Technotran

A Full Semester Internship report on

AI & ML Using Python

Submitted in a partial fulfillment for the award of the degree

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

NALLATHURU SHARAN

(20G21A04F8)

Under the esteemed Guidance of

Mrs C. SUNEETHA, M,Tech, (Ph.D)

Associate professor, Dept. of ECE

DEPARTMENT OF

ELECTRONICS AND COMMUNICATION ENGINEERING

AUDISANKARA

COLLEGE OF ENGINEERING & TECHNOLOGY

An Autonomous Institute affiliated to JNTUA, Ananthapuram & Accredited by NAAC with 'A+' Grade

(AUTONOMOUS)

NH5 Bypass Road, Gudur – 524101, Tirupati (DT.)

Andhra Pradesh www.audisankara.ac.in

2023-2024

AUDISANKARA
COLLEGE OF ENGINEERING & TECHNOLOGY
An Autonomous Institute affiliated to JNTUA, Ananthapuram & Accredited by NAAC with 'A+' Grade

(AUTONOMOUS)

NH5 Bypass Road, Gudur – 524101, Tirupati (DT.) Department of Electronics and Communication Engineering

AUDISANKARA

CERTIFICATE

This is to certify that the Full Semester Internship report entitled **" AI & ML Using Python "** is the bonafide work done by the student **NALLATHURU SHARAN , REGD NO : 20G21A04F8,** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics and Communication Engineering,** from Jawaharlal Nehru Technological University Anantapur, Anantapuramu, during the year 2023-2024.

Internship Guide
Mrs C. SUNEETHA, M. Tech, (Ph.D.)
Associate Professor,
Department of ECE
ASCET, GUDUR – TIRUPATI (DT).

Head of the Department
Prof. J. AMARENDRA M.Tech., (Ph.D.)
Associate Professor & HOD,
Department of ECE,
ASCET, GUDUR-TIRUPATI(DT).

Submitted for the viva-voce Examination held on:

Internal Examiner **External Examiner**

DECLARATION

I, **NALLATHURU SHARAN, REGD-NO:20G21A04F8,** hereby declare that the Project Work entitled **" AI & ML Using Python"** done by us under the esteemed Guidance of **C. SUNEETHA, M,Tech, (Ph.D).,** Associate professor Department of ECE and **R.Vikas Reddy Director of Technotran.** The Full Semester Internship report is submitted in partial fulfillment of the requirements for the award of the bachelor's degree in Electronics and Communication Engineering.

Date:

Place:

Signature of the candidate

NALLATHURU SHARAN

(20G21A04F8)

ACKNOWLEDGEMENT

The satisfaction and elation that accompany the successful completion of any task would be incomplete without the mention of the people who have made it a possibility. It is our great privilege to express our gratitude and respect to all those who have guided us and inspired us during the course of this project Working towards Full Semester Internship has been a period of various challenges that have led to a great deal of learning and professional growth. Making it through would not have been possible without the help and support of family and friends. First and Foremost, I would like to express my deep and sincere thanks to the team of Directors of **Technotran** India for giving me the opportunity to do an internship within the organization

I express my sincere gratitude and thanks to our honorable Chairman **Dr. VANKI PENCHALAIAH, M.A.,M.L.,Ph.D** for providing facilities and necessary encouragement during the Full semester Internship Program.

I am highly indebted to Director **Dr. A. MOHAN BABU, Ph.D.,** and Principal Prof. **K.DHANUNJAYA, M. Tech, (Ph.D.),** for the facilities provided to accomplish this internship. I would like to thank my Head of the Department **Prof. J.AMARENDRA,M.Tech, (Ph.D),** for hisconstant support and guidance throughout my internship. I would like to thank **Dr. G. Chenchu Krishnaiah, M.E, Ph.D, MISTE, IAENG, SDIWC.,** Internship coordinator, Department of ECE for their support and advice to get and complete internship in above said organization.

I would like to convey my heartfelt gratitude to external supervisor and mentor, **R. Vikas Reddy, Director** for having accepted me as Full Semester Internship report student and providing unconditional support and guidance regarding all aspects of internship and career. I also would like all the people that worked along with me in **Technotran**, Hyderabad with their patience and openness they created an enjoyable and learning oriented ambience online. It is indeed with a great sense of pleasure and immense sense of gratitude that I acknowledge the help of these individuals. I am extremely great full to my department staff members and friends who helped me in successful completion of this internship.

NALLATHURU SHARAN

(20G21A04F8)

PROFILE OF THE COMPANY

Name of the Company With Address:

Technotran -Nellore-Andhra Pradesh

Email: admin@technotran.in

TECHNOTRAN (ISO 9001:2015 Certified Company) is an Innovative Educational Technology Company in the fields of Embedded systems, Robotics,3d-printing, IoT & Artificial Intelligence. We serve universities, colleges, and schools nationwide. Our unwavering commitment revolves around delivering top-tier services, encompassing DIY Robotic kit design, electronic product design, PCB Design, prototyping, and Manufacturing, all meticulously customized to precise specifications. Our ultimate mission is to empower both students and institutions with state-of-the-art technology solutions.

Website	http://www.technotran.in
Industry	Appliances, Electrical, and Electronics Manufacturing
Company size	11-50 employees
Headquarters	Nellore, Andhra Pradesh
Type	Educational
Founded	2013
Specialties	Workshops for Engineering students, Summer Internship & training, Embedded & Robotic kits/Project Kits, Robotics Labs, IoT labs, and School Robotics

Technotran offers specialized internships in the domains of IoT, Robotics, Embedded Systems, Artificial Intelligence, Machine Learning using python and more.

Internships at Technotran are AICTE-approved, ensuring participants receive a valuable certificate upon successful completion. The program offers industry-oriented training, providing hands-on experience and exposure to diverse tech domains, including IoT, Robotics, Embedded Systems, and AI. This comprehensive approach equips participants with the skills needed to pursue dream jobs globally.

Technotran offers specialized internships in the following domains IoT, Robotics, Embedded Systems, Artificial Intelligence, Machine Learning, and Python.

The target market includes:

Aspiring Professionals: Individuals looking to enhance their skills and knowledge in specific professional domains, such as IoT, Robotics, Embedded Systems, and AI **Students in Higher Education:** Undergraduates and postgraduates aiming to supplement their academic curriculum with hands-on experiences and practical learning opportunities.

Working Professionals: Professionals in the industry seeking to upskill and stay abreast of the latest industry trends and technologies.

Colleges and Educational Institutions: Technotran collaborates with colleges under Memorandum of Understanding (MoU), offering tailored courses, placement preparation programs, and industry connections to enhance the overall educational experience

Business Segments and Activities:

1. B2B Team:

-Outreach to Colleges: The B2B team proactively reaches out to colleges, presenting and offering Technotran's comprehensive training courses, professional courses, and curated programs. This involves establishing connections with academic institutions and showcasing the benefits of integrating Technotran's educational offerings.

-Tech Events: Organising and hosting tech events targeted at colleges to promote innovation, hands-on learning, and industry connections.

-Tech Webinars: Conducting webinars to educate colleges about the latest advancements in technology and industry best practices.

-Curated Placement Preparation: Offering specialized programs to colleges for preparing students for placements, including mock interviews and resume building workshops.

CERTIFICATE OF INTERNSHIP COMPLETION :

AICTE Internship ID :INTERNSHIP_169909378365461d1777c43

Technotran
Technology.Training.Transformation

CERTIFICATE OF COMPLETION

This is to certify that

Sharan Nallathuru

student of the ECE department, Audisankara College Of Engineering And Technology, Gudur,has Successfully Completed the 16-week internship certification program at Technotran Company on AI & ML Using Python from 8th January 2024 to 27th April 2024.

Certificate ID No: **TT-ICP-241313**
Roll No: **20G21A04F8**

R.VIKAS REDDY
Director
Technotran

To verify the authenticity of this certificate, please email us at admin@technotran.in

CONTENTS

CONTENTS

Project title:

Corona Virus Infection Probability Using Machine Learning

Abstract

This project revolves around predicting COVID-19 infection probabilities by analyzing clinical symptoms like cough, fever, and cold using a provided dataset. Leveraging the Jupyter software environment, the project entails a comprehensive journey through exploratory data analysis (EDA), feature engineering, model training, and evaluation. In the initial phase of EDA, the dataset's structure is scrutinized to identify missing values, outliers, and the distribution of variables. Particular attention is given to understanding how symptoms are distributed among infected and non-infected individuals. Following EDA, feature engineering steps are undertaken to enhance the dataset's representation, potentially creating new features or transforming existing ones. Categorical variables such as symptoms are encoded into numerical representations, and numerical features may be scaled or normalized. Moving to model training, various machine learning algorithms are explored, including logistic regression, decision trees, random forests, and gradient boosting models, to predict infection status accurately. Hyperparameter tuning and handling class imbalance are essential considerations during this phase. Finally, model evaluation involves assessing model performance using metrics like accuracy, precision, recall, and F1 score, while techniques like cross-validation help gauge model generalization. Documentation throughout the project, combined with clear communication of methodologies and findings using Jupyter notebooks, ensures a comprehensive and transparent approach to understanding and predicting COVID-19 infection probabilities based on clinical symptoms.

