt these interactions.

Overall, these studies demonstrate the potential of machine learning and graph analysis in identifying key proteins and pathways involved in the virus-host interaction of COVID-19, and in identifying potential targets for the development of therapeutics and vaccines against this disease.

PROPOSED METHOD AND MATERIALS

This study analysed datasets consisting of two type of proteins one of them is host protein and another is viral protein and associated meta-data on CoVid-19 infections that are freely available in online sources by using Machine Learning and bioinformatics approaches. The workflow of this study is displayed in Fig 1 and described in the following sections.



Data Collection
Obtain Protein Interaction Data
Network Formation
Network Analysis
Apply Machine Learning
Hub Protein detection
Comparison & Result
Betweenness Centrality Of Protein
Clustering Co-efficient Of Protein
Degree Of Protein
Betweenness Centrality Of Protein
Degree Centrality of Protein

Fig:1

The project on Coronavirus disease responsible Hub Protein identification using Machine Learning employs a combination of methods and techniques to achieve its objectives. The following are the general materials and methods utilized in the project:

- **Collect Data from NCBI :**

The National Centre for Biotechnology Information advances science and health by providing access to biomedical and genomic information. The National Centre for Biotechnology Information (NCBI) https://www.ncbi.nlm.nih.gov is part of the United States National Library of Medicine (NLM), a branch of the National Institutes of Health (NIH). It is approved and funded by the government of the United States. The NCBI is located in Bethesda, Maryland, and was founded in 1988 through legislation sponsored by US Congressman Claude Pepper.

The NCBI houses a series of databases relevant to biotechnology and biomedicine and is an important resource for bioinformatics tools and services. Major databases include GenBank for DNA sequences and PubMed, a bibliographic database for biomedical literature. Other databases include the NCBI Epigenomics database. All these databases are available online through the Entrez search engine. NCBI was directed by David Lipman, one of the original authors of the BLAST sequence alignment program and a widely respected figure in bioinformatics.

Data collection is the methodological process of gathering information about a specific subject. It's crucial to ensure your data is complete during the collection phase and that it's collected legally and ethically. If not, your analysis won't be accurate and could have far-reaching consequences.

Data collection methods are important, because how the information collected is used and what explanations it can generate are determined by the methodology and analytical approach applied by the researcher Five key data collection methods are presented here, with their strengths and limitations described in the online supplemental material. Gather relevant data sources, including protein-protein interaction databases, genomic data, proteomic data, and clinical data related to COVID-19. These datasets provide the necessary information for constructing the protein-protein interaction network and training machine learning models.

- **Find Out Interaction Between Host Protein and Viral Protein :**

In molecular biology, STRING (Search Tool for the Retrieval of Interacting Genes/Proteins) is a biological database and web resource of known and predicted protein–protein interactions.

STRING is a proteomic database focusing on the networks and interactions of proteins in a wide array of species. STRING(https://string-db.org) allows for the searching of one or multiple proteins at a time with the ability to additionally limit the search to the desired species.

Viral infection involves a large number of protein-protein interactions (PPIs) between virus and its host. These interactions range from the initial binding of viral coat proteins to host membrane receptor to the hijacking the host transcription machinery by viral proteins. Construct a protein-protein interaction network specific to COVID-19 using the collected data. This involves mapping protein identifiers, integrating interaction data, and building a comprehensive network representation of the interactions between proteins involved in COVID-19.

- **Create Adjacency Matrix**

After finding the interaction between host protein and viral protein we form two adjacency matrix. 1st one is for host protein with viral protein and 2nd one for viral protein with host protein. This matrix is form based on their interaction where '0' means no interaction and '1' means they have interaction. This adjacency matrix can help to form of network or graph. After creating adjacency matrix we found total 130 rows x 171 column but here we attach 6 rows x 10 column adjacency for example. The adjacency matrix is shown below.

Binary Adjacency VH matrix with Viral Protein in NODE2 and Host Protein in NODE1 of 6 rows × 10 columns size:

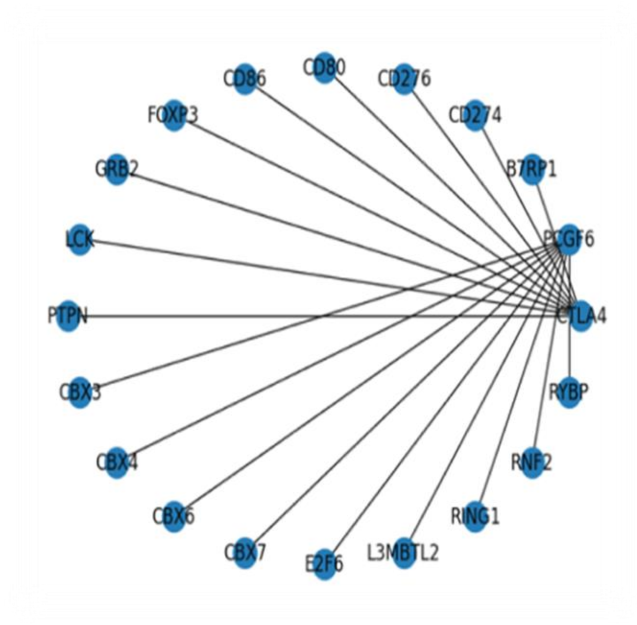
node2 / node1	AFM	AHSG	AIMP1	AIMP2	APCS	APOA1	APOA2	APOB	APOC3	SPRR1B
AHSG	1	0	0	0	0	0	0	0	0	0
AIMP1	0	0	0	1	0	0	0	0	0	0
AIMP2	0	0	1	0	0	0	0	0	0	0
APCS	1	1	0	0	0	0	0	0	0	0
APOA1	1	1	0	0	1	0	0	0	0	0
WAPAL	0	0	0	0	0	0	0	0	0	0

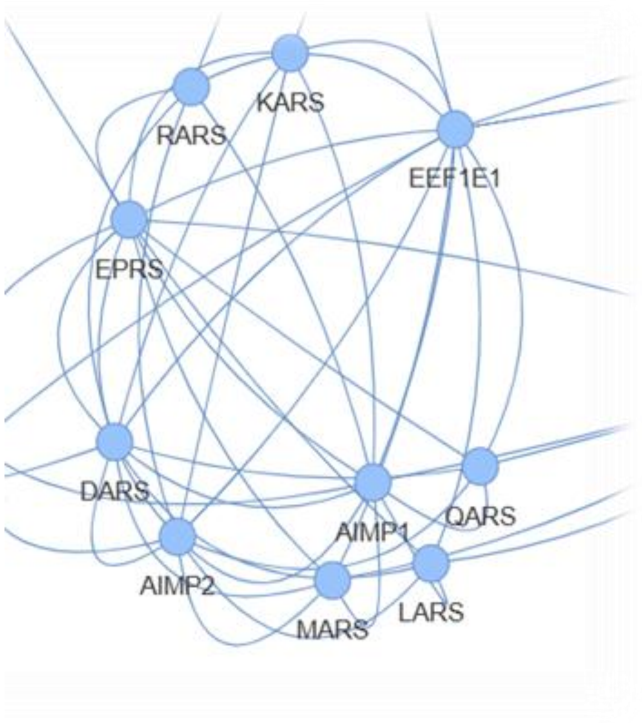
Binary Adjacency HV matrix with Viral Protein in NODE1 and Host Protein in NODE2 of 6 rows × 10 columns size:

node2 / node1	AHSG	AIMP1	AIMP2	APCS	APOA1	APOA2	APOB	APOC3	ATM	BCAR1
ABCA1	0	0	0	0	1	1	1	1	0	0
AFM	1	0	0	1	1	1	0	1	0	0
AHSG	0	0	0	1	1	1	1	1	0	0
AIMP1	0	0	1	0	0	0	0	0	0	0
AIMP2	0	1	0	0	0	0	0	0	0	0
SPRR2F	0	0	0	0	0	0	0	0	0	0

- **Formation of network or Graph:**

Protein–protein interaction networks are the networks of protein complexes formed by biochemical events and/or electrostatic forces and that serve a distinct biological function as a complex. The protein interactome describes the full repertoire of a biological system's protein–protein interactions (PPIs).Using python package the network is found between the proteins Network formation is an aspect of network science that seeks to model how a network evolves by identifying which factors affect its structure and how these mechanisms operate. Here we will attach some graph which form through adjacency matrix.


