Statistical Analysis of Plasma Proteomics Data on Covid-19 Patients using High Dimensional Data Reduction Techniques

M.Sc. Project – Report

Stage-1

by

Subhojit Kayal

Roll No: 205280022

Guide: Prof. Sanjeev V. Sabnis
Department of Mathematics
IIT Bombay

Co-guide: Prof. Sanjeeva Srivastava

Department of Biosciences and Bioengineering

IIT Bombay

2021-22

APPROVAL SHEET

This report entitled **"Statistical Analysis of Plasma Proteomics Data on Covid-19 Patients using High Dimensional Data Reduction Techniques"** by Subhojit Kayal is approved for the degree of Master of Science in 'Applied Statistics and Informatics'.

____

Prof. Monika Bhattacharya
(Examiner)

____

Prof. Sanjeev V. Sabnis
(Guide)

____

Prof. Sanjeeva Srivastava
(Co-guide)

____

Prof. Ayan Bhattacharya
(Chairman)

Date: _______________

Place: _______________

ACKNOWLEDGEMENT

I would like to thank and express my sincere gratitude towards my project guide, Professor Sanjiv V. Sabnis, Department of Mathematics, IIT Bombay for his continuous help, support and encouragement. I am extremely thankful to him for guiding me through the entire journey starting from reading papers together to learn new statistical techniques, or brushing up the basic concepts to the valuable suggestions at each and every step. Without his excellent guidance this report would not have been materialized.

I wish to thank my co-guide Professor Sanjeeva Srivastava, Department of Biosciences and Bioengineering, IIT Bombay for making available the Plasma Proteomics Dataset, which gave a practical aspect to this project and for his constant support.

I also wish to thank Dr. Arup Acharjee, Department of Biosciences and Bioengineering, IIT Bombay for helping me understand the biology behind the data and guide through all difficulties.

Subhojit Kayal

November 22, 2021

TABLE OF CONTENTS

Introduction.........6

Steps involved in this project.........8

Chapter 1: A Brief Introduction to Big Data 9-13

- 1.1 What is Big Data?.........9
- 1.2 Examples of Big Data.........10
- 1.3 Importance of Big Data.........11
- 1.4 Big Data Challenges.........11
- 1.5 Application of Big Data in Proteomics.........12
- 1.6 Salient Features of Big Data.........13
  - 1.6.1 Heterogeneity.........13
  - 1.6.2 Noise Accumulation.........13
  - 1.6.3 Spurious Correlation.........14
  - 1.6.4 Incidental endogeneity.........15

Chapter 2: Understanding the Data.........16-17

- 2.1 Description of the Data and Objective.........16
- 2.2 About Plasma Proteomics.........17

Chapter 3: Missing Value Imputation.........18-21

- 3.1 Initial missing value Checking 18
- 3.2 KNN Method: Theory.........19
- 3.3 Mean and Median Imputation.........20
- 3.4 Application on Our Dataset.........21

Chapter 4: Outlier Detection and Normalization.........22-27

- 4.1 Outlier Detection.........22
  - 4.1.1 Method of Box-plot.........23
  - 4.1.2 Application on our Dataset.........24
  - 4.1.3 Comment.........24
  - 4.1.4. Remove Outliers using IQR Method 25
  - 4.1.5 Conclusion of Outlier Detection.........26
- 4.2 Normalization.........26-28
  - 4.2.1 Min-Max Scaling.........26
  - 4.2.2 Standardization.........26
  - 4.2.3 Robust Scaling.........26
  - 4.2.4 Application on our dataset.........27-28

Chapter 5: Dimension Reduction Techniques........29-32

5.1 Sure independent screening........29
5.1.1Background........29
5.1.2 SIS: Theory........29
5.1.3 First Stage Dimension Reduction using SIS........29
5.2 LASSO: 30
5.2.1 LASSO: Theory........30
5.2.2 K-fold cross validation........31
5.2.3 Shrunken Centroids method........31
5.2.2 Second Stage Dimension Reduction using LASSO........32

Chapter 6: Conclusion........33

Chapter 7: Future Scope........33

References........33

Chapter 8: Python codes used........34-38

Introduction

- **History of Covid-19 and its application on big data analytics:**

On 30 January 2020, the World Health Organization (WHO) declared the spread of the COVID-19 pandemic as a cause of concern and called for raising the level of health emergencies. Afterward, the government of the Kingdom of Saudi Arabia urgently took several strict measures to limit the spread of the pandemic within the regions of Saudi Arabia. The Saudi Ministry of Health (MoH) and many other countries have implemented WHO recommendations related to the identification and isolation of suspected COVID-19 cases.

Nevertheless, the pandemic has spread dramatically, with the number of infected people over 82 million, and the number of deaths exceeding one million. The rapid spread of the pandemic, with its continuous evolving patterns and the difference in its symptoms, makes it more difficult to control. Moreover, the pandemic has affected health systems and the availability of medical resources in several countries around the world, contributing to the high death rate.

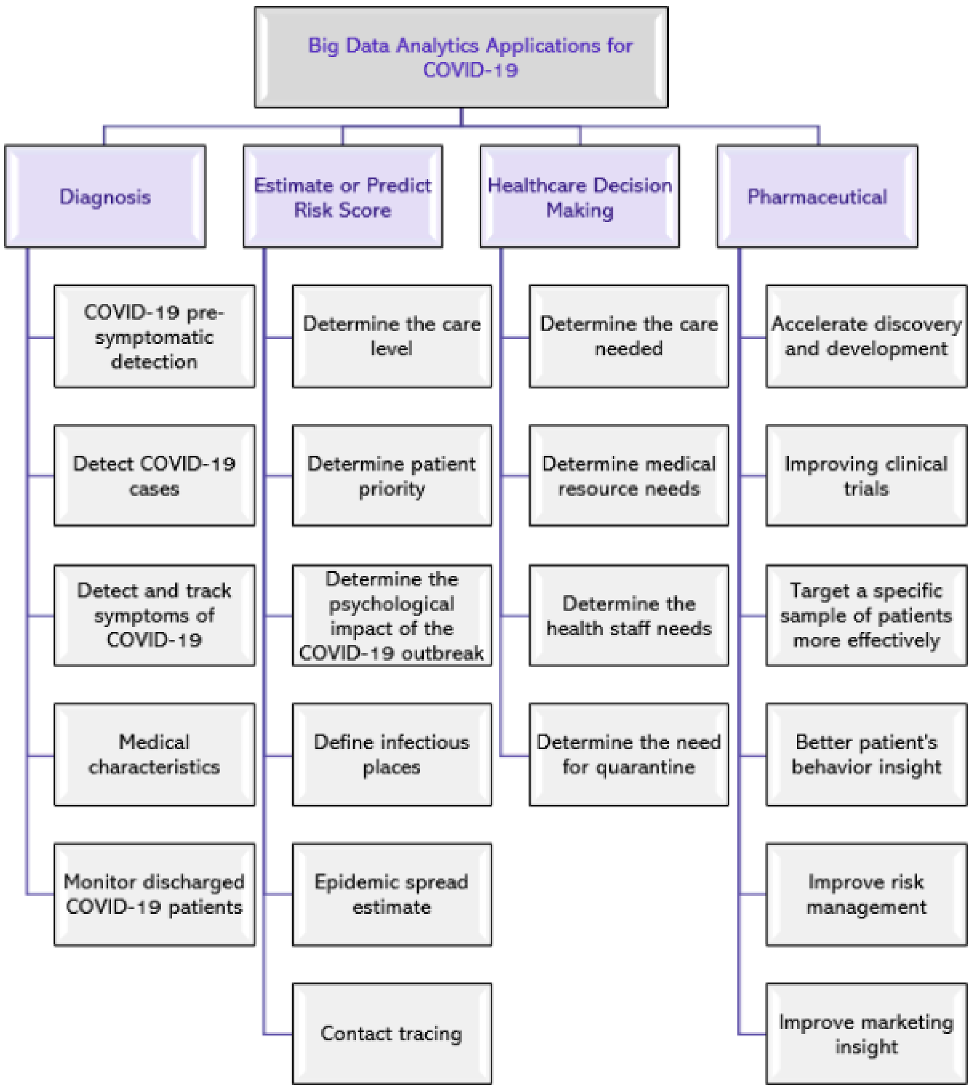
The big data characteristics apply to data acquired from the healthcare sector, which increases the tendency to use big data analysis tools to improve sector services and performance. There are wide applications of big data analytics in the healthcare sector, including genomics, drug discovery and clinical research, personalized healthcare, gynaecology, nephrology, oncology, and several other applications found in the literature. However, in this paper, we present the contributions of the most important review papers found in the literature that cover the field of big data in healthcare. We also investigate the opportunities and challenges for applying big data analytics tools to COVID-19 data and provide findings and future directions at the end of the paper.

Promising wearable technology is expected to be one of the primary sources of health information, given its widespread availability and acceptance by people. Based on a survey conducted in January 2020, 88% of 4600 subjects included in the study indicated a willingness to use wearable technology to measure and track their vital signs. While 47% of chronically ill patients and 37% of non-chronically ill patients reported a willingness to blindly share their health information with healthcare research organizations. Of the same group, 59% said they would likely use artificial intelligence (AI)-based services to diagnose their health symptoms [14]. People sharing such data routinely will greatly increase the volume of data, which calls for planning to design and implement data analysis tools and models in this sector.

Several studies used big data for sentiment analysis, such as Reference, which linked between social media behaviour and political views, opinions, and expressions. The study consisted of a representative survey conducted on 62.5% of adults from Chile and it showed the huge effect of social media on changing people’s opinions regarding political views and elections. Similarly, the authors of Reference [16] had studied how the management responding to customer satisfaction online review affects the choice of the customers for some facilities or hotels. It showed a positive correlation between the response and customer satisfaction. The authors of Reference [17] had reviewed the classification techniques, including deep and convolutional, to identify the writer from their handwriting. They discussed several challenges in identification related to language characteristics, scripts, and the lack of datasets. Also, the authors of Reference [18] had reviewed and analyzed the latest papers about big data analytics latest developments, capabilities, and profits. Their study showed that big data can support business industries in many functionalities including prediction, planning, managing, decision-making, and traceability. The limitation of their study is the data sources, which were hard to find due to privacy and conservation of the information. Moreover, the authors of Reference [19] had surveyed numerous papers about mathematical models to improve the efficiency in detecting and predicting COVID-19. Their survey suggested using artificial intelligence to detect COVID-19 cases, big data to trace cases, and nature-inspired computing (NIC) to select suitable features to increase the accuracy of detection. Some surveys studied heart-related diseases and suggested some recommendations and guidelines, such as Reference [20], to help people in understanding heart failure causes, symptoms, and the most affected group. They declared that heart failure can escalate the patient's injuries, especially the ones with serious illnesses.

- **Applications of Data Analytics in COVID-19**

The spread of the global pandemic, COVID-19, has generated a huge and varied amount of data, which is increasing rapidly. This data can be used by applying big data analytics techniques in multiple areas, including diagnosis, estimate or predict risk score, healthcare decision-making, and pharmaceutical industry [38]. Figure 1 shows examples of potential application areas.



Big Data Analytics Applications for COVID-19
Diagnosis
COVID-19 pre-symptomatic detection
Detect COVID-19 cases
Detect and track symptoms of COVID-19
Medical characteristics
Monitor discharged COVID-19 patients
Estimate or Predict Risk Score
Determine the care level
Determine patient priority
Determine the psychological impact of the COVID-19 outbreak
Define infectious places
Epidemic spread estimate
Contact tracing
Healthcare Decision Making
Determine the care needed
Determine medical resource needs
Determine the health staff needs
Determine the need for quarantine
Pharmaceutical
Accelerate discovery and development
Improving clinical trials
Target a specific sample of patients more effectively
Better patient's behavior insight
Improve risk management
Improve marketing insight

Figure 1. Potential application areas of big data analytics for COVID-19.

Big data analytics in the healthcare sector from different aspects and analyses the challenges that may hinder its application, then provides our future predictions in terms of using big data in the healthcare field, in addition to several recommendations.

Steps involved in this project:

Our project being based on such high dimensional real-life data also falls under the category.

Chapter 1 is dedicated to understand *Big Data* in more detailed manner including its Definition, characteristics, examples, importance, challenges, application in proteomics, special salient features, the unique computational and statistical challenges, and the subsequent remedies.

Chapter 2 focuses on thorough statistical analysis on the available Plasma Proteomics data of Covid-19 patients. It is always very important task in any statistical analysis to understand the nature of the data to be dealt with which includes its source, features etc. The *biological background* and all necessary information to clearly understand the data.

Chapter 3 provides the idea to deal with missing value using different missing value thresholds and different missing value imputation methods for the best protein intensity value.

In *Chapter 4* we discuss briefly the theory behind the initial data processing stage which includes *Box-Plot method of outlier detection* and some popular *normalization* methods.

Chapter 5 discusses our first approach towards dimension reduction using *Sure Independence Screening (SIS)* followed by *LASSO* and the subsequent theory K-fold cross validation and shrunken centroid methods.

Chapter 1: A BRIEF INTRODUCTION TO BIG DATA

1.1 What is Big Data?

Big data refers to the large, diverse sets of information that grow at ever-increasing rates. It encompasses the volume of information, the velocity or speed at which it is created and collected, and the variety or scope of the data points being covered (known as the "three v's" of big data). Big data often comes from data mining and arrives in multiple formats.

Characteristics:

- Big data is a great quantity of diverse information that arrives in increasing volumes and with ever-higher velocity.
- Big data can be structured (often numeric, easily formatted and stored) or unstructured (more free-form, less quantifiable).
- Nearly every department in a company can utilize findings from big data analysis, but handling its clutter and noise can pose problems.
- Big data can be collected from publicly shared comments on social networks and websites, voluntarily gathered from personal electronics and apps, through questionnaires, product purchases, and electronic check-ins.
- Big data is most often stored in computer databases and is analyzed using software specifically designed to handle large, complex data sets.

Features:

It is widely accepted that the characteristics of big data are defined by three major features, commonly known as the 3Vs: volume, variety, and velocity.

