

# **Genomic Sequence Analysis Using Machine Learning**

## **An Engineering Project in Community Service**

### **Phase – II Report**

*Submitted by*

- 1. 19BCE10054 Shreyash Mall**
- 2. 19BCE10177 Ananya Singh**
- 3. 19BCE10061 Mudit Sorikh**
- 4. 19BCE10336 K Aniket Prusty**
- 5. 19BOE10081 Chetna Bisen**
- 6. 19BAI10112 Abhishek J Nair**
- 7. 19MIM10048 Pranay Pratap Singh**
- 8. 19MIM10113 Varnika Singh**

*in partial fulfillment of the requirements for the degree of*

*Bachlore of Engineering and Technology*



**VIT<sup>®</sup>**  
**BHOPAL**  
[www.vitbhopal.ac.in](http://www.vitbhopal.ac.in)

**VIT Bhopal University**  
**Bhopal**  
**Madhya Pradesh**

**February, 2022**



### **Bonafide Certificate**

Certified that this project report titled **“Genomic Sequence Analysis Using Machine Learning”** is the bonafide work of **“19BCE10054 Shreyash Mall, 19BCE10177 Ananya Singh, 19BCE10061 Mudit Sorikh, 19BCE10336 K Aniket Prusty, 19BOE10081 Chetna Bisen, 19BAI10112 Abhishek J Nair, 19MIM10048 Pranay Pratap Singh, 19MIM10113 Varnika Singh”** who carried out the project work under my supervision.

This project report (Phase II) is submitted for the Project Viva-Voce examination held on 01/03/22

**Supervisor**

**Comments & Signature ( Reviewer 1)**

**Comments & Signature ( Reviewer 2)**

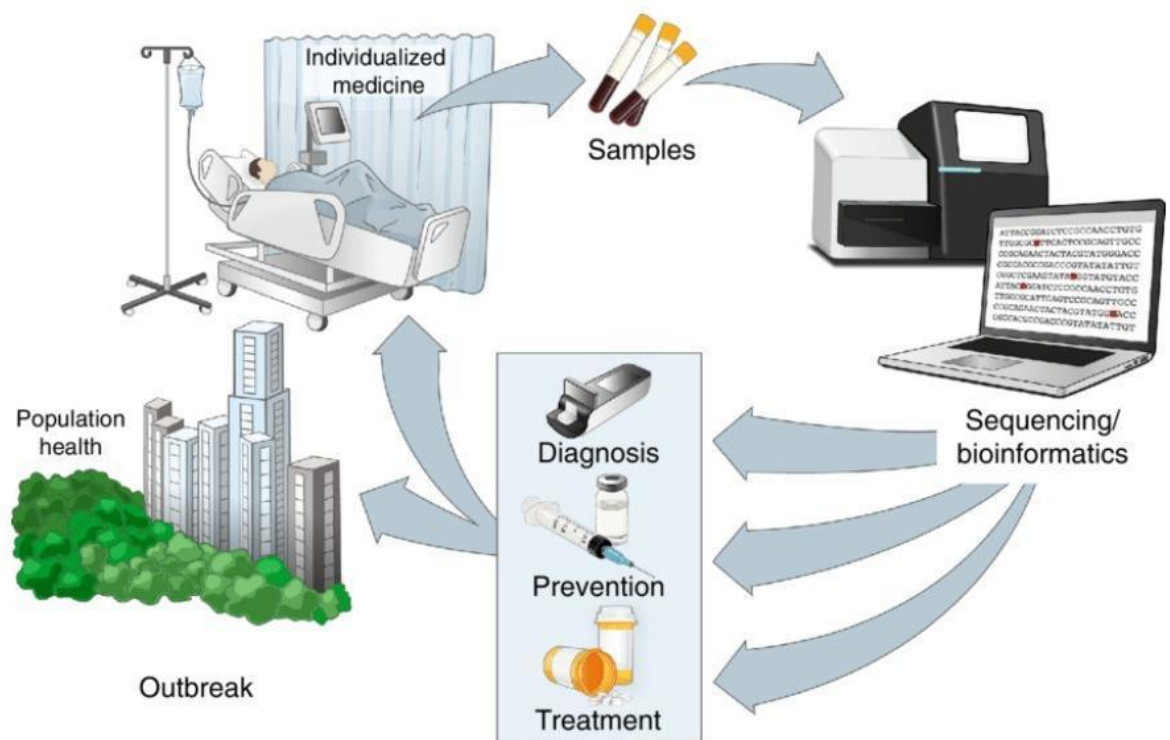
## INTRODUCTION

Bioinformatics is an interdisciplinary field that merges molecular biology and genetics with computer science, mathematics, and statistics. To solve data-intensive, large-scale biological concerns, computational techniques are applied. The most common problems are molecular modeling of biological processes and drawing inferences from data. Typically, a bioinformatics solution comprises the following steps: Statistical information may be derived from biological data. Construct a computational model. Resolve a problem through computational modeling. Put a computer algorithm through its paces and see how it fared.

Machine learning has grown in popularity in bioinformatics and computational biology. Recently, its popularity has grown, and it is now employed in a variety of sectors such as genetic sequencing, categorization, and many others.

A genome is a full collection of genetic instructions for an organism. Each genome includes all of the information required to construct the organism and allow it to grow and develop.

A laboratory technique for determining the whole genetic make-up of an organism or cell type. This approach may be used to detect changes in specific regions of the genome. These modifications may aid scientists in understanding how certain illnesses, like cancer, develop. Genomic sequencing results may potentially be utilized to diagnose and treat illness.



Machine learning (ML) is the study of the autonomous learning of computers that are not explicitly programmed. It is used in bioinformatics and focuses on producing data-driven predictions. This technique enables algorithms to make complex predictions on massive datasets.

### 1.1 Motivation

Prior to the introduction of machine learning, bioinformatics algorithms had to be constructed by hand, which proved difficult for problems like protein structure prediction. Deep learning and other machine learning algorithms may be able to identify elements of data sets without requiring the programmer to define them individually.

#### Artificial Intelligence (ML)

ML is a subfield of AI that uses statistical learning techniques to develop intelligent systems. ML systems may learn and evolve on their own without being explicitly written. ML is used in music and movie streaming businesses' recommendation systems. Machine learning algorithms are classified into three types: supervised, unsupervised, and reinforcement learning.

#### Intensive Learning (DL)

This AI subset is based on an approach influenced by how the human brain processes information. It is related to learning by example. DL systems

help a computer model forecast and categorize data by filtering incoming data across layers. Deep Learning handles data in the same manner as the human brain. It is used in technologies like self-driving cars. Deep learning network designs are classified into three types: convolutional neural networks, recurrent neural networks, and recursive neural networks.

The algorithm may then learn to combine low-level characteristics to produce more abstract features, and so on. This multi-layered approach, when correctly taught, enables such computers to make sophisticated predictions. These methods vary from earlier computational biology approaches in that, while they make use of existing datasets, they do not allow the data to be analyzed and appraised in novel ways. The volume and variety of biological datasets accessible has grown substantially in recent years.

## **1.2 Objective**

Gene sequencing using machine learning that can help identify various genetic defects and mutations in an organism.

## **Existing Work / Literature Review**

**PlasClass (PLOS ONE, 2020)** uses logistic regression on k-mer frequency vectors to determine whether they are derived from a plasmid sequence or a chromosomal region. This is a tool for binary classification.

### **PlasFlow (published 2018, Nucleic Acid Research)**

This tool determines phylum classification and if a given contig is a plasmid or a chromosome. On top of k-mer frequency vectors, a neural network is used.

**MetaBCC-LR (Bioinformatics, 2020)** uses t-distributed Stochastic Neighbour Embedding (t-SNE) to dimension decrease genomic long reads in order to execute trimer vector binning of metagenomic data.

**Whole genome genotyping** Typing of Human leukocyte antigen (HLA) by HT-NGS: A three step procedure of HLA typing was introduced. In first step, HLA-A, -B, -C, -DRB1, and -DQB1 were amplified with long-range PCR. In second step, amplicons were sequenced using the 454 GS-FLX platform. In third step, sequencing data were analyzed with Assign-NG software. Lind et al. 2010

HT-NGS in prenatal diagnosis tests: A comprehensive review on impact of HT NGS on prenatal diagnosis tests. Raymond et al. 2010

In utero disease screening: A new study demonstrates the feasibility of genome-wide fetal genotyping using non-invasive next-generation sequencing of the mother's blood Burgess 2011

### **De Novo assembling and re-assembling of the human genome.**

Re-sequencing of genome by DNA pools: study proposed a novel statistical approach, CRISP (Comprehensive Read analysis for Identification of SNPs from Pooled sequencing] that is able to identify both rare and common

variants. The CRISP approach can detect 80-85% of SNPs identified using individual sequencing while achieving a low false discovery rate (3-5%).

Bansal 2010

Re-sequencing of genome and HT-NGS platform: study evaluated the comparative performance of the Illumina Genome Analyzer and Roche 454 GS FLX for the re-sequencing of 16 genes associated with hypertrophic cardiomyopathy (HCM). Study concluded the feasibility of combining LR-PCR with NGS platforms for targeted re-sequencing of HCM-associated genes. Dames et al. 2010

De novo assembling of the human genome: study proposed a novel method for de novo assembly of human genomes from short read sequences. Method successfully assembled N50 contig size of 7.4 and 5.9 kilobases (kb) and scaffold of 446.3 and 61.9 kb of Asian and African human genome. Li et al. 2010a

Assembling of the human genome: A comprehensive review on recent development of software packages in analyzing new generation sequencing data. Nagarajan and Pop 2010

HT-NGS in ancient genome research: study sequenced the complete genome of a 4,000-year-old human with 20-fold coverage which providing a fresh look at human population history. Shapiro and Hofreiter 2010

**Epigenetics** DNA methylation and HT-NGS: study compared two different bisulfite conversion whole methylome sequencing methods using NGS SOLiD platform. Bormann et al. 2010

HT-NGS in epigenomics: study presented the methylation detection reagents and their application to microarray and sequencing platforms. Study also proposed an international coordination to standardize methylome platforms and to create a full repository of methylome maps from tissues and unique cell types. Fouse et al. 2010

Profiling genome methylation patterns at single-base resolution: Study provides new insights into the conservation and divergence of DNA methylation in eukaryotes and their regulation of gene expression.

Bhaijee et al. 2011

Database for whole genome methylation maps at single-cytosine resolution: NGS methylation database (NGSmethDB: <http://bioinfo2.ugr.es/NGSmethDB/gbrowse/>) for human, mouse and Arabidopsis genome, comprised of wide range of tissues including the differential tissue methylation or the changes occurring along pathological conditions. Hackenberg et al. 2011

**ChIP-seq** Study of gene expression regulation through HT-NGS: Study indentified both coding and regulatory regions of PPARG gene-novel nucleotide variations and haplotypes associated to human diseases by DNA-seq, defining a PPAR $\gamma$  binding map by ChIP-Seq, and unraveling the wide and intricate gene pathways regulated by PPARG by RNA-Seq. Costa et al. 2010a

Advance statistical methods for Chip-seq mapping: Improved method to predict the de novo motif discovery in the peak environments by investigating the human growth-associated binding protein (GABPalpha) based on ChIP-seq observations. Jiao et al. 2010

### **Genome wide structural variation detection in human population**

HT-NGS in 1000 genome project: Pilot study by whole-genome sequencing of 179 individuals from four populations, to develop and compare different strategies for genome-wide sequence variation using HT NGS platforms. Durbin et al. 2010

Study of fine scale human population structural variation: to implicate in population structure for the distribution and discovery of disease-causing genetic variants in diverse human genomes, using HT-NGS sequencing data. Henn et al. 2010

Detection of disease-causing mutations in patients with monogenic inherited diseases: The Retinitis pigmentosa (RP): Study demonstrates that next-generation sequencing is an effective approach for detecting novel, rare mutations causing heterogeneous monogenic disorders such as RP. With the addition of this technology, disease-causing mutations can now be identified in 65% of autosomal dominant RP cases Bowne et al. 2011

Detecting structural variations in the human genome using next generation sequencing: A comprehensive review on application of HT-NGS



technology in identification of sequencing-based algorithms for detection of structural variations of human genome. Xi et al. 2010

**Mutation detection and carrier screening** "Functional genomic fingerprinting" (FGF) in mutation detection: Study proposed a selective enrichment of functional genomic regions (the exome, promoterome, or exon splice enhancers) approach (FGF) to HT-NGS, in response to discovery of causal mutations for disease and drug response. Senapathy et al. 2010