LIST OF FIGURES

Chapter-1
Introduction

Introduction

In this era of automation, artificial intelligence and data science have important role in the health care industry. These technologies are so well-connected that medical professionals can easily manage their roles and patient care. All health care organizations work hard to develop an automated system that can be used to accept the challenges faced in health care. Scientists are working on machine learning (ML) to develop smart solutions to diagnose and treat disease. ML is capable of detecting disease and virus infections more accurately so that patients' disease can be diagnosed at an early stage, the dangerous stages of diseases can be avoided, and there can be fewer patients. In the same manner, ML can be used to automate the task of predicting COVID-19 infection and help forecast future infection tallies of COVID-19. In this chapter, we include methods for forecasting future cases based on existing data. ML approaches are used and two solutions, one for predicting the chances of being infected and other for forecasting the number of positive cases, are discussed. A trial was done for different algorithms, and the algorithm that gave the results with the best accuracy is covered in the chapter. The chapter discusses autoregressive integrated moving average (ARIMA) time series for forecasting confirmed cases for various states in India. Two classifiers, random forest and extra tree classifier (ETC) are selected; both have an accuracy of more than 90%. Of the two, ETC has 93.62% accuracy. These results can be used to take corrective measures by different government bodies. The availability of techniques for forecasting infectious disease can make it easier to fight against infectious disease such as COVID-19.

The objective of the chapter is to find the best-performing ML model for predicting and forecasting COVID-19. Afterward, reader will obtain a glimpse of some ML fundamentals and how ML can be used to predict and forecast COVID-19, which may help in future health care automation tasks using ML and data science.

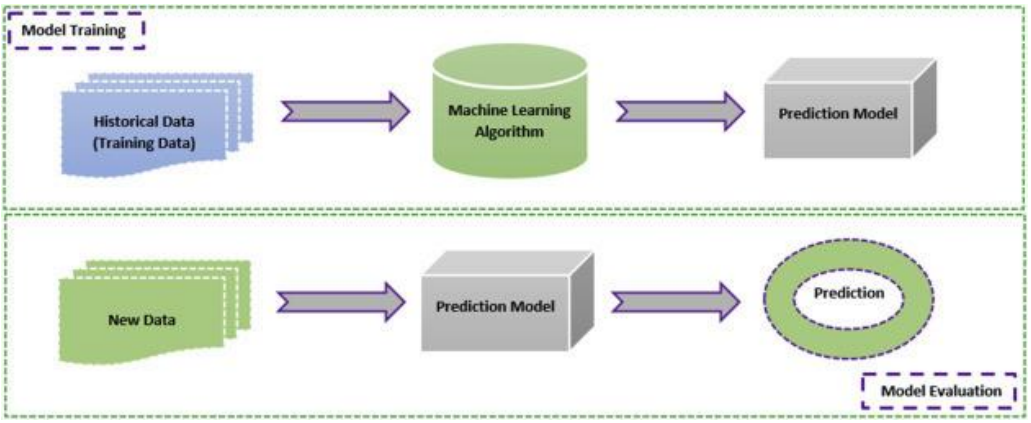
The chapter is divided into eight sections. Section 2 introduces COVID-19, the incubation period of COVID-19, and other details about COVID-19. Section 3 gives a brief overview of ML and its methods. Section 4 describes how ML can be used in COVID-19. Section 5 describes different ML techniques for prediction and forecasting, including a general ML process flowchart. describe the proposed symptoms-based prediction model for classification of COVID-19 infection and the ARIMA model for forecasting the future confirmed case count of COVID-19 in India. Section 8 focuses on conclusions and future work.

COVID-19 is not just a name now. It has become a deadly widespread virus that has affected tens of thousands of people all over the world. Its origin was Wuhan City, China in Dec. 2019. When people were unaware of the virus, COVID-19 started to spread from one person to another; it has slowly reached almost all countries and has become a pandemic

COVID-19 is the short form for coronavirus disease 2019, an illness caused by a novel coronavirus (nCoV) now known as severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2); formerly called 2019-nCoV. COVID-19 was not the formal name of this virus; it was called SARS-CoV-2 by the International Committee on Taxonomy of Viruses because its symptoms were related to the virus that caused the SARS outbreak in 2003. However, this virus had not previously appeared in humans, and this time, they were severely infected by the virus, so to avoid confusion with other viruses, the World Health Organization (WHO) named it COVID-19 to communicate with the public

Chapter-2
Machine Learning

2.1 Introduction to Machine Learning:



2.1.1 Introduction to Machine Learning

ML is the field of study that gives computers the ability to learn without being explicitly programmed. Thus, we can define ML as the field of computer science in which machines can be designed that can program themselves.

The process of learning is simply learning from experience or observations from previous work, such as examples, or instruction, to look for patterns in data and with the help of examples, provided the system can make better decisions. The basic aim of ML is to make computers learn automatically with no human intervention and to adjust perform actions accordingly.

Past data are used to train the model, and then this trained model is used to test new data and then for prediction. The trained ML model's performance is evaluated using some portion of available past data (which is not present during training). This is usually referred as the validation process. In this process, the ML model is evaluated for its performance measure, such as accuracy. Accuracy describes the ML model's performance over unseen data in terms of the ratio of the number of correctly predicted features and total available features to be predicted.

2.2 Some machine learning methods

ML algorithms can be divided into supervised or unsupervised learning:

(1) Supervised ML algorithms is a type of ML technique that can be applied according to what was previously learned to get new data using labeled data and to predict future events or labels. In this type of learning, supervisor (labels) is present to guide or correct. For this first analysis, the known training set and then the output values are predicted using the learning algorithm. The output defined

by the learning system can be compared with the actual output; if errors are identified, they can be rectified and the model can be modified accordingly [20].

(2) Unsupervised ML algorithms: In this type, there is no supervisor to guide or correct. This type of learning algorithm is used when unlabeled or unclassified information is present to train the system. The system does not define the correct output, but it explores the data in such a way that it can draw inferences (rules) from datasets and can describe hidden structures from unlabeled data.

(3) Semisupervised ML algorithms are algorithms that are between the category of supervised and unsupervised learning. Thus, this type of learning algorithm uses both unlabeled and labeled data for training purposes, generally a small amount of labeled data and a large amount of unlabeled data. This type of method is used to improve the accuracy of learning.

(4) Reinforcement ML algorithms is a type of learning method that gives rewards or punishment on the basis of the work performed by the system. If we train the system to perform a certain task and it fails to do that, the system might be punished; if it performs perfectly, it will be rewarded. It typically works on 0 and 1, in which 0 indicates a punishment and 1 indicates a reward.

It works on the principle in which, if we train a bird or a dog to do some task and it does exactly as we want, we give it a treat or the food it likes, or we might praise it. This is a reward. If it did not perform the task properly, it might be scolded as a punishment by us

2.3 Use of Machine Learning for Covid-19

ML is used in various fields, including medicine to predict disease and forecast its outcome. In medicine, the right diagnosis and the right time are the keys to successful treatment. If the treatment has a high error rate, it may cause several deaths.

For this task, ML achieved a milestone in the field of health care. ML techniques are used to interpret and analyze large datasets and predict their output. These ML tools were used to identify the symptoms of disease and classify samples into treatment groups. ML helps hospitals to maintain administrative processes and treat infectious disease.

ML techniques were previously used to treat cancer, pneumonia, diabetes, Parkinson disease, arthritis, neuromuscular disorders, and many more diseases; they give more than 90% accurate results in prediction and forecasting.

The pandemic disease known as COVID-19 is a deadly virus that has cost the lives of many people all over the world. There is no treatment for this virus. ML techniques have been used to predict whether patients are infected by the virus based on symptoms defined by WHO and CDC.

ML is also used to diagnose the disease based on x-ray images. For instance, chest images of patients can be used to detect whether a patient is infected with COVID-19.

Moreover, social distancing can be monitored by ML; with the help of this approach, we can keep ourselves safe from COVID-19

2.4 Analysis of Literature Review

According to the results obtained from the Systematic Literature Review (SLR), RQ1 could not be answered thoroughly. In many works, a clear comparison between various machine learning algorithms has been conducted deliberately but the conclusion couldn't be achieved. A comparison model was suggested.Considering the results from a set of literature, a particular set of algorithms that include: Support Vector Machine (SVM), Artificial Neural Networks (ANNs) and Random Forests (RF) were chosen to perform an experimental evaluation to select the most suitable algorithm to predict COVID-19.

Chapter-3
Proposed Methodology

3.1 Data Collection and Preprocessing:

Gather the dataset containing patient records with clinical symptoms (e.g., cough, fever, cold) and corresponding COVID-19 test results.
Perform data preprocessing tasks such as handling missing values, encoding categorical variables, and scaling numerical features.