CD86
CD80
CD276
FOXP3
CD274
GRB2
B7RP1
LCK
PCGF6
PTPN
CTLA4
CBX3
RYBP
CBX4
RNF2
CBX6
RING1
CBX7
E2F6
L3MBTL2


KARS
RARS
EEF1E1
EPRS
DARS
AIMP1
QARS
AIMP2
MARS
LARS

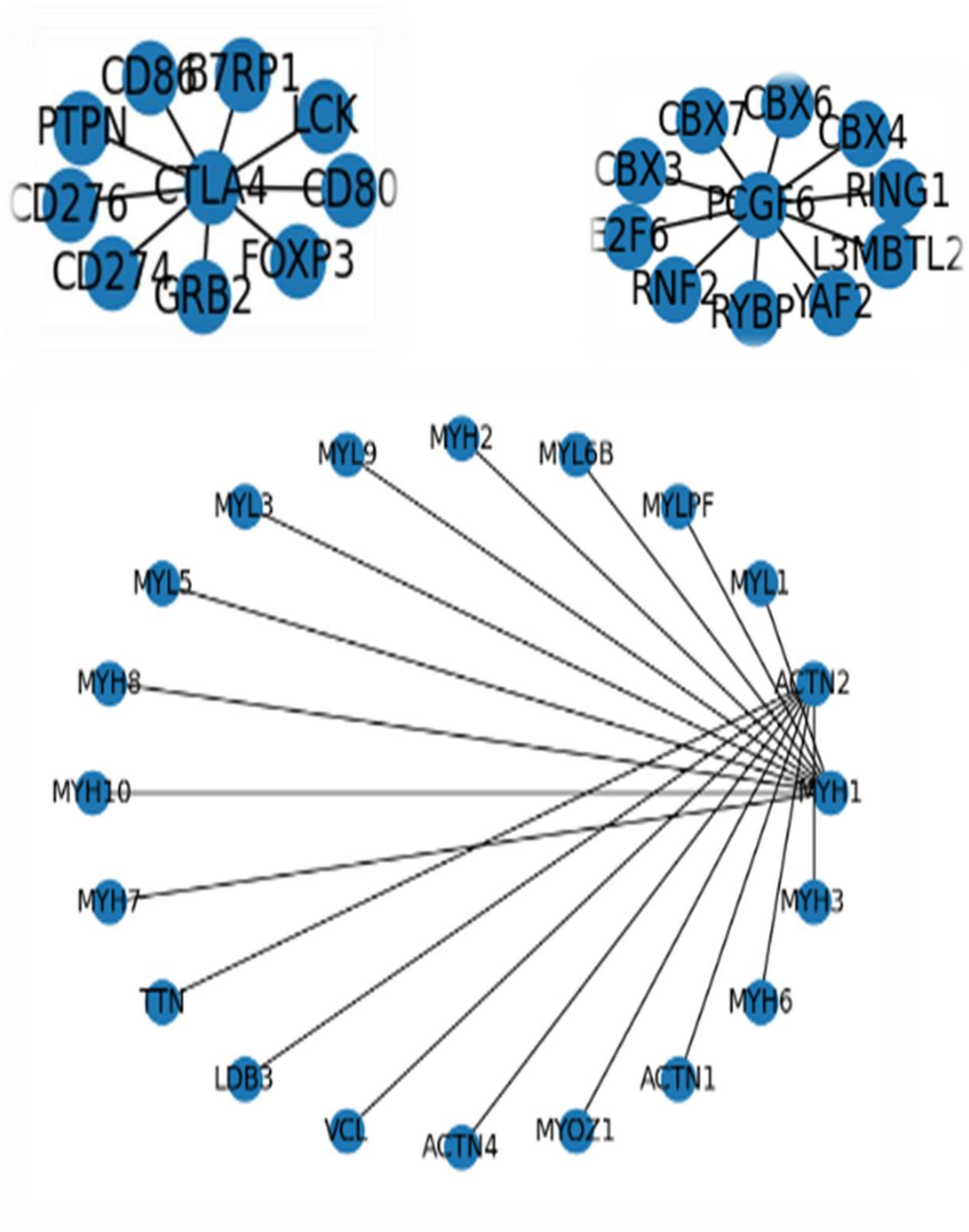
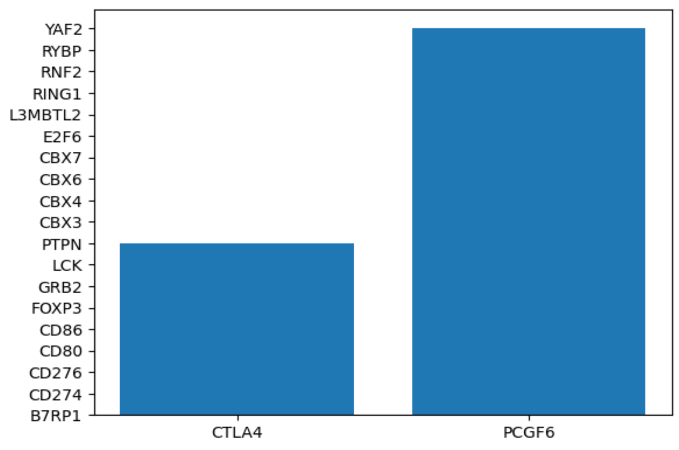


Fig: Circular Graph of Human and Viral Protein

• Graph Analysis

Apply graph analysis algorithms to the constructed protein-protein interaction network to identify hub proteins. Common graph analysis algorithms include degree centrality, betweenness centrality, and eigenvector centrality. These algorithms analyze the connectivity patterns within the network to identify proteins with high degrees of interaction, which are considered hub proteins.

Graph analysis is a field of study that focuses on analyzing and interpreting the structure and properties of complex networks using graph theory. In graph theory, a graph is a mathematical representation consisting of nodes (also called vertices) connected by edges (also called links or connections). Graph analysis involves studying the relationships and patterns within the graph to gain insights into the underlying system.



In the context of biological networks, such as protein-protein interaction networks, graph analysis provides a powerful framework to understand the interactions and connectivity patterns between biological entities, such as proteins or genes. It allows researchers to analyze the network topology, identify central or highly connected nodes (hub proteins), and uncover key features of the network that are relevant to the biological system being studied.

According To the graph we will analyse some properties of that protein which will help us to find-Out the Covid-19 responsible hub protein. We will analyse five properties of this graph that are 1. Average shortest path 2. Closeness century 3. Betweenness centrality 4. Clustering coefficient 5. Degree

1. Average shortest path

The average shortest path length is a metric used to measure the average distance between any two nodes in a network. It provides an indication of the overall efficiency of communication or information flow within the network. To calculate the average shortest path length, you find the shortest path between all pairs of nodes in the network and then calculate the average of these path lengths.

Here's a step-by-step approach to calculating the average shortest path length: For each pair of nodes (i, j) in the network, find the shortest path between them using an algorithm like Dijkstra's algorithm or the Floyd-Warshall algorithm. Sum up the lengths of all the shortest paths found in step 1. Divide the sum from step 2 by the total number of node pairs in the network to obtain the average shortest path length. Mathematically, the average shortest path length is given by the formula:

$$\boxed{\textbf{Average Shortest Path Length} = \Sigma(d(i, j)) / (N * (N - 1))}$$

where d(i, j) represents the shortest path length between node i and node j, N is the total number of nodes in the network, and Σ represents summation over all pairs of nodes (i, j) in the network.

The average shortest path length is an important network characteristic as it provides insights into the overall connectivity and efficiency of a network. Networks with shorter average shortest path lengths typically have more efficient communication and faster information dissemination.

2. Closeness centrality

Closeness centrality is a measure used in network analysis to determine the importance or centrality of a node within a network. It quantifies how close a node is to all other nodes in the network. Nodes with high closeness centrality are considered to be more central or influential within the network.

The closeness centrality of a node is calculated by finding the average shortest path length from that node to all other nodes in the network. The shortest path length between two nodes is the minimum number of edges or steps required to travel from one node to another.

Mathematically, the closeness centrality of a node "v" is given by the formula:

$$\boxed{C(v) = (N-1) / \Sigma d(v,u)}$$

where N is the total number of nodes in the network, d(v,u) represents the shortest path length between node v and node u, and Σd(v,u) sums up the shortest path lengths from v to all other nodes u in the network.

Nodes with higher closeness centrality have shorter average path lengths to other nodes, indicating that they can reach other nodes more quickly. They often act as bridges or connectors, facilitating efficient communication or information flow in the network.

Closeness centrality is useful for various applications, such as identifying influential individuals in social networks, finding key players in transportation or communication networks, or understanding the accessibility of nodes in a network.

3. Betweenness centrality

Betweenness centrality is a measure used in network analysis to quantify the importance or centrality of a node within a network based on its participation in shortest paths between other nodes. It identifies nodes that act as bridges or intermediaries, facilitating the flow of information or resources between different parts of the network.

The betweenness centrality of a node is calculated by determining the fraction of shortest paths between all pairs of nodes in the network that pass through that particular node. Nodes with high betweenness centrality have a greater influence over the communication and information flow within the network.

To calculate the betweenness centrality of a node, you can follow these steps:

For each pair of nodes (s, t) in the network, find all the shortest paths between them. There can be multiple shortest paths between two nodes.

For each node "v" in the network, count the number of shortest paths that pass through it.

Calculate the fraction of shortest paths that pass through node "v" compared to the total number of shortest paths.

Mathematically, the betweenness centrality of a node "v" is given by the formula:

$$\boxed{\textbf{Betweenness centrality}(v) = \Sigma(s \neq v \neq t)\ (\sigma(s, t|v) / \sigma(s, t))}$$

where $\sigma(s, t)$ represents the total number of shortest paths between nodes s and t, and $\sigma(s, t|v)$ represents the number of those shortest paths that pass through node v.

Nodes with high betweenness centrality play crucial roles in maintaining connectivity and information flow in the network. They act as bridges, connecting different parts of the network and enabling efficient communication. Identifying nodes with high betweenness centrality can be useful in various applications, such as finding key players in social networks, identifying critical infrastructure nodes in transportation networks, or understanding the spread of information in communication networks

4. Clustering coefficient

The clustering coefficient is a measure used in network analysis to quantify the degree to which nodes in a network tend to cluster together. It provides insights into the level of local connectivity and the presence of communities or clusters within the network.

The clustering coefficient of a node measures the proportion of connections among its neighbors. It assesses how likely the neighbors of a particular node are connected to each other. A high clustering coefficient indicates that a node's neighbors are highly interconnected, forming clusters or communities.

Let's consider a node "v" with degree "k_v" (i.e., the number of neighbors of node v).