Target HT-NGS in disease mutation detection: Study identified a mutation in a gene and have shown its association with autosomal-recessive cerebellar ataxia, by combining SNP array-based linkage analysis and targeted resequencing of relevant sequences in the linkage interval with the use of next-generation sequencing technology. Vermeer et al. 2010

Microarray-based target enrichment in HT-NGS: Study allowed the parallel, large-scale analysis of complete genomic regions for multiple genes of a disease pathway and for multiple samples simultaneously, thus provides an efficient tool for comprehensive diagnostic screening of mutations. Amstutz et al. 2011

Pre-conceptional carrier screening of 448 severe recessive childhood diseases: An economic way of carrier screening by HT-NGS is possible and available to the general population with severe recessive childhood disorders Bell et al. 2011

**Detection of inherited disorders** Detection of monogenic inherited disorders: Study revealed the identification of human monogenic disorders by sequencing of all exons in the human genome (exome sequencing). Kuhlenbäumer et al. 2011

Role of HT-NGS neurogenetics and psychiatric disorders: Comprehensive review of impact of HT-NGS on last two decades on brain research including large number of neurological and psychiatric disorders. Zoghbi and Warren 2010

Impact of HT-NGS to understand the genetic causes of disorders of sex development (DSD): a combined approach of comparative genomic

hybridization, sequencing by hybridization with HT-NGS was presented to understand the genetic basis of human sexual determination and differentiation. Bashamboo et al. 2010

**Complex human diseases** HT-NGS in exploiting the complex disease traits: comprehensive review on the experimental design considerations, data handling issues and required analytical developments tools in mapping genetic traits using NGS. Day-Williams and Zeggini 2010

Genome-wide association studies (GWAS) using HT-NGS: Systematically identifying the genetic risks that lead or predispose to complex diseases by HT-NGS. Singleton et al. 2010

HT-NGS in clinical diagnosis: principles of sequencing library preparation, sequencing chemistries, and NGS data analysis for targeted re-sequencing of genes implicated in hypertrophic cardiomyopathy. Voelkerding et al. 2010

HT-NGS in identifying the causal variants of human disease: A comprehensive review on identification of causal variant typically involves in the vicinity of disease-associated SNPs including protein coding, regulatory, and structural sequences. Kingsley 2011

**Cancer research** Analysis of HT-NGS data in cancer genomics: Introduction to set up of an integrate database for multiple cancers and tumor genomes to understand a coherent picture of the genetic basis of cancer. Ding et al. 2010

Impact of HT-NGS on surgical oncology: Fast growing HT-NGS technology enables to identify the causal mutations responsible for driving cancer initiation and metastasis and raises significant expectations for improving oncologic outcomes. Katsios et al. 2010

Understanding the cancer genomes through HT-NGS: including somatic genome alterations, cancer biology, diagnosis and therapy through whole-genome, whole-exome and whole-transcriptome HT-NGS approaches. Meyerson et al. 2010

HT-NGS in cancer researches: A Comprehensive review on HT-NGS applications to cancer genome, particularly, the glioblastoma multiforme

that identified the gene encoding isocitrate dehydrogenase 1 (IDH1), as target for cancer-driving mutations. Pfeifer and Hainaut 2011

Understanding of the potential actions of SOX2 in carcinogenesis: identification of 4883 SOX2 binding regions in the GBM cancer genome using the HT-NGS Chip-seq technology Fang et al. 2011

**RNA sequencing** MicroRNA expressing profiling by HT-NGS: study proposed an alternative improved method to generate high quality miRNA sequencing libraries for the Illumina genome analyzer. Buermans et al. 2010

RNA-seq in HT-NGS: A comprehensive review on RNA-Seq for transcriptome studies supported by HT-NGS platforms. Study also addressed how to determine accurately the expression levels of specific genes, differential splicing, allele-specific expression of transcripts and many biological-related issues utilized in RNA-Seq experiments. Costa et al. 2010b

Classification of Small non-coding RNAs (ncRNAs) using HT-NGS: Study demonstrated a scoring system called alignment of pattern matrices score (ALPS) that only uses the relative positions and lengths of reads of NGS data, to classify ncRNAs (<http://www.bio.ifi.lmu.de/ALPS>). Erhard and Zimmer 2010

Construction of complex miRNA repertoire database: A comprehensive survey of miRNA sequence variations from human and mouse samples using next generation sequencing platforms. Study device a method to construct a database to determine the most abundant sequence and the degree of heterogeneity for each individual miRNA species that catalogs the entire repertoire of miRNA sequences (<http://galas.systemsbiology.net/cgi-bin/isomir/find.pl>) Lee et al. 2010

Analysis of miRNA profiling in HT-NGS: introduction of an efficient procedure to prepare the small RNA libraries for Illumina sequencing and analyses of the resultant sequence data for measuring microRNA abundance. Morin et al. 2010

RNA-seq and HT-NGS: study introduced an efficient procedure for performing RNA-Seq using the Illumina sequencing platform  
Nagalakshmi et al. 2010

RNA-seq and HT-NGS: a comprehensive review on RNA-Seq including the technical issues accompanying RNA-seq data generation and analysis.  
Marguerat and Bähler 2010

HT-NGS in miRNA: First complete characterization of the "miRNAome" in a primary human cancer: study identified genetic variants of miRNA genes, and screen for alterations in miRNA binding sites in a patient with acute myeloid leukemia. Ramsingh et al. 2010

HT-NGS in functional genomics: A comprehensive review on contribution NGS-based technologies in functional genomics research with a special focus on gene regulation by transcription factor binding sites. Werner 2010

Annotation and mining of HT-NGS data: study proposed a novel database (The deepBase) to facilitate the comprehensive annotation and discovery of small RNAs from transcriptomic data. Yang et al. 2011

**Library preparation, paired ends and genomic captures for NGS platforms** Library preparation in HT-NGS: study presented a robust and cost-effective preprocessing method for DNA sample library construction using a unique 6 bp DNA barcode, which allowed multiplex sample processing and sequencing of 32 libraries in a single run using Applied Biosystems SOLiD sequencer. Farias-Hesson et al. 2010

Paired-end sequencing in HT-NGS: study proposed a NovelSeq pipeline (<http://compbio.cs.sfu.ca/strvar.htm>) to detect and characterize multiple types of genetic variation (SNPs, structural variation, etc.).  
Hajirasouliha et al. 2010

Library preparations for tissue specific expression profiling in HT-NGS: study compared NGS with two alternative technologies, cap analysis of gene expression (CAGE) and serial analysis of gene expression (SAGE) and identified 196 novel regulatory regions with preferential use in proliferating or differentiated cells. These CAGE and SAGE libraries

provides consistent expression levels and can enrich current genome annotations with tissue-specific promoters and alternative 3'-UTR usage.

Hestand et al. 2010

Genomic capture in HT-NGS: study developed an accurate, thorough, and cost-effective identification of inherited mutations for breast and ovarian cancer, through a genomic assay to capture, sequence, and detect all mutations in 21 genes, including BRCA1 and BRCA2, with inherited mutations that predispose to breast or ovarian cancer. Walsh et al. 2010

**Sequencing of mitochondrial genome** Annotation of mitochondrial genome HT-NGS: study proposed a high-throughput sequencing and bioinformatics pipeline for mt genomics, which have implications for the annotation and analysis of other organelles (e.g. plastid or apicoplast genomes) and virus genomes as well as long, contiguous regions in nuclear genomes. Jex et al. 2010

HT NGS in mitochondrial genome: Study developed and proposed a pipeline for sequencing and de novo assembly of multiple mitochondrial genomes without the costs of indexing. McComish et al. 2010

Sequencing of complete four F-type mitochondrial genomes (15 761 bp) from the European freshwater bivalve *Unio pictorum* (Unionidae): Comparison of mitochondrial genomes revealed very low nucleotide diversity within the species which may have the potential importance for environmental management policies. Soroka and Burzynski 2010

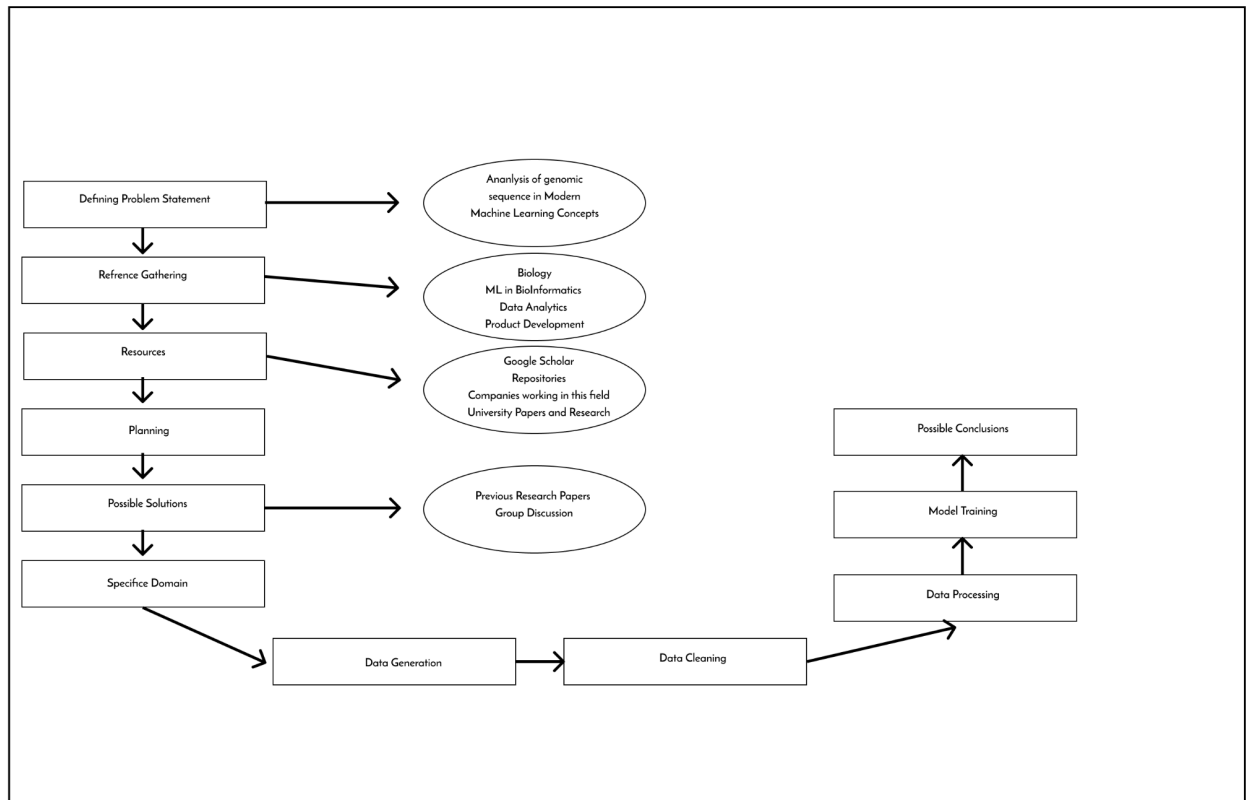
**Personal genomics** Exploring the personal human genome by total integrated archive of short-read and array (TIARA): Set up of improved database for accurate detection of personal genomic variations, such as SNPs, short indels and structural variants (SVs). Hong et al. 2011

**Deep Genomics**, for example, uses machine learning to assist researchers in interpreting genetic variation. Algorithms are specifically built based on patterns observed in big genetic data sets, which are then translated into computer models to assist customers in understanding how genetic diversity influences critical biological processes. Metabolism, DNA repair, and cell development are examples of cellular activities. Disruption in the

regular functioning of these pathways has the potential to induce illnesses like cancer.

### **Topic of the work**

#### **a. System Design / Architecture**



## b. Working Principle

- To define this model for specific genomics dataset is an important decision for Bio-informaticians, Biologists and Clinicians. In many use cases this selection is defined, for example, based on the uniformity of the DNA sequence length across the dataset.
- Traditional ML algorithms (Linear and Logistics Regressions, Decision Trees, Support Vector Machines, Random Forest, Boosting Algorithms, Bayesian Network, etc.) can be used with any DNA sequence length.
- Modern ANN algorithms like CNN and RNN require consistent DNA sequence length in the whole dataset column. The Py DNA library provides a simple function to determine if the selected DNA sequence string contains uniform length or not

- Let's look at the first use case. Suppose we need to build a classification model that can predict a gene family based on a human DNA sequence dataset.
- Genes are categorized into families based on shared nucleotide or protein sequences. A gene family is a set of several similar genes, formed by duplication of a single original gene, and generally with similar biochemical functions.