3.2 Exploratory Data Analysis (EDA):

Conduct exploratory data analysis to gain insights into the distribution of symptoms, the prevalence of COVID-19 cases, and potential correlations between symptoms and infection status.
Visualize key relationships using plots, histograms, and other statistical summaries to understand the underlying patterns in the data.

3.3 Feature Engineering:

Engineer new features or transform existing ones to enhance the predictive power of the model. Extract relevant information from clinical symptoms and create additional features that may aid in predicting COVID-19 infection probabilities.

3.4 Model Selection and Training:

Choose appropriate machine learning algorithms for the task, considering factors such as interpretability, scalability, and performance.
Split the dataset into training and testing sets to evaluate model performance effectively. Train multiple models, such as logistic regression, decision trees, random forests, or gradient boosting machines, using the training data.

3.5 Model Evaluation:

Evaluate the performance of each model using appropriate evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC-AUC.
Compare the performance of different models and select the one that achieves the highest predictive accuracy and generalization to unseen data.

3.6 Hyperparameter Tuning:

Fine-tune the hyperparameters of the selected model using techniques such as grid search or random search to optimize performance further.
Conduct cross-validation to ensure the robustness of the model and mitigate overfitting.

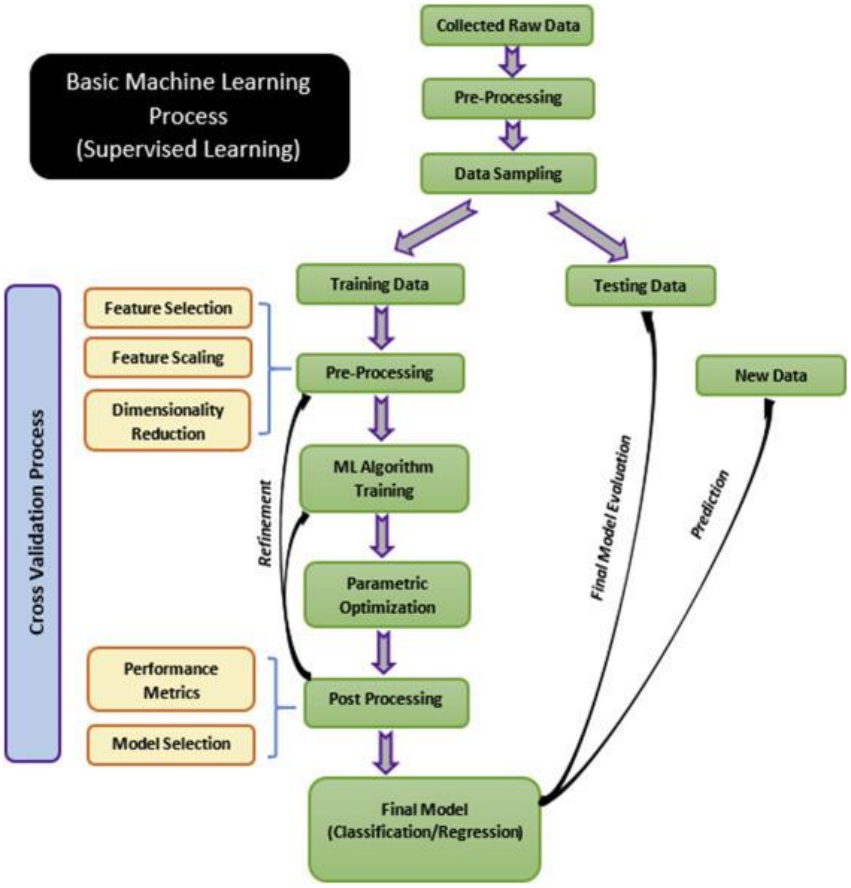
3.7 Model Interpretation:

Interpret the trained model to understand the relative importance of features and how they contribute to predicting COVID-19 infection probabilities.
Visualize model decision boundaries, feature importances, and other relevant insights to provide actionable information for healthcare professionals.

3.8 Deployment and Monitoring:

Deploy the trained model into a production environment, such as a web application or API, to make predictions on new patient data.
Implement monitoring mechanisms to track model performance over time, detect drift, and ensure the continued accuracy and reliability of predictions.

Chapter-4
Supervised Learning



4.0.1 Flow Diagram

A symptom-based predictive model was proposed to predict COVID-19 based on symptoms defined by the WHO and CDC.

Because there is no proper description of symptoms declared by the WHO, based on some existing symptoms, we defined a model used to predict the disease according to the accuracy given by the model.

We created a symptom database in which rules were created and used as input. Then, these data were used as raw data. Then, feature selection took place as part of preprocessing data. The data were divided into training data (80% of data) and test data (20% of data), usually known as the train-test split process. This split is generally done in a stratified or random manner so that population distribution in both groups consists of shuffled data, which leads minimized bias or skewness in the data. Training data were used to train the ML classifier that we used in the model, and test data were used to test that classifier in terms of accuracy received over a predefined unseen portion of the dataset.

In our work, the symptoms and patient's class dataset was defined on the basis of symptoms such as fever, cough, and sneezing, whether the patient had traveled to an infected place, age, and whether the patient had a history of disease that could increase the possibly of being infected by the virus.

This dataset was then further divided into two sets (training set and testing set) using the test-train split method. The system was trained on the basis of training set data and the accuracy of the ML classifier, and then evaluated over the testing set. Finally, the model was used to predict the probability of infection from the disease using new patient data in terms of positive or negative.

4.1 Supervised Machine Learning

Supervised Learning is a Machine Learning model that is built to give out predictions. This algorithm is performed by taking a labelled set of data as input and also known responses as output to learn the regression/classification model. It develops predictive models from classification algorithms and regression techniques.

Classification predicts discrete responses. Here, the algorithm labels by choosing two or more classes for each example. If it is done between two classes then it is called binary classification and if it is done between two or more classes then it is called multi- class classification. Applications of classification includes hand writing recognition, medical imaging etc.

Regression predicts continuous responses. Here, the algorithms returns a statistical value. For example, a set of data is collected such that the people are happy when considered the amount of sleep. Here, sleep and happy are both variables. Now, the analysis is done by making predictions. The types of popular regression techniques are:

- Linear regression.
- Logical regression.

4.2 Unsupervised Machine Learning

Unlike the supervised learning, there is no supervisor here and we only have input
data. Here, the basic aim is to find certain patterns in the data that occur more than others. According to the statistics, it is called density estimation. One of the methods for the density estimation is called clustering. Here, the input data is formed into clusters or groupings. Here, the assumptions are made such that the clusters are discovered which will match reasonably well with a classification. This is

a datadriven approach that works better when provided with sufficient data. For example, the movies in Netflix.com are suggested based on the principal of clustering of movies where several similar movies are grouped based on customer's recently watched movie list. It mostly discovers the unknown patterns in the data but most of the time these approximations are weak when compared with the supervised learning.

4.3 Semi-supervised Machine Learning

The name "semi-supervised learning" comes from the fact that the data used is between supervised and unsupervised learning. Semi-supervised algorithm has the tendency to learn both from labelled and unlabelled data. Semi-supervised machine learning gives high accuracy with a minimum annotation work. Semi-supervised machine learning uses mostly unlabelled data together combined with labelled data to give better classifiers. As less annotation work is enough to give good accuracy, humans have less work to do here.

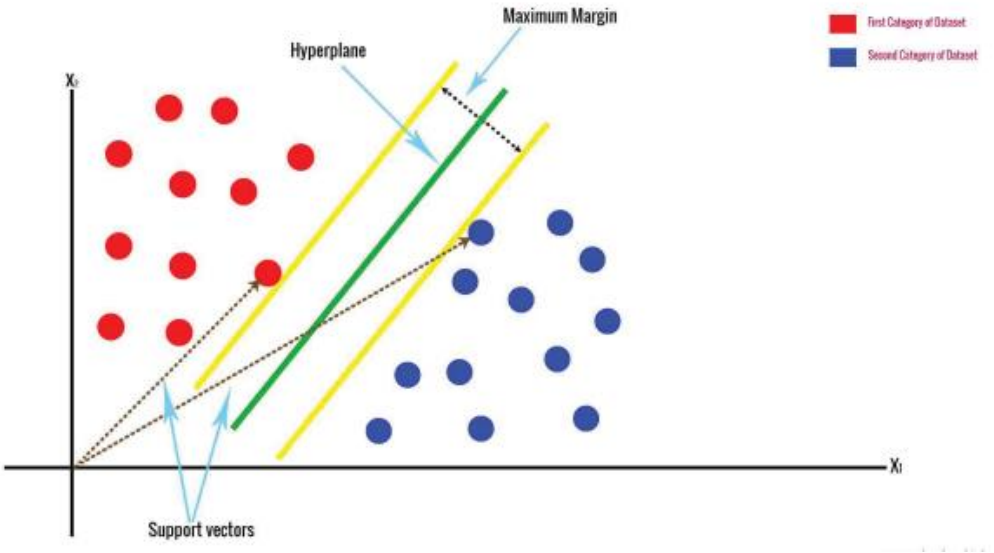
4.4 Algorithms and Support Vector Machines(SVM)

During our research, we have investigated three algorithms through which we have performed supervised classification.