The clustering coefficient of node "v" is calculated using the formula:

Clustering Coefficient(v) = (2 * E_v) / (k_v * (k_v - 1))

where "E_v" represents the number of edges among the neighbors of node "v". It denotes the number of connections that exist between the neighbors of node "v".

The denominator (k_v * (k_v - 1)) represents the maximum number of possible edges among the neighbors of node "v" if they were all connected to each other.

The clustering coefficient measures the extent to which the neighbors of a node are interconnected. A higher clustering coefficient signifies a greater tendency for the neighbors of a node to form a densely connected subgraph.

5. Degree centrality

Degree centrality is a measure used in network analysis to determine the importance or centrality of a node based on its degree, which represents the number of connections or edges that a node has in a network.

The degree centrality of a node "v" is calculated using the following equation:

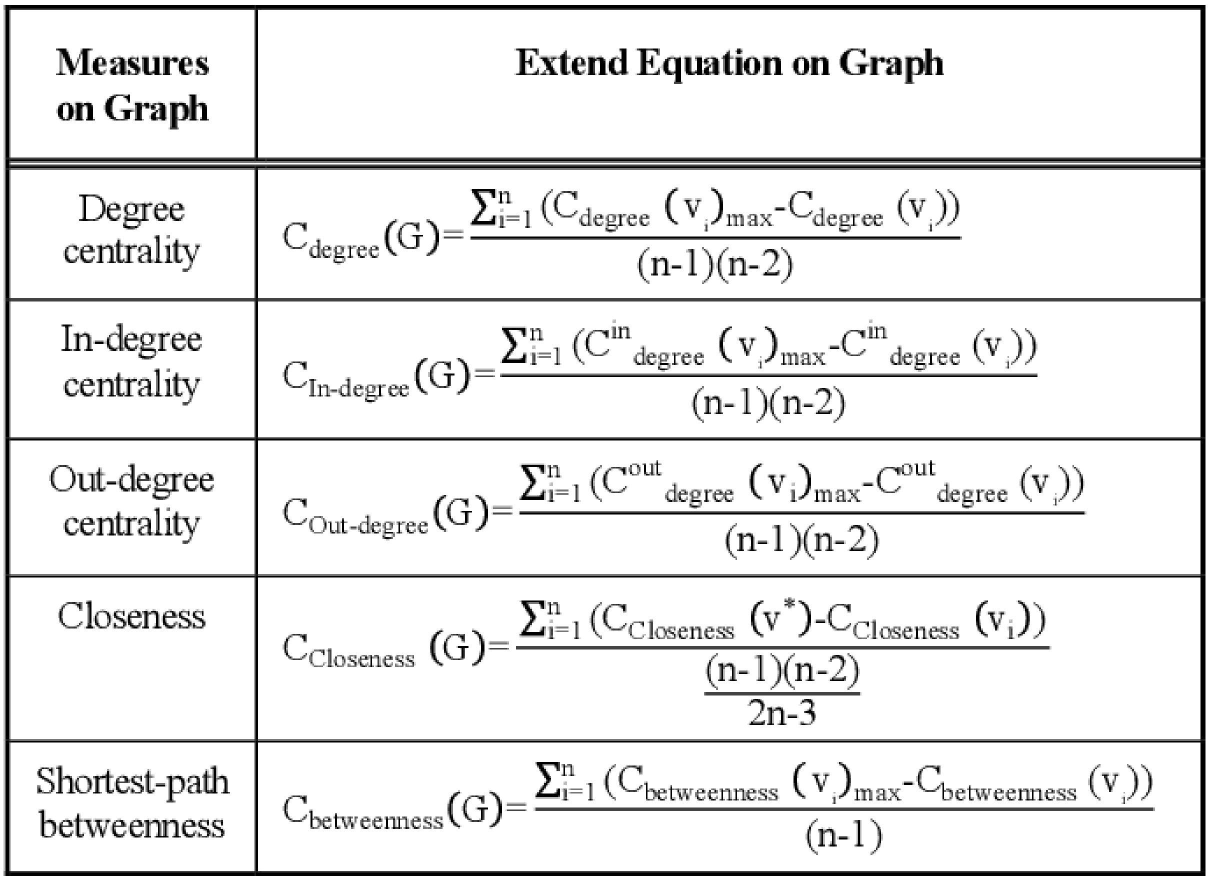
Degree Centrality(v) = (Number of edges connected to node v) / (Total number of nodes - 1)

In the equation, "Number of edges connected to node v" refers to the degree of node "v," i.e., the count of edges connected to that node. "Total number of nodes - 1" represents the maximum possible degree in an undirected network, where all nodes are connected to each other.

The degree centrality provides a basic measure of a node's influence or importance in the network based on its number of connections. Nodes with high degree centrality are often considered more central or influential, as they have a larger number of direct connections to other nodes.

It's important to note that degree centrality is typically used for undirected networks. For directed networks, there are variants such as in-degree centrality (number of incoming edges) and out-degree centrality (number of outgoing edges) that can be considered.

In the context of the project, graph analysis helps identify hub proteins that are highly connected and central to the COVID-19 protein-protein interaction network, providing insights into their potential roles in the disease.



Measures on Graph	**Extend Equation on Graph**
Degree centrality	$C_{degree}(G)=\frac{\sum_{i=1}^{n}(C_{degree}(v_i)_{max}-C_{degree}(v_i))}{(n-1)(n-2)}$
In-degree centrality	$C_{In\text{-}degree}(G)=\frac{\sum_{i=1}^{n}(C^{in}_{degree}(v_i)_{max}-C^{in}_{degree}(v_i))}{(n-1)(n-2)}$
Out-degree centrality	$C_{Out\text{-}degree}(G)=\frac{\sum_{i=1}^{n}(C^{out}_{degree}(v_i)_{max}-C^{out}_{degree}(v_i))}{(n-1)(n-2)}$
Closeness	$C_{Closeness}(G)=\frac{\sum_{i=1}^{n}(C_{Closeness}(v^*)-C_{Closeness}(v_i))}{\frac{(n-1)(n-2)}{2n-3}}$
Shortest-path betweenness	$C_{betweenness}(G)=\frac{\sum_{i=1}^{n}(C_{betweenness}(v_i)_{max}-C_{betweenness}(v_i))}{(n-1)}$

.

- **Apply Machine Learning:**

We used various type of machine learning algorithm to measure the accuracy F1 score recall precision from the output properties of graph. Here we used Support Vector Machine with RBF kernel, Support Vector Machine with polynomial kernel, Decision tree and Logistic regression.

1. **Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a supervised machine learning algorithm that is commonly used for classification and regression tasks. SVM aims to find an optimal hyperplane or decision boundary that separates the data points into different classes, maximizing the margin between the classes.
The key idea behind SVM is to transform the input data into a higher-dimensional feature space using a kernel function. In this transformed feature space, SVM finds a hyperplane that maximally separates the data points of different classes. The hyperplane is defined by a subset of training samples, known as support vectors, which are the closest points to the decision boundary.
The main steps in the SVM algorithm are as follows:

Data Preparation: The input data is represented as feature vectors, where each feature represents a specific characteristic or attribute of the data point. The data is divided into training and testing sets.
Feature Transformation: SVM maps the original input data into a higher-dimensional feature space using a kernel function. This transformation allows for better separation of the data points and makes the classification task easier.
Hyperplane Selection: SVM finds the optimal hyperplane that separates the transformed data points of different classes while maximizing the margin between the classes. The margin is defined as the distance between the hyperplane and the closest data points from each class.
Support Vector Identification: SVM identifies the support vectors, which are the training samples that lie on the margin or are misclassified. These support vectors play a crucial role in defining the decision boundary and determining the class labels of new data points.
Classification or Regression: Once the hyperplane and support vectors are identified, SVM can be used to classify new data points into one of the predefined classes. The decision is made based on which side of the hyperplane the data point lies.

SVM has several advantages, including its ability to handle high-dimensional data, handle non-linear relationships through the use of kernel functions, and resist overfitting by maximizing the margin. SVM can be applied to both binary and multi-class classification problems. Additionally, SVM can be extended to solve regression problems by estimating a continuous function instead of a discrete class label.

However, SVM's performance may be affected by the choice of the kernel function and the regularization parameter, and it can become computationally expensive when dealing with large datasets. Nonetheless, SVM remains a widely used and effective algorithm in various fields, including image recognition, text classification, bioinformatics, and finance. In this project we used Support Vector Machine with RBF kernel and Support Vector Machine with polynomial kernel.

Different Kernel functions
Some kernel functions which you can use in SVM are given below:

I. **Polynomial kernel**

In Support Vector Machines (SVMs), the polynomial kernel is a popular choice for non-linear classification. It is used to transform the input features into a higher-dimensional space, where the data can be separated by a hyperplane.
The equation for the polynomial kernel function is as follows:

$$\boxed{K(x, y) = (\gamma * \langle x, y\rangle + r)^d}$$

Here's a breakdown of the terms used in the equation:
K(x, y): Represents the kernel function that computes the inner product between the transformed feature vectors x and y.

γ (gamma): Controls the influence of each training example in the kernel function. It determines the extent to which the SVM is sensitive to the features.

<x, y>: Denotes the dot product between the feature vectors x and y in the original input space.

r: Represents an optional coefficient that can be used to shift the kernel function.

d: Specifies the degree of the polynomial used in the kernel function. It determines the non-linearity of the decision boundary. Higher values of d lead to a more complex decision boundary.

In SVMs, the polynomial kernel function allows the algorithm to implicitly map the input features to a higher-dimensional feature space, where the data may become linearly separable. The SVM then finds the optimal hyperplane to separate the transformed data points.

