A Project Report

On

PREDICTIVE ANALYSIS FOR BRAIN CANCER DETECTION USING MACHINE LEARNING TECHNIQUES

Submitted To

Amity University Uttar Pradesh

In partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

By

Siddharth Mittal

Under the Guidance of

Ms. Nidhi Chandra

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

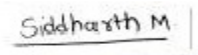
AMITY UNIVERSITY UTTAR PRADESH, NOIDA

May-June 2022

DECLARATION

I, Siddharth Mittal, student of B.Tech (CSE) hereby declare that the project titled "Predictive Analysis for Brain Cancer Detection using Machine Learning Techniques" which is submitted to Department ofComputer Science and Engineering, Amity School of Engineering and Technology, Amity University Uttar Pradesh, Noida, in partial fulfillment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering, has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition.

Noida



Date Signature of Project Team

CERTIFICATE

On the basis of declaration submitted by SIDDHARTH MITTAL, student of B.Tech (CSE), I hereby certify that the project titled "**Predictive Analysis for Brain Cancer Detection using Machine Learning Techniques**", which is submitted to Department of Computer Science & Engineering, Amity School of Engineering & Technology, Amity University Uttar Pradesh, Noida, in partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science & Engineering is an original contribution with existing knowledge and faithful record of work carried out by him/her under my guidance and supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Noida

Date

Ms. Nidhi Chandra
Department of Computer Science and Engineering
Amity School of Engineering and Technology
Amity University Uttar Pradesh, Noida

ACKNOWLEDGEMENT

It is high privilege for me to express my deep sense of gratitude to those entire faculty members who helped me in the completion of the project, specially my internal guide who was always there at hour of need. My special thanks to all other faculty members, batchmates and seniors of Amity University Uttar Pradesh for helping me in the completion of project work and its report submission.

Siddharth Mittal
A2305218422

ABSTRACT

Cancer is the 2nd most common cause of death with over 10 million deaths annually worldwide. Brain Cancer, a type of cancer affecting the brain, is reported to have affected more than 1 million people every year alone. It is also the 10th most common kind of tumour found amongst Indians. Studies done by the International Association of Cancer Registries, set the death toll of the Indian population due to brain cancer at more than 30,000 every year. These numbers represent the significant impact of this ailment. With this major impact and a very low 5-year survival rate, Brain tumours present to be the most acute disease with only early diagnosis helping the patient. This report explores the different Machine Learning algorithms employed over the years for early detection of Brain Tumours. The experimental analysis of the different algorithms was carried out to review their performance and results are presented as infographics for ease of understanding. The project also aims to give other users access to process their own MRI images of the brain, and compare the predictions made by the different algorithms used throughout the project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The unusual and uncontrollable cell growth in the brain is referred to as Brain Tumour. Brain tumours cause more pressure inside the skull as they take up the restricted space in the skull. The brain is surrounded with bones of the skull, which means it cannot inflate further to make space for the unusual cell growth which results in the unusual cells squashing the normal brain tissues which leads to life-threatening complications. Brain tumours can be segmented into primary and secondary. Primary Brain Tumours are tumours which originate in the brain and can be either benign or malignant. Secondary Brain Tumours, also termed as metastatic, grow in some other part of the body and spread to the brain through the blood. Secondary brain tumours are always cancerous as they spread in an evolved stage.

Brain tumours are threatening by nature because of the lack of knowledge about how they are caused and their prevention methods. Detecting the tumour early and starting with the diagnosis is what can save the patient's life as well as considerable reduce the cost of the however expensive treatment. People exposed to detrimental chemicals and industrial solvents for long times as well as patients having cancer in some other parts of the body are in peril the most. To spot the different irregularities in the body caused due to cancerous cell growth, Medical Imaging Analysis plays a significant role. To detect the unusual cell growth and malformed cells in the brain, MRI or Magnetic Resonance Imaging is used. This process uses magnetic fields to construct an image of the body part under question. This method can also be used to categorize the tumour as being as being primary or secondary which helps in further treatment procedure. This report experimentally analyses supervised machine learning algorithms that are used to detect brain tumours in MRI images and presents the results in a creative UI application.

2. LITERATURE REVIEW

Various sources of relevant literature were referred to understand the current state of machine learning techniques for brain cancer detection and is presented in this section.

[1] "Recent advancement in cancer detection using machine learning: Systematic survey of decades, comparisons and challenges" presents a comparative analysis of several state-of-the-art ML algorithms for different types of cancers including brain cancer, breast cancer, lung cancer, liver cancer, leukemia, etc. The experimental analysis of the different forms of cancers are done on their respective benchmark datasets. These datasets are also defined with links provided to where they can be found. The author also presents as to why these datasets are considered to be the benchmark in their fields giving due credit to the different Universities responsible for producing them. All of the mentioned datasets have labels defined by physicians which are passed to the algorithm and used as a parameter in the learning process. The author starts of by defining the cancers and what makes them chronic. They also define a generalized process pipeline which needs to be followed for every algorithm irrespective of the cancer type. This process includes: Pre-processing of the dataset, Tumor Segmentation, Feature Extraction and then Classification. This general set of steps are followed by the experimentalists while testing the different algorithms.

In the section on Brain Tumors, the author starts by discussing the various pre-processing techniques applicable to MRI Images of Brain, including Brain Surface Extractor, Partial Differential Diffusion Filter, Wiener Filter and Fast Non-Local Mean Technique. Then they go on to mention feature extraction techniques like histogram orientation gradient and local binary patterns. To choose the appropriate parameters for the model, the author describes the use of algorithms like Principle Component Analysis (PCA) and Genetic Algorithm (GA). Now that the images are ready to be fed into the algorithms as inputs, all the machine learning algorithms employed are mentioned, recording their output metrics including Accuracy, Precision, Recall and Specificity. The algorithms range from older SVM techniques to more advanced CNNs using methods like transfer learning.

[2] "A Supervised ML Applied Classification Model for Brain Tumors MRI" by Zhengyu Yu, Qinghu He, Jichang Yang and Min Luo looks into supervised machine learning algorithms and compares them based on factors found in the confusion matrix for Brain

Cancer detection. It starts of by introducing cancer and brain cancer and how MRI is one of the only Medical Imaging Analysis Techniques that is most useful for its early detection and diagnosis. It then defines the dataset that the authors used for the study, namely REMBRANDT, which can be found at The Cancer Imaging Archive (TCIA) database. The dataset is made up of MRI images of 130 patients and has around 110,000 images. The authors go onto introducing the 4 algorithms that were used for the study which include the DT Classifier, SVM, KNN and NN. The 4 factors on which the algorithms are then defined namely, Accuracy, Precision, Sensitivity, and F1-Score. The formulas for these factors are also described to show how they extracted them from the confusion matrix of the algorithms. The authors go on to describe the individual results of all 4 algorithms with tables defining the results they achieved. The authors conclude with their choice of the model, the DT classifier, which turns out to be the highest accurate model for the chosen dataset.

[3] "Brain tumor detection and classification using machine learning: a comprehensive survey" presents the findings of the author in a survey-manner. It provides all the important literature that the authors went through and the pros, limitations and developments of the algorithms reviewed for the project. The authors start by describing brain tumors and stroke lesions. The different grades of brain tumors are then defined for the clarity of the reader. The paper takes an extensive look into other Medical Imaging analysis techniques like Positron Emission Tomography (PET), Computed Tomography (CT), Magnetic Resonance Imaging (MRI) and Diffusion Weighting imaging (DWI). Since the paper goes into Deep learning methods as well, the authors describe the various segmentation and image pre-processing methods that were used during the implementation of the project. The factors used are also defined along with a list of publicly available datasets. In short, the survey covers aspects and recent work done so far with their impediments and drawbacks for the field of detection of brain cancer using machine learning techniques. It lists the limitations of the existing models being used like the intricate task of removal of noise from the MRI images before they are fed to the algorithms, as well as selecting and extracting the optimal features along with pertinent number of training/testing samples.