Support Vector Machines performs classification by constructing N-dimensional hyper plane that separates the data into two categories. In SVM, the predictor variable is called an attribute and the transformed attribute is called a feature. Selecting the most suitable representative data is called feature selection. A set of

feature describing one case is called a vector.

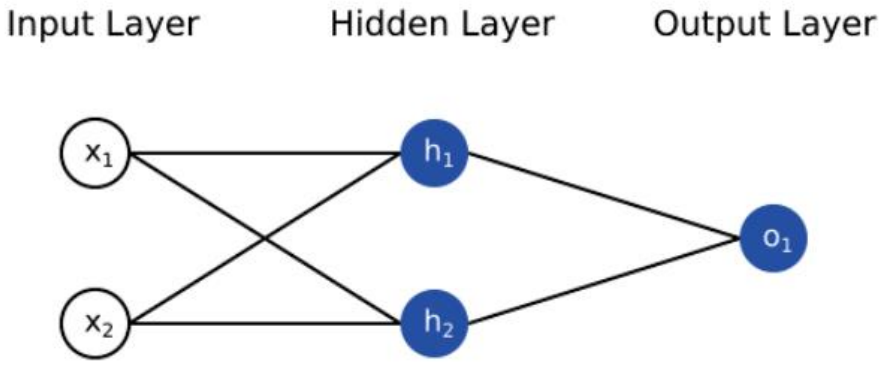
The ultimate goal of SVM modelling is to find the optimal hyper plane that separates the clusters where on one side of the plane there is target variable and on the other side of the plane other category. The vectors which are near the hyper plane are the support vectors.typical example of support vector machine is depicted.



4.4.1 Support Vector Machine

4.5 Artificial Neural Networks(ANN)

ANNs are an attempt, in the simplest way, to imitate the neural system of the human brain. The basic unit of ANN are neurons. A neuron is said to performfunctions on an input and produces an output [56]. Neurons combined together are called neural networks. Once the neural networks are formed, training of the data is started to minimize the error. In the end, an optimizing algorithm is used to further reduce the errors. The layered architecture of Artificial Neural Networks (ANNs) is represented in Figure

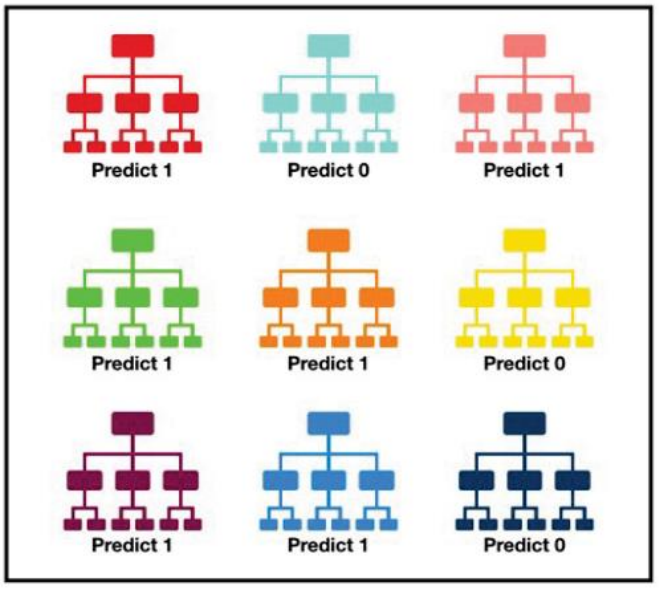


4.5.1 Neural Network

4.6 Random Forests(RF)

The random sampling and ensemble strategies utilized in RF enable it to achieve accurate predictions as well as better generalizations [40]. The random forests consists of large number of trees. The

higher the number of uncorrelated trees, the higher the accuracy . Random Forest classifiers can help filling some missing values.Prediction in Random Forests (RFs) is represented in Figure



4.6.1 Visualization of Random Forest making a prediction

Chapter-5

Software Description

5.1 Python

Python is a high level and effective general use programming language. It supports multi-paradigms. Python has a large standard library which provide tools suited to perform various tasks. Python is a simple, less-clustered language with extensive features and libraries. Different programming abilities are utilized for performing the experiment in our work. In this thesis, the following **python libraries** were used.

• **Pandas** - It is a python package that provides expressive data structures designed to work with both relational and labelled data. It is an open source python library that allows reading and writing data between data structures.

• **Numpy** - It is an open-source python package for scientific computing. Numpy also adds fast array processing capacities to python.

• **Matplotlib** - It is an open-source python package used for making plots and 2D representations. It integrates with python to give effective and interactive plots for visualization.

• **Tensorflow** - It is a mathematical open-source python library designed by Google Brain Team for Machine intelligence.

• **Sklearn** - It is an open-source python machine learning library designed to work alongside Numpy. It features various machine learning algorithms forclassification, clustering and regression.

For the development of the COVID-19 infection probability prediction project, the team will utilize the Jupyter software environment. Jupyter is an open-source web application that allows users to create and share documents containing live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, R, and Julia, making it suitable for data analysis, machine learning, and scientific computing tasks.

5.2 Key features of Jupyter software include:

Interactive Computing: Jupyter provides an interactive computing environment where users can write and execute code in a step-by-step manner. This enables iterative development and experimentation with data analysis techniques and machine learning models.

Notebook Interface: Jupyter notebooks offer a flexible and intuitive interface for organizing code, visualizations, and explanatory text in a single document. Users can create, edit, and run code cells interactively, facilitating collaborative work and reproducible research.

Rich Output Support: Jupyter notebooks support rich output formats, including HTML, LaTeX, Markdown, images, and interactive widgets. This allows for the creation of dynamic and visually appealing presentations, reports, and data visualizations within the same document.

Extensibility: Jupyter is highly extensible, with a vibrant ecosystem of third-party extensions and plugins available. Users can customize their environment with additional features and functionality to suit their specific needs and preferences.

Integration with Data Science Libraries: Jupyter seamlessly integrates with popular data science libraries and frameworks such as NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn, and TensorFlow. This enables users to leverage a rich ecosystem of tools and resources for data analysis, visualization, and machine learning model development.

Version Control and Sharing: Jupyter notebooks support version control systems such as Git, allowing users to track changes to their code and collaborate with team members effectively. Notebooks can also be shared publicly or privately via platforms like GitHub, JupyterHub, and Jupyter Notebook Viewer.

Deployment Options: Jupyter notebooks can be deployed locally on a user's machine or on cloud-based platforms such as JupyterHub, Google Colab, or Microsoft Azure Notebooks. This provides flexibility in terms of computing resources and scalability for handling large datasets and complex analyses.

By leveraging the capabilities of the Jupyter software environment, the project team aims to streamline the development and deployment of predictive models for estimating COVID-19 infection probabilities based on clinical symptoms. Jupyter's interactive and collaborative features will enable efficient exploration of data, model training, evaluation, and documentation, ultimately contributing to the project's success in addressing critical healthcare challenges.

5.3 Dataset and Used

Data Collection Data collection was an essential and protracted process. Regardless the field of research, accuracy of the data collection is essential to maintain cohesion. As the clinical information of patients was not publicly available, it was an inflexible and tedious process to collect the data. Various Hospitals and Health Institutes in Sweden and China were approached to get the most

accurate data but due to the present situation at hospitals with heavy inflow of patients with COVID-19, we couldn't get access to direct information. An intense search was conducted on various databases to gather open source clinical information of patients diagnosed with COVID-19.

The data set that was used to train the model to predict COVID-19 was gathered from an open source data shared by Yanyan Xu at a repository fig share. The data set contained information about hospitalized patients with COVID-19. It included demographic data, signs and symptoms, previous medical records, laboratory values that were extracted from electronic records. To train the model with equal records of patients with negative samples another data set from Kaggel repository was used[4]. The original data-set contained details of medications followed by the doctors to cure the disease. As our model doesn't require such data, those fields have been eliminated. The data-set is a combined multi-dimensional data. Some of the data gives information whether the patient is diagnosed with a particular disease in the past such as Renal Diseases, Digestive Diseases and other data contains precise clinical values obtained previously. It contains fields with textual data and some with precise values. Textual data was encoded with integer values for experimental setup.

Feature Number	Feature Name
1	Days from onset of symptoms to hospital admission
2	Gender
3	Clinical Classification
4	Age
5	Respiratory system disease
6	Comorbidity
7	Fatigue
8	Cardiovascular and cerebrovascular disease
9	Malignant tumor
10	Patient Condition
11	Digestive system disease
12	Renal disease
13	Chest tightness
14	Fever
15	Cough

5.3.1 Features in the dataset used.

5.4 Data Preprocessing

Data preprocessing is an important process in development of machine learning model. The data collected is often loosely controlled with out-of-range values, missing values, etc. Such data can mislead the result of the experiment.

• Imputation of missing values - In our data, missing values have been handled by using simple imputer from sklearn python package. The missing values are replaced by using mean strategy.