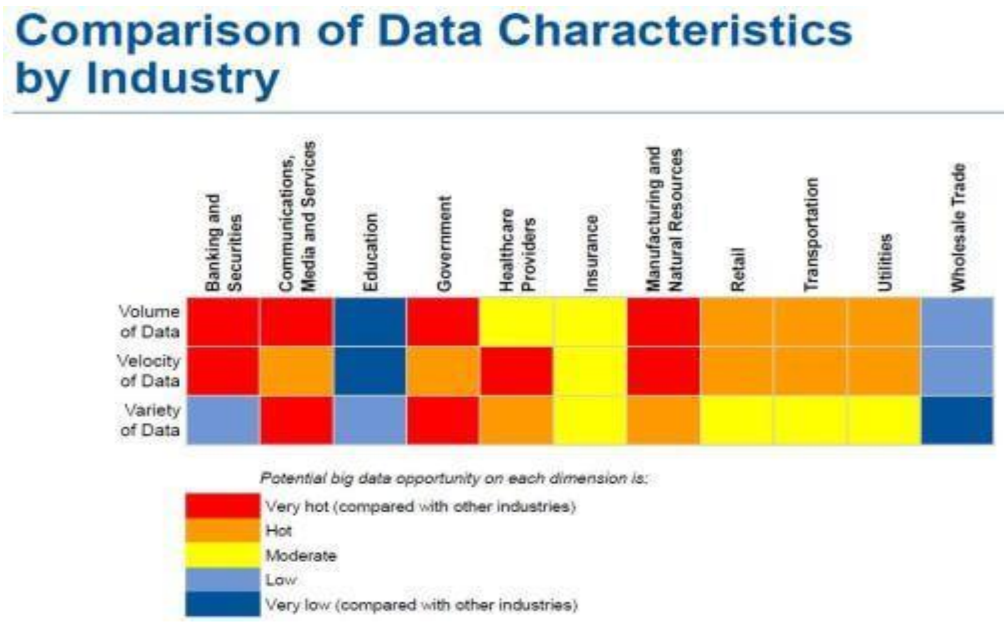
- **Volume**
  This refers to the quantity of generated and stored data. The size of the data determines the value and potential insight and whether it can actually be considered big data or not.
- **Variety**
  The second feature of big data is the *variety of data types and structures*. This helps people who analyze it to effectively use the resulting insight.
- **Velocity**
  Velocity refers to the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development.

1.2 Big Data Applications Across Industries

Here is the list of the top 10 industries using big data applications

1. Banking and Securities
2. Communications, Media and Entertainment
3. Healthcare Providers
4. Education
5. Manufacturing and Natural Resources
6. Government
7. Insurance
8. Retail and Wholesale trade
9. Transportation

10. Energy and Utilities



1.3 Importance of big data

Companies use big data in their systems to improve operations, provide better customer service, create personalized marketing campaigns and take other actions that, ultimately, can increase revenue and profits. Businesses that use it effectively hold a potential competitive advantage over those that don't because they're able to make faster and more informed business decisions.

For example, big data provides valuable insights into customers that companies can use to refine their marketing, advertising and promotions in order to increase customer engagement and conversion rates. Both historical and real-time data can be analyzed to assess the evolving preferences of consumers or corporate buyers, enabling businesses to become more responsive to customer wants and needs.

Big data is also used by medical researchers to identify disease signs and risk factors and by doctors to help diagnose illnesses and medical conditions in patients. In addition, a combination of data from electronic health records, social media sites, the web and other sources gives healthcare organizations and government agencies up-to-date information on infectious disease threats or outbreaks.

Here are some more examples of how big data is used by organizations:

- In the energy industry, big data helps oil and gas companies identify potential drilling locations and monitor pipeline operations; likewise, utilities use it to track electrical grids.
- Financial services firms use big data systems for risk management and real-time analysis of market data.
- Manufacturers and transportation companies rely on big data to manage their supply chains and optimize delivery routes.
- Other government uses include emergency response, crime prevention and smart city initiatives.

1.4 Application of Big Data in Healthcare:

Big data is very useful in the healthcare industry. Over the past decade, electronic health records (EHR) have been widely adopted in hospitals and clinics worldwide. Important clinical knowledge and a deeper understanding of patient disease patterns can be studied from such data. It will help to improve patient care and improve efficiency. There are few targeted applications of big data, such as

Healthcare Data solutions

With the help of big data, the vast amount of data can be stored systematically. Now doctors and other healthcare practitioners can make informed decisions as they have access to a wide range of data. Of course, the data generated will grow by leaps and bounds, and newer systems will be able to process it quickly and cost effectively.

Big data to fight cancer

Cancer is rapidly crippling people across the world. Big data can help to fight cancer more effectively. Healthcare providers will have enhanced ability to detect and diagnose diseases in their early stages, assigning more effectual therapies based on a patient's genetic makeup, and regulate drug doses to minimize side effects and improve effectiveness. It will also provide great support for parallelization and help in mapping the 3 billion DNA base pairs.

Monitoring patient vitals

The application of big data makes it easier for hospital staff to work more efficiently. Sensors are used besides patient beds to continuously monitor blood pressure, heartbeat and respiratory rate. Any change in pattern is quickly alerted to doctors and healthcare administrators.

Smoother Hospital Administration

Healthcare administration becomes much smoother with the help of big data. It helps to reduce the cost of care measurement, provide the best clinical support, and manage the population of at-risk patients. It also helps medical experts analyze data from diverse sources. It helps healthcare providers conclude the deviations among patients and the effects treatments have on their health.

Healthcare Intelligence

Big Data can be used for healthcare Intelligence application. This will help hospitals, payers and healthcare agencies augment their competitive advantages by developing smart business solutions.

Fraud Prevention and Detection

Big data helps to prevent a wide range of errors on the side of health administrators in the form of wrong dosage, wrong medicines, and other human errors. It will also be particularly useful to insurance companies. They can prevent a wide range of fraudulent claims of insurance.

1.4 Big data challenges

In connection with the processing capacity issues, designing a big data architecture is a common challenge for users. Big data systems must be tailored to an organization's particular needs, a DIY undertaking that requires IT and data management teams to piece together a customized set of technologies and tools. Deploying and managing big data systems also require new skills compared to the ones that database administrators and developers focused on relational software typically possess.

Both of those issues can be eased by using a managed cloud service, but IT managers need to keep a close eye on cloud usage to make sure costs don't get out of hand. Also, migrating on-premises data sets and processing workloads to the cloud is often a complex process. Other challenges in managing big data systems include making the data accessible to data scientists and analysts, especially in distributed environments that include a mix of different platforms and data stores. To help analysts find relevant data, data management and analytics teams are increasingly building data catalogues that incorporate metadata management and data lineage functions. The process of integrating sets of big data is often also complicated, particularly when data variety and velocity are factors.

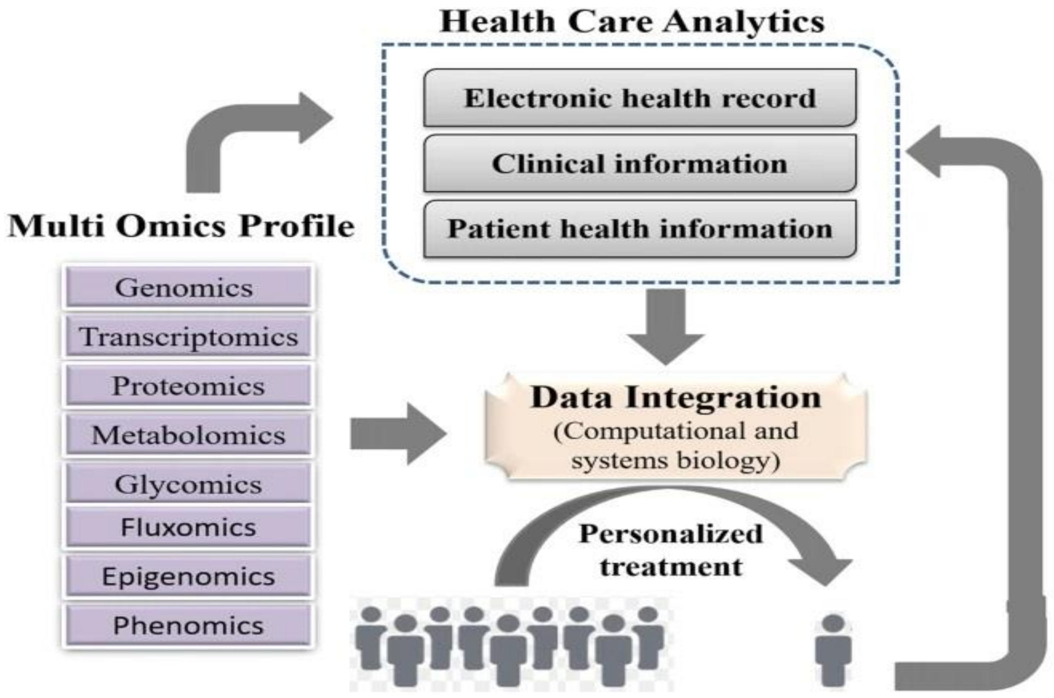
1.5 APPLICATION OF BIG DATA IN PROTEOMICS:

There are about 20,000 protein coding genes. But with modifications on multiple levels, such as non-synonymous mutation, alternative splicing and PTMs, the number of actual proteins is much larger than expected. To capture the complete information of proteome, Mass Spectrometry based technologies can be used. It is easy to generate TB of proteomics data within days. But it is difficult to analyze them in time, or even store them for a month. We are overwhelmed by digital flood of proteomics big data. To fully utilize the proteomics big data and translate them into biomedical practice, advanced computational modelling methods are urgently needed. By integrating NGS genomics data with proteomics data, we can characterize complex diseases more comprehensively and reveal the underlying proteogenomic mechanisms.

Bioinformatics research analyzes biological system variations at the molecular level. With current trends in personalized medicine, there is an increasing need to produce, store, and analyze these massive datasets in a manageable time frame. Next-generation sequencing technology enables genomic data acquisition in a short period of time. These allow biologists to generate hundreds of thousands of datasets and have shifted their primary interests from the acquisition of biological sequences to the study of biological function. The large amount of genome sequencing data now makes it possible to uncover the genetic markers of rare disorders and find associations between diseases and rare sequence variants. The emergence of publicly available genomic databases enables integrative analysis which combines information from many sources for drawing scientific conclusions. These research studies give rise to many computational methods as well as new statistical thinking and challenges.

One of the important steps in genomic data analysis is to remove systematic biases (e.g. intensity effect, batch effect, dye effect, block effect, among others). Such systematic biases are due to experimental variations, such as environmental, demographic, and other technical factors, and can be more severe when we combine different data sources. They have been shown to have substantial effects on gene/protein expression levels, and failing to take them into consideration may lead to wrong scientific conclusions.

When the data are aggregated from multiple sources, it remains an open problem on what is the best normalization practice. Even with the removed, another challenge is to conduct large-scale tests to pick important genes, proteins, or single-nucleotide polymorphism (SNP). In testing the significance of thousands of genes, classical methods of controlling the probability of making one falsely discovered gene are no longer suitable and alternative procedures have been designed to control the false discovery rates and to improve the power of the tests. These technologies, though high-throughput in measuring the expression levels of tens of thousands of genes, remain low throughput in surveying biological contexts.



1.6 Salient Features of Big Data

Big Data create unique features that are not shared by the traditional datasets because of its huge sample size and dimensionality. These unique features make traditional procedure inappropriate. The material present in this chapter are taken from the research paper.
Unique features of Big Data is as follows:

1.6.1 Heterogeneity

One of the main features of the Big Data is to make the whole sample by taking sub sample from different population. This is the main reason of heterogeneity in the Big Data. Each subpopulation might exhibit some unique features not shared by others. However, in the Big Data era, the large sample size enables us to better understand heterogeneity to explore the association between different covariates.
To better illustrate this point, we introduce the following mixture model for the population:

$$\lambda 1\ p1\ (y;\ \theta\ 1(x)) + \cdots + \lambda m\ pm\ (y;\ \theta\ m\ (x))$$

where $\lambda j \geq 0$ represents the proportion of the jth subpopulation, $p\ j\ y;\ \theta\ j\ (x)$ is the probability distribution of the response of the jth subpopulation given the covariates x with $\theta\ j\ (x)$ as the parameter vector.
In practice, . λj is very small indicating many subpopulations are rarely observed. But Big Data are characterized by large sample size n, the sample size n, λj for the jth subpopulation can be moderately large even if λj is very small. This enables us to more accurately infer about the subpopulation parameters $\theta\ j\ (\cdot)$.
Big Data also allow us to unveil weak commonality across whole population, thanks to large sample sizes. In low dimensions, standard techniques as the expectation maximization algorithm for finite mixture models can be applied, while in high dimensions, we need to carefully regularize the estimating procedure to avoid over fitting or noise accumulation and to devise good computation algorithms.

1.6.2 Noise Accumulation

Analyzing Big Data requires us to simultaneously estimate or test many parameters. In high dimension, a large number of parameters are need to be estimated in the model, as a result of which estimation error occurs. A noise accumulation effect is especially severe in high dimensions and may even dominate the true signals. It is usually handled by the sparsity assumption. Poor classification is due to the existence of many weak features that do not contribute to the reduction of classification error.
Consider the following example as an illustration:
We have two population from two different normal population and n no of sample is taken from both the population.

$$X1,\dots,Xn \sim Nd\ (\mu 1,\ Id\ )$$

$$Y1,\dots,Yn \sim Nd\ (\mu 2,\ Id\ )$$

We want to construct a classification rule which classifies a new observation $Z \in R^d$in one of the two classes.
To illustrate the impact of noise accumulation in classification, we set n = 100 and d = 1000. We set $\mu_1 = 0$ and μ_2 to be sparse, i.e. only the first 10 entries of $\mu 2$ are nonzero with value 3, and all the other entries are zero.
The figure 1 plots the first two principal components by using the first m = 2, 40, 200 features and the whole 1000 features. Now it is clear from the picture that for m=2 it has large discriminative power and as the m increases for the presence of weak features the discrimination power continues to fall and the plot become more and more noisy, that is clearly visible from the picture.

Therefore, the large the m is the more the noise accumulation. Therefore, variable selection and sparse models play a pivotal role in overcoming noise accumulation.

1.6.3 Spurious correlation:

Spurious correlation simply means that data is showing some correlation between predictors which is actually not present in the population. Spurious correlation may cause false scientific discoveries and wrong statistical inferences. High dimensionality brings spurious correlation, referring to the fact that many uncorrelated random variables may have high sample correlations in high dimensions.
Consider the problem of estimating the coefficient vector β of a linear model

$$Y = X\beta+\in$$

$$Var(\in) = \sigma^2 I_d$$

where $y \in R_n$ represents the response vector, $X = [X_1,..., X_n]^T \in R_n \times d$ represents the design matrix $\in$ belongs to R_n represents an independent random noise vector & I_d is the d × d identity matrix.
In high dimensions variable selection is challenging due to the presence of spurious correlation. When the dimensionality is high, the important variables can be highly correlated with several spurious variables which are scientifically unrelated.
We consider the following example to illustrate this phenomenon of spurious correlation:
Let, $x_1,..., x_n$ be n independent observations of a d-dimensional Gaussian random vector $X = (X_1,..., X_d)^T \sim N_d (0, I_d)$. We repeatedly simulate the data with n=60 and d=800 and 6400 for 1000 times.
Figure 2a shows the empirical distribution of the maximum absolute sample correlation coefficient between the first variable with the remaining ones defined as

$$r_{hat}= \max_{j>=2} |Corr (X_1, X_j) |$$

where $Corr (X_1, X_j)$ is the sample correlation between the variables X_1 and X_j. We see that the maximum absolute sample correlation becomes higher as dimensionality increases. we can compute the maximum absolute multiple correlation between X1 and linear combinations of several irrelevant spurious variables:

$$R_{hat}= \max |S|=4 \max \{\beta j\} 4 j=1 | Corr (X1,sumj\in S \beta j Xj)|$$

a, Fig. 2 b plots the empirical distribution of the maximum absolute sample correlation coefficient between X_1 and sum $\beta_j X_j$, where S is any size four subset of {2, ..., d} and β_j is the least-squares regression coefficient of X_j when regressing X_1 on $\{X_j \}j \in S$.
If X_1 represents the expression level of a gene that is responsible for a disease, we cannot distinguish it from the other four genes in S that have a similar predictive power although they are scientifically irrelevant.
Besides variable selection, spurious correlation may also lead to wrong statistical inference.