[4] "Brain Tumour Detection based on Machine Learning Algorithms" by Komal Sharma, Akwinder Kaur, and Shruti Gujral, compares two algorithms, namely Multi-Layer Perceptron

(MLP) and Naive Bayes. This paper lays the foundation to understand the methodology process as it explains all the steps involved in such algorithms in a detailed manner. The paper also explains pre-processing operations and feature extraction operations that can be used to solve such problems. It works on around 210 samples and compares the results of the 2 algorithms based on their classification rate and model build time. It describes feature extraction methods like Gray Level Co-occurrence matrix (GLCM). The texture features that are described include energy, contrast, correlation, and homogeneity.

[5] The React documents on the official ReactJS page provides a great tutorial to learn the basic as well as advanced topics of react. It introduces the syntax used in react and which has to be used while building a react project. It helps get familiar with the starter code generated by npx when a react app is created. The file hierarchy is defined which makes it easier to understand where to make changes and where all the dependencies go. It also links to the community support resources which can be utilized whenever some question arises or one gets stuck. Another topic very useful to understand react is props (or properties) which are clearly defined and taught in the docs. The docs introduces us to react hooks like useState and useEffect which are majorly used in the project. It teaches how react utilizes a Virtual DOM and how and when it re-renders the interfaces. One of major functions of JavaScript i.e. providing interactivity to the interfaces is also explained clearly along with how state of object gets passed. Lastly, it also explains how the front-end code built using react should be tested to detect bugs and fix them. Testing is an important part for any code, as it helps weed out malfunctioning of the software and fix bugs before they get into production.

[6] The D3js documentation helps understand the variety of infographics that can be made using the library. This is a good starter path to learn the different types of charts, graphs, maps, and diagrams that can be made.

[7] The Flutter Docs provides great documentation to learn the flutter framework. It also helps first understand the Dart language which is used to create flutter applications. The Dart developer tutorial helps understand from the basic to advanced concepts involved in dart. They help realize dart fundamentals using references to other languages so that users already familiar with other languages have something to compare it to and learn the language faster. The flutter tutorial helps learn the fundamentals of Widgets: Stateless and Stateful widgets. This helps understand the process of re-rendering the interface of the app. This makes it

easier to relate it to the native level apps. The docs also carry all the Widgets that can be used to build a mobile application along with examples. This helps understand the flutter application building process in an efficient way. The documentation also explains the colour schemes that can be used in flutter along with the package management system used by flutter. It explains the yaml file that contains the dependency list and shows how to edit it for the developer's use. It also shows how to add assets like images and fonts to the project and how to use them to create a custom interface.

[8] freecodecamp provides great free courses to learn react and d3. The d3 curriculum on the freecodecamp website helps understand and build simple charts and graphs using vanilla JavaScript. The 2-part D3 course by Curran Kelleher hosted on freecodecamp's YouTube channel helps combine react and d3 knowledge into one to create modern and reactive infographics using d3. This helps provide dynamic retrieval of data to create charts and graphs on the fly. It also helps create dynamic user interfaces so that the application provides interactivity to the user.

3. FRAMEWORKS AND TOOLS:

3.1 Machine Learning Frameworks:

For the Machine Learning part, python was used as the primary language because of its support of extensible libraries which provide a wide variety of tools to tackle machine learning problems. It also comes packaged with plotting libraries that help plot input and output points of the ML algorithms for better understanding using visual aid. Other tools utilized in tandem with python for the machine learning implementation part are:

i) Jupyter Notebooks: Jupyter notebooks is a web-based platform which is most useful in machine learning implementation as it provides a birds-eye view of the situation. It provides the power to understand the variables being used as well as the series or dataset in question. This basically acts as debuggers used in other programming languages as it helps analyse the program at different points of time and easily make changes to make the program function as wanted.

ii) Scikit-learn: scikit-learn is a python ML library that provides tools to make machine learning implementation easier. Instead of creating models from scratch on our own, it provides these models which just needs the input data to fit it according to the problem and give the solution. It also provides common methods that can be used on these models to get results for the algorithms using one-line of code.

iii) OpenCV: OpenCV is a third-party library provided for python using the pip package manager. This library basically provides tools and functions for computer vision tasks. Its functions can be used to detect objects on an image as well as process images before they are entered as input to any ML algorithm. The project uses OpenCV functions to read the brain MRI images and resize them. Since the supervised algorithms do not require much pre-processing, only the resizing feature of OpenCV is used.

The jupyter notebooks describing the implementations of the algorithms and the dataset are all stored on github for transparency and ease of access.

3.2 Web Development Frameworks:

For the website design and development part, React, a front-end JavaScript framework was used. Developed at Facebook, it helps ease the development of single-page applications using JavaScript, HTML and CSS. To display the results, a third-party JavaScript library of D3.js was used. The back-end API which helps provide user input functionality to the website and application are built using Node.js, Express which are JavaScript frameworks. These frameworks are just used to create the endpoints, the actual code which processes the input images is written in python.

i) - React: React is a JavaScript client-side framework which makes it easier to build SPAs (single-page applications) for the web using JavaScript, HTML and CSS. Developed by the engineers at Facebook (now Meta), it helps simplify the task of building interactive front-end user interfaces. It also has access to third-party libraries which makes it easier to communicate with the back-end to make full-fledged web applications.

ii) - D3.js: Another JavaScript library which is used in the project. This helps in manipulating UIs based on data. It basically provides tools to make mathematical computations easier on data, which can be further used to create infographics like bar charts, scatter plots, line charts, donut charts, and the like.

iii) - Node.js: Neither a framework nor a library, Node.js is a JavaScript runtime built on a JavaScript engine. This is used in the project to create the back-end API. It makes it easier to create endpoints for the API, that can be easily accessible by the front-end client website and the mobile application.

The source code for the website and the different data being displayed on the website is also stored on github. The website and the API are deployed using the Heroku platform.

3.3 Mobile Development Frameworks:

For the mobile application development, Google's Flutter framework was used as it provides cross-platform development. Other third-party libraries provided in flutter are also used for ease of development.

i) Flutter: An open-source framework by Google, Flutter is used to create cross-platform applications. It helps overcome the cons of other cross-platform modules and get closer to the processing power of native mobile apps.

ii) http: This is a third-party library for Flutter applications which makes it easier to fetch results from the back-end API and communicate with the web.

iii) charts_flutter: Similar to D3.js, this is a third-party library developed for Flutter applications. Makes it easier to create infographics like bar charts and donut charts to represent the data and provide visual aid.

iv) image_picker: This module is used in Flutter applications to provide system calls at the root level so that the application can get access to the gallery and the user's camera to upload an MRI image as input.

3.4 Other Tools Used:

The general-purpose tools used throughout the project are:

i) Github: GitHub uses the inclusive technology of Git, an open-source project, for version control of software projects and managing and storing revisions of projects. GitHub helps in collaboration among team members for big projects for active software development. The team used GitHub to develop the website and keep track of the machine learning implementations. Through GitHub and its Github Gists, we were able to host datasets and our results. These datasets are then referenced and called on the website and application.

ii) Heroku: Heroku is a container-based service that helps developers build and run web applications on the cloud. It helps spin up servers for free provided that they may work slower than other paid distributors. This is used to host the website as well as the back-end API. Hosting the API provides us an endpoint to call everywhere. This is utilized in the mobile application as well.