**c. Expected Results:**

The expected results depend on six main calculated metrics used in classification models validation: accuracy score, precision, recall, f1 score, confusion matrix and classification report. To find an appropriate algorithm to train the model for best results.

**d. Individual Contribution:**

The development of the whole project was a team effort, each individual came to their best mind and cooperation to make this project come to the very near end of the completion. To divide the individuality of them to this project we can look at Table 1.1

Reg. No.	NAME	PHASE 1	PHASE 2
19BOE10081	Chetna Bisen	Research conducts on Sequencing.	Model Validation & Sequencing Method
19MIM10113	Varnika Singh	Algorithm Selection	Human Reference Genome Model
19BCE10177	Ananya Singh	FAST API Development	UI/UX Design & iOS App development



19BCE10054	Shreyash Mall	Flutter Mobile Development	AI as an API
19BCE10336	K Aniket Prusty	Web Development	Webflow & Flask.
19BCE10061	Mudit Sorikh	Dataset gathering for model training.	UX/UI Design Website.
19MIM10048	Pranay Pratap Singh	Algorithm Selection	SARS Cov-19 Model
19BAI10112	Abhishek J Nair	Algorithm Selection	Human Evolution Model

Table 1.1

## Individual Contributions

### Chetna Bisen (19BOE10081)

Humans have around 20,000 genes and three billion base pairs. Surprisingly, just approximately 2% of the human genome encodes proteins. Currently, there are two major impediments to broader precision medicine implementation: high prices and technological restrictions. Many researchers are applying machine learning approaches to deal with the massive quantity of patient data that must be collected and evaluated, as well as to assist minimize expenses.

The categorization and analysis of diseases and illnesses is one of the most significant advantages of machine learning in healthcare. Because it was difficult to diagnose, it made it more controllable. It can include everything from difficult-to-find cancers in the early stages to other transmitted illnesses.

A genome is a full collection of genetic instructions for an organism. Each

genome includes all of the information required to construct the organism and allow it to grow and develop.

A laboratory technique for determining the whole genetic make-up of an organism or cell type. This approach may be used to detect changes in specific regions of the genome.

When you read a phrase, the meaning is more than simply the order of the letters. It is also present in the words formed by those letters, as well as in the grammar of the language. Similarly, the human genome is more than just a series of letters.

As an individual, my duty in this project was to ensure that clinical techniques were carried out correctly, as most of the others did not have the same background as me. As a biologist and an engineer, I must concentrate on all ML model creation and their results or prediction validation of the model since there must be support for the outcomes that the model has projected across the dataset presented. To make other team members aware of the problems that a researcher faces when doing clinical experiments that are hassle-free and straightforward to conduct while producing accurate and promising data.

### **Ananya Singh(19BCE10177)**

Bioinformatics is an interdisciplinary discipline that combines molecular biology, genetics, computer technology, mathematics, and statistics. Computational strategies are used to address data-intensive, large-scale biological issues. The most typical issues are molecular modeling of biological processes and data inference. A bioinformatics solution typically consists of the following steps: Biological data may be used to generate statistical information. Create a computational model. Solve an issue by using computational modeling. Put a computer algorithm to the test and see how it performs.

Machine learning is becoming increasingly prominent in bioinformatics and computational biology. Its popularity has recently expanded, and it is currently used in a range of industries such as genetic sequencing, classification, and many others.

My position on this project was as a mobile application developer and front-end designer. I have successfully completed all of the UI/UX design campaigns with all of the needed aesthetic, sturdy, and user-friendly design.

My responsibility as a member of the mobile application development team was to ensure that the deployment system runs as smoothly as the predicting system; in the end, users will interact with the front end rather than the model directly, which is important because the user may not be as technically savvy to understand the model predictions.

**Varnika Singh (19MIM10113)**

Researched the DNA sequences in different datasets to prepare ML applications starting with DNA data handling, data encoding, data loading, algorithm implementation, and model performance matrix with my peers in the AI Model Development Team.

Also provided assistance in data loading and analytical implementation of the model 1 with human, dog and chimpanzee dataset and in model 2 with COVID -19 patient dataset to my peers in the AI Model Development Team.

**Shreyash Mall (19BCE10054)**

Prior to the introduction of machine learning, bioinformatics algorithms had to be constructed by hand, which proved difficult for problems like protein structure prediction. Deep learning and other machine learning algorithms may be able to identify elements of data sets without requiring the programmer to define them individually.

APIs are important from a technical viewpoint because they allow one computer program's capabilities to be utilized by another. They allow two separate programmes to communicate with one another. APIs are fostering a new wave of innovation based on sharing services, similar to how the Web expanded the Internet's possibilities. APIs and their potential to alter business processes are attracting the attention of organizations across all industries.

With the immense possibility of the automation of the whole sequencing process the agility and accuracy of the faster clinical test results can be assured. The most important things like mutation synthesis can be of a faster rate than before which helps in taking right measures as a precaution as can be assured in the current situation of COVID-19.

My role in the team was to assure the API development and integration for both the deployment platform that is website and a mobile application on both os android and iOS. Using Fast API and Flutter as a tool for mobile application development for cross platform integration. For the implementation of API we need two type of APIs number one is the AI model as the API to be hosted on the database and second to send and receive the query request to the model for prediction.

### **K Aniket Prusty(19BCE10336)**

After deciding the algorithm to use and implementing to make the model it is very important to make the model easily usable and accessible by the people or organizations of different domains such as health care sectors(medical), research organizations etc. To make the model easily accessible we have to deploy it as a website or as a mobile application. By doing so it will enable every single person, whether he/she knows coding or not, to use the model without taking the stress of learning and implementing the coding and ML knowledge.

My role for this project is to develop a website which can perform all the mentioned functionalities:

1. Making the model accessible to the community.
2. Making the website usable with ease.
3. Making the website scalable and robust.
4. Deploying the ML model.
5. Developing an API to interact with the database.
6. Making an API to connect the frontend with Backend.
7. Making the website fetch the file from frontend and give it to the backend and training the model, testing it and getting the classification report, confusion matrix and accuracy data as output and rendering it on the website.
8. Hosting the website.

In phase 2 we have worked in a modular pattern to make the working, testing, deployment of the model, website and mobile application easy. For this phase I have used Flask to deploy the model(70 percent of which is done), and I have also developed the UI design of the home page using webflow which will later fetch the files from user as input, developed the API to connect the backend with database. Flask is a python online application framework that allows end users to interact with your python

code (in this case our machine learning models).

The work left is to connect the frontend and backend using REST API, integrating the model completely and hosting the website so that it will be functional.

### **Mudit Sorikh(19BCE10061)**

With the advent of technology in the twenty-first century, everyone wants to enjoy the latest technologies without wasting time or straining their busy minds. The same is true for surfing websites or mobile applications, where the faster and more efficiently the website or mobile application reacts, the better the results. In a nutshell, it's all about consumers these days. When it comes to customer happiness with mobile applications or websites, most technology companies focus on the applications' User Interface (UI) and User Experience Design (UX). However, most people mix up the phrases user interface and user experience. As a result, we've come to decode the UI and UX design concepts.

Many people are looking for programmes that are visually appealing and feel great to use. A visually appealing and engaging app is usually the product of good User Experience (UX) and User Interface design (UI). The best developers will tell you, an app has to be developed while putting much emphasis on its UI/UX design since that's where success of an app begins.

A great User Interface will draw people to the app right away, while a great User Experience will leave a lasting impression on your users. As a result, if you want your app to be successful, you must get both of them correct.

This project needed me to work as a front-end designer and a Web app developer. All of the UI/UX design initiatives were completed successfully, with all of the needed beauty, sturdiness, and user-friendliness. My responsibility as a member of the Web Application development team was to ensure that the deployment system ran as smoothly as the predicting system; in the end, users will interact with the front end rather than the model directly, which is important because the user may not be as technically savvy as the model predicts.

### **Abhishek J Nair (19BAI10112)**

My contribution towards the project was to ensure the development of the model so I created a topic related sample program to check out which Algorithm would be more accurate to be used in ML-Python integrated model (Algorithm Selection). Prepared classification model that is trained on the human DNA sequence and can predict a gene family based on the DNA sequence of the coding sequence. Also used the chosen algorithm to implement the classification model from phase-1. Researched the DNA sequences to prepare ML applications starting with DNA data handling, data encoding, data loading, algorithm implementation, and model performance matrix with my peers in the AI Model Development Team. Also provided assistance in data loading and analytical implementation of the SARS COVID-19 model to my peers in the AI Model Development Team.

### **Pranay Pratap Singh (19MIM10048)**

To train a classification classifier on the human DNA sequence such that it can identify a gene family based on the coding sequence's DNA sequence. To put the model to the test, we will use genetic data from a Covid-19 patient discovered in Wuhan, China. The dataset we utilized comprises the genes of a Covid-19 patient located in Wuhan, China, as well as some additional information such as id and nucleotide details.

I analyzed the data after importing it from the NCBI site, put the patient's gene sequence in a variable, and then estimated its molecular weight. Following that, we tallied the number of nucleotides in the gene sequence that was available.

I have worked on model -2 with the dataset of Covid-19 patient and explored the gene sequence of the patient with the help of a biopython library and will also calculate the accuracy of the prediction using the Multinomial Naive Bayes algorithm.

Researched the DNA sequences in different datasets to prepare ML applications starting with DNA data handling, data encoding, data loading, algorithm implementation, and model performance matrix with my peers in the AI Model Development Team.

Also provided assistance in data loading and analytical implementation of the model 1 with human and chimpanzee dataset and in model 3 with human reference genome dataset to my peers in the AI Model

Development.

## Phase 1 Development

### a. The Machine Learning Classification Models used:

#### 1. Logistic regression:

It is similar to linear regression because both of these involve estimating the values of parameters used in the prediction equation based on the given training data. Linear regression predicts the value of some continuous, dependent variable. Whereas logistic regression predicts the probability of an event or class that is dependent on other factors. Thus, the output of logistic regression always lies between 0 and 1. Because of this property it is commonly used for classification purpose.

Consider a model with features  $x_1, x_2, x_3 \dots x_n$ . Let the binary output be denoted by  $Y$ , that can take the values 0 or 1. Let  $p$  be the probability of  $Y = 1$ , we can denote it as  $p = P(Y=1)$ . The mathematical relationship between these variables can be denoted as:

$$\ln\left(\frac{p}{1-p}\right) = b_0 + b_1x_1 + b_2x_2 + b_3x_3 \dots b_nx_n$$

Here the term  $p/(1-p)$  is known as the odds and denotes the likelihood of the event taking place. Thus  $\ln(p/(1-p))$  is known as the log odds and is simply used to map the probability that lies between 0 and 1 to a range between  $(-\infty, +\infty)$ . The terms  $b_0, b_1, b_2 \dots$  are parameters (or weights) that we will estimate during training.

So, this is just the basic math behind what we are going to do. We are interested in the probability  $p$  in this equation. So, we simplify the equation to obtain the value of  $p$ :

The log term  $\ln$  on the LHS can be removed by raising the RHS as a power of  $e$ :

$$\frac{p}{1-p} = e^{b_0+b_1x_1+b_2x_2+b_3x_3\dots b_nx_n}$$

Now we can easily simplify to obtain the value of  $p$ :



$$p = \frac{e^{b_0+b_1x_1+b_2x_2+b_3x_3...b_nx_n}}{1 + e^{b_0+b_1x_1+b_2x_2+b_3x_3...b_nx_n}}$$

or

$$p = \frac{1}{1 + e^{-(b_0+b_1x_1+b_2x_2+b_3x_3...b_nx_n)}}$$

This actually turns out to be the equation of the Sigmoid Function which is widely used in other machine learning applications. The Sigmoid Function is given by:

$$S(x) = \frac{1}{1 + e^{-x}}$$

## **2. Decision Tree:**

It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves

Entropy is nothing but the uncertainty in our dataset or measure of disorder  
In a decision tree, the output is mostly “yes” or “no”

The formula for Entropy is shown below:

$$E(S) = -p_{(+)} \log p_{(+)} - p_{(-)} \log p_{(-)}$$

Here  $p_{+}$  is the probability of positive class  $p_{-}$  is the probability of negative class  
 $S$  is the subset of the training example  
Information gain measures the reduction of uncertainty given some feature and it is also a deciding factor for which attribute should be selected as a decision node or root node.

$$\text{Information Gain} = E(Y) - E(Y|X)$$

It is just entropy of the full dataset – entropy of the dataset given some feature.