• Encoding Categorical Data - We used the package of OneHotEncoder in python, this package handles categorical data by one-hot or dummy encoding scheme.

5.5 Implementation

The experiment was conducted in the Python IDLE, which is a default integrated development and learning environment for python. The experiment was conducted in various phases that are mentioned below:

• After data collection, the patients data is divided into record sets containing 100 records, 150 records, 200 records, 250 records, 300 records, 355 records
respectively.

• A 5-fold cross validation technique is used to randomize the testing data-set to get accurate results. Experiment on each machine learning algorithm is
conducted by 5-fold cross validation with each of the record sets.

• The prediction accuracy of each algorithm at each record set is compared and evaluated for selecting the suitable algorithm for this dataset.

• A feature importance experiment is conducted to evaluate the importance of each attribute on the artificial classification task.

5.6 Algorithm Configurations

In this section, the configuration of the algorithms is mentioned. Changes made to the configuration of the algorithm can effect the results.

• Support Vector Machines: SVC (kernel = 'linear', random_state = 0)

• Artificial Neural Networks:

Layers:

ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

Compiling the ANN: ann.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

• Random Forests: RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)

5.7 Performance Metrics

It is an essential task to measure the performance of a machine learning model. As our model requires classification, we have used accuracy as the performance metric.

5.8 Accuracy

Accuracy is the metric used in this thesis for evaluation of the algorithms. It is the most used performance metric to evaluate classification techniques. This measure allows us to understand which model is best at identifying patterns in training set to give better predictions in the unknown test data-set.

Chapter -6

Observations/Results

The performance metric mentioned in Section 4.2.5 is utilized to evaluate the performance of the algorithms that were selected after the Literature Review. Three algorithms that were identified as the most suitable for the classification task to predict COVID-19 are:

• SVM (Support Vector Machine).

• RF (Random Forests).

• ANN (Artificial Neural Networks).

6.1 Experiment Results

Each of the above stated algorithms were trained with the data-set that was collected and results were interpreted. Performance of each algorithm was evaluated at different stages of training set. Each algorithm was trained with records sets containing 100 records, 150 records ,200 records, 250 records, 300 records, 355 records respectively. This experiment is performed to obtain which algorithm would be the most suitable for prediction of COVID-19. Also, as the data is split into smaller sets,we could also asses which algorithm would perform better with different datasets available.

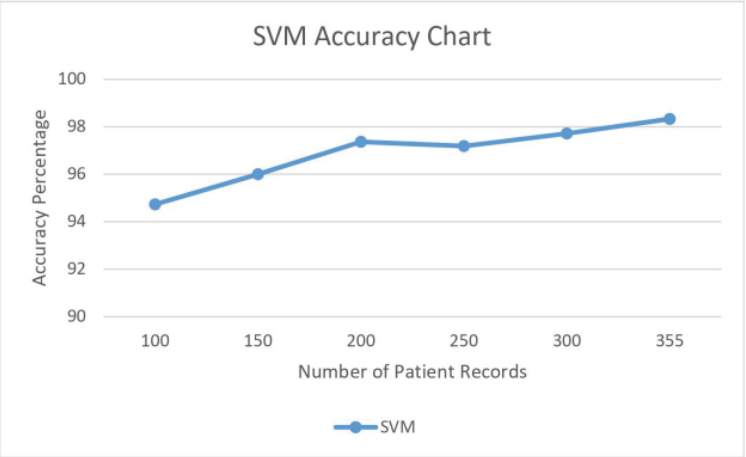
6.2 Support Vector Machine (SVM) Results

Support Vector Machine (SVM) algorithm is trained with each record sets to identify its accuracy at all stages. At all stages, the data was divided into training and test data by using k-fold cross validation (5-folds). SVM achieves an accuracy of 98.33%. represent the accuracy for every set of records achieved by Support VectorMachine (SVM) algorithm.

Number of Patient Records	Accuracy
100	94.73%
150	96%
200	97.36%
250	97.18%
300	97.71%
355	98.33%

6.2.1 Support Vector Machine (SVM) Accuracy Results

The classification accuracy of Support Vector Machine (SVM) at each record set can be clearly identified from the chart



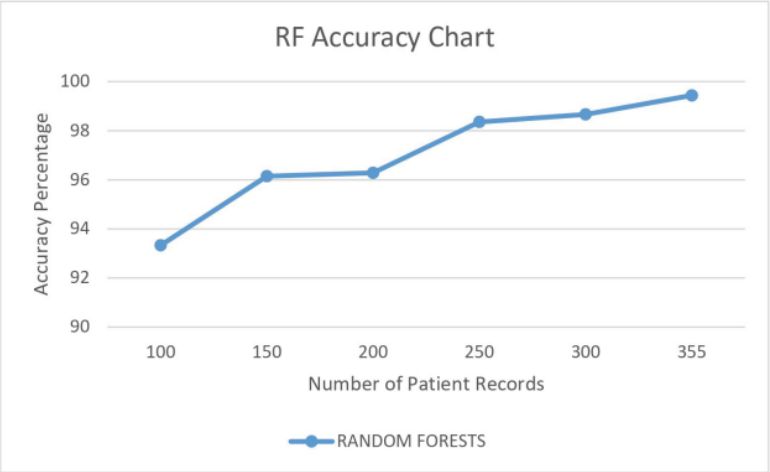
6.2.2 Support Vector Machine (SVM) Accuracy Chart

6.3 Random Forest (RF) Results

Random Forest (RF) algorithm is trained in a similar way with each records set to identify its accuracy at all stages. At all stages, the data was divided into training and test data by using k-fold cross validation (5-folds). RF achieves an accuracy of 99.44%. The classification accuracy of Random Forest (RF) algorithm for every set of records is represented.

Number of Patient Records	Accuracy
100	93.33%
150	96.15%
200	96.29%
250	98.36%
300	98.66%
355	99.44%

6.3.1 Random Forest (RF) Accuracy Results



6.3.2 Random Forest (RF) Accuracy Chart

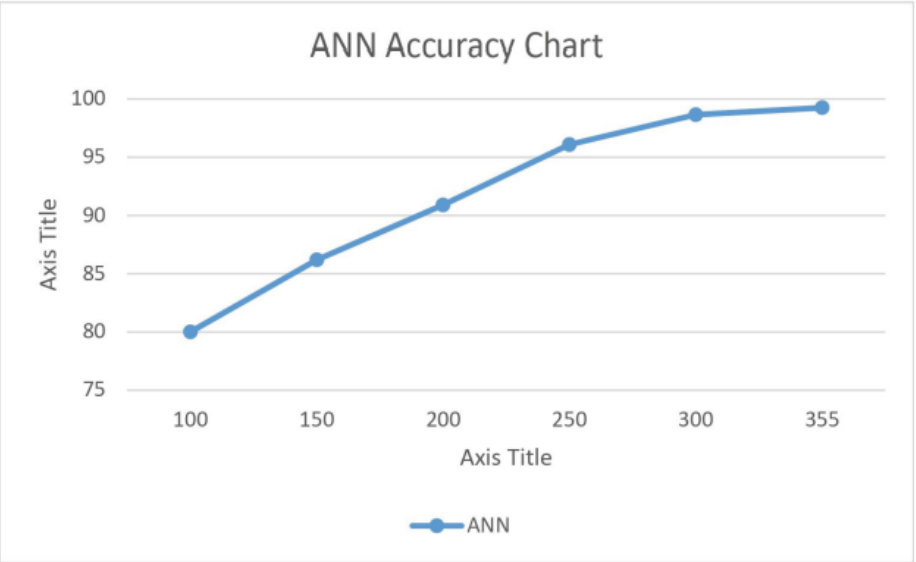
The classification accuracy of Random Forest (RF) at each record set can beidentified from the chart in Figure. The figure represents the change in accuracy while using each record set as training data.

6.4 Artificial Neural Networks (ANN) Results

Artificial Neural Networks (ANN) Algorithm is trained on data with record sets and tested. On implementing ANN Algorithm, it achieves an classification accuracy of 99.25%. The classification accuracy reported with each record set is tabulated in Table.

Number of Patient Records	Accuracy
100	80.00%
150	86.20%
200	90.90%
250	96.07%
300	98.65%
355	99.25%

6.4.1 Artificial Neural Networks (ANN) Accuracy Results



6.4.2 Artificial Neural Networks (ANN) Accuracy Chart

The accuracy of Artificial Neural Networks (ANN) with each record set is represented in Figure.