4. METHODOLOGY

The brain of a human is modelled through application and design of the neural network. Brain images that are extracted from the MRI Image are in the first level and any slice in the area is then segmented to get tumors.

4.1 MACHINE LEARNING IMPLEMENTATION:

After thorough research about supervised machine learning algorithms previously used in the industry for brain cancer detection from research papers, 6 algorithms were chosen for experimentation for the project. The 6 algorithms include Logistic Regression, Support Vector Machine, K-Nearest Neighbours, Naive Bayes algorithm, Decision Tree Classification, and Random Forest algorithm. All the algorithms fall in the category of Classification meaning they are used to solve classification problems. The 4 factors of Accuracy, Precision, Recall and Processing Time were chosen to analyse these algorithms on and compare the results. The dataset chosen consists of brain MRI images of 4 classes. The 4 classes include no tumour, glioma tumour, meningioma tumour and pituitary tumour. The input images do not require much pre-processing as the supervised learning algorithms only require simple matrices defining the images. For these functions, OpenCV can be used which is a third-party python module. The resized images are then transformed to matrices representing the RGB values and finally passed to the algorithms for fitting and prediction.

Logistic Regression is one of the oldest and most simple classification algorithms. It is used to classify the input data points into output classes. The algorithm was developed by Joseph Berkson and works using logit functions. The model can be imported from the scikit-learn library and used out of the box. The input points needs to be pre-processed so they fit the model perfectly.

```
from sklearn.linear_model import LogisticRegression
lg = LogisticRegression(C=0.1)
lg.fit(xtrain, ytrain)
print("Training Score:", lg.score(xtrain, ytrain))
print("Testing Score:", lg.score(xtest, ytest))
start = time.time()
pred = lg.predict(xtest)
end = time.time()
print("Time:",end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision:",precMa)
print("Recall:",recMa)
```

```
Training Score: 1.0
Testing Score: 0.759581881533101
Time: 0.06283450126647949
Precision: 0.7590645955351838
Recall: 0.7560042567585671
```

Fig 4.1: Implementing Logistic Regression on the chosen dataset

Support Vector Machine (SVM) is a traditional ML algorithm which is one of the best algorithms that fall under the classification ML algorithms. It works by creating boundaries between the different data points based on the output class. This decision boundary is called hyperplane and is used to segregate the data points properly. New input points are then classified according to this boundary and given an output class. This algorithm still performs at par in comparison to the latest complex problems in terms of accuracy and precision, but takes a huge amount of processing time. The SVM model can be imported from the scikit-learn package and can be tuned according to the problem at hand using parameters that can be set for the algorithm.

```
from sklearn.svm import SVC
svm = SVC()
svm.fit(xtrain, ytrain)
print("Training Score:", svm.score(xtrain, ytrain))
print("Testing Score:", svm.score(xtest, ytest))
start = time.time()
pred = svm.predict(xtest)
end = time.time()
pritn("Time: ", end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision:", precMa)
print("Recall:", recMa)
```

```
Training Score: 0.9299119111451551
Testing Score: 0.8346094946401225
Time: 118.37529706954956
Precision: 0.8369716980345494
Recall: 0.8729093461428791
```

Fig 4.2: Implementing SVM algorithm

KNN or K-Nearest Neighbours is another basic supervised algorithm used to solve classification problems. The basic idea behind the algorithm is to assume similarity or resemblance between the data points and then group them under output classes. Since it is a supervised algorithm and uses the given dataset and training output points to train the algorithm, the algorithm uses it to find the likeness between the data points. It is not much efficient in terms of image classification and does not give good results.

```
from sklearn import neighbors
clf = neighbors.KNeighborsClassifier()
clf.fit(xtrain, ytrain)
print("Training Score: ",clf.score(xtrain,ytrain))
print("Testing Score: ", clf.score(xtest,ytest))
start = time.time()
pred = clf.predict(xtest)
end = time.time()
print("Time:",end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision:",precMa)
print("Recall:",recMa)
```

```
Training Score:  0.8575258521639219
Testing Score:  0.7718223583460949
Time: 1.7676193714141846
Precision: 0.7659961858886672
Recall: 0.7715966710856849
```

Fig 4.3: KNN Algorithm Implementation and Results

Naive Bayes classification algorithm is another supervised learning algorithm which as the name suggests is used to solve classification problems and uses the Bayes theorem. The word “Naive” means that it uses the Bayes theorem in a naive way as it helps get rid of the bias and variance which cause problems of under-fitting and over-fitting.

```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(xtrain, ytrain)
print("Training Score:", gnb.score(xtrain, ytrain))
print("Testing Score:", gnb.score(xtest, ytest))
start = time.time()
pred = gnb.predict(xtest)
end = time.time()
print("Time: ", end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision:",precMa)
print("Recall:",recMa)
```

```
Training Score: 0.5381080045959402
Testing Score: 0.5053598774885145
Time:  1.50433349609375
Precision: 0.5405972045045856
Recall: 0.5166759871026606
```

Fig 4.4: Naive Bayes algorithm in action

Decision Tree method is an advanced and complex ML algorithm used to solve classification problems by creating a tree-like structure. The tree consists of nodes which are the data points and weights and the branches which act as the conditions. The algorithm tries to traverse the tree through the different branches to understand the input data point and its output class. Through this the algorithm learns and fits the dataset to the model. This is a much recent algorithm compared to the above 4 algorithms and provides a better result in terms of accuracy, precision as well as processing time. This algorithm may not be able to perform against the modern deep learning models but gives above average results.

```
from sklearn.tree import DecisionTreeClassifier
clf= DecisionTreeClassifier(criterion='entropy', random_state=0)
clf.fit(xtrain, ytrain)
start = time.time()
pred = clf.predict(xtest)
end = time.time()
print("Training Score:", clf.score(xtrain, ytrain))
print("Testing Score:", clf.score(xtest, ytest))
print("Time: ", end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision: "  + str(precMa))
print("Recall: "  + str(recMa))
```

```
Training Score: 1.0
Testing Score: 0.8376722817764165
Time:  0.08507418632507324
Precision: 0.8500785779723781
Recall: 0.8369716980345494
```

Fig 4.5: Above Average results returned by the Decision Tree Classifier

Random Forest algorithm is another modern and complex supervised learning algorithm that was used in the experimentation analysis in the project. The algorithm uses a forest (or combination) of algorithms. This concept is better known as Ensemble Learning. This combination of multiple classifiers helps solve complex problems like object detection in images improves the overall performance of the model. The classifier works through various Decision Tree Classifiers on subsets of the dataset and uses the average to improve the overall prediction accuracy of the model.

```
from sklearn.ensemble import RandomForestClassifier
clf= RandomForestClassifier(n_estimators= 10, criterion="entropy")
clf.fit(xtrain, ytrain)
print("Training Score:", clf.score(xtrain, ytrain))
print("Testing Score:", clf.score(xtest, ytest))
start = time.time()
pred = clf.predict(xtest)
end = time.time()
print("Time:",end-start)
precMa = precision_score(ytest,pred,average='macro')
recMa = recall_score(ytest,pred,average='macro')
print("Precision:",precMa)
print("Recall:",recMa)
```