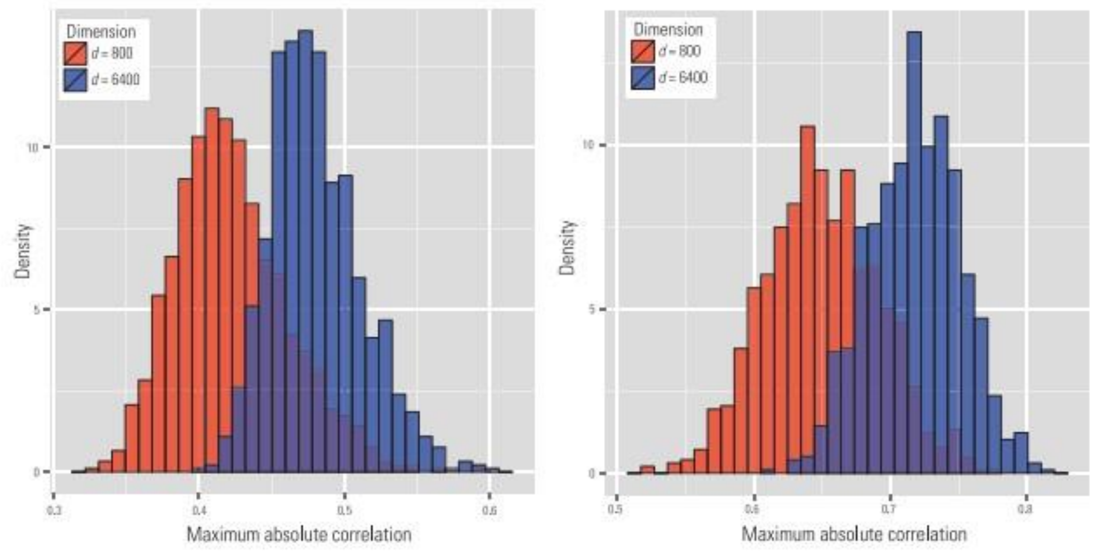


Figure 1. Illustration of spurious correlation. (a) Distribution of the maximum absolute sample correlation coefficients between X_1 and $\{X_j\}_{j\neq 1}$.
(b) Distribution of the maximum absolute sample correlation coefficients between X_1 and the closest linear projections of any four members of $\{X_j\}_{j\neq 1}$ to X_1

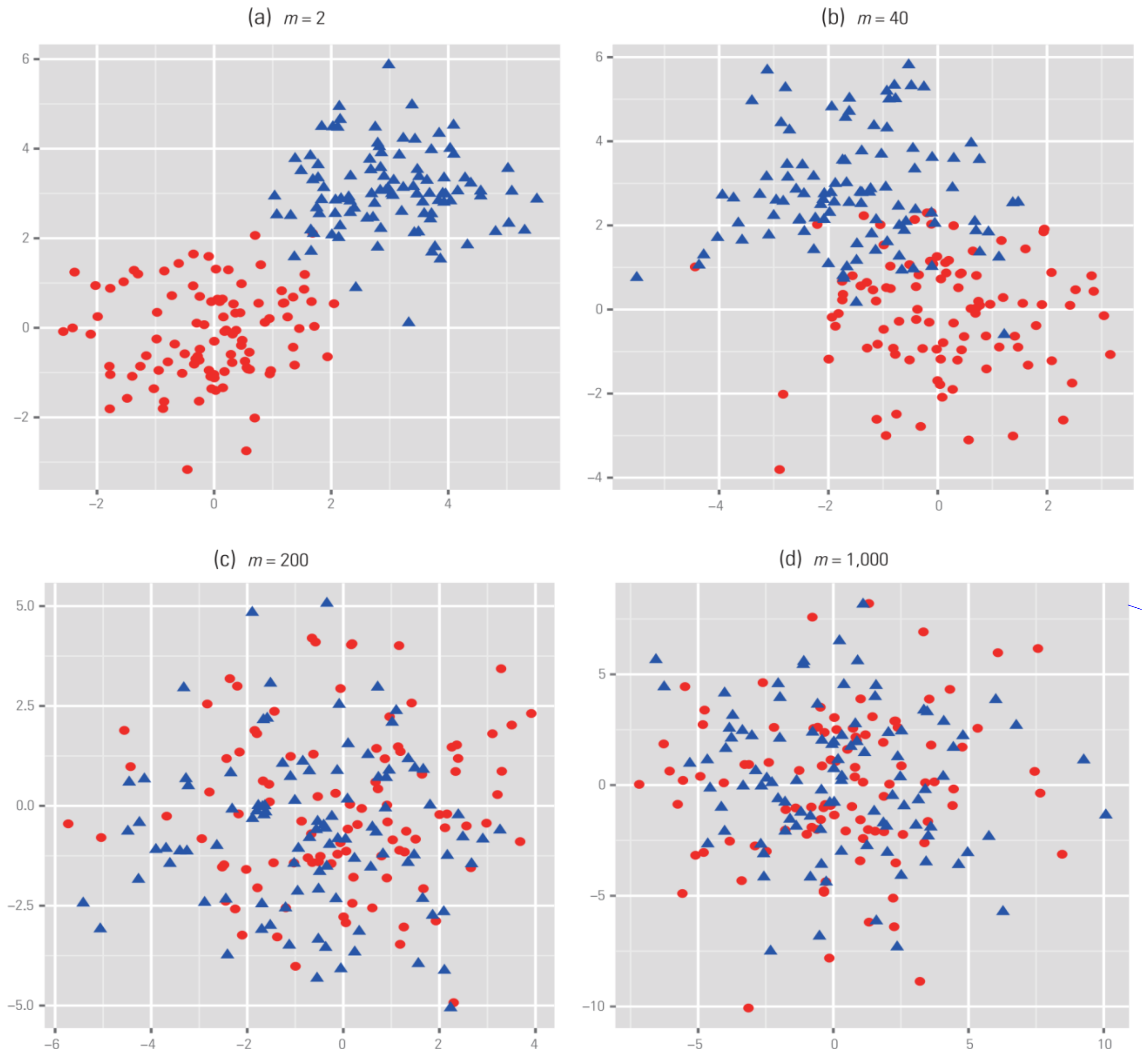


Figure 2. Scatter plots of projections of the observed data (n = 100 from each class) onto the first two principal components of the best m-dimensional selected feature space. A projected data with the filled circle indicates the first class and the filled triangle indicates the second class.

1.6.4 <u>Incidental Endogeneity</u>

In a regression setting

$$Y = \sum \beta_j X_j + \varepsilon$$

The term 'endogeneity' means that some predictors $\{X_j\}$ correlate with the residual noise ε. The conventional sparse model assumes

$$Y = \sum \beta_j X_j + \varepsilon$$

and $E(\varepsilon X_j) = 0$ for $j = 1, \ldots, d$, with a small set $S = \{j: \beta_j = 0\}$

This assumption is easy to be violated in high dimensions as some of variables $\{X_j\}$ are incidentally correlated with ∈, making most available statistical procedure invalid.

Endogeneity generally occurs due to selection biases, measurement errors and omitted variables, especially in the analysis of Big Data.

In order to illustrate the existence of endogeneity, an L1-penalized least-squares regression (Lasso) on the data is fitted, and the penalty is automatically selected via 10-fold cross validation.

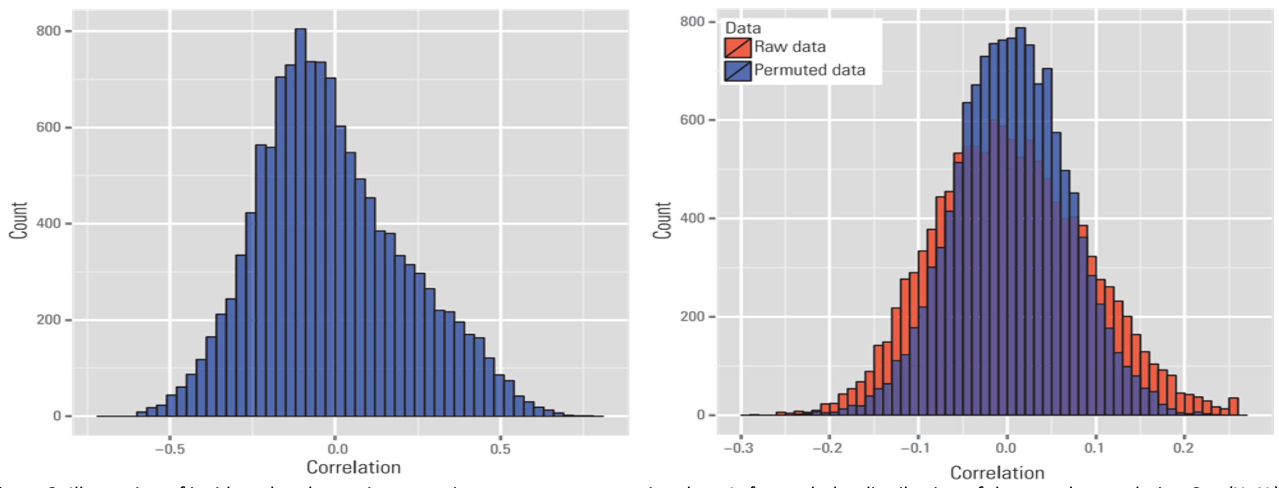


Figure 3. Illustration of incidental endogeneity on a microarray gene expression data. Left panel: the distribution of the sample correlation Corr(X_j, Y) (j = 1, . . . , 12 718). Right panel: the distribution of the sample correlation Corr(X_j, ε). Here ε represents the residual noise after the Lasso fit. We provide the distributions of the sample correlations using both the raw data and permuted data

Chapter 2: UNDERSTANDING THE DATA

2.1 DESCRIPTION OF THE DATA AND OBJECTIVE

I've been provided two Micro-array Plasma Proteomics datasets of Covid-19 patients. The first dataset contains 40 patient ids along with 1394 proteins and the second dataset contains 37 patient ids along with 1310 proteins. The disease considered is 'Covid-19'. For the first dataset, out of the 40 patients 16 belongs to 'Severe' category, 24 belongs to 'non-severe' category. For the second dataset, out of the 37 patients 14 belongs to 'Severe' category & 23 belongs to 'non-severe' category. Corresponding to each patient there are protein intensity readings of 1394 proteins (for the first dataset) and 1310 proteins (for the second dataset), provided as numerical figures.

Our main objective of this project with those real-life datasets is to identify those important proteins or biomarkers which are responsible for severe covid-19 patients.

Our Dataset (Small Part)

Protein IDs	NSOP_13	NSOP_15	OP1	OP10	OP19	OP2	OP20	OP21	OP3	OP4
	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever	Non Sever
P62851	0	0	0	0	0	0	0	0	0	0
P22392	0	0	0	0	0	0	0	0	0	0
O60869	0	0	0	0	0	0	420140	0	0	0
Q9H0K6	5865400	3445900	9311000	6671700	3153200	0	2271500	2116400	7179200	11305000
Q14974	0	0	3374700	0	0	0	0	0	2807200	0
Q96RW7	0	0	0	26332000	0	0	0	0	0	0
P47897	0	0	0	0	0	0	0	0	0	0
O95399	0	1487200	23681000	0	0	0	0	0	6819200	0
Q86X29	0	0	0	0	0	0	0	0	0	0
Q5SYC1	0	0	0	0	0	0	0	0	0	0
Q86Y82	0	0	0	0	0	0	0	0	0	0
Q9Y3U8	0	0	1785400	0	0	0	0	0	4506900	0
Q9Y623	0	0	0	0	0	0	0	0	0	0
Q01105	0	0	0	0	0	0	0	0	0	0
Q9UNZ2	0	0	0	0	0	0	0	0	0	0
O60268	0	0	0	0	0	0	0	0	0	0
Q13200	0	0	0	0	0	0	0	0	0	0
P49321	0	0	4032800	0	0	6158400	0	0	5075000	0

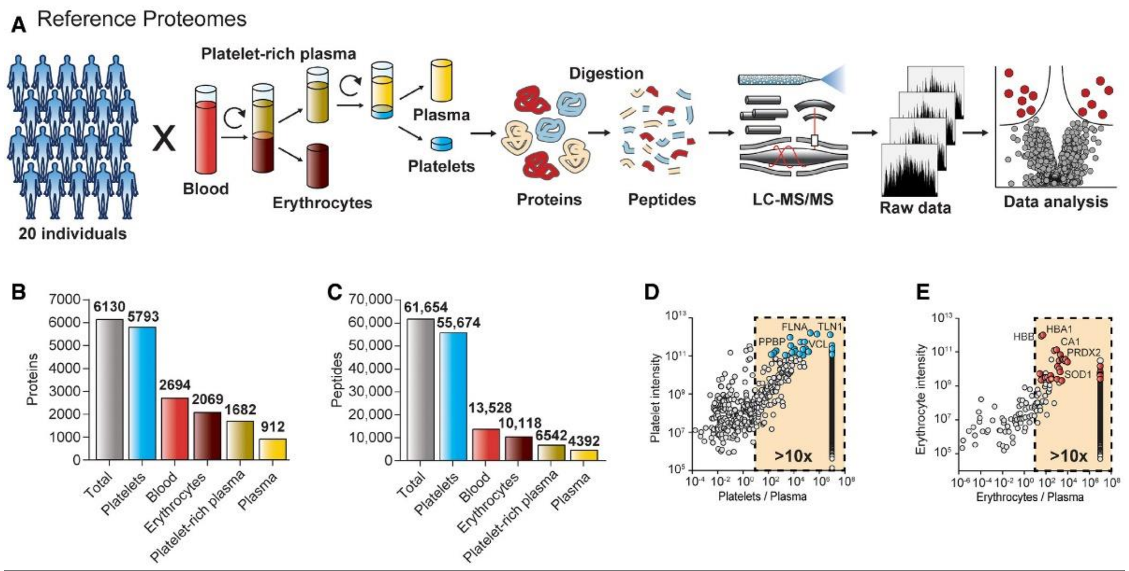
2.2 ABOUT PLASMA PROTEOMICS:

Plasma proteomics is a term which describes the high-throughput analysis of plasma biomarkers using very powerful, sensitive and specific instruments. The proteomic analysis of human blood and blood-derived products (e.g., plasma) offers an attractive avenue to translate research progress from the laboratory into the clinic. However, due to its unique protein composition, performing proteomics assays with plasma is challenging. Plasma proteomics has regained interest due to recent technological advances, but challenges imposed by both complications inherent to studying human biology (e.g., interindividual variability) and analysis of biospecimens (e.g., sample variability), as well as technological limitations remain. As part of the Human Proteome Project (HPP), the Human Plasma Proteome Project (HPPP) brings together key aspects of the plasma proteomics pipeline. Here, we provide considerations and recommendations concerning study design, plasma collection, quality metrics, plasma processing workflows, mass spectrometry (MS) data acquisition, data processing, and bioinformatic analysis. With exciting opportunities in studying human health and disease though this plasma proteomics pipeline, a more informed analysis of human plasma will accelerate interest while enhancing possibilities for the incorporation of proteomics-scaled assays into clinical practice.