Chapter -7

Conclusion & Future Scope

Conclusion

In this research, a systematic literature review has been conducted to identify the suitable algorithm for prediction of COVID-19 in patients. There was no pure evidence found to summarize one algorithm as the suitable technique for prediction. Hence, a set of algorithms which include Support Vector Machine (SVM), Artificial Neural Networks (ANNs) and Random Forests (RF) were chosen. The selected algorithms were trained with the patient clinical information. To evaluate the accuracy of machine learning models, each algorithm is trained with record sets of varying number of patients. Using accuracy performance metric, the trained algorithms were assessed. After result analysis, Random Forest (RF) showed better prediction accuracy in comparison with both Support Vector Machine (SVM) and Artificial Neural

Networks (ANNs). The trained algorithms were also assessed to find the features that affect the prediction of COVID-19 in patients. There is a lot of scope for Machine Learning in Healthcare. For Future work, it is recommended to work on calibrated and ensemble methods that could resolve quirky problems faster with better outcomes than the existing algorithms. Also an AI-based application can be developed using various sensors and features to identify and help diagnose diseases. As healthcare prediction is an essential field for future, A prediction system that could find the possibility of outbreak of novel diseases that could harm mankind through socio-economic and cultural factor consideration can be developed

Future Scope

In response to the ongoing global COVID-19 pandemic, the development of accurate predictive models for infection risk assessment has emerged as a critical area of research. Leveraging machine learning techniques, our project aims to create a robust framework for estimating the probability of COVID-19 infection for individuals and populations. While our initial focus lies in developing an accurate model for current infection probabilities, the future scope of our project extends far beyond.

Moving forward, our endeavor includes real-time forecasting and monitoring capabilities, enabling timely interventions and resource allocation based on evolving infection dynamics. By integrating geospatial data, we seek to elucidate regional variations in infection probabilities, facilitating targeted containment strategies and healthcare resource allocation tailored to local needs. Furthermore, temporal analysis of infection trends will allow us to identify emerging patterns and

adapt intervention strategies accordingly, ensuring responsiveness to evolving epidemiological dynamics.

Continual refinement of our model will be achieved through feature engineering efforts, incorporating new data sources such as demographic information, vaccination rates, and mobility patterns to enhance predictive accuracy and reliability. Integration with existing healthcare systems will streamline the deployment of our model, providing healthcare practitioners with actionable insights for risk assessment and decision support.

Exploring ensemble learning approaches will further enhance the robustness of our predictions by leveraging diverse modeling techniques and data sources. Longitudinal studies will enable us to assess the long-term effectiveness of interventions, informing future pandemic preparedness strategies and policy decisions. Additionally, behavioral analysis will shed light on the influence of human behavior on infection probabilities, guiding the development of targeted public health campaigns aimed at promoting adherence to preventive measures.

.

Chapter -8

Reference

[1] Confirmed cases of Covid 19. Available from: https://www.covid19india.org/. (Accessed 26 April 2020).

[2] David J Cennimo Discusses Coronavirus Disease 2019 (COVID 19). Available from: https://emedicine.medscape.com/article/2500114-overview.
(Accessed 25 April 2020).

[3] Wang J., Liu Y., Wei Y., Xia J., Yu T., Zhang X., Zhang L. Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in Wuhan, China: a descriptive study. Lancet. 2020;395(10223):507–513. [PMC free article] [PubMed] [Google Scholar]

[4] Coronavirus Disease (COVID 19) Outbreak. Available from: http://www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/novel-coronavirus-2019-ncov.
(Accessed 25 April 2020).

[5] COVID 19 Article. Available from: https://www.newscientist.com/term/covid-19/.
(Accessed 30 April 2020).

[6] Erica Hersh Discusses How Long Is the Incubation Period for the Coronavirus?. Available from: https://www.healthline.com/health/coronavirus-incubation-period#incubation-period.
(Accessed 25 April 2020).

[7]Symptoms Coronavirus. Available from: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html. (Accessed 24 April 2020).

[8] Anulekha Ray Discusses ABOUT Coronavirus: India's biggest Concerns are COVID19 Patients with No Symptoms .Available from: https://www.livemint.com/news/india/coronavirus-india-s-biggest-concerns-are-covid-19-patients-with-no-symptoms-11587533159071.html.
(Accessed 26 April 2020).

[9] Teena Thacker Discusses About No Symptoms in 80% of COVID Cases Raises Concern. Available from: https://economictimes.indiatimes.com/industry/healthcare/biotech/healthcare/no-symptoms-in-80-of-covid-cases-raise-concerns/articleshow/75260387.cms?from=mdr.
(Accessed 26 April 2020).

[10] Praveen Duddu Discusses About COVID 19 Coronavirus: Top Ten Most Affected Countries. Available from: https://www.pharmaceutical-technology.com/features/covid-19-coronavirus-top-ten-most-affected-countries/.

[11] Covid 19 Cases in China. Available from:
https://www.worldometers.info/coronavirus/country/china/.
(Accesses 26 April 2020).
[12] V. Wang, Coronavirus Epidemic Keeps Growing, But Spread in China Slows. New York Times. https://www.nytimes.com/2020/02/18/world/asia/china-coronavirus-cases.html?referringSource=articleShare.
(Accessed 26 April 2020).
[13] Covid 19 cases in Italy. Available from:
https://www.worldometers.info/coronavirus/country/italy/.
(Accessed 26 April 2020).
[14] Covid 19 cases in Spain. Available from:
https://www.worldometers.info/coronavirus/country/Spain/.
(Accessed 26 April 2020).
[15] Covid 19 cases in US. Available from:
https://www.worldometers.info/coronavirus/country/US/.
(Accesses 26 April 2020).
[16] Covid 19 cases in Germany. Available from:
https://www.worldometers.info/coronavirus/country/Germany/.
(Accesses 26 April 2020).
[17] Covid 19 cases in France. Available from:
https://www.worldometers.info/coronavirus/country/France/.
(Accesses 26 April 2020).
[18] Covid 19 cases in Iran. Available from:
https://www.worldometers.info/coronavirus/country/Iran/.
(Accesses 26 April 2020).
[19] Covid 19 cases in Turkey. Available from:
https://www.worldometers.info/coronavirus/country/Turkey/.
(Accesses 26 April 2020).
[20] Gavin Edwards discusses about Machine Learning: An Introduction. Available from:
https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0.
(Accessed 27 April 2020).

APPENDIX

Projrct Source Code

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
import time
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report,confusion_matrix, roc_auc_score, mean_squared_error,r2_score, accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn import tree
from sklearn.naive_bayes import GaussianNB
from sklearn import svm

covid_data = pd.read_csv("Covid Dataset.csv")
covid_data
covid_data.shape
covid_data.columns
covid_data.info()
covid_data.describe().T
covid_data.head()
# create a table with data missing
missing_values=covid_data.isnull().sum() # missing values
```

```
percent_missing = covid_data.isnull().sum()/covid_data.shape[0]*100 # missing value %

value = {
   'missing_values ':missing_values,
   'percent_missing %':percent_missing
}
frame=pd.DataFrame(value)
frame
sns.heatmap(covid_data.isnull(),yticklabels=False,cbar=False,cmap='Pastel1')
ax = sns.countplot(x='COVID-19',data=covid_data, palette="PuRd")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
covid_data["COVID19"].value_counts().plot.pie(explode=[0.1,0.1],autopct='%1.1f%%',shadow=True)
plt.title('Covid Positive');
ax = sns.countplot(x='Breathing Problem',data=covid_data, palette="Set1")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Breathing Problem',hue='COVID-19',data=covid_data, palette="Set2")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Fever',data=covid_data)
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Fever',hue='COVID-19',data=covid_data, palette="PuRd")
for p in ax.patches:
```

```
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Dry Cough',hue='COVID-19',data=covid_data, palette="Set1")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Sore throat',hue='COVID-19',data=covid_data, palette="Set2")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Abroad travel',hue='COVID-19',data=covid_data)
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Contact with COVID Patient',hue='COVID-19',data=covid_data, palette="PuRd")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
ax = sns.countplot(x='Attended Large Gathering',hue='COVID-19',data=covid_data, palette="Set1")
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.2, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
e=LabelEncoder()
covid_data['Breathing Problem']=e.fit_transform(covid_data['Breathing Problem'])
covid_data['Fever']=e.fit_transform(covid_data['Fever'])
covid_data['Dry Cough']=e.fit_transform(covid_data['Dry Cough'])
covid_data['Sore throat']=e.fit_transform(covid_data['Sore throat'])
```