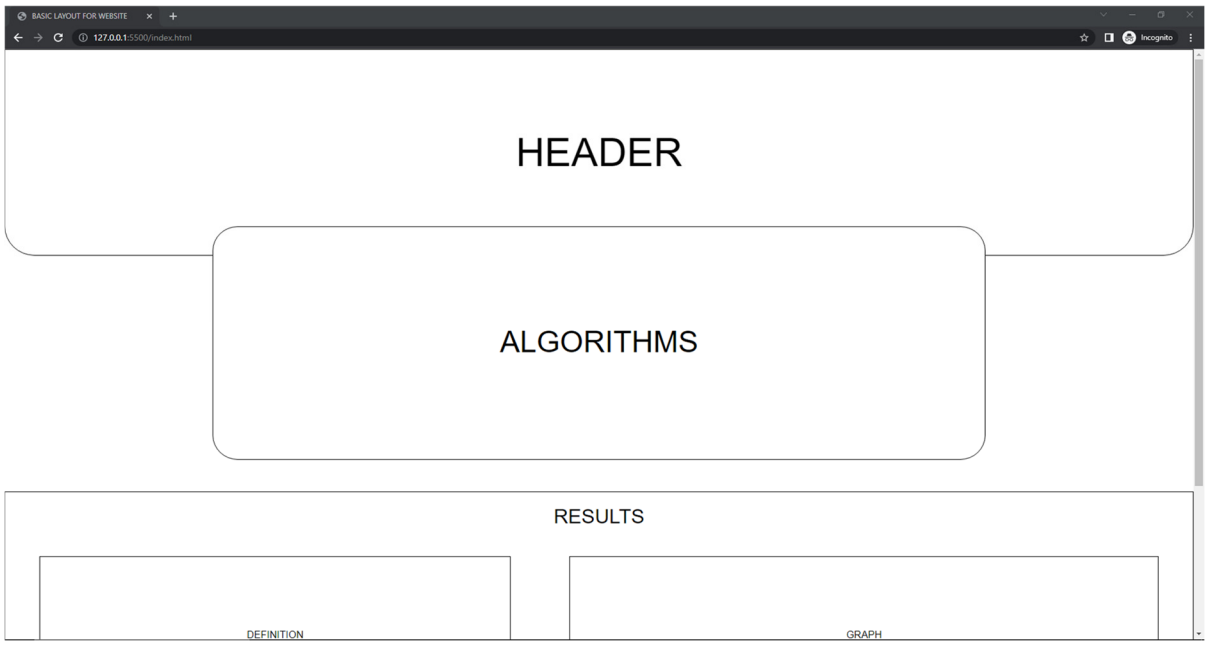
```
Training Score: 0.9969360398314822
Testing Score: 0.8728943338437979
Time: 0.099090576171875
Precision: 0.8729093461428791
Recall: 0.8707151603293658
```

Fig 4.6: Random Forest algorithm made up of 10 classifiers

4.2 WEBSITE DEVELOPMENT:

For the website and mobile application development Agile methodology was followed. This helped segregate the development process into many phases with every week being treated as a sprint. There was one meeting for every sprint which acted as both sprint planning and sprint review. The group members were informed on what was planned to be done during the next sprint as well as caught up with the progress made.

The starting couple of sprints were used for creating a basic layout of the website. The different interfaces were discussed and ideas on how to present the Machine Learning results



were brainstormed.

Fig 4.7: Initial Layout Design for the website

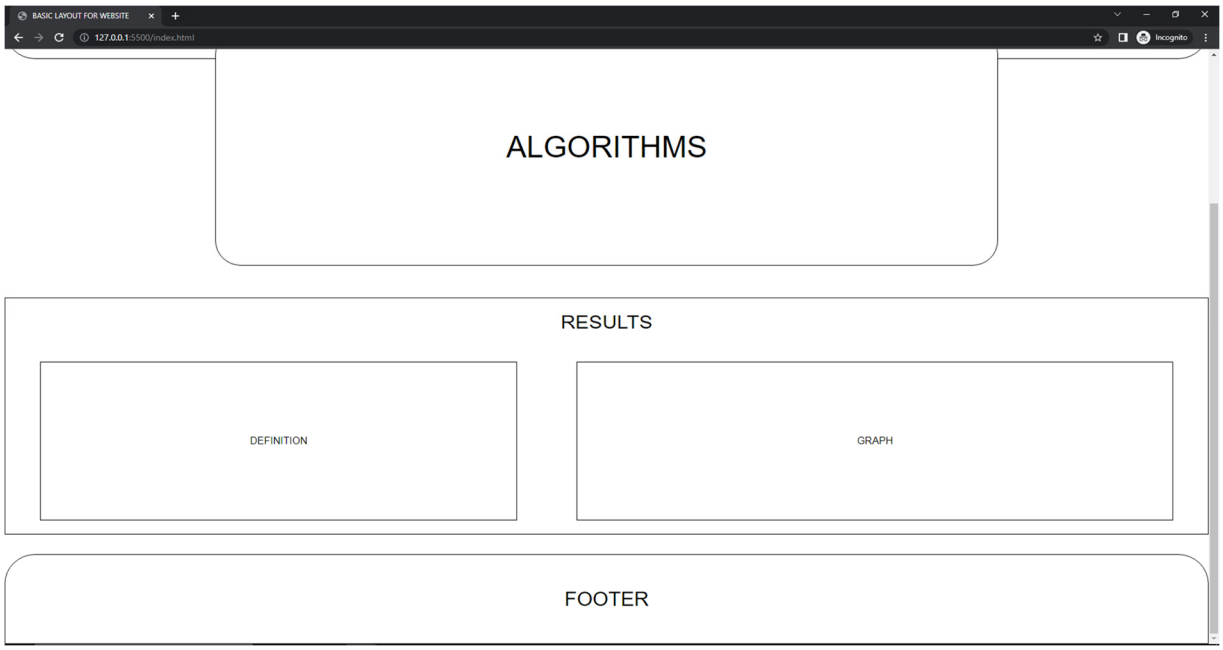


Fig 4.8: Continued Initial Layout Design

The next sprints started with choosing the front-end framework for the website development as well as starting with website development. This consisted of designing the basic layout using HTML and CSS and making some design changes along the way. The static parts of the

website were designed first like the Header and Footer. The initial designs of the header and footer changed quite a lot of times during the development cycle because of feedback and design choices. This is what defines agile methodology and makes it so powerful as it includes incremental updates of the project upon feedback.

Starting with the dynamic part took a little while as some functional choices had to be made regarding the API lookup. The project backlog started with data retrieval functionality as when the data is available, the designing part of the components is made easy. React hooks were used to retrieve and set data, particularly useState and useEffect. The data retrieved from the endpoints was also pre-processed so that it can be more easily accessible during the design phase. This pre-processing consisted of type casting data types as well introducing new data points that may be relevant later on. It would help lower the time taken for the designing phase.

The designing phase started with displaying of results of the algorithms as infographics as this was the main objective of creating web and mobile applications. To help create infographics from data points D3js, a JavaScript library, was used as it provides various tools and functions to solve the mathematical and computational part of the graphs and charts and model them onto svg's. To make the process easier, only the accuracy results were first graphed to ease into the library and understand it's basics. After thorough understanding, the bar charts were made for the other factors as well. User interactivity was also added using React and simply JavaScript. This included filtering between factors, hover to show more information, and filtering between efficient and inefficient results.