## **3. Random Forest:**

The Random Forest Algorithm is composed of different decision trees, each with the same nodes, but using different data that leads to different leaves. It merges the decisions of multiple decision trees in order to find an answer, which represents the average of all these decision trees.

When using the Random Forest Algorithm to solve regression problems, you are using the mean squared error (MSE) to how your data branches from each node.

$$MSE = \frac{1}{N} \sum_{i=1}^N (f_i - y_i)^2$$

Where  $N$  is the number of data points,  
 $f_i$  is the value returned by the model and  
 $y_i$  is the actual value for data point  $i$ .

This formula calculates the distance of each node from the predicted actual value, helping to decide which branch is the better decision for your forest. Here,  $y_i$  is the value of the data point you are testing at a certain node and  $f_i$  is the value returned by the decision tree.

#### 4. Support Vector Machine:

A Support Vector Machine or SVM is a machine learning algorithm that looks at data and sorts it into one of two categories.

Support Vector Machine is a supervised and linear Machine Learning algorithm most commonly used for solving classification problems and is also referred to as Support Vector Classification.

Lagrange stated that if we want to find the minimum of  $f$  under the equality constraint  $g$ , we just need to solve for:

$$\nabla f(\mathbf{x}) - \alpha \nabla g(\mathbf{x}) = 0$$

$\alpha$  is called the Lagrange multiplier.

In terms of the SVM optimization problem,  $f(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$ ,  $g(\mathbf{w}, \mathbf{b}) = \min_i [\mathbf{w} \cdot \mathbf{x}_i + b - 1]$ . The Lagrangian function is then

$$L(\mathbf{w}, \mathbf{b}, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i [\mathbf{w} \cdot \mathbf{x}_i + b - 1]$$

To solve for  $\nabla L(\mathbf{w}, \mathbf{b}, \alpha) = 0$  analytically, the number of examples has to be small. Thus, we will rewrite the problem using the duality principle.

Equivalently, we need to solve the following Lagrangian primal problem:

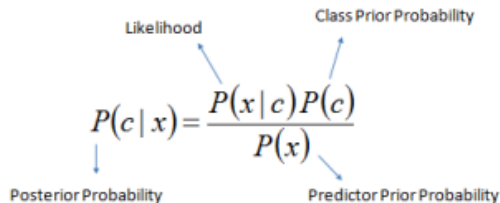
$$\min_{\mathbf{w}, \mathbf{b}} \max_{\alpha} L(\mathbf{w}, \mathbf{b}, \alpha) \text{ subject to } \alpha_i \geq 0, i=1 \dots m$$

More accurately,  $\alpha$  should be KKT (Karush-Kuhn-Tucker) multipliers because we are dealing with inequality constraints here. But we will use the term Lagrangian multipliers for continuity.

There is an  $\alpha$  for each example, we need to maximize  $L(\mathbf{w}, \mathbf{b}, \alpha)$  for all examples. And there is a  $(\mathbf{w}, \mathbf{b})$  for each hyperplane, we need to minimize the  $\max_{\alpha} L(\mathbf{w}, \mathbf{b}, \alpha)$  in the meantime.

## 5. Multinomial Naïve Bayes:

Bayes theorem provides a way of calculating the posterior probability,  $P(c|x)$ , from  $P(c)$ ,  $P(x)$ , and  $P(x|c)$ . Naive Bayes classifier assume that the effect of the value of a predictor ( $x$ ) on a given class ( $c$ ) is independent of the values of other predictors. This assumption is called class conditional independence.



The diagram shows the formula  $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$  with arrows pointing from labels to the terms: 'Likelihood' points to  $P(x|c)$ , 'Class Prior Probability' points to  $P(c)$ , 'Posterior Probability' points to  $P(c|x)$ , and 'Predictor Prior Probability' points to  $P(x)$ .

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

- $P(c|x)$  is the posterior probability of class (target) given predictor (attribute).
- $P(c)$  is the prior probability of class.
- $P(x|c)$  is the likelihood which is the probability of predictor given class.
- $P(x)$  is the prior probability of predictor.

## 6. Gradient Boosting:

Let's say the output model  $y$  when fit to only 1 decision tree, is given by:

$$y = A_1 + (B_1 * X) + e_1$$

where,  $e_1$  is the residual from this decision tree. In gradient boosting, we fit the consecutive decision trees on the residual from the last one. So when gradient

boosting is applied to this model, the consecutive decision trees will be mathematically represented as:

$$e_1 = A_2 + (B_2 * X) + e_2$$

and

$$e_2 = A_3 + (B_3 * X) + e_3$$

Note that here we stop at 3 decision trees, but in an actual gradient boosting model, the number of learners or decision trees is much more. Combining all three equations, the final model of the decision tree will be given by:

$$y = A_1 + A_2 + A_3 + (B_1 * x) + (B_2 * x) + (B_3 * x) + e_3$$

## 7. Ada Boost:

---

Given:  $(x_1, y_1), \dots, (x_m, y_m)$  where  $x_i \in \mathcal{X}$ ,  $y_i \in \{-1, +1\}$ .

Initialize:  $D_1(i) = 1/m$  for  $i = 1, \dots, m$ .

For  $t = 1, \dots, T$ :

- Train weak learner using distribution  $D_t$ .
- Get weak hypothesis  $h_t : \mathcal{X} \rightarrow \{-1, +1\}$ .
- Aim: select  $h_t$  with low weighted error:

$$\epsilon_t = \Pr_{i \sim D_t} [h_t(x_i) \neq y_i].$$

- Choose  $\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$ .
- Update, for  $i = 1, \dots, m$ :

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where  $Z_t$  is a normalization factor (chosen so that  $D_{t+1}$  will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign} \left( \sum_{t=1}^T \alpha_t h_t(x) \right).$$


---

**Fig. 1** The boosting algorithm AdaBoost.

Adaboost, shortened for Adaptive Boosting, is a machine learning approach that is conceptually easy to understand, but less easy to grasp mathematically. Part of the reason owes to equations and formulas not being broken down into simple terms with basic math as demonstration of the equations. This essay intends to do just that with Adaboost, with newcomers into data science as the primary target audience.

## **8. K- Nearest Neighbour:**

A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If  $K = 1$ , then the case is simply assigned to the class of its nearest neighbor.

### **Distance functions**

Euclidean	$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$
Manhattan	$\sum_{i=1}^k  x_i - y_i $
Minkowski	$\left( \sum_{i=1}^k (x_i - y_i)^q \right)^{1/q}$

It should also be noted that all three distance measures are only valid for continuous variables. In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and

categorical variables in the dataset.

#### Hamming Distance

$$D_H = \sum_{i=1}^k |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

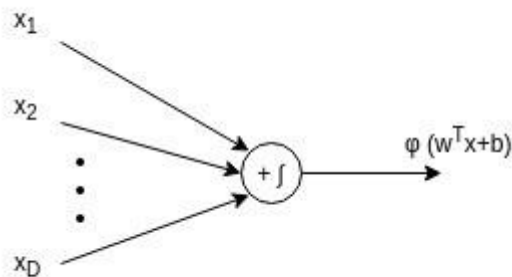
$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise but there is no guarantee. Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. Historically, the optimal K for most datasets has been between 3-10. That produces much better results than 1NN.

### 9. Multilayer Perceptron:

in the perceptron we just multiply with weights and add Bias, but we do this in one layer only.



We update the weight when we found an error in classification or miss-classified. Weight update equation is this...

$$\text{weight} = \text{weight} + \text{learning\_rate} * (\text{expected} - \text{predicted}) * x$$

### 10. Extreme Gradient Boost:

For each node, there is a factor  $\gamma$  with which  $hm(x)$  is multiplied. This accounts for the difference in impact of each branch of the split. Gradient boosting helps

in predicting the optimal gradient for the additive model, unlike classical gradient descent techniques which reduce error in the output at each iteration. The following steps are involved in gradient boosting: •  $F_0(x)$  – with which we initialize the boosting algorithm – is to be

defined:

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^n L(y_i, \gamma)$$

- The gradient of the loss function is computed iteratively:

$$r_{im} = -\alpha \left[ \frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)}, \text{ where } \alpha \text{ is the learning rate}$$

- Each  $h_m(x)$  is fit on the gradient obtained at each step
- The multiplicative factor  $\gamma_m$  for each terminal node is derived and the boosted model  $F_m(x)$  is defined:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

## b. Results and Conclusions

The results are based on 50 epoch cycles run on Google Colab for various regression algorithms, with the best results obtained with "Multinomial Naive Bayes and Multi-layer Perceptron classifier models" with a test accuracy score of more than 97 percent and the idea that these algorithms can be used on both continuous and discrete data because genomic sequences can be continuous or discrete. Other logistic regression and random forest models were investigated, however they only gave roughly 92 percent of the test accuracy score because to the large number of nucleotide substrings that required to be classified (i.e. sequences of A=Adenine, G=Guanine, T=Thymine, C=Cytosine). Some of the test results and confusion matrices are displayed here for your convenience

## **Phase 2 Development**

### **a. Model Development**

#### **Model-1**

Modules, Working and Conclusion Report:

DNA data handling using Biopython:

We worked with a DNA sequence in fasta format using Biopython. The sequence object will contain attributes such as id and sequence and the length of the sequence that we can work with directly.

We will use Bio.SeqIO from Biopython for parsing DNA sequence data(fasta). It provides a simple uniform interface to input and output assorted sequence file formats.

Now that we can load and manipulate biological sequence data easily, how can we use it for machine learning or deep learning?

Now since machine learning or deep learning models require input to be feature matrices or numerical values and currently we still have our data in character or string format. So the next step is to encode these characters into matrices.

There are 3 general approaches to encode sequence data:

1. Ordinal encoding DNA Sequence
2. One-hot encoding DNA Sequence
3. DNA sequence as a “language”, known as k-mer counting

Here comes machine learning...

Now that we have extracted the featured matrix from the DNA sequence, we will apply the acquired knowledge to the real-life machine learning use case.

### **Our Github Code Link**

[https://github.com/DarkTitan007/Human\\_Chimp\\_Dog\\_DNA\\_Sequencing/blob/main/DNA\\_Sequencing\\_with\\_ML.ipynb](https://github.com/DarkTitan007/Human_Chimp_Dog_DNA_Sequencing/blob/main/DNA_Sequencing_with_ML.ipynb)

### **Objective and Methodology:**

To build a classification model that is trained on the human DNA sequence and can predict a gene family based on the DNA sequence of the coding sequence. To test the model, we will use the DNA sequence of humans, dogs, and chimpanzees and compare the accuracies.

Gene families are groups of related genes that share a common ancestor. Members of gene families may be paralogs or orthologs. Gene paralogs are genes with similar sequences from within the same species while gene orthologs are genes with similar sequences in different species.

The dataset contains human DNA sequence, Dog DNA sequence, and Chimpanzee DNA sequence.

After we have loaded the dataset, the next step is to convert a sequence of characters into k-mer words, default size = 6 (hexamers). The function `Kmers_func()` will collect all possible overlapping k-mers of a specified length from any sequence string.

We need to now convert the lists of k-mers for each gene into string sentences of words that can be used to create the Bag of Words model. We will make a target variable `y` to hold the class labels. We will do the same for chimp and dog.



We will create the Bag of Words model using `CountVectorizer()`. This is equivalent to k-mer counting. The n-gram size of 4 was previously determined by testing.

We will convert our k-mer words into uniform length numerical vectors that represent counts for every k-mer in the vocabulary

Transformation our DNA sequences into uniform length numerical vectors in the form of k-mer counts and ngrams.

So, for humans we got 6890 genes converted into uniform length feature vectors of 4-gram k-mer (length 6) counts. For chimp and dog, we have the same number of features with 1682 and 820 genes respectively.

So now that we know how to transform our DNA sequences into uniform length numerical vectors in the form of k-mer counts and ngrams, we can now go ahead and build a classification model that can predict the DNA sequence function based only on the sequence itself.

Here I will use the human data to train the model, holding out 20% of the human data to test the model. Then we can challenge the model's generalizability by trying to predict sequence function in other species (the chimpanzee and dog).