```
covid_data['Running Nose']=e.fit_transform(covid_data['Running Nose'])
covid_data['Asthma']=e.fit_transform(covid_data['Asthma'])
covid_data['Chronic Lung Disease']=e.fit_transform(covid_data['Chronic Lung Disease'])
covid_data['Headache']=e.fit_transform(covid_data['Headache'])
covid_data['Heart Disease']=e.fit_transform(covid_data['Heart Disease'])
covid_data['Diabetes']=e.fit_transform(covid_data['Diabetes'])
covid_data['Hyper Tension']=e.fit_transform(covid_data['Hyper Tension'])
covid_data['Abroad travel']=e.fit_transform(covid_data['Abroad travel'])
covid_data['Contact with COVID Patient']=e.fit_transform(covid_data['Contact with COVID Patient'])
covid_data['Attended Large Gathering']=e.fit_transform(covid_data['Attended Large Gathering'])
covid_data['Visited Public Exposed Places']=e.fit_transform(covid_data['Visited Public Exposed Places'])
covid_data['Family working in Public Exposed Places']=e.fit_transform(covid_data['Family working in Public Exposed Places'])
covid_data['Wearing Masks']=e.fit_transform(covid_data['Wearing Masks'])
covid_data['Sanitization from Market']=e.fit_transform(covid_data['Sanitization from Market'])
covid_data['COVID-19']=e.fit_transform(covid_data['COVID-19'])
covid_data['Dry Cough']=e.fit_transform(covid_data['Dry Cough'])
covid_data['Sore throat']=e.fit_transform(covid_data['Sore throat'])
covid_data['Gastrointestinal ']=e.fit_transform(covid_data['Gastrointestinal '])
covid_data['Fatigue ']=e.fit_transform(covid_data['Fatigue '])
covid_data.head()
covid_data
covid_data.hist(figsize=(20,15));
print(covid_data['Wearing Masks'].value_counts())
sns.countplot(x='Wearing Masks',data=covid_data)
print(covid_data['Sanitization from Market'].value_counts())
sns.countplot(x='Sanitization from Market',data=covid_data)
covid_data=covid_data.drop('Wearing Masks',axis=1)
covid_data=covid_data.drop('Sanitization from Market',axis=1)
covid_data.columns
plt.figure(figsize=(25,20))
sns.heatmap(covid_data.corr(), annot=True, cmap="PuRd")
x=covid_data.drop('COVID-19',axis=1)
```

```
y=covid_data['COVID-19']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.20, random_state = 101)
accuracies = {}
algo_time={}
r2_scores={}
mean_squared_errors={}
roc_auc_scores={}
def print_performance2(yt,clf,clf_name):
    y_pred=clf.predict(x_test)
    roc_auc_scores[clf_name]=roc_auc_score(yt,y_pred)*100
    mean_squared_errors[clf_name]=mean_squared_error(yt,y_pred)*100
    r2_scores[clf_name]=r2_score(yt,y_pred)*100
    accuracies[clf_name]=clf.score(x_train,y_train)*100
    print('ROC_AUC value :',roc_auc_scores[clf_name],"%",'\n')
    print("Mean Squared Error :",mean_squared_errors[clf_name],"%")
    print("\nR2 score is :",r2_scores[clf_name],"%")
    print("\nAccuracy Score :",accuracies[clf_name],"%")
    print('\nClassification Report : ','\n',classification_report(yt,y_pred))

    confusionmatrix=confusion_matrix(yt,y_pred)

    fig, ax = plt.subplots(figsize=(3, 3))
    ax.matshow(confusionmatrix, cmap=plt.cm.Blues, alpha=0.3)
    for i in range(confusionmatrix.shape[0]):
        for j in range(confusionmatrix.shape[1]):
            ax.text(x=j, y=i,s=confusionmatrix[i, j], va='center', ha='center', size='xx-large')

    plt.xlabel('Predictions', fontsize=18)
    plt.ylabel('Actuals', fontsize=18)
    plt.title('Confusion Matrix', fontsize=18)
## LOGISTIC REGRESSION

print("LOGISTIC REGRESSION")
start = time.time()
lr = LogisticRegression()
```

```
lr.fit(x_train, y_train)
end = time.time()

print_performance2(y_test,lr,'LOGISTIC REGRESSION')
#acc = lr.score(x_train, y_train)*100
#accuracies['LOGISTIC REGRESSION'] = acc
algo_time['LOGISTIC REGRESSION']=end-start
## K-NEAREST NEIGHBOURS
start = time.time()
knn = KNeighborsClassifier()
# assigning the dictionary of variables whose optimium value is to be retrieved
param_grid = {'n_neighbors' : np.arange(1,50)}
knn_cv = GridSearchCV(knn, param_grid, cv=5)
# training the model with the training data and best parameter
knn_cv.fit(x_train,y_train)
end=time.time()
algo_time['K-NEAREST NEIGHBOURS']=end-start
# finding out the best parameter chosen to train the model
print("The best paramter we have is: {}" .format(knn_cv.best_params_))

# finding out the best score the chosen parameter achieved
print("The best score we have achieved is: {}" .format(knn_cv.best_score_))
print("K-NEAREST NEIGHBOURS")
print_performance2(y_test,knn_cv,'K-NEAREST NEIGHBOURS')
#acc = knn_cv.score(x_train, y_train)*100
#accuracies['K-NEAREST NEIGHBOURS'] = acc
## RANDOM FOREST
rf_start=time.time()
rfc=RandomForestClassifier(random_state=42)
param_grid = {
    'n_estimators': [200, 500],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
}
```

```
CV_rfc = GridSearchCV(estimator=rfc, param_grid=param_grid, cv= 5)
CV_rfc.fit(x_train, y_train)
rf_end=time.time()
algo_time['RANDOM FOREST TREE']=rf_end-rf_start
# finding out the best parameter chosen to train the model
print("The best paramter we have is: {}" .format(CV_rfc.best_params_))

# finding out the best score the chosen parameter achieved
print("The best score we have achieved is: {}" .format(CV_rfc.best_score_*100))
print("RANDOM FOREST TREE")
print_performance2(y_test,CV_rfc,'RANDOM FOREST TREE')
#acc = CV_rfc.score(x_train, y_train)*100
#accuracies['RANDOM FOREST TREE'] = acc
## GRADIENT BOOSTING CLASSIFIER
### Decision Tree
### Naive bayes
colors = ["purple", "green", "orange", "blue", "red", "yellow", "magenta"]

sns.set_style("whitegrid")
plt.figure(figsize=(10,5))
plt.yticks(np.arange(0,100,10))
plt.ylabel("Accuracy %")
plt.xlabel("Algorithms")
plt.xticks(rotation=90)
ax = sns.barplot(x=list(accuracies.keys()), y=list(accuracies.values()), palette=colors)
plt.show()
!conda install -c plotly plotly
!pip install colorama
import plotly.express as px
fig = px.bar(x=list(accuracies.keys()), y=list(accuracies.values()))
fig.update_traces(marker_color='teal', marker_line_color='rgb(8,48,107)', marker_line_width=1.5)
fig.update_layout(title="Accuracy Comparision", xaxis_title="Model", yaxis_title="Accuracy")
fig.show()
fig = px.bar(x=list(algo_time.keys()), y=list(algo_time.values()))
fig.update_traces(marker_color='teal', marker_line_color='rgb(8,48,107)', marker_line_width=1.5)
```

```
fig.update_layout(title="Algorithm Time Comparision", xaxis_title="Model", yaxis_title="")
fig.show()
fig = px.bar(x=list(r2_scores.keys()), y=list(r2_scores.values()))
fig.update_traces(marker_color='teal', marker_line_color='rgb(8,48,107)', marker_line_width=1.5)
fig.update_layout(title="R2 Score Comparision", xaxis_title="Model", yaxis_title="R2 Scores")
fig.show()
fig = px.bar(x=list(mean_squared_errors.keys()), y=list(mean_squared_errors.values()))
fig.update_traces(marker_color='teal', marker_line_color='rgb(8,48,107)', marker_line_width=1.5)
fig.update_layout(title="Mean Squared Error Comparision", xaxis_title="Model",
yaxis_title="Mean Squared Error")
fig.show()
fig = px.bar(x=list(roc_auc_scores.keys()), y=list(roc_auc_scores.values()))
fig.update_traces(marker_color='teal', marker_line_color='rgb(8,48,107)', marker_line_width=1.5)
fig.update_layout(title="ROC Score Comparision", xaxis_title="Model", yaxis_title="ROC
Scores")
fig.show()
import plotly.graph_objects as go
Algos=list(roc_auc_scores.keys())

fig = go.Figure(data=[
    go.Bar(name='Accuracies', x=Algos, y=list(accuracies.values())),
    go.Bar(name='R2 scores', x=Algos, y=list(r2_scores.values())),
    go.Bar(name='Mean Squared Errors', x=Algos, y=list(mean_squared_errors.values())),
    go.Bar(name='ROC Auc Scores', x=Algos, y=list(roc_auc_scores.values()))
])
# Change the bar mode
fig.update_layout(barmode='group')
fig.show()
from sklearn.metrics import roc_curve
plt.figure(figsize=(25,16))
# Logistic Regression Classification
Y_predict1_proba = lr.predict_proba(x_test)
Y_predict1_proba = Y_predict1_proba[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, Y_predict1_proba)
plt.subplot(441)
```

```
plt.plot([0,1],[0,1],'k--')
plt.plot(fpr,tpr, label='ANN')
plt.xlabel('fpr')
plt.ylabel('tpr')
plt.title('ROC Curve Logistic Regression')
plt.grid(True)

Y_predict1_proba = knn_cv.predict_proba(x_test)
Y_predict1_proba = Y_predict1_proba[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, Y_predict1_proba)
plt.subplot(442)
plt.plot([0,1],[0,1],'k--')
plt.plot(fpr,tpr, label='ANN')
plt.xlabel('fpr')
plt.ylabel('tpr')
plt.title('ROC Curve K-Nearest Neighbours')
plt.grid(True)