Plasma proteomics may help significantly to predict the onset of and treat, metabolic and other chronic illnesses in a personalized manner, and to identify adverse effects of drugs. The evolution of novel biomarkers for disease conditions requires the identification of potential biomarkers by case-control studies, or currently by longitudinal studies. This is then verified by targeted analysis of a greater number of plasma samples. Finally, the candidate markers are tested in large cohorts of subjects to evaluate their sensitivity, specificity, and usefulness by clinical validation.

2.3 Methods Used in Plasma Proteomics

Serum albumin, globulins, and complement factors make up over 99% of the total of plasma proteome, making it among the most complicated and diverse sub-proteomes in the human organism. Novel methods of sample preparation have also evolved to help detect even those proteins which are present at extremely low concentrations in plasma. These include affinity purification to obtain a specific protein and immunochemistry to eliminate proteins in high concentrations. Moreover, high-speed MS tools and the launch of LC platforms for durable nanoscale flows have enabled robust workflows in proteomics to deepen the analysis.



Chapter 3: Missing value detection

3.1 Missing value pattern:

The absence of values is a cause of concern for real-life datasets. When collecting observations about variable, missing values can occur due to reasons as diverse as –

- an error in machinery/equipment
- error on part of the researcher
- unavailable respondents
- accidental deletion of observations
- forgetfulness on part of the respondents
- error in accounting, etc.

Types of missing values can generally be classified as:

- **Missing Completely at Random (MCAR)**

  This happens when the missing values have no hidden dependency on any other variable or any characteristic of observations. If a doctor forgets to record the age of every tenth patient entering an ICU, the presence of missing value would not depend on the characteristic of the patients.

- **Missing at Random (MAR*)***

  In this case, the probability of missing value depends on the characteristics of observable data. In survey data, high-income respondents are less likely to inform the researcher about the number of properties owned. The missing value for the variable number of properties owned will depend on the income variable.

3.2 Missing value ratio and percentage:

$$Missing\ value\ percentage = \frac{Number\ of\ missing\ values}{Total\ number\ of\ observations} \times 100$$

Once we have the missing value ratio and percentage on the variables. Now we can decide a significant threshold, we can use this threshold and drop all the variables which have a missing value percentage more than this threshold. For our dataset, we have used 30% missing value threshold, i.e., if a particular protein id is missing more than 30% of the total patients counts then we can remove those protein ids since it doesn't have any biological significance effect on severe or non-severe patients.

We've separated the whole dataset into severe patients' dataset and non-severe patients' dataset. Then, we've started calculating the missing value percentage with respect to each protein id for severe and non-severe patient id separately. We've also performed the same task for the whole dataset to make the clear idea and to identify the biological significant proteins for Covid-19 patients.

The table below shows the missing value count and percentage with respect to each patient id and each protein id:

TABLE FOR MISSING VALUE COUNT AND PERCENTAGE FOR EACH PATIENT & PROTEIN WISE

Patient Ids	Null_counts	percentage
NSOP_13	1033	78.97553517
NSOP_15	885	67.66055046
OP1	593	45.33639144
OP10	1026	78.44036697
OP19	1106	84.55657492
OP2	1029	78.66972477
OP20	1103	84.32721713
OP21	1099	84.02140673
OP3	591	45.18348624
OP4	932	71.25382263
OP5	1044	79.81651376
OP6	1009	77.14067278
OP7	1040	79.51070336
OP8	1035	79.12844037
OP9	1019	77.90519878
SOP28	718	54.89296636
SOP29	1001	76.52905199
SOP30	872	66.66666667
SOP31	934	71.40672783
SOP32	744	56.88073394
SOP35	979	74.8470948
SOP36	917	70.10703364
SOP38	1020	77.98165138
SOP62	1045	79.89296636
SOP67	997	76.22324159
SOP73	1027	78.51681957
SOP79	1063	81.26911315
SOP80	1034	79.05198777
SOP82	454	34.70948012
SOP83	1012	77.37003058
SOP84	462	35.32110092
SOP85	931	71.17737003
OP34	1053	80.50458716
OP37	1049	80.19877676
OP40	1054	80.58103976
OP42	1007	76.98776758
OP43	895	68.42507645

No.	Proteins	S	% (S)	N-S	%(N-S)
1	P62851	12	85.714286	23	95.833333
2	P22392	13	92.857143	22	91.666667
3	O60869	14	100	21	87.5
4	Q9H0K6	5	35.714286	6	25
5	Q14974	11	78.571429	18	75
6	Q96RW7	11	78.571429	20	83.333333
7	P47897	13	92.857143	21	87.5
8	O95399	12	85.714286	19	79.166667
9	Q86X29	14	100	23	95.833333
10	Q5SYC1	14	100	23	95.833333
11	Q86Y82	14	100	23	95.833333
12	Q9Y3U8	14	100	21	87.5
13	Q9Y623	14	100	23	95.833333
14	Q01105	14	100	23	95.833333
15	Q9UNZ2	14	100	23	95.833333
16	O60268	12	85.714286	23	95.833333
17	Q13200	12	85.714286	23	95.833333
18	P49321	13	92.857143	13	54.166667
19	Q96GW7	12	85.714286	23	95.833333
20	Q9NYW8	4	28.571429	2	8.3333333
21	P25789	10	71.428571	21	87.5
22	O60635	13	92.857143	21	87.5
23	P07203	14	100	23	95.833333
24	P0C711	14	100	15	62.5
25	O75569	14	100	23	95.833333
26	Q01523	14	100	21	87.5
27	P82909	14	100	20	83.333333
28	P59594	4	28.571429	14	58.333333
29	P49221	14	100	23	95.833333
30	Q14165	12	85.714286	22	91.666667
31	P14550	13	92.857143	21	87.5
32	P49588	14	100	23	95.833333
33	Q5JPH6	10	71.428571	16	66.666667
34	P0DPH7	14	100	23	95.833333
35	Q9H0B8	14	100	23	95.833333
36	Q8WXA2	14	100	23	95.833333
37	Q9UBC2	14	100	23	95.833333
38	O15540	14	100	21	87.5
39	Q969T9	14	100	23	95.833333
40	Q9ULE3	11	78.571429	11	45.833333

* S: Missing value count for Severe patient & N-S: Missing value count for non-severe patients

Using 30% missing value threshold, we got 203 protein ids {Q9NYW8, P59594, A0A0C4DH42, O75150, Q16762, Q8N1C3, P00742, ……, Q71U36} out of 1310 protein ids which are significant for the Severe patients, 215 protein ids {Q9H0K6, Q9NYW8, Q569K6, O75150, Q6IC98, Q8N1C3, P00742, P55884, …., Q03591, Q14624, Q71U36} out of 1310 protein ids which are significant for the Non-Severe patients.

3.3 K-Nearest Neighbour(KNN) Algorithm for Missing value imputation

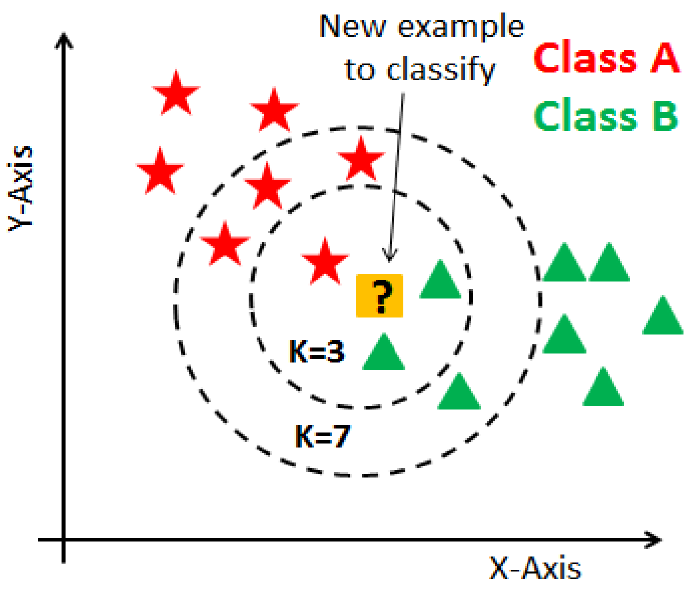
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

3.3.1 How does K-NN work?

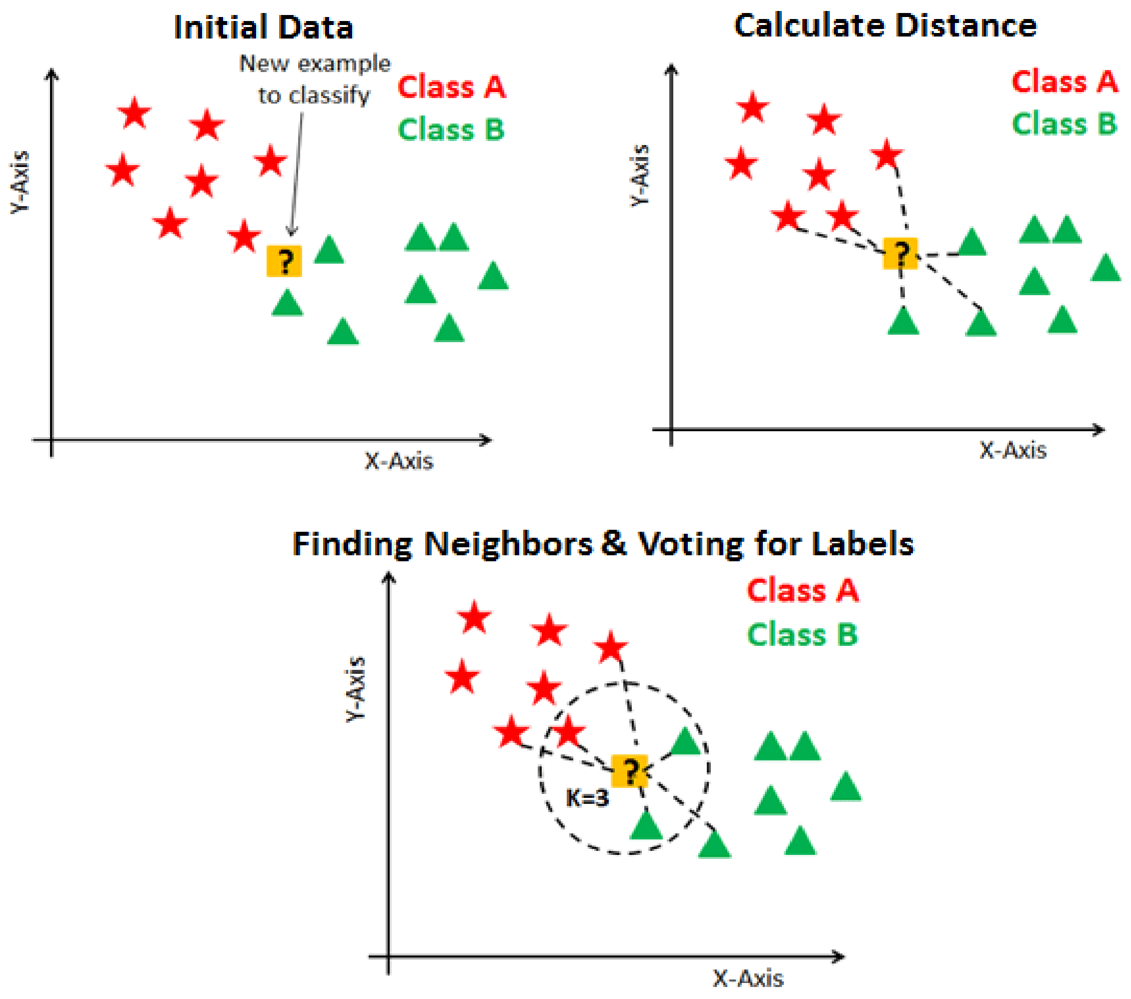
The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbours
- **Step-2:** Calculate the Euclidean distance of **K number of neighbours**
- **Step-3:** Take the K nearest neighbours as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbours, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



- Firstly, we will choose the number of neighbours, so we will choose the k=5.
- Next, we will calculate the **Euclidean distance** between the data points.
- By calculating the Euclidean distance, we got the nearest neighbours, as three nearest neighbours in Class A and two nearest neighbours in category B. Consider the below image:



- As we can see the 3 nearest neighbours are from class B, hence this new data point must belong to class B.

We've performed this method for the remaining missing values imputation for the whole dataset, severe dataset and non-severe dataset.

3.4 <u>Missing value imputation using mean and median:</u>

Mean or median imputation consists of replacing all occurrences of missing values (NA) within a variable with the mean or median of that variable. Missing values can be imputed using the mean and median value of each column in which the missing value is located. This method provides the constant value (mean or median) for all missing values of a particular feature.

<u>Limitations of mean or median imputation:</u>

- It distorts the original variable distribution and variance.
- It distorts the covariance with the remaining dataset variables.
- The higher the percentage of missing values, the higher the distortions.