Next, train/test split human dataset and build simple multinomial naive Bayes classifiers.

You might want to do some parameter tuning and build a model with different ngram sizes, here I'll go ahead with an ngram size of 4 and a model alpha of 0.1.

## Algorithm from Phase-1.

We will create a multinomial naive Bayes classifier as selection by trial and error method from phase-1 review. We previously did some parameter tuning and found the ngram size of 4 (reflected in the Countvectorizer() instance) and a model alpha of 0.1 did the best.

## Observations and Conclusions:

Now we will look at some model performance metrics like the confusion matrix, accuracy, precision, recall and f1 score. We are getting really good results on our unseen data, so it looks like our model did not overfit to the training data. In the real model I would go back and sample many more train test splits since we have a relatively small data set.

Confusion matrix for predictions on human test DNA sequence

Predicted	0	1	2	3	4	5	6
Actual							
0	99	0	0	0	1	0	2
1	0	104	0	0	0	0	2
2	0	0	78	0	0	0	0
3	0	0	0	124	0	0	1
4	1	0	0	0	143	0	5
5	0	0	0	0	0	51	0
6	1	0	0	1	0	0	263

accuracy = 0.984  
precision = 0.984  
recall = 0.984  
f1 = 0.984

Confusion matrix for predictions on Chimpanzee test DNA sequence

```
Predicted    0    1    2    3    4    5    6
Actual
0           232    0    0    0    0    0    2
1           0   184    0    0    0    0    1
2           0    0   144    0    0    0    0
3           0    0    0   227    0    0    1
4           2    0    0    0   254    0    5
5           0    0    0    0    0   109    0
6           0    0    0    0    0    0   521
accuracy = 0.993
precision = 0.994
recall = 0.993
f1 = 0.993
```

Confusion matrix for predictions on Dog test DNA sequence

```
Predicted    0    1    2    3    4    5    6
Actual
0           127    0    0    0    0    0    4
1           0   63    0    0    1    0   11
2           0    0   49    0    1    0   14
3           1    0    0   81    2    0   11
4           4    0    0    1   126    0    4
5           4    0    0    0    1   53    2
6           0    0    0    0    0    0   260
accuracy = 0.926
precision = 0.934
recall = 0.926
f1 = 0.925
```

The model seems to produce good results on human data. It also does on Chimpanzee which is because the Chimpanzee and humans share the same genetic hierarchy. The performance of the dog is not quite as good which is because the dog is more diverging from humans than the chimpanzee.

## Model-2

Modules, Working and Conclusion Report-

## **DNA data handling using Biopython:**

In the model 2 we worked with a DNA sequence of Covid-19 patient found in Wuhan, China, using the Biopython library. We used the 'Entrez' module from the biopython library to read the data of the patient which was available on the NCBI site.

We used the SeqIO module from Biopython for parsing DNA sequence data. Using this parsed data we will be analyzing and working further in this model.

## **Using parsed data in machine learning:**

Machine Learning or Deep Learning models require input to be feature matrices or numerical values but currently we have our data in character or we can say string format. So the next step is to encode these characters into matrices, and we will be encoding the data by following the below given approaches.

There are three general approaches to encode sequence data available in string format:

1. Ordinal encoding DNA Sequence
2. One-hot encoding DNA Sequence
3. DNA sequence as a "language", known as k-mer counting

Now that we have extracted the featured matrix from the DNA sequence, we will apply the acquired knowledge to the real-life machine learning use case.

## **Objective and Methodology:**

To build a classification model that is trained on the human DNA sequence and can predict a gene family based on the DNA sequence of the coding sequence. To test the model, we will use the genomic data of Covid-19 Patient found in Wuhan, China.

The dataset we used contains genes of Covid-19 Patient found in Wuhan, China and some other details like id and nucleotides detail.

After we have imported the dataset from the NCBI site, we have parsed the data and stored the gene sequence of the patient in a variable and then calculated its molecular weight. After this, we have counted the number of nucleotides in the gene sequence available.

Later on we transcribed the DNA sequence into a RNA sequence and calculated the GC content available in the DNA sequence because using this value we will get to know the stability of the molecule.

After calculating counts of nucleotides in the sequence we have plotted a graph which shows the frequency of nucleotides. Later on we translated the RNA sequence into a series of amino acids. And then calculated the counts of amino acids.

After completing the above steps we have plotted the graph which gives the detail about the frequency of amino acids and then formulated these amino acids into a sequence of proteins.

Now that we have explored the genomic sequence of the patient we will be implementing the prediction part using few machine learning libraries. The algorithm that we will be working on will be a Multinomial Naive Bayes classifier as selection by trial and error method from phase-1 review.

### **Observations and Conclusion:**

We will calculate some model performance metrics like the confusion matrix, accuracy, precision, recall and f1 score to get the results of the model. As after working on Model 1, we are getting really good results with the algorithm that we have used. In the real model we would go back and sample many more train test splits since we have relatively small data.

### **Our Github Code Link:**

[https://github.com/DarkTitan007/COVID-19-Genome-Analysis-using-ML/blob/main/COVID\\_19\\_Genome\\_Analysis\\_using\\_ML.ipynb](https://github.com/DarkTitan007/COVID-19-Genome-Analysis-using-ML/blob/main/COVID_19_Genome_Analysis_using_ML.ipynb)

### **Model-3**

Modules, Working and Conclusion Report

### **DNA sequence handling using Biopython:**

In the Model 3 we have worked with the latest Human Reference Genome Sequence which was available on the National Cancer Institute GDC site using the Biopython library.

We used the SeqIO module from Biopython using which we have parsed the dataset of Human Reference Genome Sequence. Using this parsed data we will be analyzing and working further in this model.

### **Using parsed data in machine learning:**

Machine Learning or Deep Learning models require input to be feature matrices or numerical values but currently we have our data in character or we can say string format. So the next step is to encode these characters into matrices, and we will be encoding the data by following the below given approaches.

There are three general approaches to encode sequence data available in string format:

1. Ordinal encoding DNA Sequence
2. One-hot encoding DNA Sequence
3. DNA sequence as a “language”, known as k-mer counting

Now that we have extracted the featured matrix from the DNA sequence, we will apply the acquired knowledge to the real-life machine learning use case.

### **Objective and Methodology:**

To build a classification model that is trained on the human DNA sequence and can predict a gene family based on the DNA sequence of the coding sequence. To test the model, we will use the Human Reference Genome Sequence dataset.

The dataset we used contains a huge number of genomic sequences of humans and their Sequence ID. The dataset was in the fasta format so we have used the biopython library to parse the data from this dataset. We have parsed the sequence, sequence id and sequence length from the dataset.

After parsing the dataset we have transformed the sequence string that has been parsed from the dataset into a numpy array as it is required for the machine learning algorithm.

After transforming, we have encoded the sequence data with the help of three encoding approaches that have been mentioned in the above section of the report.

Now, after transforming and encoding the genomic sequence we will be implementing the prediction part using a few machine learning libraries. We will be implementing the Multinomial Naive Bayes classifier algorithm. We are selecting this algorithm by trial and error method from phase-1 review.

## Observations and Conclusions:

We will calculate some model performance metrics like the confusion matrix, accuracy, precision, recall and f1 score to get the results of the model. As after working on Model 1, we are getting really good results with the algorithm that we have used. In the real model we would go back and sample many more train test splits to get much more accurate results.

## Our Github Code Link:

<https://github.com/pps-07/Genome-Sequence-Project/blob/main/Epics%20code%20to%20run.ipynb>

### b. Mobile Application Development

The collection of techniques and procedures involved in building software for tiny, wireless computing devices, such as smartphones and other hand-held devices, is known as mobile application development.

Mobile application development, like web application development, has its origins in older software development methods. Mobile applications, on the other hand, are frequently designed particularly to take use of the unique capabilities of a particular mobile device. For example, a game app might be created to take advantage of the iPhone's accelerometer, while a mobile health app could be written to take advantage of the temperature sensor on a wristwatch.

### Technology used:

#### 1. Flutter:

Flutter is a mobile user interface framework developed by Google for creating high-quality native interfaces for iOS, Android, the web, and the desktop. Flutter is free and open source, and it works with existing code. It is used by developers and organizations all around the world. In this project it is used as the cross platform development tool for the deployment application.

#### 2. FAST API:

FastAPI is a Web framework for developing RESTful APIs in Python. FastAPI is based on Pydantic and type hints to validate, serialize, and deserialize data, and automatically auto-generate OpenAPI documents. It

fully supports asynchronous programming and can run with Uvicorn and Gunicorn. For the record in this project we need to implement most of our AI models as API so as to perform that we need to make use of this framework.

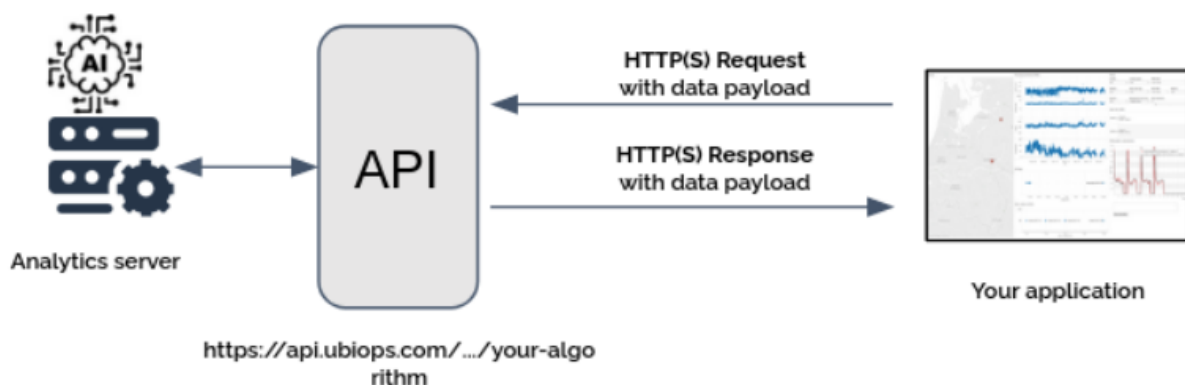
### 3. DART:

Dart is a programming language designed for client development, such as for the web and mobile apps. It is developed by Google and can also be used to build server and desktop applications. Dart is an object-oriented, class-based, garbage-collected language with C-style syntax.

### 4. Jupyter Notebook:

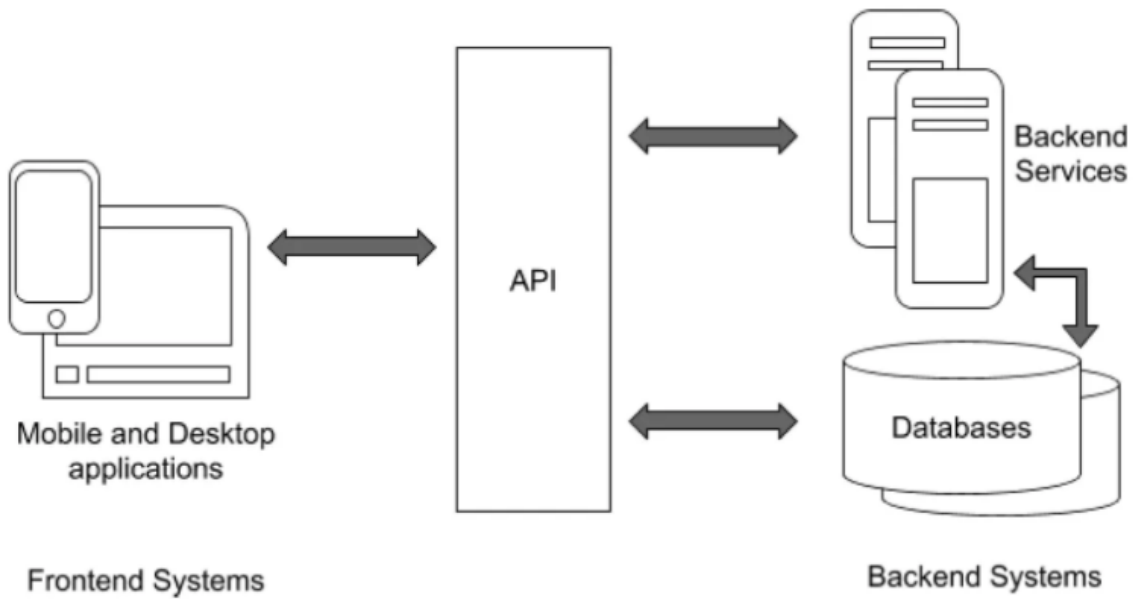
Project Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages". It was spun off from IPython in 2014 by Fernando Pérez and Brian Granger.