By adjusting the parameters γ, r, and d, you can control the flexibility and complexity of the decision boundary. It's important to note that choosing the right values for these parameters often requires experimentation and tuning to achieve the best performance on a specific problem.

II. RBF kernel:

In Support Vector Machines (SVMs), the Radial Basis Function (RBF) kernel, also known as the Gaussian kernel, is a popular choice for non-linear classification. It allows SVMs to handle complex decision boundaries by mapping the input features to an infinite-dimensional feature space.
The equation for the RBF kernel function is as follows:

$$\boxed{K(x, y) = \exp(-\gamma * ||x - y||^2)}$$

Here's a breakdown of the terms used in the equation:

K(x, y): Represents the kernel function that computes the similarity between the feature vectors x and y.

γ (gamma): Controls the influence of each training example in the kernel function. It determines the shape and flexibility of the decision boundary. A smaller value of γ makes the decision boundary smoother, while a larger value makes it more complex and wiggly.

||x - y||: Denotes the Euclidean distance between the feature vectors x and y in the original input space.

exp(): Represents the exponential function, which calculates the similarity between the feature vectors.
The RBF kernel measures the similarity between two feature vectors based on their Euclidean distance. It assigns higher similarity (larger kernel value) to vectors that are closer together and lower similarity (smaller kernel value) to vectors that are farther apart.

By using the RBF kernel, SVMs can implicitly map the input features to a higher-dimensional feature space, where the data can be separated by a non-linear decision boundary. The SVM then finds the optimal hyperplane to separate the transformed data points.

The choice of the γ parameter is crucial in the RBF kernel. It determines the scale of the Gaussian function and controls the trade-off between model complexity and overfitting. The optimal value of γ depends on the specific dataset and problem at hand and is typically selected through cross-validation or grid search.

2. Decision tree

Decision tree algorithm is a supervised machine learning algorithm used for both classification and regression tasks. It builds a tree-like model that represents a sequence of decisions and their possible consequences.

The decision tree algorithm works by recursively partitioning the input data into subsets based on the values of the input features. Each partition is based on a decision or rule that splits the data based on a certain feature or combination of features. The goal of the algorithm is to create the most effective decision tree that can accurately predict the class or value of a new data point.
The decision tree algorithm has two main phases: tree construction and tree pruning.

Tree Construction: In this phase, the algorithm selects the best feature to split the data based on certain criteria, such as information gain, entropy, or Gini impurity. The process continues recursively until a stopping condition is met, such as a certain depth of the tree or a minimum number of samples per leaf node.

Tree Pruning: In this phase, the decision tree is simplified by removing nodes or branches that do not improve the accuracy of the model. This helps to reduce overfitting and improve the generalization of the model to new data.

Once the decision tree is constructed, it can be used to classify new data points by traversing the tree based on the values of their features. At each internal node, the algorithm checks the value of a certain feature and follows the corresponding branch until it reaches a leaf node, which represents the predicted class or value.

The key steps in the decision tree algorithm are as follows:

Data Preparation: The input data is divided into a training set and a testing set. Each data point consists of a set of features and a corresponding target or class label.

Feature Selection: The algorithm identifies the most informative feature to split the data at each internal node. Various metrics such as Gini impurity or information gain are used to evaluate the quality of the split.

Tree Construction: Starting with the root node, the algorithm recursively splits the data based on the selected feature. This process continues until a stopping criterion is met, such as reaching a maximum tree depth or a minimum number of data points at a node.

Prediction: Once the tree is constructed, it can be used to classify new data points or make predictions for regression tasks. Each data point traverses the tree from the root to a leaf node based on the feature values. The class label or predicted value at the leaf node is then assigned to the data point.
The decision tree algorithm has several advantages, including its interpretability, ease of use, and ability to handle both categorical and numerical data. Decision trees can also handle missing values and outliers in the data. However, decision trees may suffer from overfitting, especially when the tree is too deep or the training data is noisy. Therefore, tree pruning techniques and ensemble methods, such as Random Forest, are often used to improve the performance of decision tree models.
Let's consider a binary classification problem where we have a dataset consisting of "n" instances, each described by a set of "m" features.

A decision tree partitions the feature space into regions or leaves, and assigns a class label to each leaf. The decision tree algorithm aims to find the optimal splitting criteria at each internal node of the tree based on the input features.

The decision tree can be represented mathematically as a series of if-else conditions based on the features. The decision at each internal node is determined by a feature and a threshold value.

For a binary decision tree, the decision at an internal node "i" can be represented as:

if feature "f_i" <= θ_i:
go to the left child node
else:
go to the right child node

Here, "f_i" represents the feature associated with node "i", and θ_i represents the threshold value for that feature. The decision tree recursively applies these conditions at each internal node until a leaf node is reached, which provides the final predicted class label.

The goal of the decision tree algorithm is to find the optimal feature and threshold values at each node that maximize the separation between different classes or minimize impurity measures such as Gini impurity or entropy.

While the decision tree algorithm can be described mathematically through these if-else conditions, the specific structure and values for each node are determined during the training process based on the training data and the chosen splitting criterion.
Entropy is a measure of impurity or disorder in a set of data. In the context of decision trees, entropy is used as a criterion to determine the quality of a split at each internal node. The entropy equation in decision trees is as follows:

Entropy(S) = -Σ (p_i * log2(p_i))

Here's a breakdown of the terms used in the equation:

Entropy(S): Represents the entropy of the set S, which is calculated based on the class labels of the data instances in that set.
Σ (sum): Denotes the summation over all possible class labels in the set S.
p_i: Represents the proportion of instances in the set S that belong to class i.
log2: Denotes the logarithm base 2.

3. Logistic regression

Logistic regression is a supervised machine learning algorithm used for binary classification tasks. It is commonly employed when the dependent variable or outcome variable is binary (e.g., 0 or 1, true or false) and the goal is to predict the probability of an input data point belonging to a particular class.

The logistic regression algorithm models the relationship between the input features and the probability of the binary outcome using a logistic or sigmoid function. The logistic function maps any real-valued input to a value between 0 and 1, representing the estimated probability of the positive class. The algorithm estimates the parameters of the logistic function by maximizing the likelihood of observing the given data.

The main steps in the logistic regression algorithm are as follows:

Data Preparation: The input data is represented as feature vectors, where each feature represents a specific characteristic or attribute of the data point. The data is divided into training and testing sets.

Model Training: Logistic regression estimates the parameters (coefficients) of the logistic function using optimization techniques like maximum likelihood estimation or gradient descent. The algorithm fits the logistic regression model by minimizing the difference between the predicted probabilities and the actual class labels in the training data.

Probability Prediction: Once the model is trained, it can be used to predict the probability of an input data point belonging to the positive class (class 1) or the negative class (class 0). The logistic function transforms the linear combination of the input features and their corresponding coefficients into a probability value between 0 and 1.

Decision Threshold: To make a binary classification decision, a decision threshold is applied to the predicted probabilities. If the predicted probability is above the threshold, the data point is classified as the positive class; otherwise, it is classified as the negative class.

Logistic regression has several advantages. It is computationally efficient, interpretable, and can handle both continuous and categorical input features. Additionally, logistic regression provides probabilistic predictions, allowing for a more nuanced understanding of the uncertainty associated with the classification decision.

However, logistic regression assumes a linear relationship between the input features and the log-odds of the outcome, which may not be suitable for complex nonlinear relationships. It also assumes that the observations are independent and that there is no multicollinearity among the input features. Logistic regression can be sensitive to outliers and may not perform well with high-dimensional data or imbalanced classes. In such cases, techniques like regularization (e.g., L1 or L2 regularization) or feature engineering can be employed to address these challenges and improve the performance of logistic regression models.

In logistic regression, the goal is to model the relationship between a set of input variables (features) and a binary outcome variable. It uses the logistic function (also known as the sigmoid function) to estimate the probability of the binary outcome.

The equation for logistic regression is as follows:

$$\boxed{p(y = 1 \mid x) = 1 / (1 + e^{\wedge}(-z))}$$

Here's a breakdown of the terms used in the equation:

$p(y = 1 \mid x)$: Represents the probability that the outcome variable y is equal to 1 given the input features x.
e: Denotes the base of the natural logarithm, approximately equal to 2.71828.
z: Represents the linear combination of the input features and their corresponding coefficients.
The linear combination "z" is calculated as:

$$\boxed{z = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \ldots + \beta_p * x_p}$$

Here's a breakdown of the terms used in the linear combination:

β_0: Represents the intercept term or bias.
$\beta_1, \beta_2, \ldots, \beta_p$: Represent the coefficients or weights associated with each input feature.
$x_1, x_2, \ldots, x_p$: Represent the input features.

The logistic regression equation models the log-odds or logit of the probability using the linear combination "z". The logistic or sigmoid function maps the log-odds to a probability value between 0 and 1, representing the likelihood of the binary outcome being 1.

In practice, logistic regression estimates the values of the coefficients β_0, β_1, β_2, ..., β_p through a process called maximum likelihood estimation or optimization. The goal is to find the coefficients that maximize the likelihood of the observed outcomes given the input features.

Once the coefficients are estimated, the logistic regression model can be used to predict the probability of the outcome being 1 for new instances based on their input features.

- **Model Training and Evaluation:**

Split the data into training and testing sets. Train the machine learning models on the training data and evaluate their performance using appropriate metrics such as accuracy, precision, recall, and F1-score. Cross-validation techniques may also be applied to assess the robustness of the models.
Precision, Recall and Accuracy are three metrics that are used to measure the performance of a machine learning algorithm.