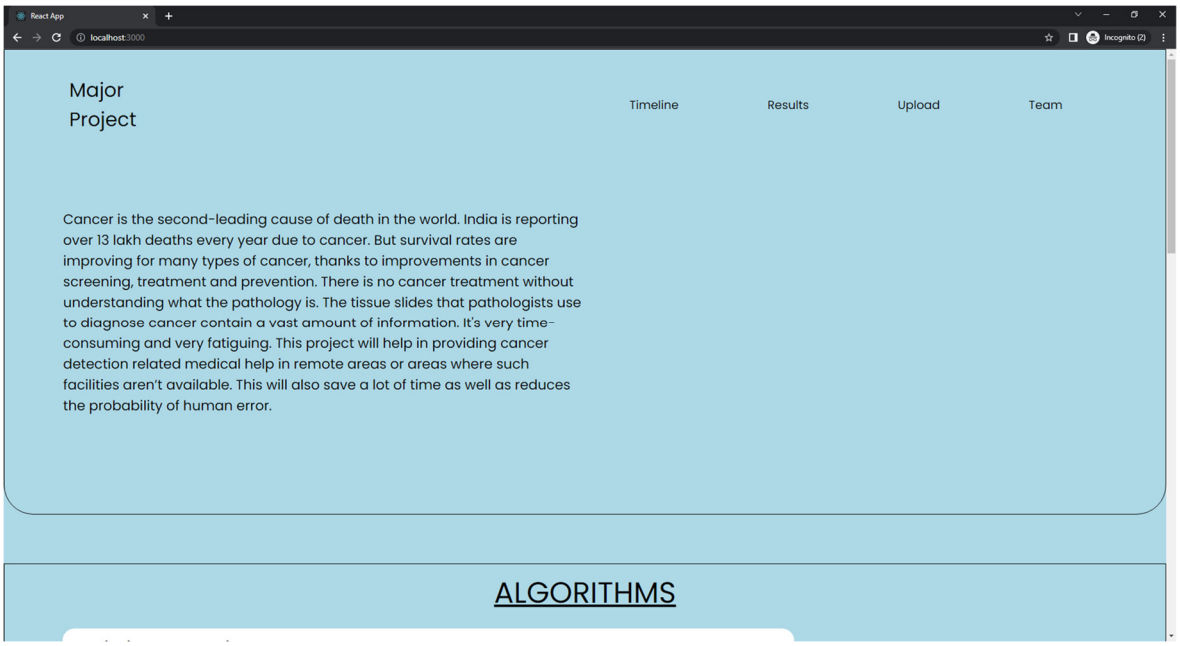


Fig 4.9: First fully developed version of the website

Fig 4.10: Results and Upload section of first iteration

During all these sprints, changes were being made on the existing sections upon feedback and minor bugs were being resolved. This is one of the main practices followed under Agile methodology and was very useful during the development stage.

Focus was also to display overall results of the machine learning implementation in a tabular form for a general comparison amongst the algorithms. User interactivity was added for the section describing the algorithms.