### Architecture:



**Fig 1.1**





**Fig 1.2**

### **Results and Screenshots:**

From Fig 1.3 to 1.6 are the code snippets for the API development and client integration.

```

1 import json
2 import numpy as np
3
4 from typing import Optional, List
5 from pathlib import Path
6 from dataclasses import dataclass # pip install dataclasses
7
8 from tensorflow.keras.models import load_model
9 from tensorflow.keras.preprocessing.sequence import pad_sequences
10 from tensorflow.keras.preprocessing.text import tokenizer_from_json
11
12 from . import encoders
13
14 @dataclass
15 class AIModel:
16     model_path: Path
17     tokenizer_path: Optional[Path] = None
18     metadata_path: Optional[Path] = None
19
20     model = None
21     tokenizer = None
22     metadata = None
23
24     def __post_init__(self):
25         if self.model_path.exists():
26             self.model = load_model(self.model_path)
27         if self.tokenizer_path:
28             if self.tokenizer_path.exists():
29                 if self.tokenizer_path.name.endswith("json"):
30                     tokenizer_text = self.tokenizer_path.read_text()
31                     self.tokenizer = tokenizer_from_json(tokenizer_text)
32         if self.metadata_path:
33             if self.metadata_path.exists():

```

**Fig 1.3**

```

1 import os
2 import pathlib
3 from typing import Optional
4 from fastapi import FastAPI
5 from fastapi.responses import StreamingResponse
6
7 from cassandra.query import SimpleStatement
8 from cassandra.cqlengine.management import sync_table
9
10 from . import (
11     config,
12     db,
13     models,
14     ml,
15     schema
16 )
17
18 app = FastAPI()
19 settings = config.get_settings()
20
21 BASE_DIR = pathlib.Path(__file__).resolve().parent
22
23 MODEL_DIR = BASE_DIR.parent / "models"
24 SMS_SPAM_MODEL_DIR = MODEL_DIR / "spam-sms"
25 MODEL_PATH = SMS_SPAM_MODEL_DIR / "spam-model.h5"
26 TOKENIZER_PATH = SMS_SPAM_MODEL_DIR / "spam-classifier-tokenizer.json"
27 METADATA_PATH = SMS_SPAM_MODEL_DIR / "spam-classifier-metadata.json"
28
29 AI_MODEL = None
30 DB_SESSION = None
31 SMSInference = models.SMSInference
32
33 @app.on_event("startup")

```

```

1 import pathlib
2 from . import config
3 from cassandra.cluster import Cluster
4 from cassandra.auth import PlainTextAuthProvider
5 from cassandra.cqlengine import connection
6
7 BASE_DIR = pathlib.Path(__file__).resolve().parent
8 CLUSTER_BUNDLE = str(BASE_DIR / 'encrypted' / 'astradb_connect.zip')
9
10 settings = config.get_settings()
11
12 ASTRA_DB_CLIENT_ID = settings.db_client_id
13 ASTRA_DB_CLIENT_SECRET = settings.db_client_secret
14
15
16 def get_cluster():
17     cloud_config = {
18         'secure_connect_bundle': CLUSTER_BUNDLE
19     }
20     auth_provider = PlainTextAuthProvider(ASTRA_DB_CLIENT_ID, ASTRA_DB_CLIENT_SECRET)
21     return Cluster(cloud=cloud_config, auth_provider=auth_provider)
22
23
24 def get_session():
25     cluster = get_cluster()
26     session = cluster.connect()
27     connection.register_connection(str(session), session=session)
28     connection.set_default_connection(str(session))
29     return session

```

**Fig 1.5**

```

1 import uuid
2 from cassandra.cqlengine import columns
3 from cassandra.cqlengine.models import Model
4
5
6 class Human_chimp(Model):
7     __keyspace__ = "Human_chimp"
8     uuid = columns.UUID(primary_key=True, default=uuid.uuid1) # uuid.uuid1 -> timestamp
9     query = columns.Text()
10    label = columns.Text()
11    confidence = columns.Float()
12    model_version = columns.Text(default='v1')
13
14 class SARS_COV(Model):
15     __keyspace__ = "Covid 19"
16     uuid = columns.UUID(primary_key=True, default=uuid.uuid1) # uuid.uuid1 -> timestamp
17     query = columns.Text()
18     label = columns.Text()
19     confidence = columns.Float()
20     model_version = columns.Text(default='v1')
21
22 class Genome_Seq(Model):
23     __keyspace__ = "Geneome sequencing"
24     uuid = columns.UUID(primary_key=True, default=uuid.uuid1) # uuid.uuid1 -> timestamp
25     query = columns.Text()
26     label = columns.Text()
27     confidence = columns.Float()
28     model_version = columns.Text(default='v1')

```

**Fig 1.6**

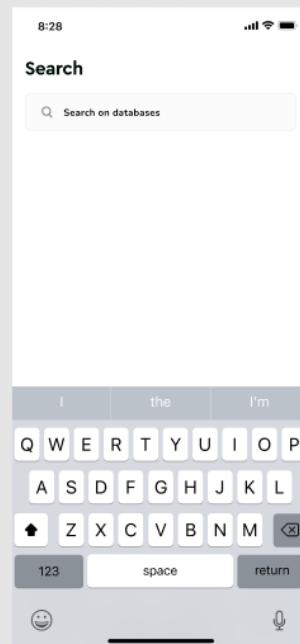
## 05

## Search

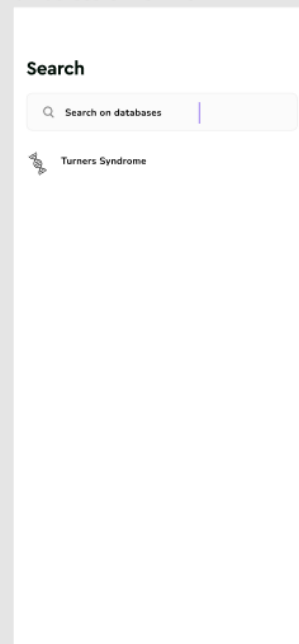
07-01-search



07-02-search-active

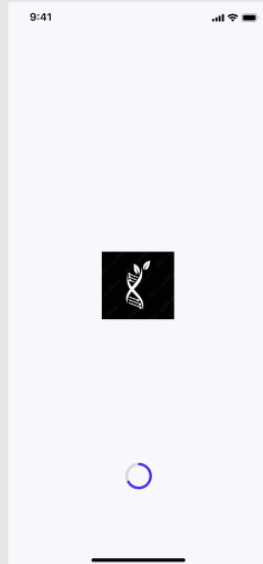


07-03-search-returns



## 01 Splash Screen

01-02-splash-screen-loading



## 02 Login

Login

8:28




### Log In

Please enter your email or phone number used to create your account

Phone or Email address

Continue

Or continue using

Don't have an account? [Sign Up](#)

Login

8:28




### Log In

Please enter your email or phone number used to create your account

Email address

Continue

Or continue using

Don't have an account? [Sign Up](#)

Login

8:28

### Log In




Please enter your Password

Password

[Forgot password?](#)

Login

Or continue using

Don't have an account? [Sign Up](#)

Login

8:28




### Login

Please enter your email or phone number used to create your account

Password

Continue

Or continue using

Don't have an account? [Sign Up](#)

## 04 Sign Up

05-01-sign-up-bl...

**Sign up**  
Please enter your email or phone number to create an account

Phone or Email address

Continue

By Signing up you agree to our Terms Conditions & Privacy Policy.

Or continue using

Google Facebook Apple

Already have an account? Login

05-02-sign-up-fill...

**Sign up**  
Please enter your email or phone to create an account

Email address  
Xgen

Continue

By Signing up you agree to our Terms Conditions & Privacy Policy.

Or continue using

Google Facebook Apple

Already have an account? Login

05-03-sign-up-ty...

**Sign up**  
Please enter your email or phone to create an account

Email address  
Xgen

Continue

By Signing up you agree to our Terms Conditions & Privacy Policy.

Or continue using

Google Facebook Apple

Already have an account? Login

05-04-sign-up-ot...

**Verify email address**  
Enter the 4-digit code sent to you at walka3019@gmail.com

4

Continue

Didn't receive code? Resend Again.

By Signing up you agree to our Terms Conditions & Privacy Policy.

05-05-sign-up-ot...

**Verify email address**  
Enter the 4-digit code sent to you at walka3019@gmail.com

4 7 9 3

Continue

Didn't receive code? Resend Again.

By Signing up you agree to our Terms Conditions & Privacy Policy.

05-06-sign-up-se...

**Set up account**  
Finish setting up your account

Full name

Phone number

Date of birth

Password

Create account

### c. Website Development:

#### Technology Used:

1. Webflow
2. Flask:
- 3.

Flask is a python online application framework that allows end users to interact with your python code (in this case our machine learning models) directly from their web browser without the need for any libraries, code files, or other software.

Flask makes it simple to build web apps, allowing you to spend your efforts on more critical aspects of the ML lifecycle, like as EDA and feature engineering. In this article, I'll show you how to create a simple web application using your machine learning model and then launch it.

#### Development and Deployment Process:

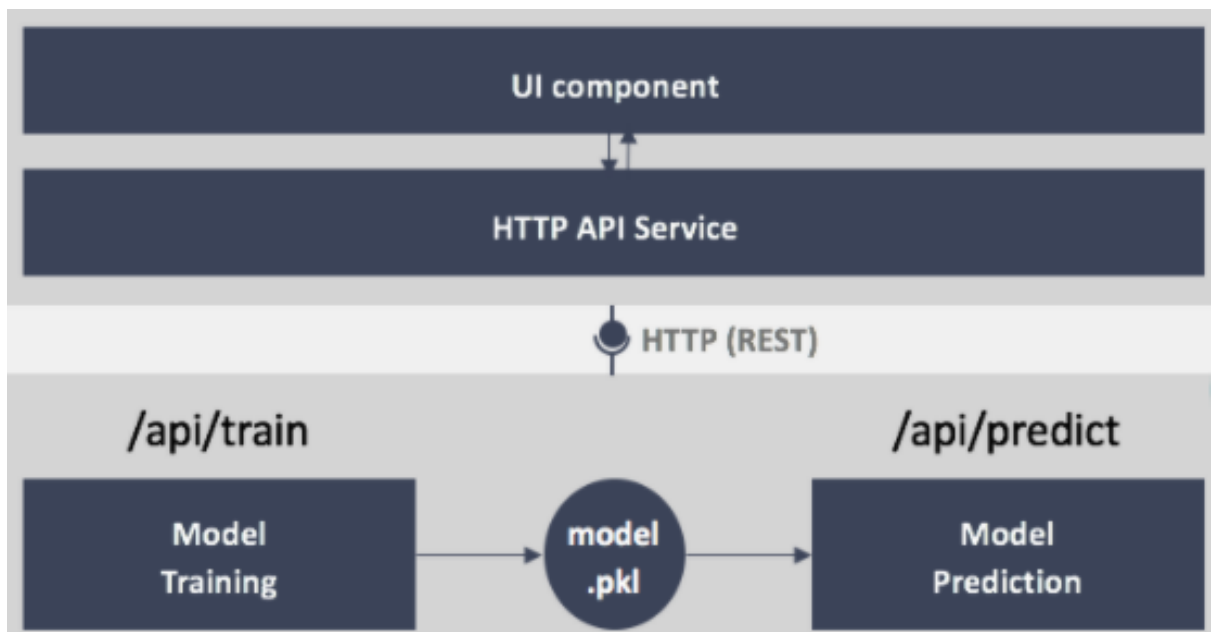
1. Make a model for machine learning.
2. Create a web app with Flask and incorporate your model into it.
3. Heroku Cloud Platform is where you may deploy your web app.
  - A. Use the 'pip install flask' command to install Flask. To create flask applications, I use the VS code.
  - B. Import required libraries, start the flask app, and load our machine learning model: We'll start by initialising our app and then loading the "model.pkl" file.
  - C. #import libraries  
import numpy as np  
from flask import Flask, render\_template, request  
import pickle  
#Initialize the flask App  
app = Flask(\_\_name\_\_)  
model = pickle.load(open('model.pkl', 'rb'))
  - D. Define an app route for the web app's default page: Routes are an app's URL patterns. @app.route("/") is a Python decorator provided by Flask that allows us to quickly assign URLs in our app to functions.  
  