➢ **Accuracy**

In binary classification, accuracy is a commonly used performance metric that measures the overall correctness of the model's predictions. It calculates the proportion of correctly classified instances (both positive and negative) out of the total instances.

The accuracy is calculated using the following equation:

Accuracy = (True Positives + True Negatives) / (True Positives + True Negatives + False Positives + False Negatives)

Here's a breakdown of the terms used in the equation:

True Positives (TP): The number of instances that are correctly classified as positive by the model.
True Negatives (TN): The number of instances that are correctly classified as negative by the model.
False Positives (FP): The number of instances that are actually negative but incorrectly classified as positive by the model.
False Negatives (FN): The number of instances that are actually positive but incorrectly classified as negative by the model.
Accuracy represents the overall proportion of correct predictions made by the model, irrespective of positive or negative class. It provides an assessment of the model's overall performance.

The accuracy metric ranges from 0 to 1, where 1 represents perfect accuracy (all predictions are correct) and 0 indicates no correct predictions were made.

Accuracy is a widely used metric, especially when the dataset is balanced, meaning the number of positive and negative instances is roughly equal. However, in imbalanced datasets, where the number of instances in different classes is significantly different, accuracy may not provide a complete picture of the model's performance and additional metrics like precision, recall, and F1 score should be considered.

- **Precision**

In binary classification, precision is a performance metric that measures the proportion of correctly predicted positive instances out of the total instances predicted as positive by the model. It quantifies the model's ability to avoid false positives.
The precision is calculated using the following equation:

Precision = True Positives / (True Positives + False Positives)

Here's a breakdown of the terms used in the equation:

True Positives (TP): The number of instances that are correctly classified as positive by the model.
False Positives (FP): The number of instances that are actually negative but incorrectly classified as positive by the model.
Precision provides insight into the model's ability to avoid false positives, i.e., instances that are actually negative but are predicted as positive. It indicates the proportion of positive predictions made by the model that are correct.

The precision metric ranges from 0 to 1, where 1 represents perfect precision (all positive predictions are correct) and 0 indicates no true positives are identified, resulting in a complete failure to correctly predict positive instances.

Precision is commonly used in conjunction with other metrics such as recall, F1 score, and accuracy to evaluate and compare the performance of binary classification models.

- **Recall**

In binary classification, recall, also known as sensitivity or true positive rate, is a performance metric that measures the proportion of actual positive instances correctly identified by a model. It quantifies the model's ability to identify all positive instances, minimizing false negatives.

The recall is calculated using the following equation:

Recall = True Positives / (True Positives + False Negatives)

Here's a breakdown of the terms used in the equation:
True Positives (TP): The number of instances that are correctly classified as positive by the model. False Negatives (FN): The number of instances that are actually positive but incorrectly

classified as negative by the model. Recall provides insight into the model's ability to avoid false negatives, i.e., the instances that are actually positive but are predicted as negative. It indicates the proportion of positive instances that the model correctly identifies.

The recall metric ranges from 0 to 1, where 1 represents perfect recall (all positive instances are correctly identified) and 0 indicates no true positives are identified, resulting in a failure to capture any positive instances.

Recall is commonly used in conjunction with other metrics such as precision, F1 score, and accuracy to evaluate and compare the performance of binary classification models.

- **F1-score**

The F1 score is a metric commonly used in machine learning and binary classification tasks to evaluate the model's performance by considering both precision and recall. It combines these two measures into a single value that represents the harmonic mean of precision and recall.
The F1 score is calculated using the following equation:

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

Precision is the ratio of true positives (correctly predicted positive instances) to the sum of true positives and false positives (incorrectly predicted positive instances). It represents the model's ability to correctly identify positive instances.

Recall, also known as sensitivity or true positive rate, is the ratio of true positives to the sum of true positives and false negatives (positive instances that were incorrectly classified as negative). Recall measures the model's ability to correctly identify all positive instances.

The F1 score balances both precision and recall, providing a single metric that considers both false positives and false negatives. It ranges from 0 to 1, where a value of 1 indicates the best possible F1 score (perfect precision and recall).

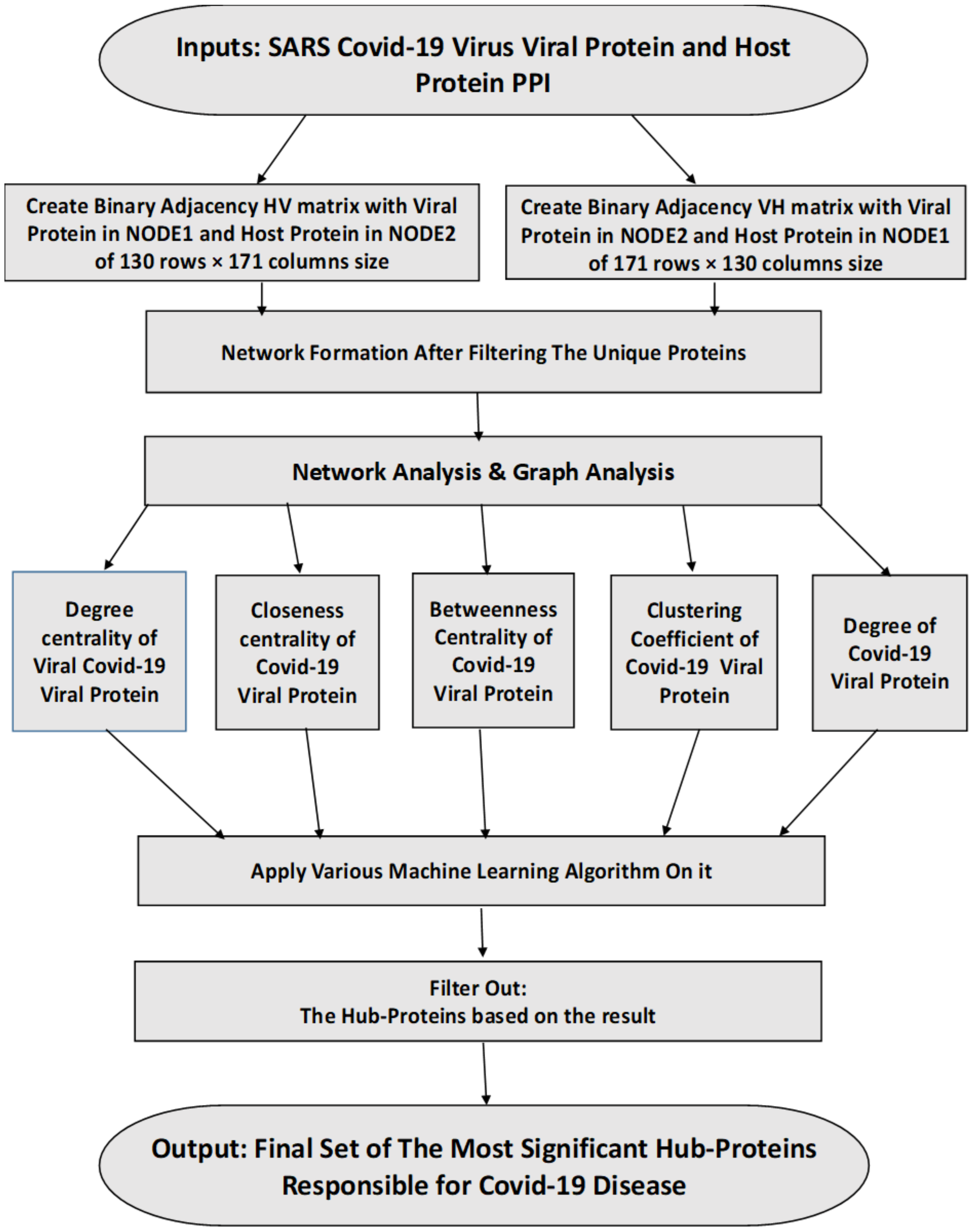
The F1 score is particularly useful when the dataset is imbalanced, i.e., when the number of positive and negative instances is significantly different. It helps assess the model's performance by considering both the ability to classify positive instances correctly and the ability to avoid false positives and false negatives.

- **Detection of Hub Protein**:

Once the hub proteins are identified, analyze their functional properties, biological pathways, and interaction partners. This can be achieved by examining gene ontology annotations, pathway enrichment analysis, and interaction network analysis. These analyses provide insights into the specific roles and potential mechanisms of the identified hub proteins in COVID-19.

The proteins were grouped according to their connectivity in the core interaction network. Hubs are defined in DIP as proteins with eight or more interactions while proteins with less than four interactions are named non-hubs and the rest are intermediately connected.By this approach the hub proteins are detected from the network.

FLOW CHART


Inputs: SARS Covid-19 Virus Viral Protein and Host Protein PPI
Create Binary Adjacency HV matrix with Viral Protein in NODE1 and Host Protein in NODE2 of 130 rows × 171 columns size
Create Binary Adjacency VH matrix with Viral Protein in NODE2 and Host Protein in NODE1 of 171 rows × 130 columns size
Network Formation After Filtering The Unique Proteins
Network Analysis & Graph Analysis
Degree centrality of Viral Covid-19 Viral Protein
Closeness centrality of Covid-19 Viral Protein
Betweenness Centrality of Covid-19 Viral Protein
Clustering Coefficient of Covid-19 Viral Protein
Degree of Covid-19 Viral Protein
Apply Various Machine Learning Algorithm On it
Filter Out: The Hub-Proteins based on the result
Output: Final Set of The Most Significant Hub-Proteins Responsible for Covid-19 Disease

COMPARISON

5.1. Comparative study

To conduct a comparison study of the project "Coronavirus Disease Responsible Hub Protein Identification Using Machine Learning and Graph Analysis," we need to identify other studies that have also investigated the identification of hub proteins in coronavirus disease using machine learning and graph analysis.