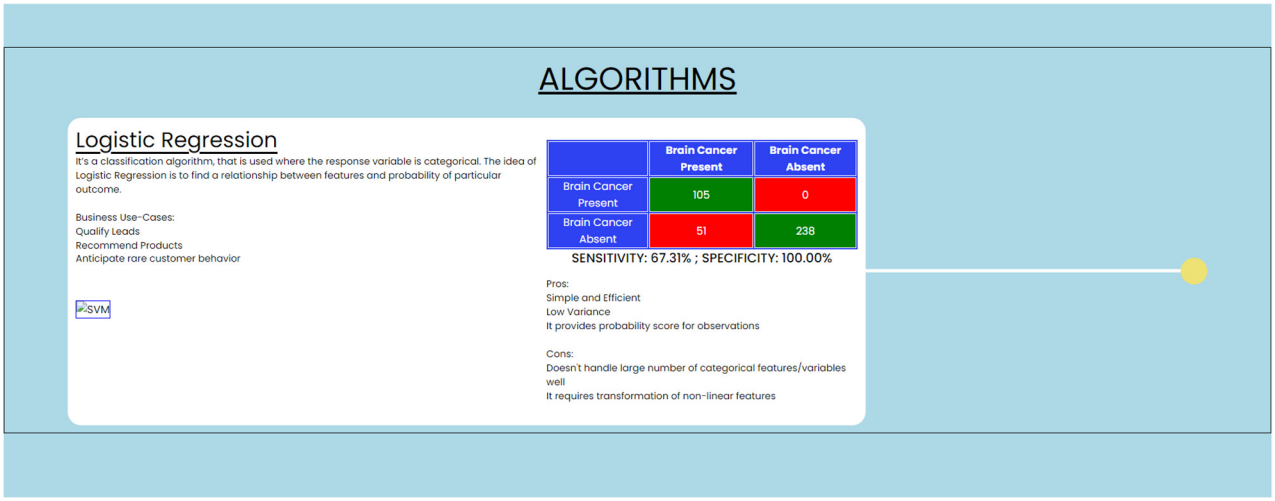


Fig 4.11: Algorithms Section of the first iteration

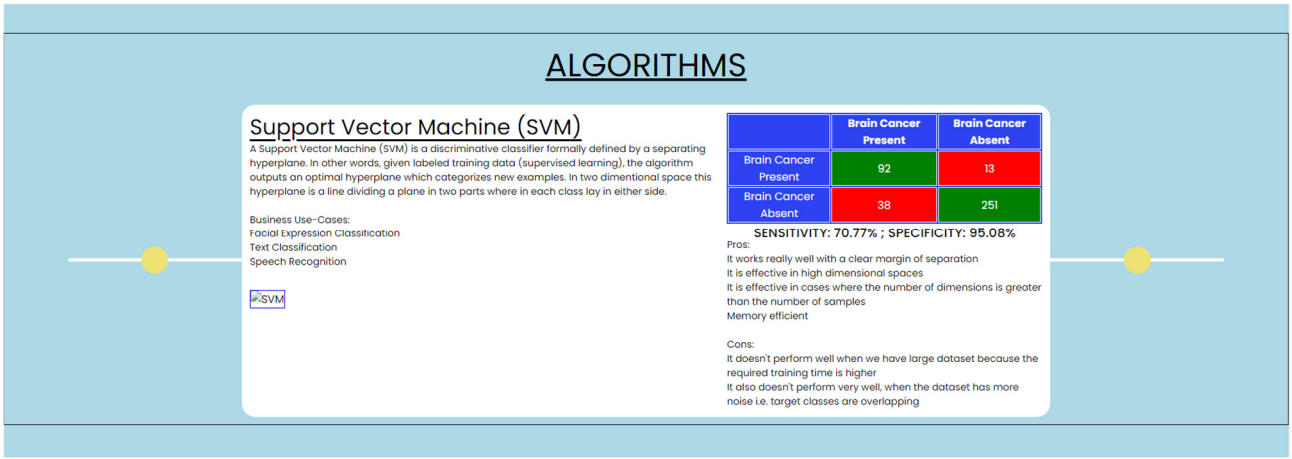


Fig 4.12: User Interactivity Provided in the Algorithms Section

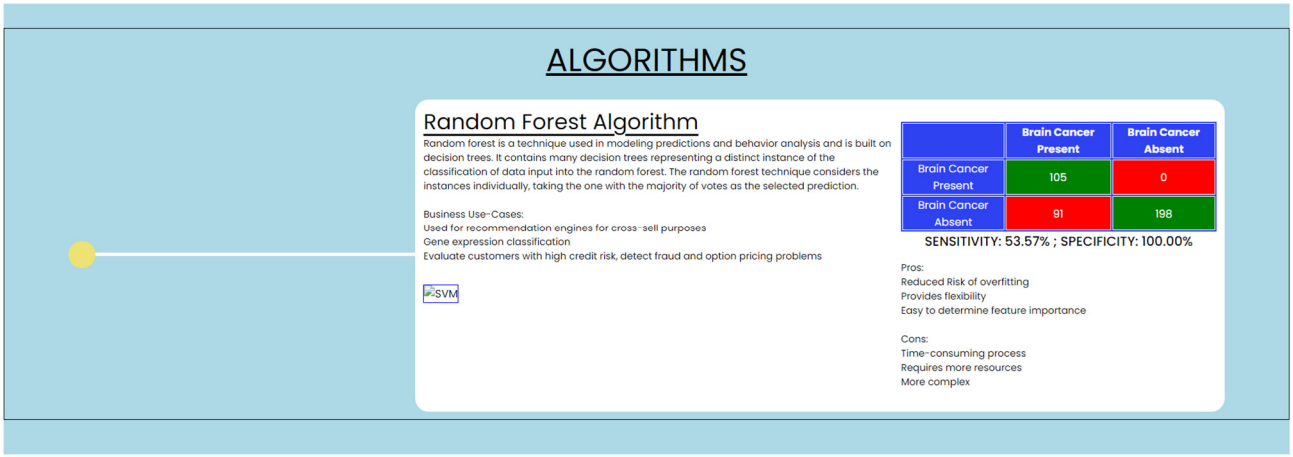


Fig 4.13: All 6 algorithms are defined in the Algorithm section

To provide a brief summary about cancer, another section was created. Since, the project relies heavily on infographics, the decision to display cancer statistics from certified sources as choropleth maps (heat maps), bar charts and line graphs was made. A description on brain cancer is also presented to help the user understand how the ailment happens. The section displaying the infographics is made similar to the interactive section describing the algorithms to maintain consistency in the design choices of the website.

Lastly, the much-needed User Input Upload Functionality is added to let the user input a brain MRI image and process that on the algorithms used during the implementation of the project. This shows the predictability of the algorithms under question in the form of a donut chart.

Fig 4.14: Upload Section

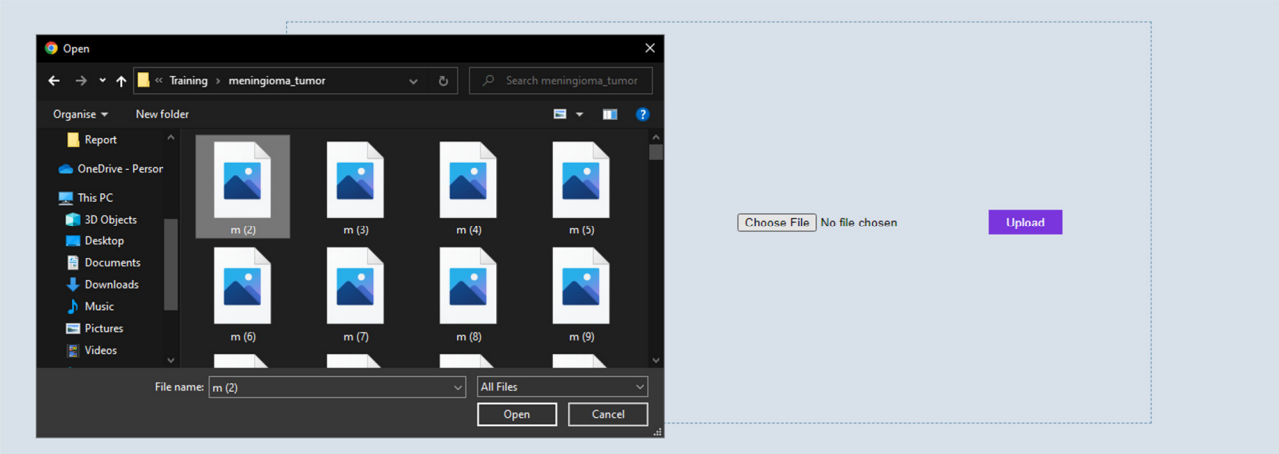


Fig 4.15: The form let's the user choose an image

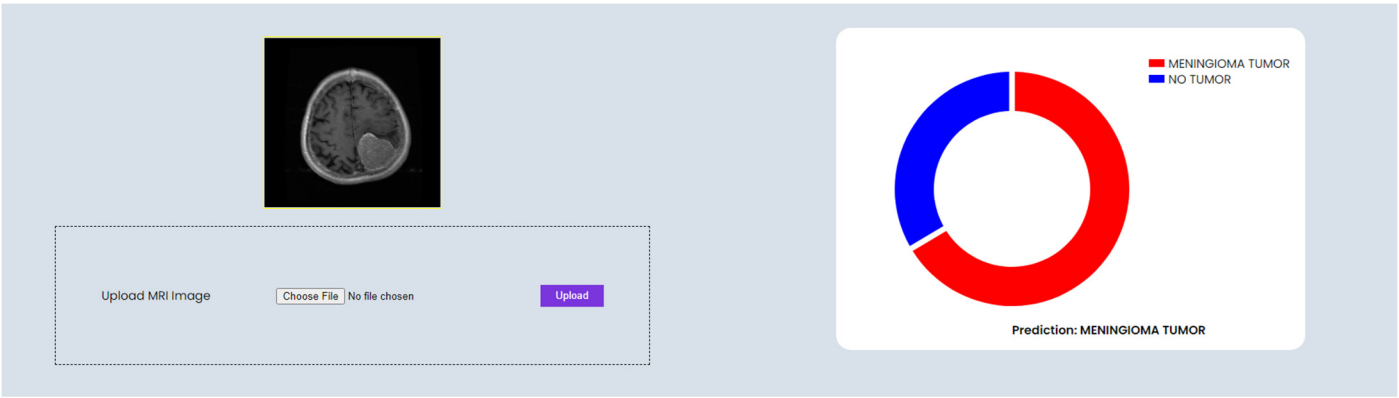


Fig 4.16: Predictions are retrieved and displayed in an appropriate fashion

4.3 MOBILE APPLICATION DEVELOPMENT:

Moving onto the mobile application, the first sprint was used to research about frameworks used in mobile development and to make a decision on which one to use. Due to its various advantages and a little bit of bias because of prior knowledge, Flutter, a cross-platform mobile framework developed by Google, was chosen. The mobile application is basically built to provide the users the same website experience but on a mobile device. The alternative was to make the website responsive to mobile dimensions, but because of graphs and interfaces, doing that would harm the website design and may cause it to malfunction. The website is responsive till the mobile tablet dimensions, but below that, the interfaces would

just not look as intended, and changing any code would cause problems for desktop and laptop views.