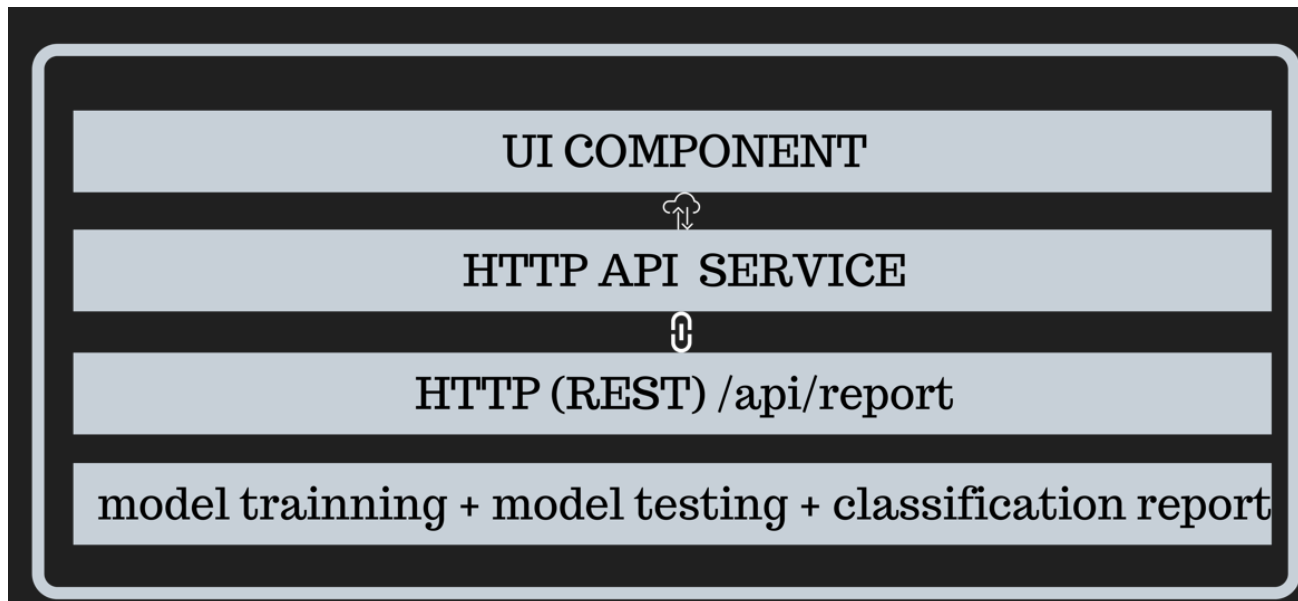
```
@app.route('/')  
def home():  
    return render_template('index.html')
```
  - E. **Templates:** This folder contains the html files (index.html, predict.html) that our main code (app.py) will use to construct the front end of our application.



- F. **app.py**: This is the primary application file, which contains all of our code and connects everything.
- G. **requirements.txt**: This file contains all of the project's dependencies/libraries (once a virtual environment is established, it can utilise this requirements file directly to download all of the dependencies; you don't need to manually install all of the libraries, just put them in this file)
- H. **model.pkl**: This is our classification model, which in this case is a Logistic Regression Model that I had previously trained.
- I. **Pickle** is a Python package that allows you to serialise and de-serialize Python object structures. Pickling, serialisation, flattening, or marshallng is the process of converting any type of Python object (list, dict, etc.) into byte streams (0s and 1s). Unpickling is a technique that turns the byte stream (produced by pickling) back into Python objects. Pickle file is is like a check point where we have educated the entire model and loaded into a file.

### Architecture:

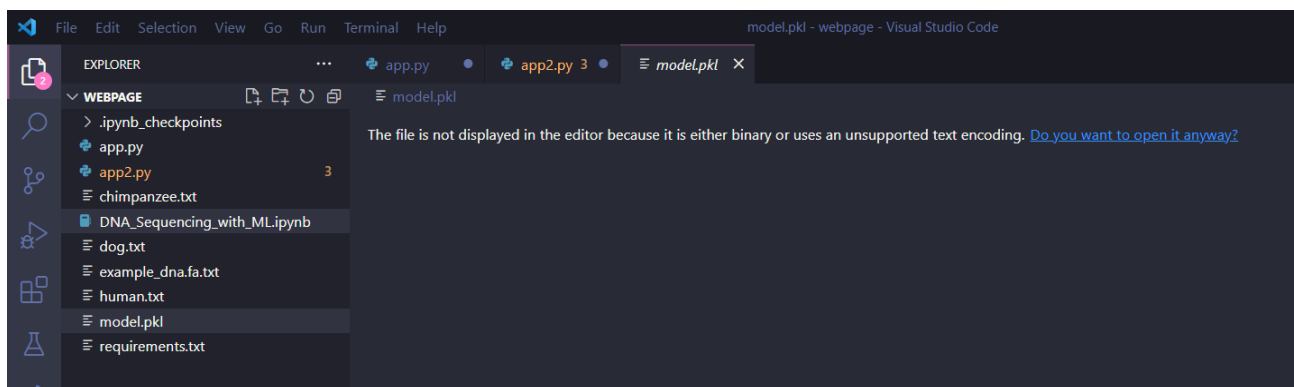
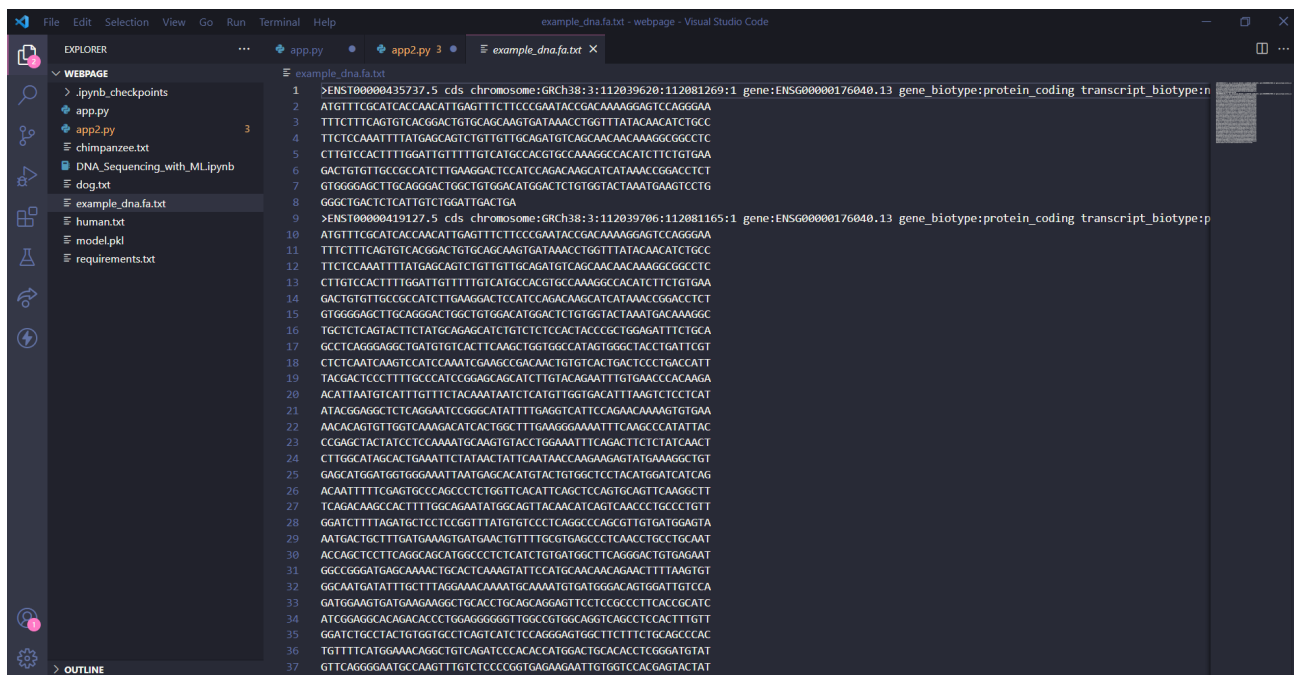
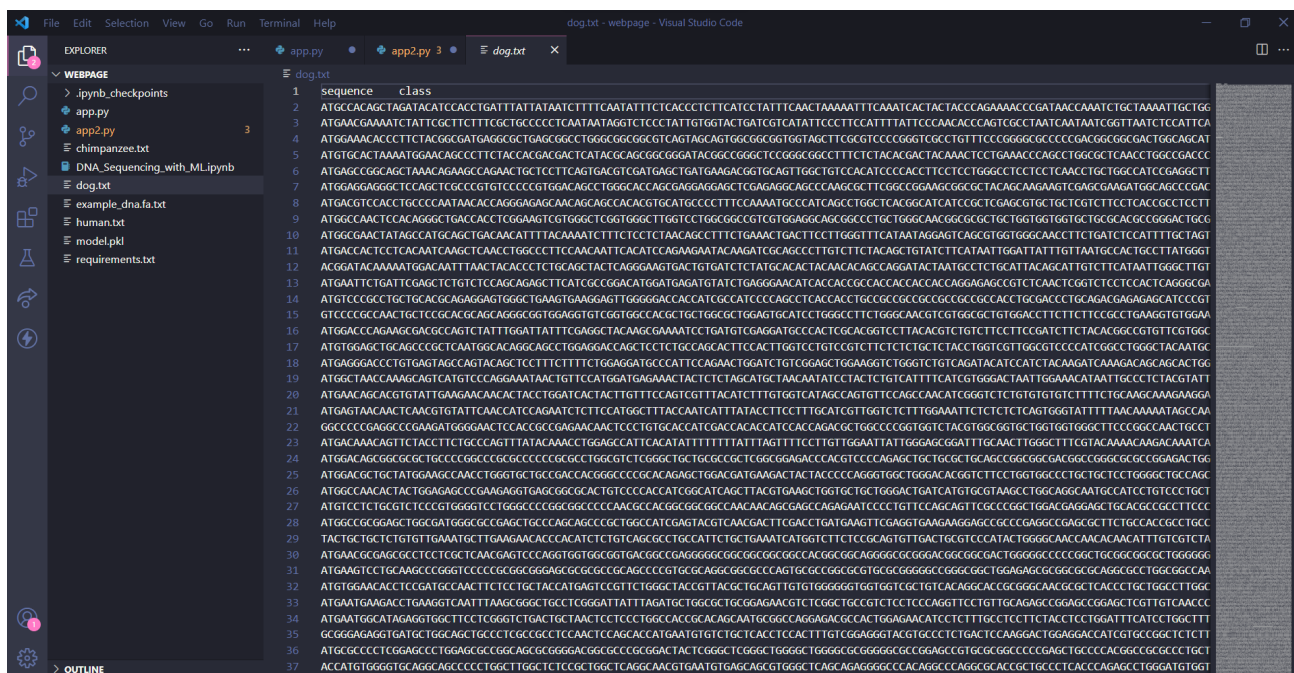