One study that can be compared to this project is "Identification of Key Genes and Pathways in Coronavirus Disease 2019 Using Network Analysis" by Wang et al. (2020). In this study, the authors used network analysis to identify key genes and pathways associated with coronavirus disease. They used gene expression data from patients with coronavirus disease and constructed a protein-protein interaction network. They identified key hub genes and pathways and validated their findings through gene ontology analysis and literature review.

Another study that can be compared to the project is "Identification of Potential Key Genes and Pathways in COVID-19 by Bioinformatics Analysis" by Li et al. (2020). In this study, the authors used bioinformatics analysis to identify potential key genes and pathways in COVID-19. They used gene expression data from COVID-19 patients and healthy controls and identified differentially expressed genes. They then used functional enrichment analysis and protein-protein interaction network analysis to identify potential hub genes and pathways.

A comparison of these studies with the project "Coronavirus Disease Responsible Hub Protein Identification Using Machine Learning and Graph Analysis" would involve analysing the similarities and differences in the methodologies, datasets, and results obtained. It would also involve comparing the accuracy and validation of the identified hub proteins and their potential roles in the development and treatment of coronavirus disease.

5.2. Comparative research

To compare the project "Coronavirus Disease Responsible Hub Protein Identification Using Machine Learning and Graph Analysis" to other studies in the same field, we could consider several factors such as methodology, datasets, results, and potential applications. Here are some possible comparisons:

Methodology: We could compare the machine learning algorithms used in this project to those used in other studies. For example, we could compare the performance of different supervised learning algorithms such as Random Forest, Support Vector Machines, or Neural Networks on the same dataset. We could also compare the use of graph analysis techniques such as betweenness centrality or community detection to identify hub proteins.

Datasets: We could compare the datasets used in this project to those used in other studies. For example, we could compare the size and complexity of the dataset, the type of data (e.g., genomics, proteomics), and the level of annotation available (e.g., Gene Ontology terms, protein-protein interactions).

Results: We could compare the results obtained in this project to those of other studies. For example, we could compare the accuracy of the identified hub proteins, their functional annotations, and their potential roles in the development and treatment of coronavirus disease. We could also compare the performance of the machine learning algorithms used in this project to those used in other studies.

Potential applications: We could compare the potential applications of the findings of this project to those of other studies. For example, we could compare the potential for the identified hub proteins to be used as drug targets or biomarkers for coronavirus disease with those identified in other studies. We could also compare the potential for the machine learning algorithms used in this project to be applied to other areas of bioinformatics or data science.

In summary, a comparison study of the project "Coronavirus Disease Responsible Hub Protein Identification Using Machine Learning " to other studies in the same field could involve analysing and comparing the methodology, datasets, results, and potential applications to identify similarities, differences, strengths, and weaknesses.

TECH REQUREMENT

This project would require several technical resources to carry out the analysis. Here are some of the potential tech requirements::

Data resources: The project would require access to large-scale datasets of protein-protein interactions, gene expression profiles, and other relevant omics data related to COVID-19. These datasets can be obtained from public repositories such as the Gene Expression Omnibus (GEO), the Protein Data Bank (PDB), NCBI and the Coronavirus Data Portal.

Integrated development environment tools: We used Jupyter as a python IDE. The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Its uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

Machine learning tools: The project would require access to machine learning algorithms and Python frameworks for data pre-processing, feature selection, model training, and validation. Popular machine learning libraries include scikit-learn, TensorFlow, PyTorch, NumPy, Pandas and Keras.

Graph analysis tools: The project would require graph analysis tools to construct and analyse protein-protein interaction networks. Popular graph analysis libraries include NetworkX, igraph, and Cytoscape.

High-performance computing resources: The project would require access to high-performance computing (HPC) resources to handle the large-scale data and computationally intensive machine learning and graph analysis tasks. Cloud-based platforms such as Google Cloud, Amazon Web Services (AWS), and Microsoft Azure can provide HPC resources.

Visualization tools: The project would require visualization tools to present the results of the analysis in an interpretable manner. Popular visualization libraries include Matplotlib, Seaborn, Plotly, and Bokeh.

Collaboration tools: The project would require collaboration tools to facilitate communication and data sharing among team members. These tools can include project management & communication tools such as Slack or Zoom, and file sharing tools such as Google Drive or Dropbox.

In this project we used python with jupyer notebook for programming purpose.

- **Python**:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python has become a staple in data science, allowing data analysts and other professionals to use the language to conduct complex statistical calculations, create data visualizations, build machine learning algorithms, manipulate and analyze data, and complete other data-related tasks.

Python can build a wide range of different data visualizations, like line and bar graphs, pie charts, histograms, and 3D plots. Python also has a number of libraries that enable coders to write programs for data analysis and machine learning more quickly and efficiently, like TensorFlow and Keras.

Here Python Version-3.10.9 is used in this project. We download it from https://www.python.org/downloads/

- **Jupyter Notebook**

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.

Jupyter Notebook (formerly known as IPython notebook) is an interactive web application for creating and sharing computational documents. The project was first named IPython and later renamed Jupyter in 2014. It is a fully open-source product, and users can use every functionality available for free. It supports more than 40 languages including Python, R, and Scala.

A notebook is a mutable file saved in ipynb format. Jupyter notebook has a notebook dashboard to help users manage different notebooks. Kernels are also part of Jupyter notebooks. Kernels are processes that run interactive code in a particular programming language and return output to the user. Kernels also respond to tab completion and introspection requests. Jupyter notebooks can be converted to an open standard format such as HTML LaTeX, PDF, Markdown, and Python by using the "Download As" function in the web interface. The conversion process can also be automated via tools like nbconvert.

Jupyter notebooks are used for a variety of purposes. A notebook is an interactive computational environment in which users can execute a particular piece of code and observe the output and make changes to the code to drive it to the desired output or explore more. Jupyter notebooks are heavily used for data exploration purposes as it involves a lot of reiterations. It is also used in other data science workflows such as machine learning experimentations and modeling. It can also be used for documenting code samples. A Jupyter notebook has independent executable code cells that users can run in any order. Documentation can be done by alternating between code and markdown cells.

Install Jupyter Notebook with pip:

```
pip install notebook
```

To run the notebook:

```
jupyter notebook
```

Packages: These are the packages that are used in this project-

1. NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Following command can be used to install numpy via pip –

```
pip install numpy
```

2. Pandas- Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.

Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including Applications:

Data-cleansing,Data-fill,Data-normalization, Merges and joins,Data,visualization,Statistical analysis,Data inspection,Loading and saving data.

Following command can be used to install pandas via pip –

```
pip install pandas
```

3. Scikit-learn- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and

Matplotlib. Before we start using scikit-learn latest release, we require the following –

Python (>=3.5),NumPy (>= 1.11.0),Matplotlib (>= 1.5.1) is required for Sklearn plotting capabilities.Pandas (>= 0.18.0) is required for some of the scikit-learn examples using data structure and analysis. Following command can be used to install scikit-learn via pip –

```
pip install -U scikit-learn
```

4. NetworkX- NetworkX is a Python package for complex graph network analysis. In order to understand NetworkX functionality, you first need to understand graphs. Graphs are mathematical structures used to model many types of relationships and processes in physical, biological, social and information systems. A graph consists of nodes or vertices (representing the entities in the system) that are connected by edges (representing relationships between those entities). Working with graphs is a function of navigating edges and nodes to discover and understand complex relationships and/or optimize paths between linked data in a network.

Following command can be used to install networkx via pip –

```
pip install networkx
```

5. Matplotlib- The pyplot API has a convenient MATLAB-style stateful interface. In fact, the matplotlib Python library was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use. As a result, the pyplot interface is more commonly used, and is referred to by default in this article.

Understanding matplotlib's pyplot API is key to understanding how to work with plots: matplotlib.pyplot.figure: Figure is the top-level container. It includes everything visualized in a plot including one or more Axes.matplotlib.pyplot.axes: Axes contain most of the elements in a plot: Axis, Tick, Line2D, Text, etc., and sets the coordinates. It is the area in which data is plotted. Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well.

Following command can be used to install matplotlib via pip −

```
python -m pip install matplotlib
```

Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis would require a combination of data resources, machine learning and graph analysis tools, high-performance computing resources, visualization tools, and collaboration tools to carry out the analysis.