Since, our dataset is a real-life dataset related to health sector, so we've performed this method but didn't use those datasets for our further statistical analysis. As, KNN method can do better approximation for large number of missing values and it takes the different neighbourhood of present protein intensity values to replace the missing values with relevant protein intensity value

3.5 <u>Application on our Dataset</u>

❖ **KNN imputed dataset obtained for severe patient dataset (small part)**

Protein ID	SOP62	SOP67	SOP73	SOP79	SOP80	SOP82	SOP83	SOP84	SOP85	OP34
Q9NYW8	10803000	130670000	94740600	101100000	94740600	100530000	81258000	87820000	166070000	9823600
P59594	9706060	11676000	10370000	8050220	4516900	7222400	12005000	12170000	9797400	8050220
A0A0C4DI	10829000	4321400	2847100	2299200	1006100	2840200	1906800	2048000	1527800	2299200
O75150	120620000	68555000	77024000	42202000	57151000	107370000	72950000	302130000	71339400	77244000
Q16762	44581000	8883600	41532000	68825000	52059400	18126000	50697000	64030000	49731000	13911000
Q8N1C3	98207000	24706000	17762000	19416000	2855600	8695500	21730000	17889000	10468000	3626700
P00742	12253000	15917000	7864660	5928380	8761900	8143000	6756800	7231900	9923500	4176000
P55884	5085980	7765300	7166840	5245500	4595000	4791100	3422600	4855900	7166840	13144000
A0A0B4J1	30453000	28197000	11691000	11608000	17099000	6423800	14230000	45569000	26956000	4863200
Q96SN8	38621000	47921000	14860000	27294000	25219000	4420500	18002000	9480900	13760000	10436000
A0A087W	18138000	27369000	11856000	8767200	9005700	10739000	8766500	13547240	12006080	3881300
P51659	5199700	3206900	2824120	2377100	4009940	4241400	2299600	8306200	4034720	2588880
P0DOX6	44177000	25795000	21172000	3230300	4984600	15571000	35172000	8486700	18623180	5145500
Q6PL18	36259000	155120000	6415100	9833400	5775100	10796600	10028000	11691000	7593320	4682200

❖ **KNN imputed dataset obtained for severe patient dataset (small part)**

Protein IDs	NSOP_13	NSOP_15	OP1	OP10	OP19	OP2	OP20	OP21	OP3	OP4
Q9H0K6	5865400	3445900	9311000	6671700	3153200	7087160	2271500	2116400	7179200	11305000
Q9NYW8	75304000	45686000	73187800	3.9E+08	42895000	1.13E+08	17927000	10829000	2.5E+08	1.93E+08
Q569K6	33081000	43359000	54723000	1.75E+08	1.95E+08	2.85E+08	1.95E+08	1.95E+08	2.48E+08	2.6E+08
O75150	21492000	83129000	63012000	1.6E+08	7036500	2.7E+08	4081800	3913500	2.05E+08	1.72E+08
Q6IC98	10242000	16112000	6834300	2.37E+08	10049000	6.21E+08	6952700	4095400	2.2E+08	2.58E+08
Q8N1C3	9330800	8708000	32652000	8868300	26132160	16550000	26132160	26132160	10357000	20433000
P00742	5955100	4984700	15534000	6680000	5496440	9969500	1273700	5496440	7829380	8492500
P55884	710810	4246882	7248920	2388700	3112108	28068380	3112108	480130	7375360	33104000
A0A0B4J1U	9507300	15473000	15772000	25192000	32292260	25910000	32292260	32292260	24337000	11096000
Q96SN8	15093000	23456000	46442000	12026000	1278400	21329000	883910	932290	6056900	4409500
A0A087WS	4493800	9397700	30245000	17152000	9531160	26180000	9531160	9531160	27502000	10586000
P51659	1852100	5526620	3082600	6191900	3178900	2905500	3160500	2408200	1718700	3972300
P0DOX6	5740800	13580000	6900500	23684200	13832360	26930000	13832360	13832360	31667000	23911000
Q6PL18	6568000	10445000	20526000	19573060	7834260	30503000	7834260	7834260	21474000	9531400

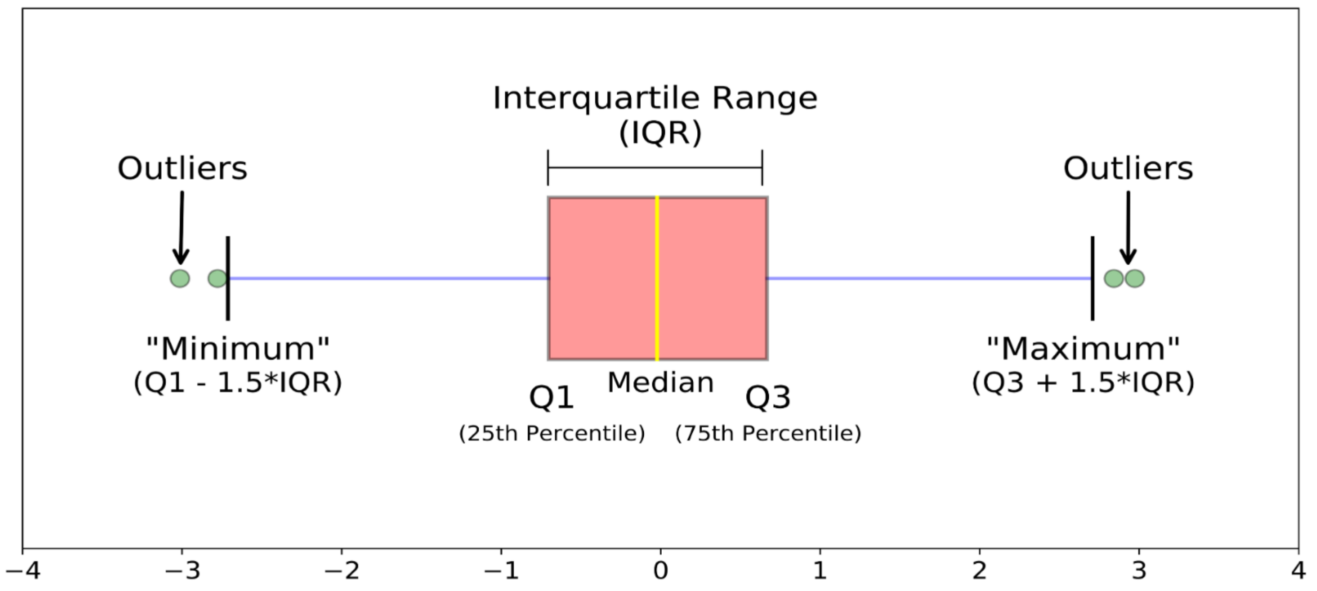
Chapter 4 OUTLIER DETECTION AND NORMALIZATION

4.1 OUTLIER DETECTION

An *outlier* is an observation that lies an abnormal distance from other values in a random sample from a population. In a sense, this definition leaves it up to the analyst (or a consensus process) to decide what will be considered abnormal. Before abnormal observations can be singled out, it is necessary to characterize normal observations.

4.1.1 METHOD OF BOX-PLOT:

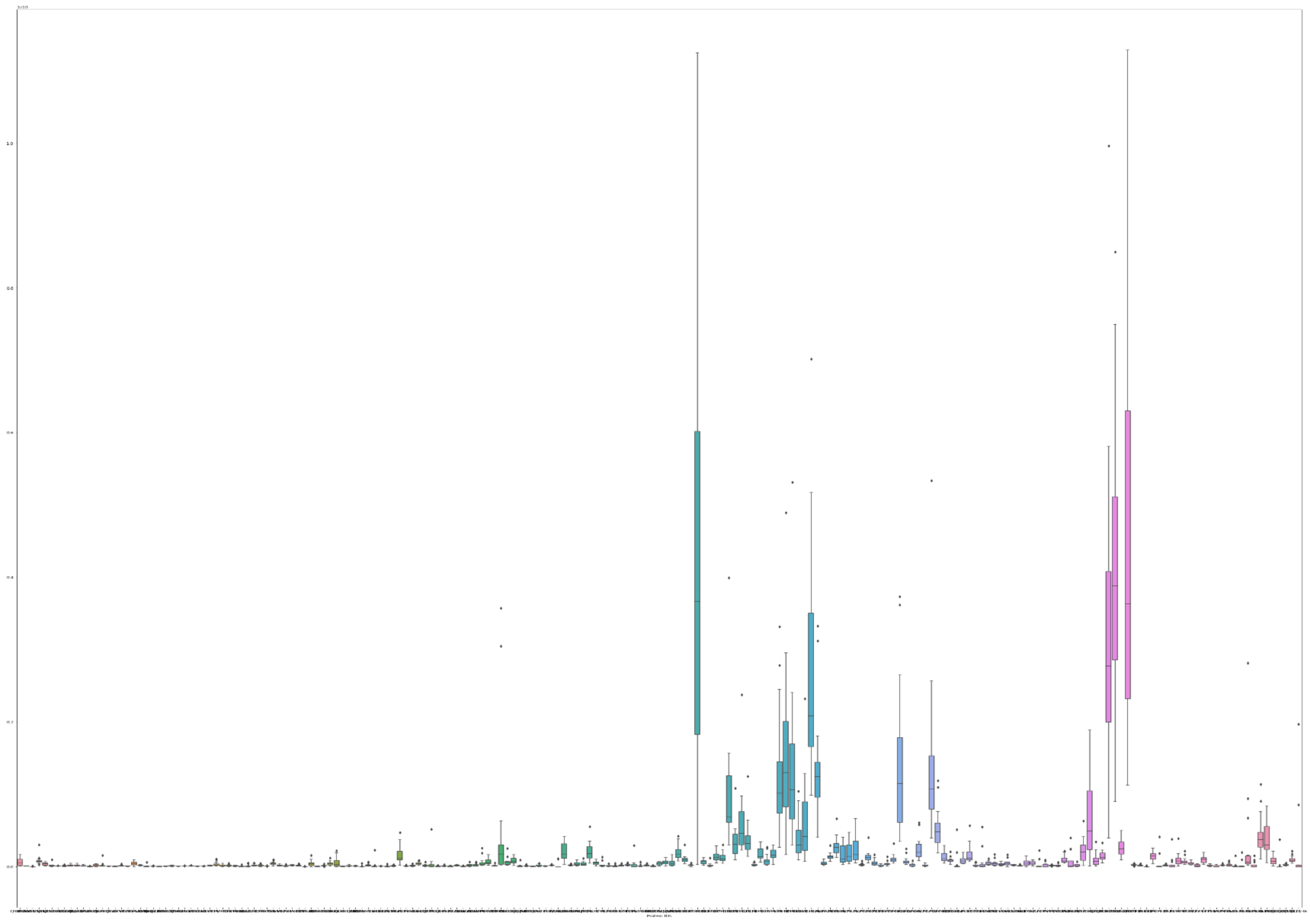
The *box plot* is a useful graphical display for describing the behaviour of the data in the middle as well as at the ends of the distributions. The box plot uses the median and the lower and upper quartiles (defined as the 25th and 75th percentiles). If the lower quartile is Q1 and the upper quartile is Q3, then the difference (Q3 - Q1) is called the *Inter-quartile Range* or *IQ.*



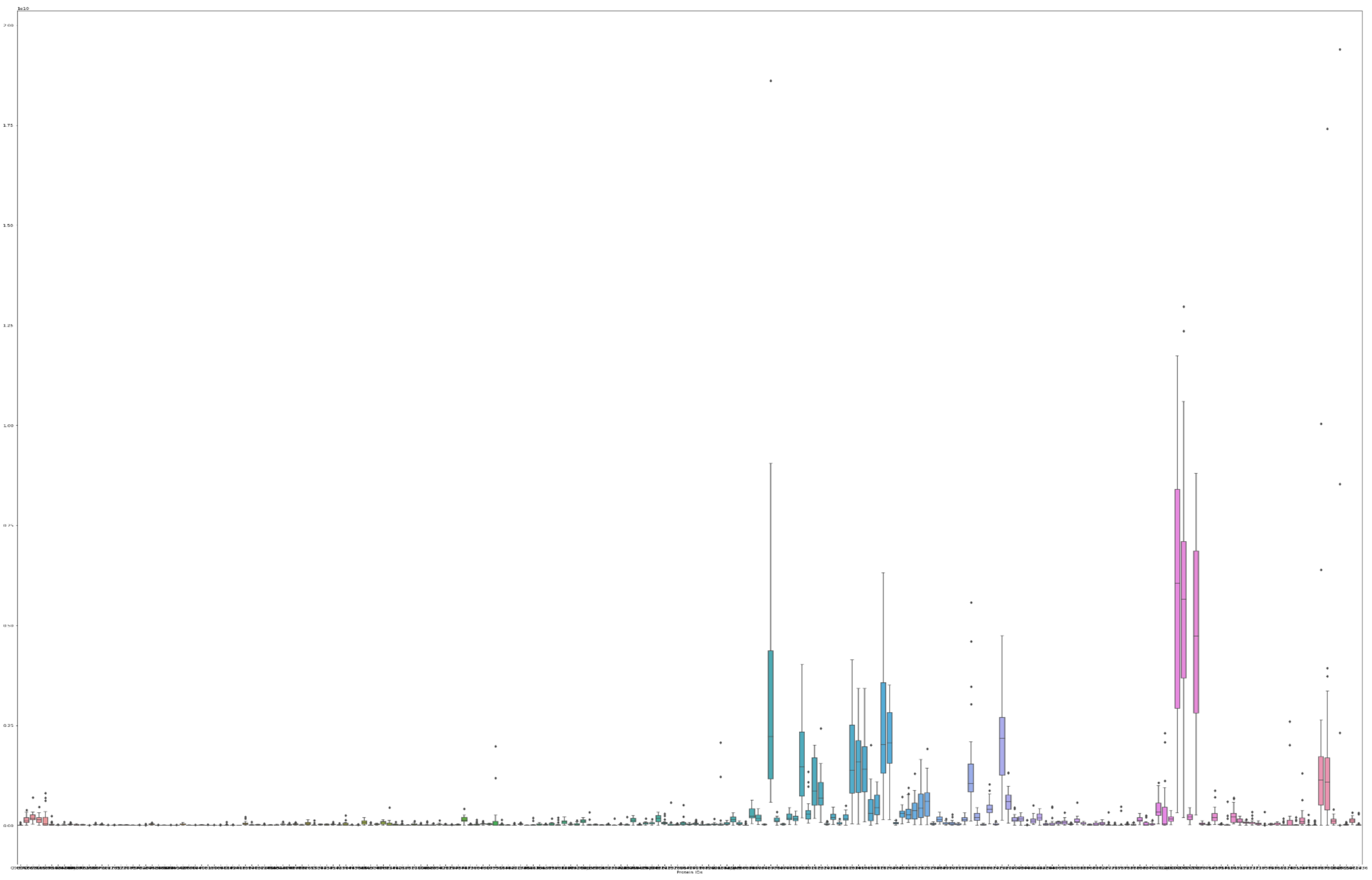
A box plot is constructed by drawing a box between the upper and lower quartiles with a solid line drawn across the box to locate the median.

Application on our dataset:

❖ **For severe patient dataset(With Outlier points):**



❖ **For non-severe patient dataset(with outlier points):**



We can see the above box plots for severe and non-severe patients with respect to each protein id and can get the idea that the range of some protein intensity value is very high and some protein intensity value is too low. There is also outlier present on some of the protein ids. So, we've performed different normalization methods to make the range of protein intensity values normal or stable.