Y_predict1_proba = CV_rfc.predict_proba(x_test)
Y_predict1_proba = Y_predict1_proba[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, Y_predict1_proba)
plt.subplot(443)
plt.plot([0,1],[0,1],'k--')
plt.plot(fpr,tpr, label='ANN')
plt.xlabel('fpr')
plt.ylabel('tpr')
plt.title('ROC Curve Random Forest Tree')
plt.grid(True)

plt.subplots_adjust(top=2, bottom=0.08, left=0.10, right=1.4, hspace=0.45, wspace=0.45)
plt.show()
import colorama
from colorama import Fore

print("COVID PREDICTION BASED ON ML ALGORITHMS")
```

```
print("Enter 1 for Yes and 0 for No")
Breathing_Problem = int(input("Does the patient have breathing problem ? "))
Fever = int(input("Does the patient have fever ? "))
Dry_Cough = int(input("Does the patient have dry cough ? "))
Sore_throat = int(input("Does the patient have sore throat ? "))
Running_Nose = int(input("Does the patient have running nose ? "))
Asthma = int(input("Does the patient have any record of asthma ? "))
Chronic_Lung_Disease = int(input("Does the patient have any records of chronic lung disease ? "))
Headache = int(input("Is the patient having headche ? "))
Heart_Disease = int(input("Does the patient have any record of any heart disease ? "))
Diabetes = int(input("Does the patient have diabetes ? "))
Hyper_Tension = int(input("Does the patient have hyper tension ? "))
Fatigue = int(input("Does the patient experience fatigue ? "))
Gastrointestinal = int(input("Does the patient have any gastrointestinal disorders ? "))
Abroad_travel = int(input("Has the patient travelled abroad recently ? "))
Contact_with_COVID_Patient = int(input("Was the patient in contact with a covid patient recently
? "))
Attended_Large_Gathering = int(input("Did the patient attend any large gathering event recently ?
"))
Visited_Public_Exposed_Places = int(input("Did the patient visit any public exposed places recently
? "))
Family_working_in_Public_Exposed_Places = int(input("Does the patient have any family member
working in public exposed places ? "))

patient =
[[Breathing_Problem,Fever,Dry_Cough,Sore_throat,Running_Nose,Asthma,Chronic_Lung_Disea
se,Headache,Heart_Disease,Diabetes,Hyper_Tension,Fatigue,Gastrointestinal,Abroad_travel,Cont
act_with_COVID_Patient,Attended_Large_Gathering,Visited_Public_Exposed_Places,Family_wo
rking_in_Public_Exposed_Places]]
result = knn_cv.predict(patient)
print("\nResults : ",result)
if result == 1:
   print(Fore.RED + 'You may be affected with COVID-19 virus! Please get RTPCR test ASAP and
stay in Quarantine for 14 days!')
   print()
```

else :

print(Fore.GREEN + 'You do not have any symptoms of COVID-19. Stay home! Stay safe!

print()

In [ ]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
import time
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report,confusion_matrix, roc_auc_score, mean_squared_error,r2_score, accuracy_sc
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn import tree
from sklearn.naive_bayes import GaussianNB
from sklearn import svm
```

In [ ]:
```
covid_data = pd.read_csv("Covid Dataset.csv")
```

In [ ]:
```
covid_data
```

Out[ ]:

	Breathing Problem	Fever	Dry Cough	Sore throat	Running Nose	Asthma	Chronic Lung Disease	Headache	Heart Disease	Diabetes	Hyper Tension	Fatigue	Gastrointestinal	AI
0	Yes	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Yes	Yes	
1	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No	Yes	No	
2	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	
3	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes	No	No	No	
4	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
5429	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes	
5430	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	
5431	Yes	Yes	Yes	No	No	No	No	No	Yes	No	Yes	No	No	
5432	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	No	No	No	No	
5433	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	

5434 rows × 21 columns

In [ ]:
```
covid_data.shape
```

Out[ ]:
```
(5434, 21)
```

In [ ]:
```
covid_data.columns
```

Out[ ]:
```
Index(['Breathing Problem', 'Fever', 'Dry Cough', 'Sore throat',
       'Running Nose', 'Asthma', 'Chronic Lung Disease', 'Headache',
       'Heart Disease', 'Diabetes', 'Hyper Tension', 'Fatigue ',
       'Gastrointestinal ', 'Abroad travel', 'Contact with COVID Patient',
       'Attended Large Gathering', 'Visited Public Exposed Places',
       'Family working in Public Exposed Places', 'Wearing Masks',
       'Sanitization from Market', 'COVID-19'],
      dtype='object')
```

In [ ]:
```
covid_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5434 entries, 0 to 5433
Data columns (total 21 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Breathing Problem                       5434 non-null   object
 1   Fever                                   5434 non-null   object
 2   Dry Cough                               5434 non-null   object
 3   Sore throat                             5434 non-null   object
 4   Running Nose                            5434 non-null   object
 5   Asthma                                  5434 non-null   object
 6   Chronic Lung Disease                    5434 non-null   object
 7   Headache                                5434 non-null   object
 8   Heart Disease                           5434 non-null   object
 9   Diabetes                                5434 non-null   object
 10  Hyper Tension                           5434 non-null   object
 11  Fatigue                                 5434 non-null   object
```

In [ ]:
```
covid_data.describe().T
```

Out[ ]:

	count	unique	top	freq
Breathing Problem	5434	2	Yes	3620
Fever	5434	2	Yes	4273
Dry Cough	5434	2	Yes	4307
Sore throat	5434	2	Yes	3953
Running Nose	5434	2	Yes	2952
Asthma	5434	2	No	2920
Chronic Lung Disease	5434	2	No	2869
Headache	5434	2	Yes	2736
Heart Disease	5434	2	No	2911
Diabetes	5434	2	No	2846
Hyper Tension	5434	2	No	2771
Fatigue	5434	2	Yes	2821
Gastrointestinal	5434	2	No	2883
Abroad travel	5434	2	No	2983
Contact with COVID Patient	5434	2	Yes	2726
Attended Large Gathering	5434	2	No	2924
Visited Public Exposed Places	5434	2	Yes	2820
Family working in Public Exposed Places	5434	2	No	3172

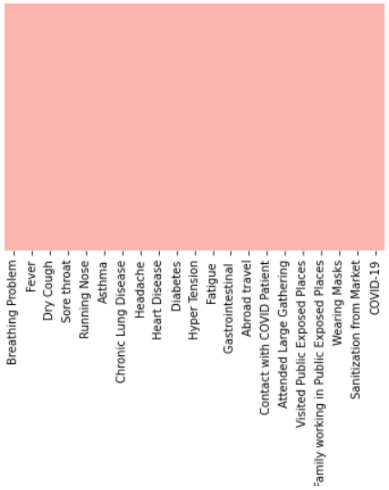
In [ ]:
```
# create a table with data missing
missing_values=covid_data.isnull().sum() # missing values

percent_missing = covid_data.isnull().sum()/covid_data.shape[0]*100 # missing value %

value = {
    'missing_values ':missing_values,
    'percent_missing %':percent_missing
}
frame=pd.DataFrame(value)
frame
```

Out[ ]:

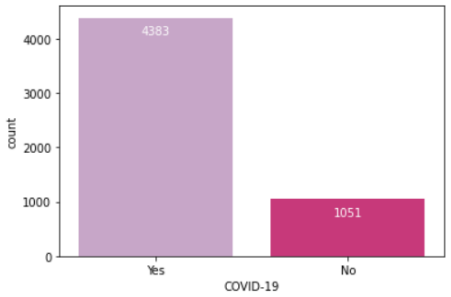
	missing_values	percent_missing %
Breathing Problem	0	0.0
Fever	0	0.0
Dry Cough	0	0.0
Sore throat	0	0.0
Running Nose	0	0.0
Asthma	0	0.0
Chronic Lung Disease	0	0.0
Headache	0	0.0
Heart Disease	0	0.0
Diabetes	0	0.0
Hyper Tension	0	0.0
Fatigue	0	0.0
Gastrointestinal	0	0.0
Abroad travel	0	0.0
Contact with COVID Patient	0	0.0
Attended Large Gathering	0	0.0
Visited Public Exposed Places	0	0.0
Family working in Public Exposed Places	0	0.0
Wearing Masks	0	0.0

In [ ]:
```
sns.heatmap(covid_data.isnull(),yticklabels=False,cbar=False,cmap='Pastel1')
```

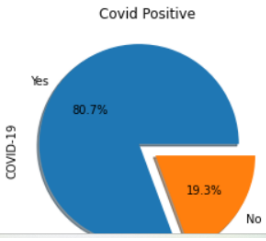
Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9d5f2a7a10>
```



In [ ]:
```
ax = sns.countplot(x='COVID-19',data=covid_data, palette="PuRd")
for p in ax.patches:
     ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()+100), ha='center', va='top', color='white', size=10)
plt.show()
```



In [ ]:
```
covid_data["COVID-19"].value_counts().plot.pie(explode=[0.1,0.1],autopct='%1.1f%%',shadow=True)
plt.title('Covid Positive');
```



In [ ]:
```
covid_data.hist(figsize=(20,15));
```