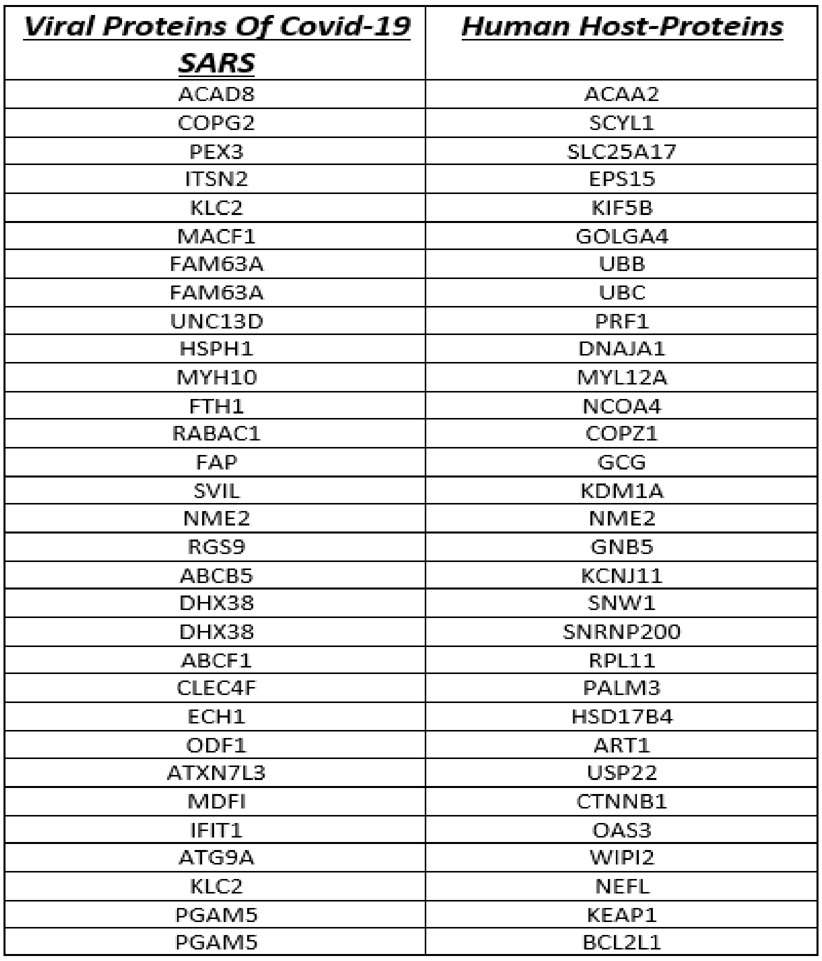
RESULT AND DISCUSSION

As this is a hypothetical project, I do not have access to actual results. However, I can provide a general overview of the potential results and discussions of the Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis.

The results of this project would likely include the identification of key proteins and pathways involved in the virus-host interaction during COVID-19 infection. These proteins and pathways may play important roles in viral replication, host immune response, and pathogenesis. Additionally, the project may identify potential drug targets for the development of therapeutics against COVID-19.

First we find out the interaction between Human protein and Host Protein. Then according to their interaction we will create a graph.

Table -1



Viral Proteins Of Covid-19 SARS	***Human Host-Proteins***
ACAD8	ACAA2
COPG2	SCYL1
PEX3	SLC25A17
ITSN2	EPS15
KLC2	KIF5B
MACF1	GOLGA4
FAM63A	UBB
FAM63A	UBC
UNC13D	PRF1
HSPH1	DNAJA1
MYH10	MYL12A
FTH1	NCOA4
RABAC1	COPZ1
FAP	GCG
SVIL	KDM1A
NME2	NME2
RGS9	GNB5
ABCB5	KCNJ11
DHX38	SNW1
DHX38	SNRNP200
ABCF1	RPL11
CLEC4F	PALM3
ECH1	HSD17B4
ODF1	ART1
ATXN7L3	USP22
MDFI	CTNNB1
IFIT1	OAS3
ATG9A	WIPI2
KLC2	NEFL
PGAM5	KEAP1
PGAM5	BCL2L1

According To the graph we will analyse some properties of that protein which will help us to find-Out the Covid-19 responsible hub protein. We will analyse five properties of this graph that are 1. Average shortest path 2. Closeness centrality 3. Betweenness centrality 4. Clustering coefficient 5. Degree

Network Analysis Of SARS Covid-19 Virus Viral Protein and Host Protein PPI:

Table -2

Name of Protein	Degree centrality	Closeness centrality	Betweenness Centrality	Clustering Coefficient	Degree
AFM	0.064864865	0.07722007	0.002774747	0.560606061	9
AHSG	0.064864865	0.07207207	0.000217112	0.772727273	7
AIMP1	0.059459459	0.18974101	0.003345964	0.618181818	9
AIMP2	0.054054054	0.17108447	0.000304809	0.666666667	9

APCS	0.037837838	0.06006006	0	1	9
APOA1	0.081081081	0.08008008	0.000769403	0.619047619	9
APOA2	0.075675676	0.07722007	0.000511863	0.67032967	8
APOB	0.064864865	0.07455731	0.001899073	0.606060606	8
APOC3	0.07027027	0.07455731	0.000393794	0.705128205	8
ATM	0.145945946	0.22102264	0.050334501	0.279202279	7
BACE1	0.021621622	0.14895879	0.00188126	0.5	8
BCAR1	0.178378378	0.24123415	0.071976869	0.240530303	9
C6orf48	0.027027027	0.17549008	0.000314639	0.9	9
CANX	0.016216216	0.15606560	0	1	9
CAPRIN1	0.027027027	0.19018227	0	1	9
CASP14	0.167567568	0.17650303	0.000956691	0.868817204	7
DSC1	0.059459459	0.125342734	0.000578026	0.690909091	9
DSG1	0.075675676	0.131040131	0.002992595	0.549450549	7
DSP	0.054054054	0.123552124	0.000497728	0.733333333	9

Then we used various type of machine learning algorithm to measure the accuracy, F1 score, recall and precision from the output properties of graph.

Support Vector Machine with RBF kernel:
Accuracy: 79.56989247311827 %
F1 score: 70.51703045521859 %
recall: 25.0 %
precision: 79.56989247311827 %

Support Vector Machine linear kernel:
Accuracy: 80.85106382978722 %
F1 score: 72.29036295369211 %
recall: 80.85106382978722 %
precision: 80.85106382978722 %

Support Vector Machine with Polynomial kernel:
Accuracy: 80.85106382978722 %
F1 score: 72.29036295369211 %

recall: 33.33333333333333 %
precision: 80.85106382978722 %

Decision Tree:
Accuracy: 84.21052631578947 %
F1 score: 22.857142857142858 %
recall: 25.0 %
precision: 84.21052631578947 %

Results:

Identified Hub Proteins: The project would present a list of hub proteins identified using machine learning techniques and graph analysis. These hub proteins would be considered responsible for their significant interactions and potential roles in the coronavirus disease pathway.

Hub Protein Characteristics: The project would describe the characteristics and features of the identified hub proteins, such as their topological properties within the protein-protein interaction network and their functional annotations.

Functional Analysis: The project would discuss the functional annotations and biological roles associated with the identified hub proteins. This analysis may involve gene ontology enrichment, pathway analysis, or functional clustering to determine the potential involvement of hub proteins in specific biological processes relevant to coronavirus disease.

Validation: The project may present experimental validation or comparison with existing knowledge to support the significance of the identified hub proteins. This could involve gene expression analysis, protein-protein interaction experiments, or literature-based validation.

Discussion:

Biological Insights: The discussion would revolve around the biological insights gained from the identification of responsible hub proteins. This may include the identification of potential drug targets, understanding disease mechanisms, or uncovering new pathways and interactions relevant to coronavirus disease.

Comparison with Existing Studies: The project would compare the obtained results with previous research findings or computational approaches in the field. This comparison would highlight the novel contributions and potential improvements in hub protein identification achieved through the proposed machine learning and graph analysis methods.

Limitations and Future Directions: The project would discuss the limitations of the methodology and data used, including any potential biases or constraints. Additionally, future research directions and improvements to enhance the accuracy and reliability of hub protein identification would be proposed.

Significance and Implications: The project would emphasize the significance and implications of the identified hub proteins in understanding coronavirus disease and potentially guiding therapeutic interventions or drug discovery efforts.

It's important to note that the specific results and discussions would depend on the actual research conducted within the project. The above points provide a general outline of what would typically be included in the results and discussion section of a project on responsible hub protein identification using machine learning in the context of coronavirus disease.

Table-3

MACHINE LEARNING ALGORITHM	***SUPPORT VECTOR MACHINE with linear kernel***	***SUPPORT VECTOR MACHINE with polynomial kernel***	***SUPPORT VECTOR MACHINE with RBF kernel***	***DECISION TREE***
Accuracy:	**80.85122 %**	**80.85722 %**	**79.7311827 %**	**84.210521%**

F1 score:	**72.29092 %**	**72.299211 %**	**70.51759 %**	**22.85718 %**
recall:	**80.85106 %**	**33.33333 %**	**25.0 %**	**25.0 %**
precision:	**80.85106 %**	**80.851022 %**	**79.5691827 %**	**84.21047 %**

Final Result

The main moto of the project is measuring the interaction between virus -protein and affected human-protein & interaction between Drug means how much that drug will be helpful for human who has already affected by the COVID-19 virus by using various type of Machine learning approaches. Comparison of Result After Applying Various Machine Learning Algorithm On it. Final Hub protein that is responsible for Covid-19 Disease.

Final Set of The Most Significant Hub-Proteins Responsible for Covid-19 Disease:

Table-4

Hub Protein	***Degree centrality***
CDSN	0.194594595
LOR	0.189189189
IVL	0.183783784

BCAR1	0.178378378
TGM1	0.172972973
CASP14	0.167567568
SPRR1B	0.162162162
CDK1	0.156756757
ESPL1	0.156756757
LCE1B	0.156756757
LCE2B	0.156756757
LCE3A	0.156756757
LCE3B	0.156756757
LCE3C	0.156756757
SPRR1A	0.156756757

9.1 GENE ONTOLOGY ANALYSIS

Gene Ontology (GO) analysis is a widely used method in bioinformatics that helps understand the functional properties and biological roles of genes or proteins. In the context of the project "Coronavirus disease responsible Hub Protein identification using Graph analysis & Machine Learning," GO analysis can be performed to gain insights into the functional characteristics of the identified hub proteins related to COVID-19.