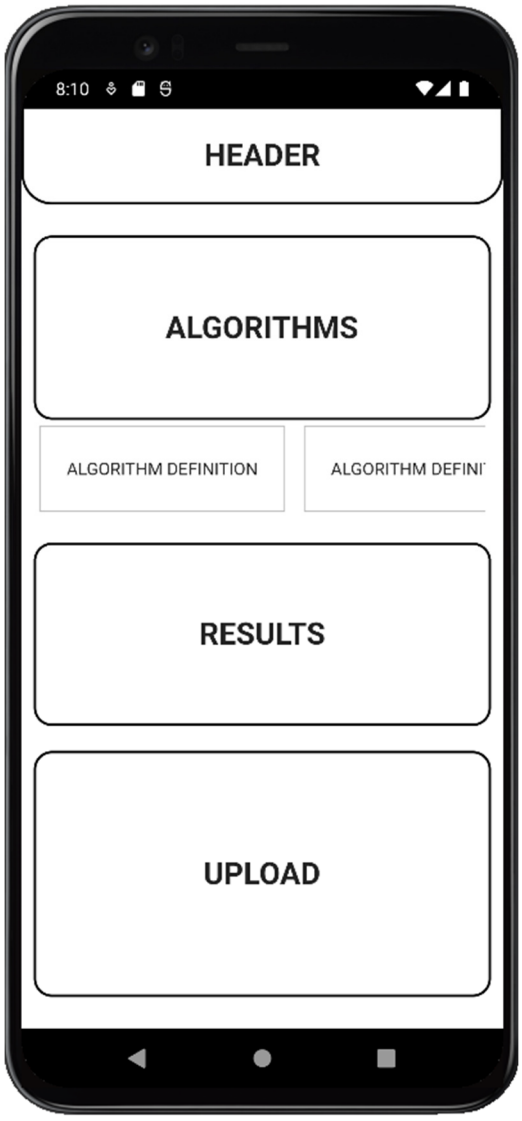


Fig 4.17: Basic Layout for the Mobile Application

The next sprints started similar to the website development stages with creation of static parts of the app, which are just rendered once and do not change overtime. The header and footer sections were created.

Moving forward, data retrieval methods were created using third-party libraries provided in flutter like http. This library helps retrieve results and set them correctly. These results were also parsed and formatted as per choice so that they can be used in an efficient manner.

The results section of the app consists of a whole new screen as putting everything on one screen would make it look messy and unprofessional. The charts_flutter library was utilized for this. Unlike the D3js library, it does not give base control to the developer but it is efficient enough for our requirements.

The user functionality section was made using three third-party libraries, namely image_picker, http, and charts_flutter. To let the user upload an image, the image_picker library is utilized as it provides system calls and makes getting the input easier. The application communicates with the API endpoint using the http library and retrieves the results for the input using it. The results retrieved are then presented as a donut chart using the charts_flutter library.

All of these sprints consisted of incremental updates of the existing interfaces as well as fixing bugs seen during the application walk throughs.

5. RESULTS AND ANALYSIS:

Rigorous testing of each and every selected algorithm with varying parameters on both the selected datasets are compared and analyzed in this section. Algorithms are picked to show the contrasting sides of traditional supervised learning algorithms and the much more recent and advanced supervised learning algorithms. Researchers opted for SVM and Logistic Regression techniques in the early days of research because of their simplicity as well as high efficiency of classifying the images based on accuracy and precision. The use of CNNs casts a shadow on the use of these traditional classifiers because they provide much more complexity and are capable of faster learning of large number of parameters.

5.1 MACHINE LEARNING RESULTS:

The contrast highlighted by the analysis shows how different the algorithms are and how supervised learning algorithms do not perform so well in object detection in images. The analysis is as below:

(1) Based on Accuracy:

The analysis based on the accuracy results retrieved by running the algorithms on the dataset shows how the much recent and more complex algorithms like Decision Tree Classifier and Random Forest Algorithm perform the best compared to the other models. It also shows how the traditional SVM model is still at par with the modern algorithms in terms of accuracy.

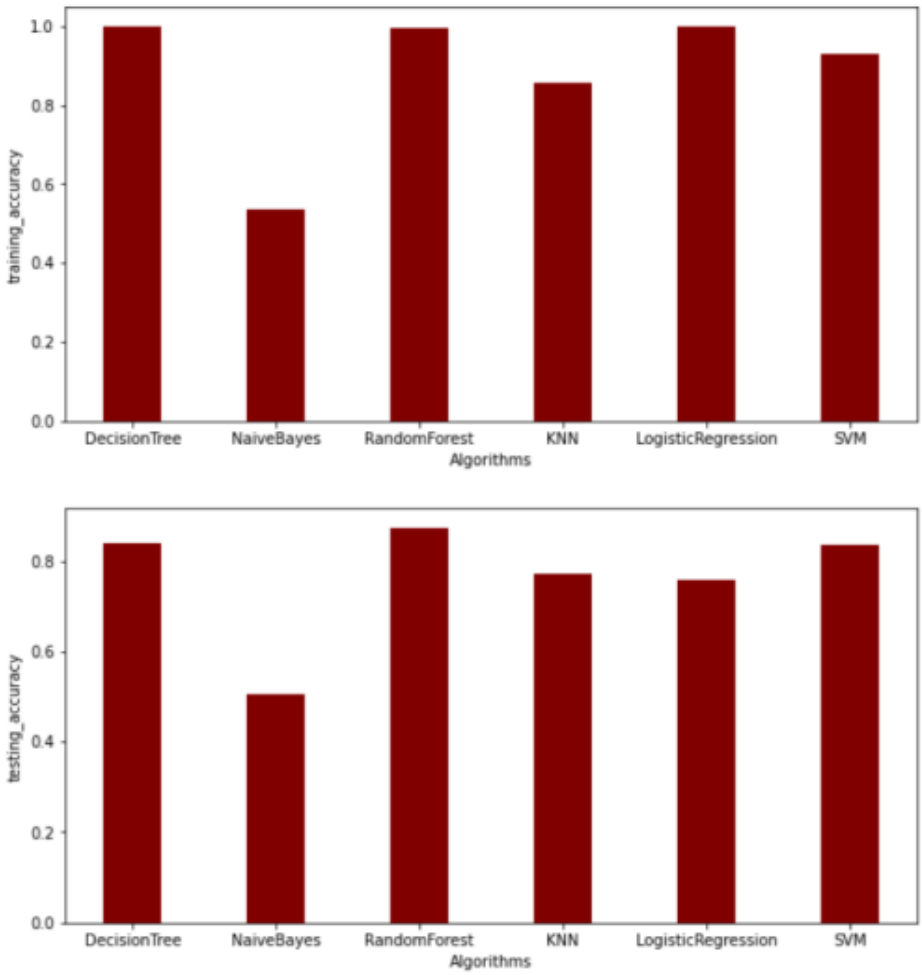


Fig 5.1: Analysis based on Accuracy

(2) Based on Precision:

The comparison results on the basis of precision reveals a similar pattern like accuracy. The complex algorithms perform much more precise and are able to better fit the dataset compared to the simple and traditional methods. SVM is the only simpler algorithm that is able to keep up with these algorithms and give a high precision score.