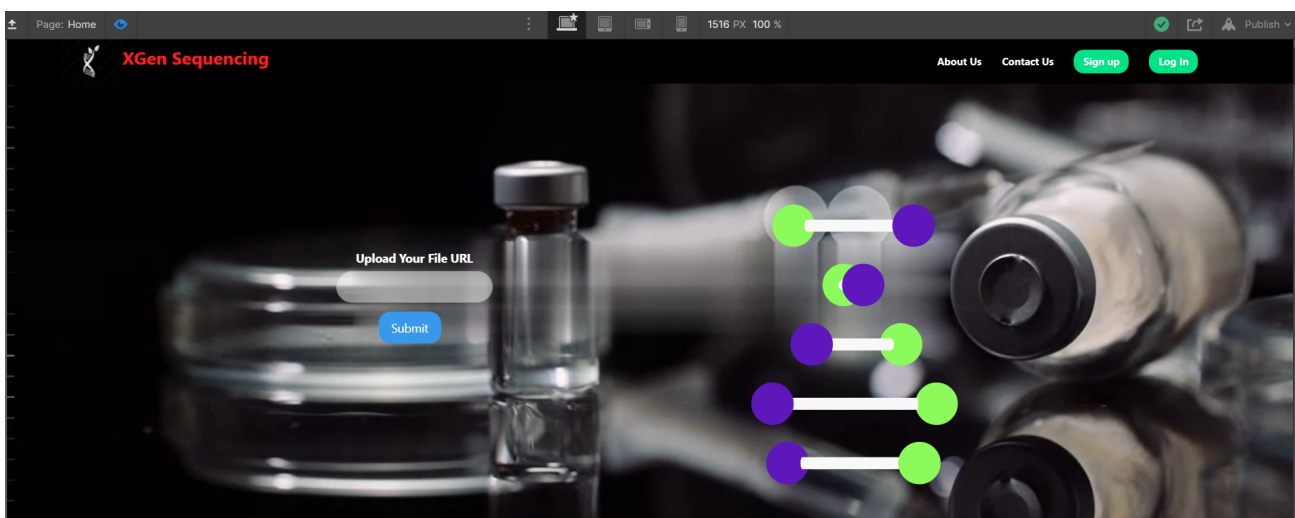
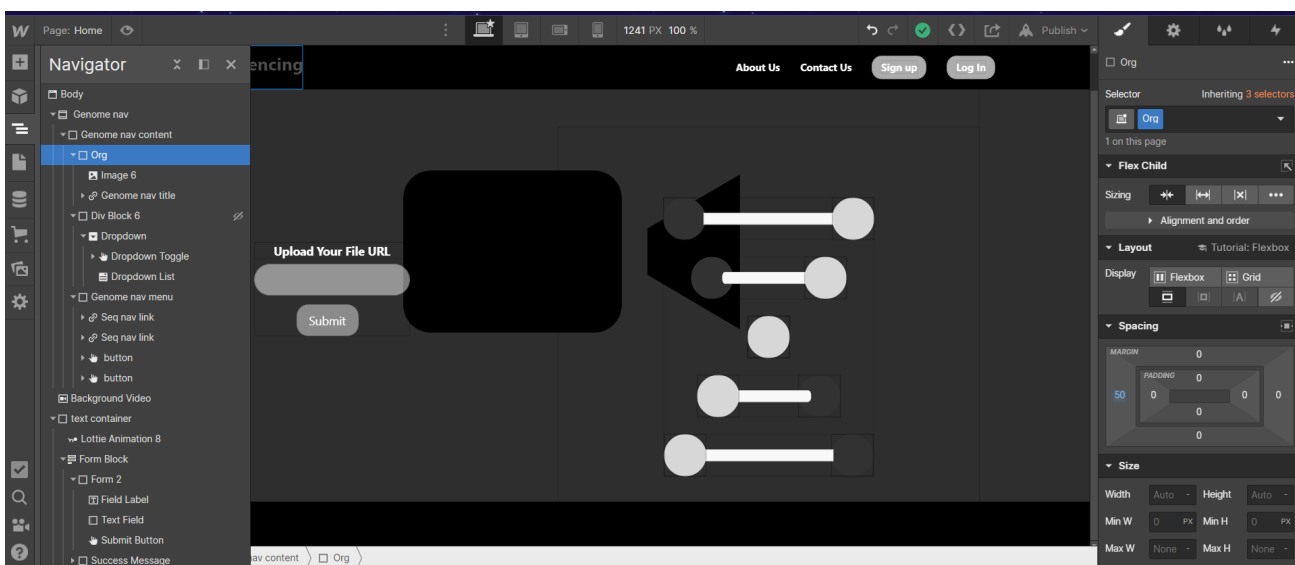
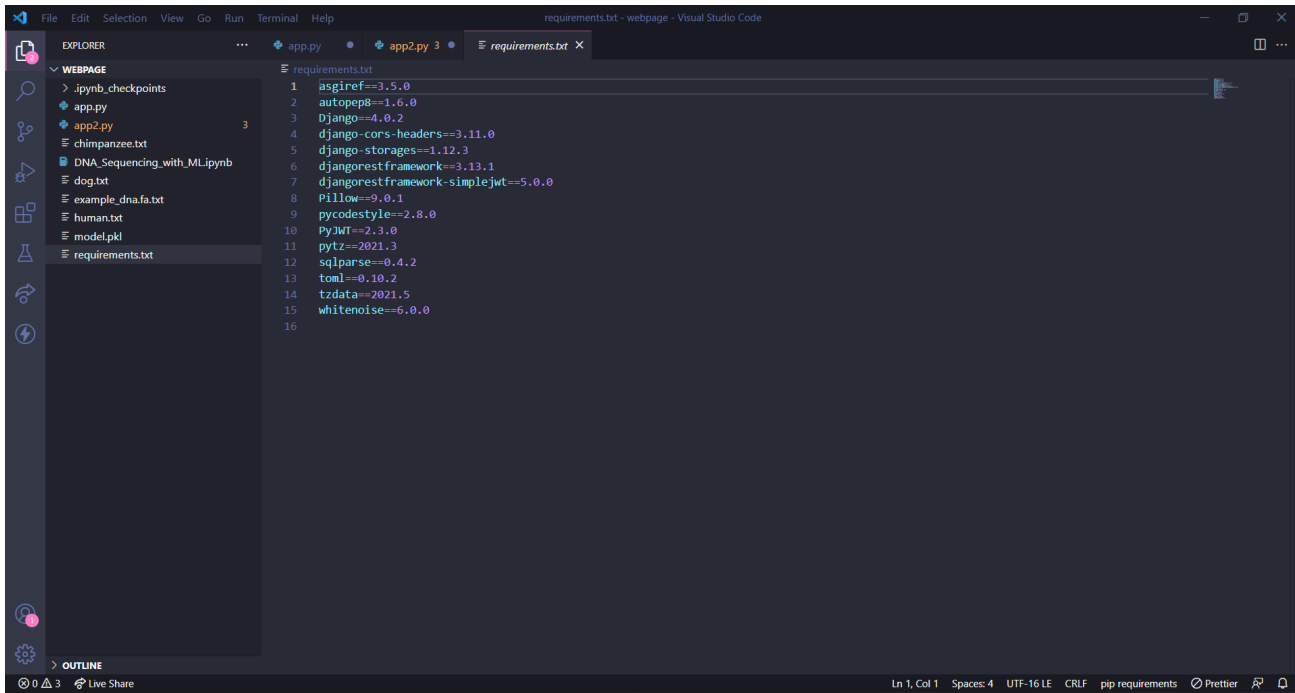


## Results and Screenshots:

```
app.py > home
1  # Using flask to make an api
2  # import necessary libraries and functions
3  from flask import Flask, jsonify, request
4
5  # creating a Flask app
6  app = Flask(__name__)
7
8  # on the terminal type: curl http://127.0.0.1:5000/
9  # returns hello world when we use GET.
10 # returns the data that we send when we use POST.
11 @app.route('/', methods = ['GET', 'POST'])
12 def home():
13     if(request.method == 'GET'):
14
15         data = "hello world"
16         return jsonify({'data': data})
17 @app.route('/home/<int:num>', methods = ['GET'])
18 def disp(num):
19
20     return jsonify({'data': num**2})
21
22
23 # driver function
24 if __name__ == '__main__':
25
26     app.run(debug = True)
27
```








<https://anikets-beautiful-site-13b10484fdb072fb.webflow.io/>




 **XGen Sequencing**


[About Us](#) [Contact Us](#)

English (IN) ▼

Create Account

Login


 Sign Up with Google

 Sign Up with Facebook

- OR -


Full Name

Email Address

Password 

Create Account

[Already have an account? Login](#)


 **XGen Sequencing**


[About Us](#) [Contact Us](#)

English (IN) ▼

Create Account


Login

 Sign in with Google

 Sign in with Facebook

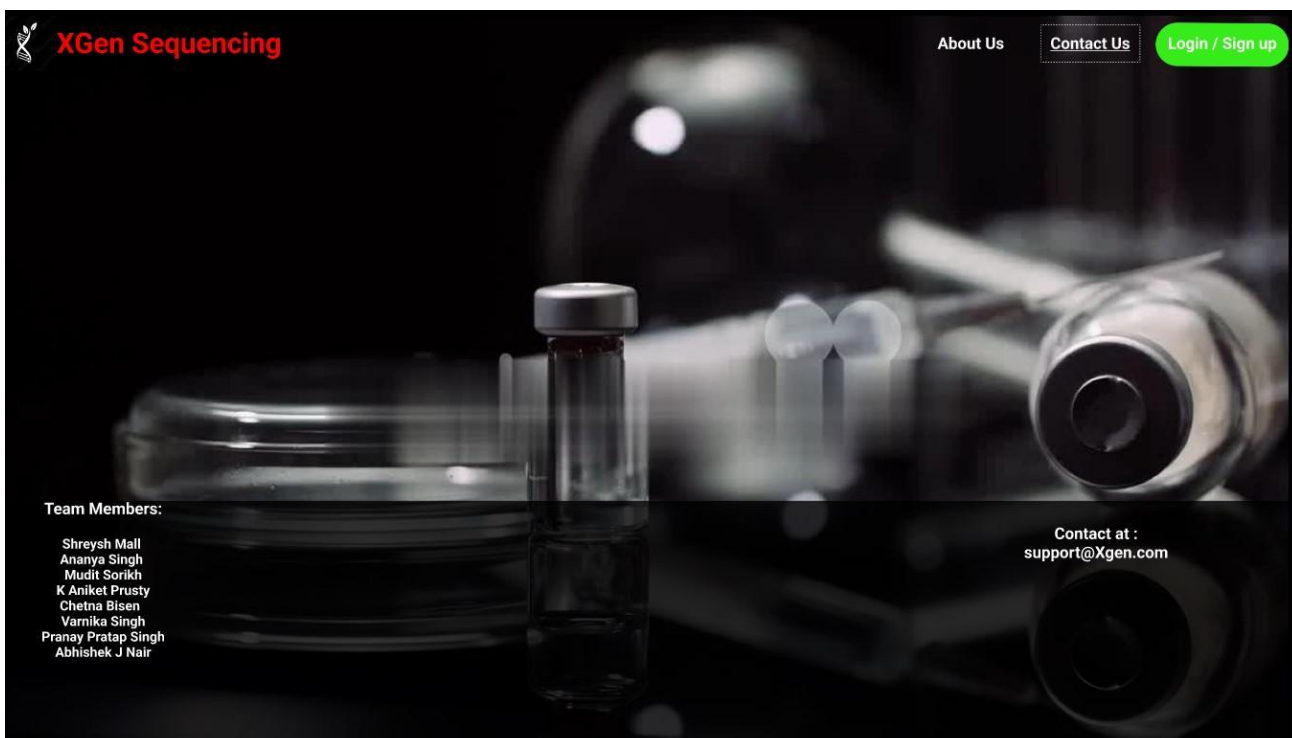
- OR -

Email Address

Password 

Login

[Forgot Password](#)



## Conclusion

In phase 1, we gathered information about several bioinformatics methods and determined that 'Multinomial Naive Bayes and Multi-layer Perceptron classifier models' is the best algorithm for developing a model based on that algorithm. We worked on all conceivable components of the prediction and deployment application throughout phase 2 development. To assure the model's durability, we set out to make it as sturdy and modular as possible, so the most crucial feature we provide to society is the capacity and agility to go forward in this rapidly changing technology environment. We are all aware that the majority of us have suffered greatly as a result of the epidemic. This project will help us get a good start because of its quicker clinical approaches, such as ease of sampling and results display, as well as the addition of Machine Learning, which will make the process faster and more accurate. There can be a lot of reports and a lot of quicker sequencing to detect novel mutations, especially deadly ones.



## Reference

1. <https://towardsdatascience.com/machine-learning-for-genomics-c02270a51795?gi=e4e0c90e2e7b>
2. <https://www.coursera.org/learn/dna-sequencing>
3. <https://link.springer.com/article/10.1186/s40364-017-0082-y>
4. <https://www.nature.com/articles/nrg3920>
5. [https://www.researchgate.net/publication/255395208\\_Machine\\_Learning\\_in\\_Computational\\_Biology](https://www.researchgate.net/publication/255395208_Machine_Learning_in_Computational_Biology)
6. <https://computationalgenomics.blogs.bristol.ac.uk/resources/software>
7. <https://www.ncbi.nlm.nih.gov/books/NBK20261/>
8. <https://www.nature.com/articles/nature04072>.
9. [https://www.researchgate.net/publication/353291346\\_Analysis\\_of\\_DNA\\_Sequence\\_Classification\\_Using\\_CNN\\_and\\_Hybrid\\_Models](https://www.researchgate.net/publication/353291346_Analysis_of_DNA_Sequence_Classification_Using_CNN_and_Hybrid_Models)