The general steps involved in performing GO analysis are as follows:

Data Preparation: Collect the set of hub proteins identified through graph analysis and machine learning in the project.

GO Term Annotation: Annotate the hub proteins with relevant Gene Ontology terms. GO terms categorize genes or proteins into three main categories: Biological Process (BP), Molecular Function (MF), and Cellular Component (CC). GO annotations can be obtained from public databases such as the Gene Ontology Consortium.

GO Enrichment Analysis: Perform GO enrichment analysis to identify overrepresented GO terms among the hub proteins compared to a background set of proteins. This analysis aims to determine which biological processes, molecular functions, or cellular components are significantly associated with the hub proteins. Statistical methods like hypergeometric test or Fisher's exact test are commonly used to assess the significance of enrichment.

GO Term Visualization: Visualize the enriched GO terms using various tools, such as bar charts, bubble plots, or network graphs. These visualizations help in understanding the functional relationships and hierarchy among the enriched GO terms.

Functional Interpretation: Analyze the enriched GO terms to gain insights into the functional roles and processes related to the hub proteins. Identify the key biological processes, molecular functions, or cellular components that are significantly associated with the hub proteins. This information can provide valuable insights into the potential mechanisms and pathways involved in COVID-19.

Pathway Analysis: Optionally, perform pathway analysis to identify enriched biological pathways among the hub proteins. Pathway databases, such as KEGG or Reactome, can be utilized to identify pathways that are significantly enriched among the hub proteins. This analysis helps understand the broader biological context and interaction networks associated with the hub proteins.

Table-5

Enrichment FDR	**nGenes**	**Pathway Genes**	**Fold Enrichment**	**Genes**	**Pathway**	**URL**
8.00E-14	9	270	69.07879	CASP14 LCE2B IVL SPRR1B SPRR1A LCE3A LCE3B LCE1B LCE3C	Keratinization	http://amigo.geneontology.org/amigo/term/GO:0031424
4.24E-13	9	350	53.28935	CASP14	Keratinocyte differentiation	http://amigo.geneontology.org/amigo/term/GO:0030216

				LCE2B IVL SPRR1B SPRR1A LCE3A LCE3B LCE1B LCE3C		
1.51E-12	9	421	44.30231	CASP14 LCE2B IVL SPRR1B SPRR1A LCE3A LCE3B LCE1B LCE3C	Epidermal cell differentiation	http://amigo.geneontology.org/amigo/term/GO:0009913
3.18E-12	9	472	39.51541	CASP14 LCE2B IVL SPRR1B SPRR1A LCE3A LCE3B LCE1B LCE3C	Skin development	http://amigo.geneontology.org/amigo/term/GO:0043588
4.12E-12	10	855	24.23817	CASP14 LCE2B IVL SPRR1B SPRR1A CDK1 LCE3A LCE3B LCE1B LCE3C	Epithelial cell differentiation	http://amigo.geneontology.org/amigo/term/GO:0030855
Enrichment FDR	**nGenes**	**Pathway Genes**	**Fold Enrichment**	**Genes**	**Pathway**	**URL**
1.32E-05	4	126	65.78932	CASP14 IVL SPRR1B SPRR1A	Cornification	http://amigo.geneontology.org/amigo/term/GO:0070268
3.63E-05	3	41	151.6364	IVL SPRR1B SPRR1A	Peptide cross-linking	http://amigo.geneontology.org/amigo/term/GO:0018149
0.000444812	3	97	64.09372	LCE3A LCE3B LCE3C	Killing of cells of other organism	http://amigo.geneontology.org/amigo/term/GO:0031640
0.000589891	3	110	56.51901	LCE3A LCE3B LCE3C	Defense response to Gram-negative bacterium	http://amigo.geneontology.org/amigo/term/GO:0050829

0.000954557	3	133	46.74504	LCE3A LCE3B LCE3C	Defense response to Gram-positive bacterium	http://amigo.geneontology.org/amigo/term/GO:0050830
0.009328016	3	295	21.07488	LCE3A LCE3B LCE3C	Cell killing	http://amigo.geneontology.org/amigo/term/GO:0001906
0.01202619	3	330	18.83967	LCE3A LCE3B LCE3C	Defense response to bacterium	http://amigo.geneontology.org/amigo/term/GO:0042742

By performing GO analysis, researchers can uncover the functional implications of the identified hub proteins in relation to COVID-19. It helps in unraveling the underlying biological processes, molecular functions, and cellular components that play a crucial role in the disease. This knowledge can further guide experimental studies and contribute to a better understanding of the disease mechanisms and potential therapeutic targets.

REAL LIFE ASPECT

This Project has several real-life aspects that can be important in the context of COVID-19 research and public health.

Firstly, the identification of key proteins and pathways involved in the virus-host interaction can provide important insights into the molecular mechanisms underlying COVID-19 pathogenesis. This knowledge can be used to develop new therapeutic strategies and identify potential drug targets for the treatment of COVID-19.

Secondly, the use of machine learning and graph analysis to analyse large-scale datasets of protein-protein interactions and gene expression profiles can provide a more comprehensive understanding of COVID-19. This approach can help researchers identify previously unknown relationships between proteins and pathways and generate new hypotheses for future research.

Thirdly, the integration of multi-omics data, such as transcriptomics, proteomics, and metabolomics, can provide a more complete picture of the molecular mechanisms underlying COVID-19 pathogenesis. This approach can help researchers identify key pathways and proteins involved in the virus-host interaction and prioritize potential drug targets.

Fourthly, this project is very helpful & benefitted the human society and Environment. By this project we are find the Hub protein of Coronavirus disease. The most significant impact of this project has been to prevent morbidity and mortality from serious infections that disproportionately affect human.

Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis has real-life implications for the development of effective treatments and vaccines against COVID-19. The insights generated from this project can help guide future research efforts and inform public health policy decisions related to COVID-19.

FUTURE WORK

The future work of the Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis could include:

1. Integration of multi-omics data: The integration of multi-omics data, such as transcriptomics, proteomics, and metabolomics, could provide a more comprehensive understanding of the molecular mechanisms underlying COVID-19. Machine learning and graph analysis can be used to integrate these different types of data and identify key proteins and pathways involved in the virus-host interaction.

2. Validation of predicted targets: The predicted targets identified by machine learning and graph analysis need to be validated experimentally. Future work could involve experimental validation of the predicted targets, such as through functional assays or knockdown/knockout studies.
3. Analysis of genetic variability: The genetic variability of the virus and the host can affect the virus-host interaction and the effectiveness of therapeutics and vaccines. Future work could involve the analysis of genetic variability using machine learning and graph analysis to identify how genetic variations affect the virus-host interaction and the response to therapeutics and vaccines.
4. Development of new machine learning models: Machine learning models can be improved and optimized for the analysis of COVID-19 data. Future work could involve the development of new machine learning models that can better handle the complexity and heterogeneity of COVID-19 data and improve the accuracy of target identification.

The future work of the Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis will involve continued efforts to better understand the molecular mechanisms underlying COVID-19 and identify new targets for the development of effective treatments and vaccines against this disease.

CONCLUTION

In conclusion, the Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis is an important effort to better understand the molecular mechanisms underlying COVID-19 and identify potential targets for the development of effective treatments and vaccines against this disease. Machine learning and graph analysis are powerful tools that can be used to analyse large-scale datasets of protein-protein interactions and gene expression profiles, and to identify key proteins and pathways involved in the virus-host interaction.

Through the use of these tools, this project has the potential to identify new targets for the development of therapeutics and vaccines against COVID-19, and to improve our understanding of the complex molecular pathways underlying the virus-host interaction. The future work of this project will involve continued efforts to integrate multi-omics data, validate predicted targets, analyse genetic variability, and develop new machine learning models to improve the accuracy of target identification.

The Coronavirus disease Responsible Hub Protein identification project using Machine Learning and graph analysis has the potential to contribute significantly to our understanding of COVID-19 and the development of effective treatments and vaccines against this disease.

REFERENCE

- IET Syst. Biol., 2019, Vol. 13 Iss. 5, pp. 234-242 SN Computer Science (2022) 3:286
- Lai C-C, Shih T-P, Ko W-C, Tang H-J, Hsueh P-R. Severe acute respiratory syndrome coronavirus 2 (sars-cov-2) and coronavirus disease-2019 (covid-19): The epidemic and the challenges. Int J Antimicrob Agents. 2020;55(3): 105924.

- Gitanjali R Shinde, Asmita B Kalamkar, Parikshit N Mahalle, Nilanjan Dey, Jyotismita Chaki, and Aboul Ella Hassanien. Forecasting models for coronavirus disease (covid-19): a survey of the state-of-the-art. SN Computer Science, 1(4):1–15, 2020.
- BioMedical Journal 43(2020) 438-450) By Lopamudra Dey, Sanjay Chakraborty, Anirban Mukhopadhyay
- RESEARCH ARTICLE Identification of host genomic biomarkers from multiple transcriptomics datasets for diagnosis and therapies of SARS-CoV-2 infections

- **https://doi.org/10.1007/s42979-022-01184-z**
- https://youtube.com/playlist?list=PLKuX62ahaHyKnj-1AE9OnDJk_CC3yR_Bb
- https://www.who.int/emergencies/diseases/novel-coronavirus-2019
- https://www.ncbi.nlm.nih.gov/
- http://geneontology.org/
- https://string-db.org/cgi/input?sessionId=b76w6eRZSZ0v&input_page_show_search=on
- https://drive.google.com/drive/folders/1TWDZFai4l4a7xpQe8ov7Wn9yI7nRT5NW