4.1.2 <u>Outlier treatmet using IQR:</u>

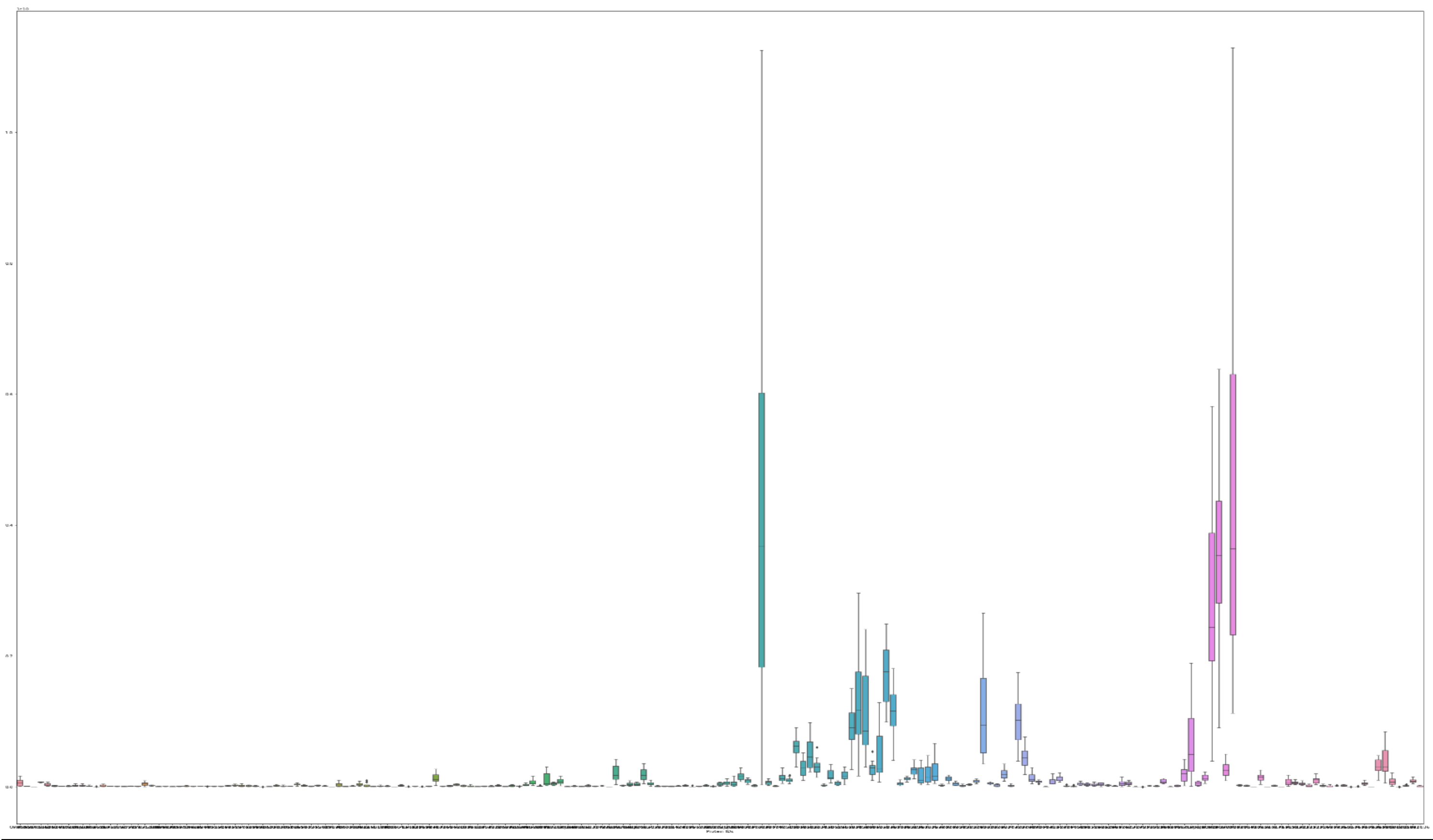
The following quantities (called *fences*) are needed for identifying extreme values in the tails of the distribution:

1. lower inner fence: Q1 - 1.5*IQ
2. upper inner fence: Q3 + 1.5*IQ
3. lower outer fence: Q1 - 3*IQ
4. upper outer fence: Q3 + 3*IQ

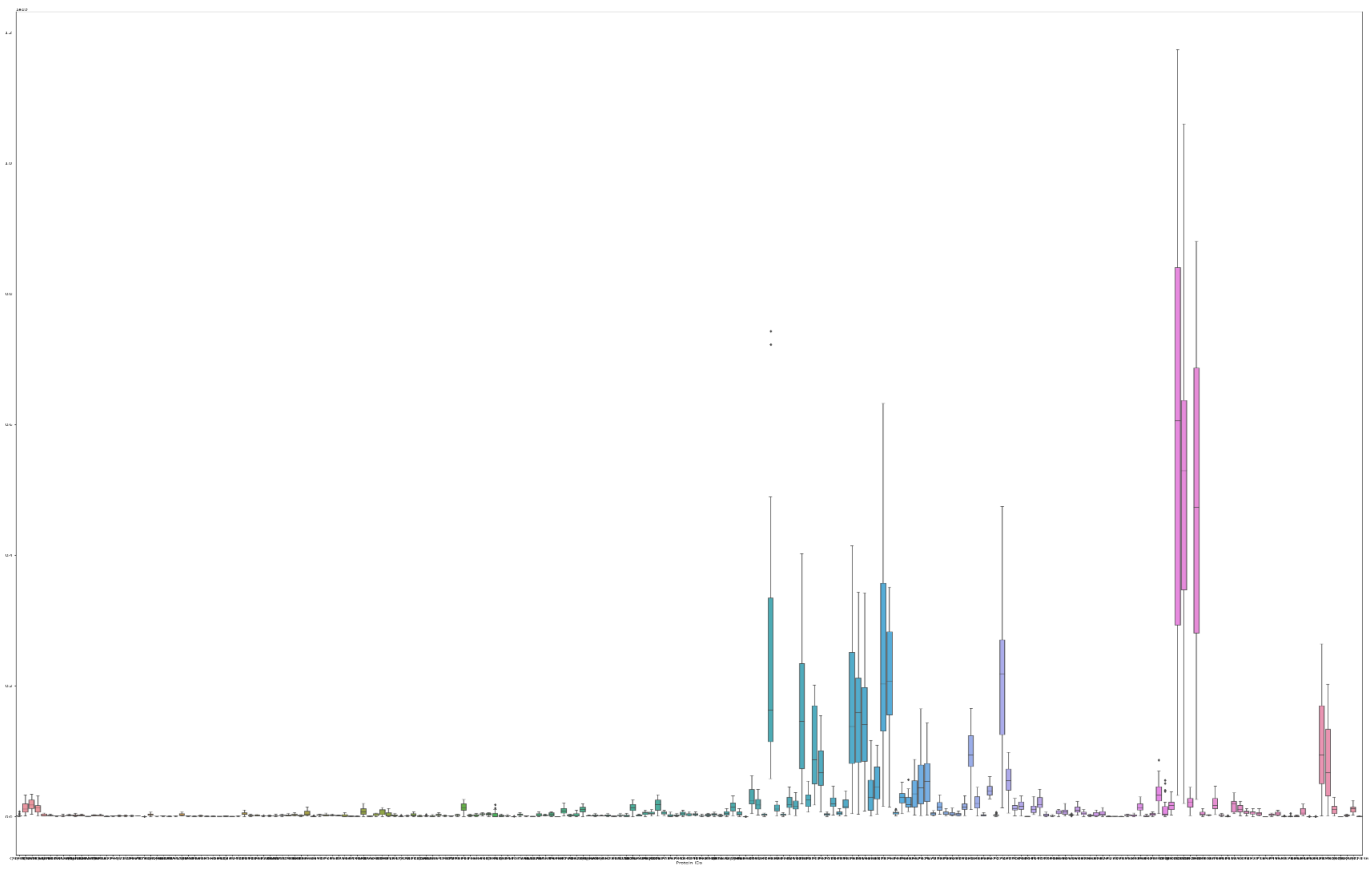
A point beyond an inner fence on either side is considered a *mild outlier*. We have to remove those points which are less than lower inner fence or greater than upper inner fence.

<u>Application on our dataset:</u>

❖ **For severe patient dataset (After removing Outlier points using IQR):**



❖ **For non-severe patient dataset (After removing Outlier points using IQR):**



4.1.3 Conclusion:

In machine learning projects, during model building it is important to remove those outliers because presence of those outliers can mislead the model. Presence of outliers may change the mean and standard deviation of the whole dataset that can badly affect the performance of the model. Outliers also increases the variance error and reduces the power of statistical test. Outliers' detection and removal is the important task in data cleaning process. These unusual data may change the standard deviation and mean of the dataset causing poor performance of the machine learning model.

Hence, outliers must be removed from the dataset for better performance of model but it is not always an easy task.

4.2 NORMALIZATION:

Normalization is a very important step in making the raw data (after removal of outliers and applying background correction) ready to apply the dimensionality reduction techniques. General data collected using measuring instruments, have errors attached to it due to manual handling of the slides and machines and also due to day-to-day variation. These errors must be taken into account before starting the analysis. Thus, *normalization* is required to diminish the day to-day or experimental errors.

We've considered three approaches for normalization of the given raw data:

1. **Min-Max Scaling:**

Here is the formula of this normalization method:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, X_{max} and X_{min} are the maximum and the minimum values of the feature respectively.

2. **Standardization:**

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

Here's the formula for standardization:

$$X' = \frac{X - \mu}{\sigma}, where\ \mu = \sum_{i=0}^{n}(X_i)\ \&\ \sigma = \sqrt{\sum_{i=1}^{n}\left(\left(X_i - \overline{X}\right)^2\right)}$$

μ is the mean of the feature values and σ is the standard deviation of the feature values.

3. **Robust Scaler:**

Robust Scaler algorithms scale features that are robust to outliers. The method it follows is almost similar to the Min-Max Scaler but it uses the interquartile range (rather than the min-max used in Min-Max Scaler). The median and scales of the data are removed by this scaling algorithm according to the quantile range.

It follows the following formula:

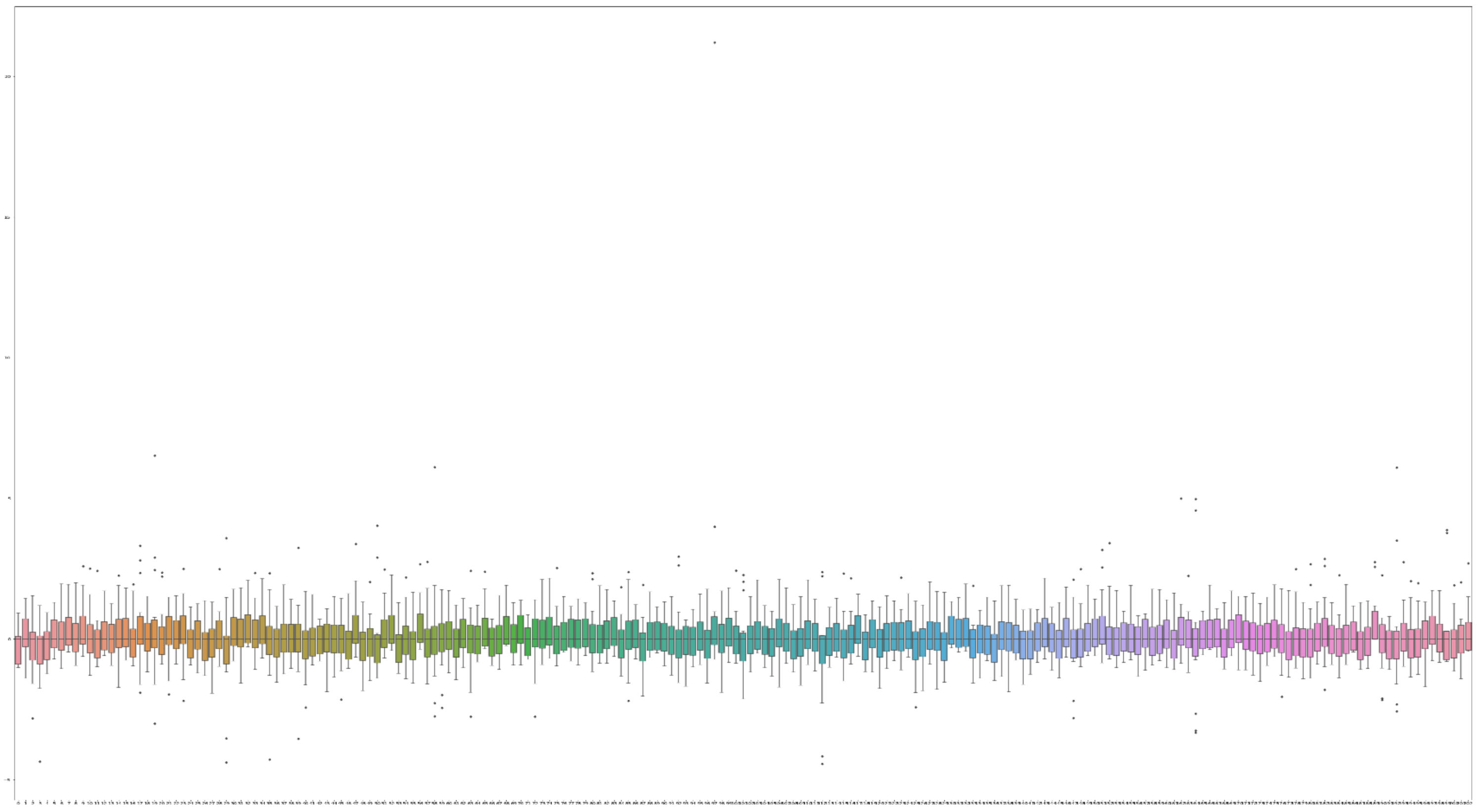
$$X' = \frac{X - \mathrm{median}(X_i)}{Q_3(X) - Q_1(X)}$$

Where, Q1 is the 1st quartile, and Q3 is the third quartile & $\mathrm{median}(X_i)$ is the median of the features.

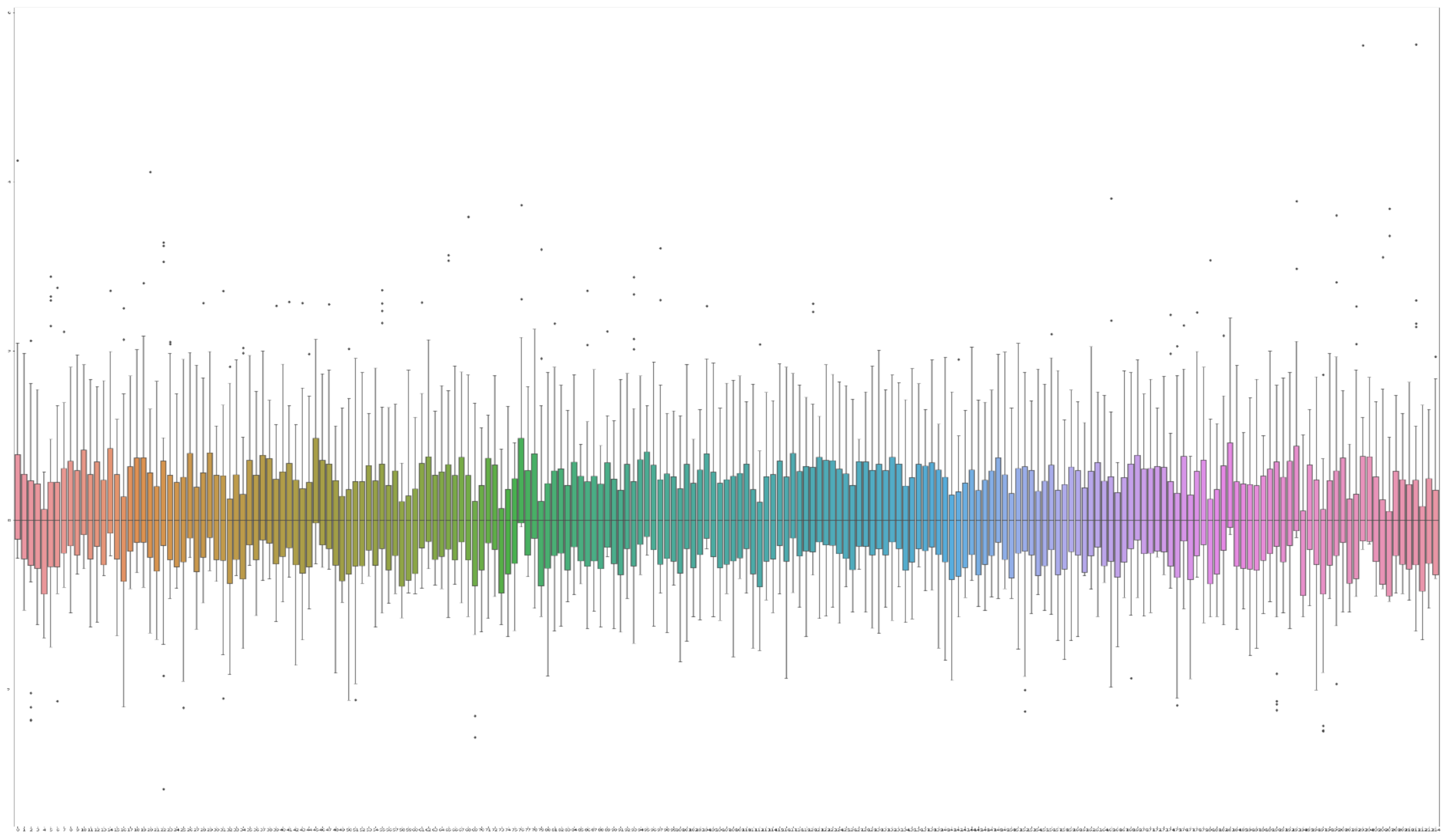
4.2.1 Application on our dataset:

We've used those 3 normalization methods on our dataset and plotted the boxplot to see the spread of the feature values of the datasets. Some significant box-plots are displayed below:

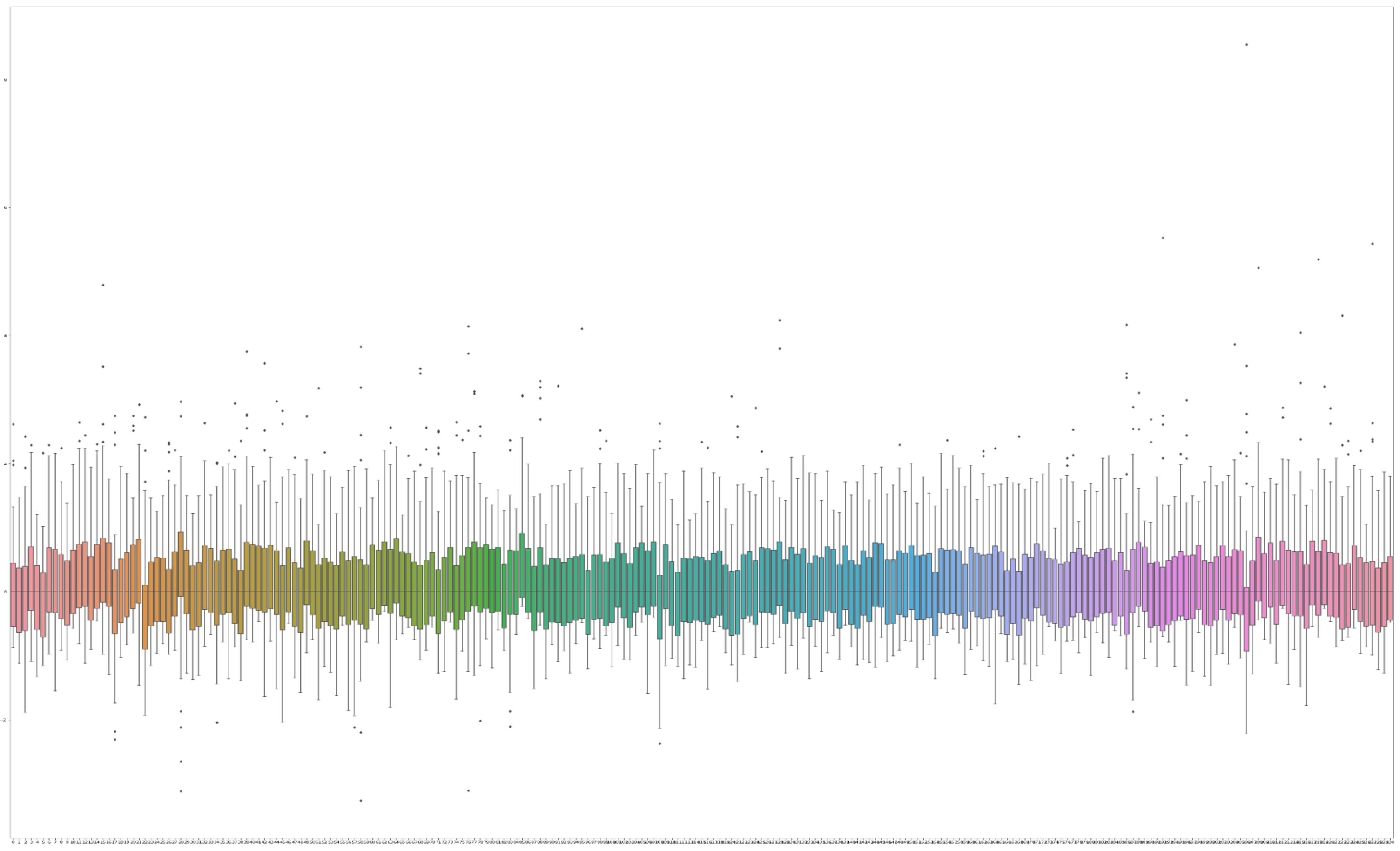
- **Boxplot on severe dataset (after normalization using Robust scaling technique)**



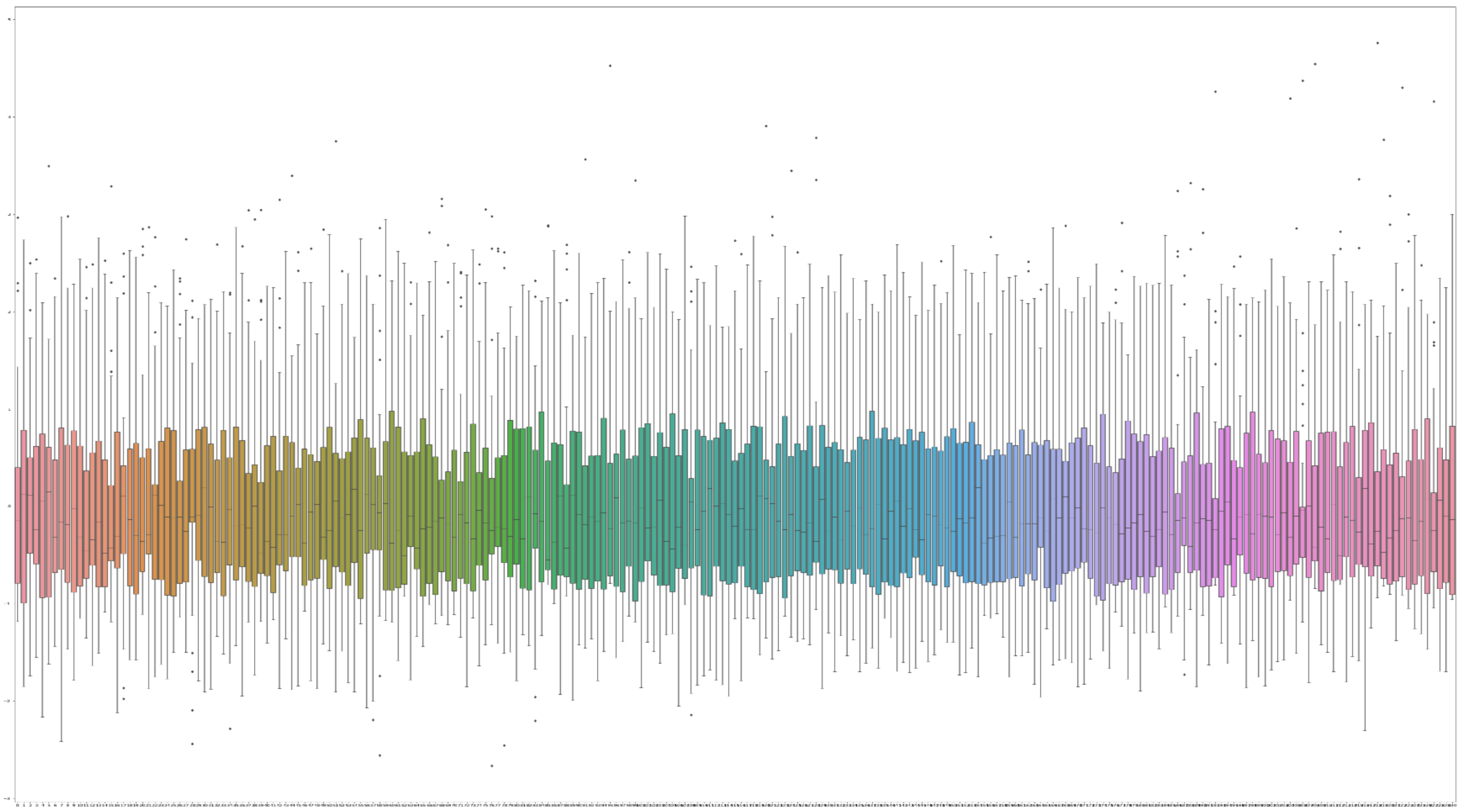
- **Boxplot on non-severe dataset (after normalization using Robust scaling technique)**



❖ **Boxplot on whole dataset (after normalization using Robust scaling technique)**



❖ **Boxplot on whole dataset (after normalization using standard scaling technique)**



Now we can see that the range of the protein intensity value is quite stable and normal. There are many outliers present on different protein ids that can be removed using IQR treatment (mentioned in 4.1.2)

Chapter 5: Dimension Reduction Techniques

5.1 SURE INDEPENDENCE SCREENING (SIS):

5.1.1 BACKGROUND

Consider the problem of estimating a p-vector of parameters β from the linear model

$$y = X\beta + \varepsilon \tag{5.1}$$

where $y = (Y_1,...,Y_n)^T$ is an n-vector of responses, $X = (x_1,...,x_n)^T$ is an n×p random design matrix with independent and identically distributed (IID) $x_1,...,x_n$, $\beta = (\beta_1,...,\beta_p)^T$ is a p-vector of parameters and $\varepsilon = (\varepsilon_1,..., \varepsilon_n)^T$ is an n-vector of IID random errors. When dimension p is high, it is often assumed that only a small number of predictors among $X_1,...,X_p$ contribute to the response, which amounts to assuming ideally that the parameter vector β is sparse. With sparsity, variable selection can improve the accuracy of estimation by effectively identifying the subset of important predictors, and also enhance model interpretability with parsimonious representation. Sparsity comes frequently with high dimensional data, which is a growing feature in many areas of contemporary statistics. The problems arise frequently in genomics such as gene expression and proteomics studies, biomedical imaging, functional MRI, tomography, tumour classifications, signal processing, image analysis, and finance, where the number of variables or parameters *p* can be much larger than sample size *n*. For instance, one may wish to classify tumours using microarray gene expression or proteomics data; one may wish to associate protein concentrations with expression of genes or predict certain clinical prognosis (e.g., injury scores or survival time) using gene expression data. For this kind of problems, the dimensionality can be much larger than the sample size, which calls for new or extended statistical methodologies and theories. See, e.g., Donoho (2000) and Fan and Li (2006) for overviews of statistical challenges with high dimensionality.

5.1.2 Sure Independent Screening: Theory

Dimension reduction or feature selection is an effective strategy to deal with high dimensionality. With dimensionality reduced from high to low, computational burden can be reduced drastically. Meanwhile, accurate estimation can be obtained by using some well-developed lower dimensional method. Motivated by this along with those concerns on the Dantzig selector, we have the following main goal in our paper:

- Reduce dimensionality p from a large or huge scale (say, $\exp(O(n^{\xi}))$ for some $\xi > 0$) to a relatively large-scale d (e.g., $o(n)$) by a fast and efficient method.

We achieve this by introducing the concept of sure screening and proposing a sure screening method Sure Independence Screening (SIS). Here and below, by sure screening we mean a property that all the important variables survive after variable screening with probability tending to one. This dramatically narrows down the search for important predictors.

By sure screening we mean a property that all the important variables survive after applying a variable screening procedure with probability tending to one. A dimensionality reduction method is desirable if it has the sure screening property. Below we introduce a simple sure screening method using component wise regression. Throughout the paper we centre each input variable so that the observed mean is zero, and scale each predictor so that the sample standard deviation is one. Let $\mathrm{M}_* = \{1 \le i \le p : \beta_i \, 6= 0\}$ be the true sparse model with non sparsity rate $s = |\mathrm{M}_*|$. The other $p - s$ variables can also be correlated with the response variable via linkage to the predictors contained in the model. Let $\omega = (\omega_1, \cdots, \omega_p)^T$ be a p-vector obtained by the component wise regression, that is,

$$\omega = X^T y,$$

where the $n \times p$ data matrix X is first standardized column wise as mentioned before.

For any given $\gamma \in (0,1)$, we sort the p component wise magnitudes of the vector ω in a decreasing order and define a sub model

$$M_\gamma = \{1 \le i \le p : |\omega_i| \text{ is among the first } [\gamma_n] \text{ largest of all}\},$$

where $[\gamma n]$ denotes the integer part of γn. This is a straightforward way to shrink the full model $\{1,\cdots,p\}$ down to a sub model M_γ with size $d = [\gamma n] < n$. We call this method Sure Independence Screening (SIS). Its computational cost is that of multiplying a $p{\times}n$ matrix with an n-vector plus getting the largest d components of a p-vector, so SIS has computational complexity $O(np)$. We remark here that d is below sample size n.

It is worth to mention that SIS uses only the order of component wise magnitudes of ω, so it is indeed invariant under scaling. Thus the idea of SIS is identical to selecting predictors using their correlations with the response. To implement SIS, we may choose $d = [\gamma n]$ to be conservative, for instance, $n - 1$ or $n/\log n$ depending on the order of sample size n. Although SIS is proposed to reduce dimensionality p from high to below sample size n, nothing can stop us applying it with final model size $d \ge n$, say, $\gamma \ge 1$. It is obvious that larger d means larger probability to include the true model M_* in the final model M_γ.

SIS is a hard-thresholding-type method. For orthogonal design matrices, it is well understood. But for general design matrices, there is no theoretical support for it, though this kind of idea is frequently used in applications. It is important to identify the conditions under which the sure screening property holds for SIS, i.e.,

$$P\,(M_* \subset M_\gamma) \to 1 \qquad\qquad \text{as } n \to \infty$$

for some given γ.

5.1.3 First Stage Dimension Reduction using SIS

We've considered the problem of identifying the bio-markers for severe and non-severe Covid-19 patients using dimension reduction through SIS. At the first step we've fitted Univariate Logistic Regression on each of the proteins separately. Here the Target variable (Y) is the two groups: Severe (1), Non-severe (0) & the co-variate(x) is intensity of a particular protein id. Thus, we have got 230 univariate logistic regressions. For each of them we test for the significance of the corresponding regression co-efficient and keep only those which are significant with 95% level of confidence.

Thus, we get the following 62 proteins out of all 230 proteins after applying SIS.

Table 5.1: *Name of the 62 proteins (out of 230 proteins) obtained by dimension reduction using SIS*

No.	Protein IDs	No.	Protein IDs	No.	Protein IDs	No.	Protein IDs	No.	Protein IDs	No.	Protein IDs	No.	Protein IDs
1	Q9NYW8	11	P02745	21	P07360	31	P00915	41	P02675	51	P07357	61	P68871
2	O75150	12	P04430	22	P00739	32	A2NJV5	42	P02679	52	P08697	62	P69905
3	Q6PL18	13	P0C0L5	23	P51884	33	P00734	43	P02751	53	P0C0L4		
4	Q9UGM5	14	A0A0C4DH35	24	P61626	34	P00747	44	P02765	54	P0DOX5		
5	Q7Z406	15	P0DP03	25	P22792	35	P00748	45	P02774	55	P0DP08		
6	P43251	16	P01780	26	Q06830	36	P01009	46	P02787	56	P19652		
7	A0A0A0MS15	17	P02654	27	Q96PD5	37	P01023	47	P04004	57	P25311		
8	P00488	18	P32119	28	P05452	38	P01024	48	P04406	58	P27169		
9	P01602	19	P02743	29	P02042	39	P02652	49	P06312	59	P43652		

5.4 LASSO: THEORY

LASSO (Least Absolute Shrinkage Selector Operator) is a very popular shrinkage method. (*Ref: [1]*)

Consider the model: $y_i = \beta_0 + \sum \beta j\ x_{ij}$, where least squares fitting procedure estimates $\beta_0, \beta_1, \ldots, \beta_p$ using the values that minimize:

$$\text{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

The lasso coefficients, β_λ^L minimize the quantity:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|$$

The lasso uses an l_1 penalty. The l_1 norm of a coefficient vector β is given by

$$||\beta||_1 = \sum |\beta j|.$$

The lasso shrinks the coefficient estimates towards zero. The l_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. Hence, much like best subset selection, the lasso performs variable selection. Thus, lasso yields sparse models -that is, models that involve only a subset of the variables. Selecting a good value of λ for the lasso is critical; which is done using "K-fold cross-validation" and "Shrunken Centroids".

5.4.1 K-fold cross-validation:

This approach involves randomly k-fold CV dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k − 1 folds. The mean squared error, MSE_1, is then computed on the observations in the held-out fold. This procedure is repeated k times; each time, a different group of observations is treated as a validation set. This process results in k estimates of the test error, $MSE_1, MSE_2, \ldots, MSE_k$. The k-fold CV estimate is computed by averaging these values,

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k}(MSE_i)$$

5.4.2 Shrunken Centroids Method:

There are two factors to consider in selecting good proteins within class distance and between classes distance. When expression levels of a protein for all samples in the same class are fairly consistent with a small variance, but are largely different among samples of different classes, the protein is considered as a good candidate for classification. The protein has discriminant information for different classes. In the nearest shrunken centroid method, variance within a class was further taken into consideration to measure the goodness of a protein within class. The difference between a class centroid and the overall centroid for a protein is divided by the within class variance to give a greater weight to a protein whose expressions are stable among samples in the same class. A threshold value is applied to the resulting normalized class centroid differences. If it is small for all classes, it is set to zero, meaning the protein is eliminated. This reduces the number of proteins that are used in the final predictive model. The mathematics involved in the shrunken centroid algorithm can be found from Tibshirani's work. Assume there are n patients, p proteins and K classes where s_i is the within class standard deviation for protein i.

$$S_i{}^2 = \frac{1}{n-K}\sum_{k=1}^{K}(\sum_{j\in C_k}((X_{ij}-\overline{X_{ij}})^2)) \ and\ m_k = \sqrt{\frac{1}{n_k}-\frac{1}{n}}$$

S_0 is a small positive constant to avoid having a large d_{ij} value when a protein has a small standard deviation S_i. In the nearest shrunken centroid, for each protein i, d_{ik} is evaluated in each class k to see if it is smaller than a chosen value, threshold Δ. If it is too small for all classes, the protein i is considered not very significant and is eliminated from the protein lists. Otherwise update d_{ik} by $sign(d_{ik})(|d_{ik}|-\Delta)$. The updated d_{ik} is called d_{ik} afterwards. Once the optimal threshold is found, all centroids are updated accordingly (shrunken centroid) using the equation

$$\overline{X}_{ik} = \overline{X}_i + m_k(S_i - S_0)d_{ik}$$

where d_{ik} is the updated d_{ik}. If a protein is shrunk to zero for all classes, then it is considered not important and is eliminated. After the centroids are determined through the training stage, classification can take place with new samples. Test samples are classified to belong to the class whose shrunken centroid is nearest to it.

<u>5.4 SECOND STAGE DIMENSION REDUCTION USING LASSO:</u>

We've applied LASSO with tuning parameter λ=0.02 on the 62 proteins obtained via first stage dimension reduction using SIS. We've used various methods including K-fold cross-validation for finding the best tuning parameter λ as the tuning parameter plays a very crucial role for LASSO. Finally, we've got the dimension further reduced to 17 proteins after applying LASSO. They are given in the following table.

Table 5.2: *Name of the 17 proteins (out of 106 proteins after SIS) obtained by 2nd stage dimension reduction using LASSO*

NO.	Protein IDs	NO.	Protein IDs
1	Q9NYW8	11	P00748
2	Q9UGM5	12	P01023
3	P01602	13	P02652
4	P32119	14	P02675
5	P51884	15	P0DP08
6	P61626	16	P25311
7	Q96PD5	17	P68371
8	P02042		
9	P00915		
10	A2NJV5		

So, we can conclude from our 1st stage of dimension reduction technique that the above mentioned 17 proteins has a significant effect on severe Covid-19 patients.

Chapter 6: Summary & Conclusion

- ❖ We've studied different broad areas where big data is being generated and about its different applications and scope of applications in future.
- ❖ The project focused on study of different statistical techniques which can be used to analyze big data with high dimension.
- ❖ The techniques studied throughout the course of the project were used on a real life micro array dataset with high dimensionality.
- ❖ We've imputed the missing values of the dataset using KNN imputation, Mean and median imputation.
- ❖ Concept of Outlier analysis using Box-plot method was discussed in details.
- ❖ Different Normalization methods including Max-Min Normalization, Standardization and Quantile Normalization were covered and applied on the real micro-array data set.
- ❖ We've tried three different and independent approaches to reach our objective, i.e to identify a finite number of bio-marker proteins using a dimension reduction technique namely:
  a. SIS followed by LASSO: ***17 protein ids identified.***

Chapter 7: Future Scope

- ❖ We've used the dataset obtained by using Robust scaling normalization. Other datasets can be used for further statistical analysis.
- ❖ Other outlier detection and treatment techniques can be performed.
- ❖ Different dimension reduction techniques Sparse Hierarchical Clustering, Test for inequality of Means (t-test, Welch Test, Wilcoxon Rank Sum Test), can be applied to identify more significant bio-marker.
- ❖ Using Random Forest Classifier with Nearest Shrunken centroids can be applied for dimension reduction.

References:

- ❖ Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani | An Introduction to Statistical Learning with Applications in R (Ver. 2)
- ❖ Jianqing Fan and Jinchi Lv | Sure independence screening for ultrahigh dimensional feature space, 2008
- ❖ Jianqing Fan, Fang Han and Han Liu | Challenges of Big Data analysis, National Science Review 1:293-314, 2014
- ❖ Shikah J. Alsunaidi , Abdullah M. Almuhaideb ,, Nehad M. Ibrahim , Fatema S. Shaikh , Kawther S. Alqudaihi , Fahd A. Alhaidari , Irfan Ullah Khan , Nida Aslam | Applications of Big Data Analytics to Control COVID-19 Pandemic.
- ❖ Jure Leskovec Stanford Univ. Anand Rajaraman Milliway Labs Jeffrey D. Ullman Stanford Univ. | Mining of Massive Datasets
- ❖ Ian H. Witten , Eibe Frank , Mark A. Hall | Data Mining , Practical Machine Learning Tools and Techniques.
- ❖ Scikit-learn Documentation, Python Documentation.

Python Code used:

Missing value treatment:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('E:/3rd sem project/new data/Plasma2021_37-Metabo_Input.csv')
df.head()
df.replace('0', np.nan, inplace=True)
df

null_counts = df.isnull().sum()[1:]/len(df[1:])*100

d1={'Patient Ids':df.columns[1:],'Null_counts':df.isnull().sum()[1:]}
df_n=pd.DataFrame(d1)
df_n

d3={'Proteins':df['Protein IDs'][1:],'Null_counts':df.isnull().sum(axis=1)[1:]}
df_null = pd.DataFrame(d3)
df_null
df_null[df_null['Null_counts']==37]

P_id=df['Protein IDs'].to_numpy()
T=df.index.to_numpy()
p=df_t[0].to_numpy()
I=df_t.index.to_numpy()
keys = I
vals = p
d2=dict(zip(keys, zip(vals)))
keys_p=P_id
vals_p=T
d3=dict(zip(keys_p,zip(vals_p)))

def getprotein_col_index(ProteinIDs):
    protein_id_column_index=d3[ProteinIDs][0]
    return protein_id_column_index

def getprotein_intensity(protein_id_column_index,patient_id):
    return df_t[protein_id_column_index][patient_id]

Proteins_to_remove_2=df_null[df_null['Null_counts']>18]['Proteins'].to_numpy()
Proteins_to_remove_2

ins2=[]
for i in range(len(Proteins_to_remove_2)):
    ins2.append(getprotein_col_index(Proteins_to_remove_2[i]))
ins2
```

```
for i in range(len(Proteins_to_remove_2)):
    index_to_remove=getprotein_col_index(Proteins_to_remove_2[i])
    df.drop(index_to_remove,axis=0,inplace=True)
```

KNN Imputation:

```
from sklearn.impute import KNNImputer
imputer = KNNImputer(n_neighbors=5)
df_knn = pd.DataFrame(imputer.fit_transform(df_trans),columns = df_trans.columns)
```

Mean-Median imputation:

```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan, strategy='median')
imp = imp.fit(df_trans_1)
dfmedianimp = imp.transform(df_trans_1)

from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan, strategy='mean')
imp = imp.fit(df_trans_1)
dfmedianimp = imp.transform(df_trans_1)
```

Outlier Detection using Boxplot:

```
#For severe patients
plt.figure(figsize=(50,50))
sns.boxplot(data=df_new_s_t)

#For non-severe patients
plt.figure(figsize=(50,50))
sns.boxplot(data=df_new_ns_t)
```

Outlier treatment using IQR:

```
df_s_copy=df_new_s_t
df_ns_copy=df_new_ns_t
b=[df_s_copy,df_ns_copy]

for item in b:
  cols=item.columns
  for i in range(1,len(cols)):
    percentile25 = item[cols[i]].quantile(0.25)
    percentile75 = item[cols[i]].quantile(0.75)
    iqr=percentile75-percentile25
    upper_limit = percentile75 + (1.5* iqr)
    lower_limit = percentile25 - (1.5* iqr)
    item[item[cols[i]] > upper_limit]
    item[item[cols[i]] < lower_limit]
    item[cols[i]]=item[(item[cols[i]]<=upper_limit)&(lower_limit<=item[cols[i]])][cols[i
]]
```

Normalization:

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import RobustScaler
```

```
#MIN-MAX Scaling
scaler = MinMaxScaler()
# transform data
Min_Max_scaled = scaler.fit_transform(df_knn_s)
df_minmax_s = pd.DataFrame(Min_Max_scaled)
df_minmax_s

#standard Scaling
scaler_1 = StandardScaler()
# transform data
Standard_scaled_s = scaler_1.fit_transform(df_knn_s)
df_standard_scaled_s = pd.DataFrame(Standard_scaled_s )
df_standard_scaled_s

#robust Scaling
trans = RobustScaler()
df_robust = trans.fit_transform(df_knn_s)
# convert the array back to a dataframe
df_robust_s = pd.DataFrame(df_robust)
df_robust_s
```

Sure independent screening:

```
df_new=pd.read_csv('/content/gdrive/My Drive/new covid project work/df_w_robust_remove_o
utlier_wout_con.csv')
df_new=df_new.set_index("Protein IDs")
df_new_t=df_new.T
df_new_t

df_new_t["target"]=[0]*23+[1]*14
df_new_t

#Fitting logistic regression for each protein as feature and condition(S, NS) as
response
import statsmodels.api as sm
pvals=[]
prot=[]
y=df_new_t["target"]
for i in df_new_t.columns:
    prot.append(i)
    sm_model = sm.Logit(df_new_t["target"],df_new_t[i]).fit(disp=0)
    pvals.append(sm_model.pvalues[i])

sis=pd.DataFrame()
sis["prot"]=prot
sis["pvals"]=pvals
sis

sum(sis["pvals"]<0.05)

sis=sis[sis["pvals"]<0.05] #using 95% confidence interval

sis_data["target"]=[0]*23+[1]*14
sis_data.to_csv("/content/gdrive/My Drive/sis.csv")
```

Lasso Fit:

```
df_new=pd.read_csv('/content/gdrive/My Drive/new covid project work/sis.csv')
#df_new=df_new.set_index("Protein IDs")
df_new

Y=df_new["target"]
Y

X= df_new.drop(['target','Protein IDs'],axis=True)
X

#For best tuning parameter alpha

from sklearn.linear_model import LassoCV
from yellowbrick.regressor import AlphaSelection
# Create a list of alphas to cross-validate against

alphas = np.logspace(-10, 1, 400)
# Instantiate the linear model and visualizer
model = LassoCV(alphas=alphas)
visualizer = AlphaSelection(model)
visualizer.fit(X, Y)
visualizer.show()

from sklearn.linear_model import LassoCV
from yellowbrick.regressor.alphas import alphas

# Use the quick method and immediately show the figure
alphas(LassoCV(random_state=0), X, Y)

#K-fold cross validation for best tuning parameter
# grid search hyperparameters for lasso regression
from numpy import arange
from pandas import read_csv
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RepeatedKFold
from sklearn.linear_model import Lasso
model = Lasso()
# define model evaluation method
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
# define grid
grid = dict()
grid['alpha'] = arange(0, 1, 0.01)
# define search
search = GridSearchCV(model, grid, scoring='neg_mean_absolute_error', cv=cv, n_jobs=-1)
# perform the search
results = search.fit(X, Y)
# summarize
print('MAE: %.3f' % results.best_score_)
print('Config: %s' % results.best_params_)
```

```
#LASSO fit
from sklearn.linear_model import Lasso
lasso = Lasso(alpha = 0.02)
lasso_f=lasso.fit(X,Y)
lasso_coeff=lasso_f.coef_

protein=[]
coefficients=[]
for i in X.columns:
    protein.append(i)
for i in lasso_coeff:
    coefficients.append(i)
lasso_df= pd.DataFrame()
lasso_df["Protein"]=protein
lasso_df["Coefficients"]=coefficients
lasso_df

np.sum(lasso_df["Coefficients"]==0) #no of proteins whose coefficients are zero

lasso_df_1=lasso_df[lasso_df["Coefficients"]!=0]

df_new_copy=df_new.set_index("Protein IDs")
df_new_copy

lasso_data=pd.DataFrame()
for i in lasso_df_1["Protein"]:
    lasso_data[i]=df_new_copy[i]

lasso_data["target"]=[0]*23+[1]*14
lasso_data.to_csv("/content/gdrive/My Drive/lasso.csv")
```