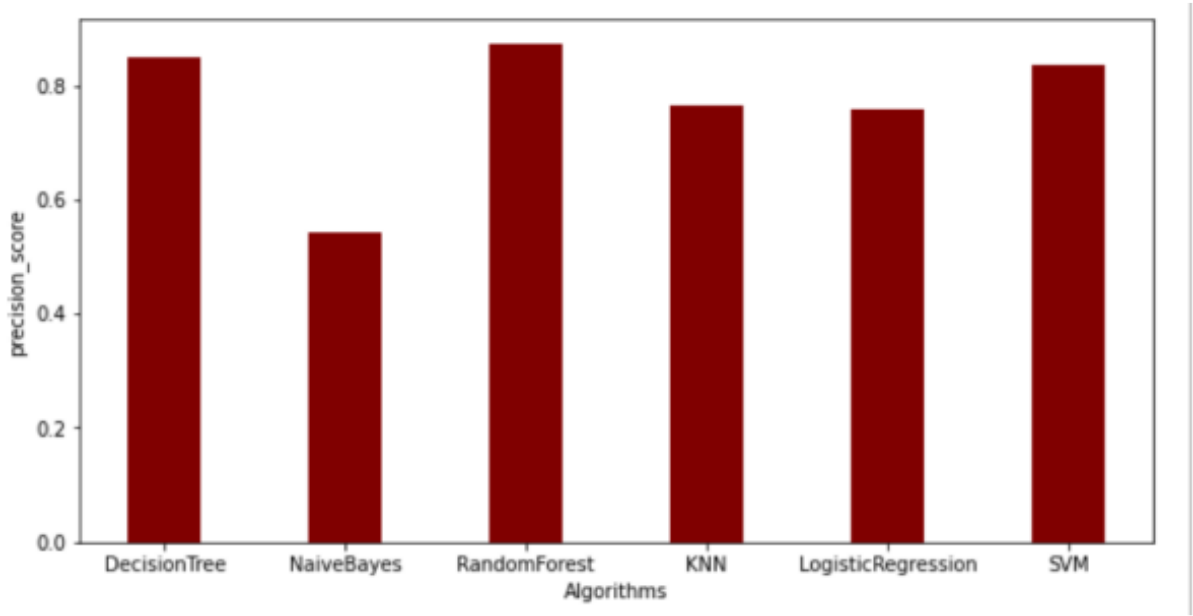


Fig 5.2: Comparison based on Precision

(3) Based on Processing Time:

Processing Time can be defined as the time taken by the algorithm to process the supplied input and give the predicted output. For an algorithm to be efficient, it should have a low processing time and operate on the input faster. Even though SVM has been able to keep up with the advance algorithms on the basis of accuracy and precision, its high processing time makes it infeasible for the industry and highlights its limitation.

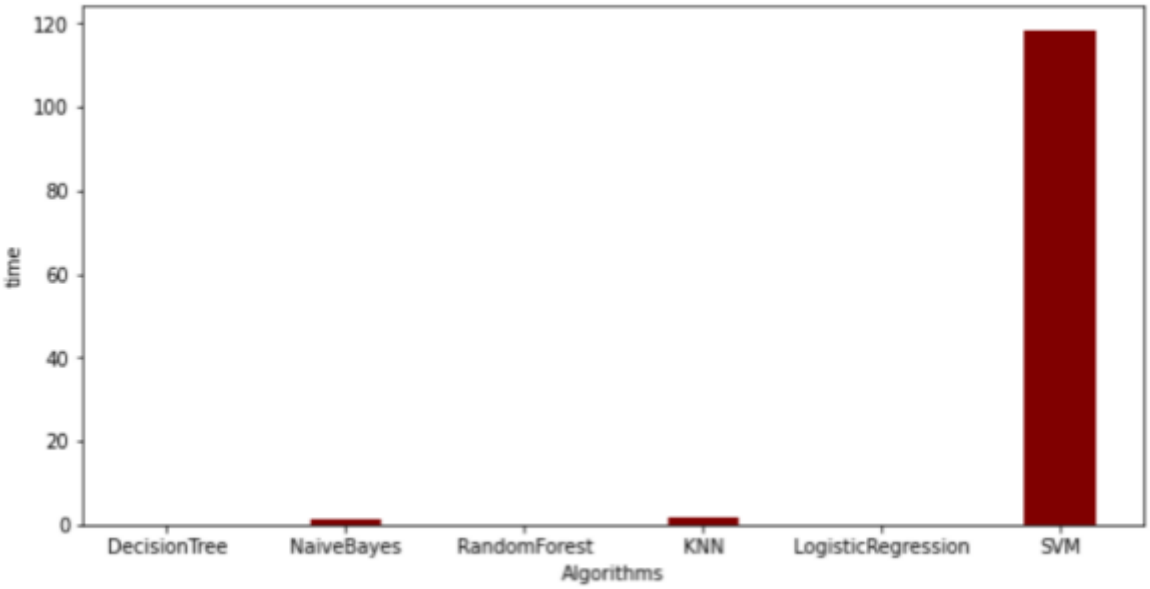
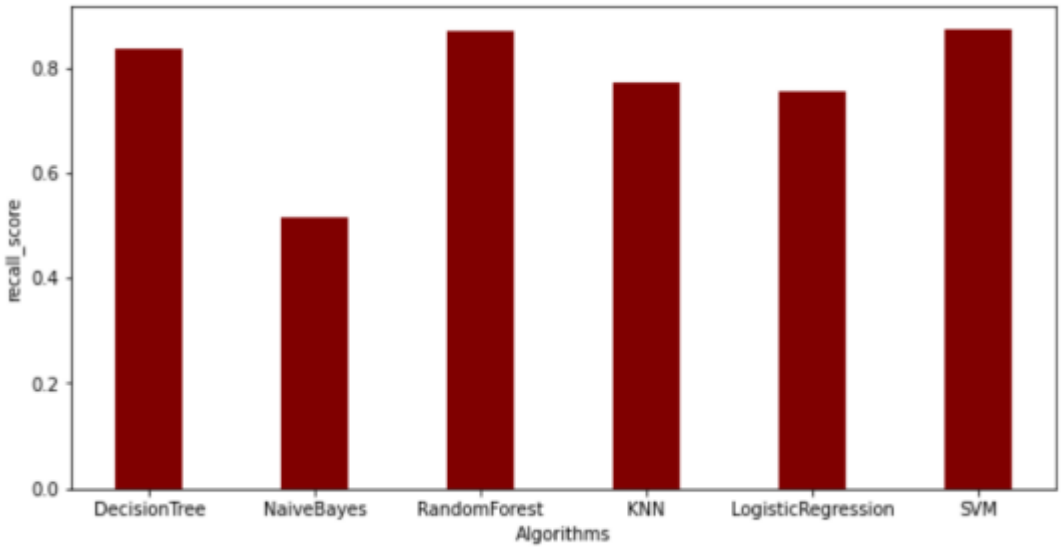


Fig 5.3: Analysis on the basis of Time

(4) Based on Recall Score:

Recall score defines the ratio of true positives predicted by the algorithm to the total number of correct predictions. This metric can be calculated from the confusion matrix of the algorithm. The analysis based on recall score paints a similar picture to the based-on accuracy and precision, where the complex and much more advanced algorithms of Decision Tree



Classifier and Random Forest Algorithm perform much better compared to the others.

Fig 5.4: Comparison based on Recall Score

ALGORITHM	ACCURACY	PRECISION	PROCESSING TIME	RECALL SCORE
Logistic Regression	72.84	0.7940	0.0628	0.7095
Support Vector Machine	83.46	0.8369	118.375	0.8729
K-Nearest Neighbors	77.18	0.7659	1.7676	0.7715
Naïve Bayes Classification	50.535	0.5405	1.5043	0.5166
Decision Trees Classifier	83.76	0.85	0.0850	0.8369
Random Forest Algorithm	87.28	0.8729	0.0990	0.8707

Table-1

5.2 WEB DEVELOPMENT RESULTS:

The web page was developed using the React front-end framework which makes it easy to create interactive UIs for Single-Page Applications using JavaScript, HTML and CSS. The src folder containing the source code of the website is segmented into 2 more folders namely: views and components. The views folder contains the declaration of the different views being displayed on the home page of the website. The components folder consists of reusable components that are being displayed and used in the different views. The various views are mentioned here:

1. Header:

This is the top-most section of the website page and contains the Navigation bar and the title of the project. The website boasts a user functionality which requires user input as an MRI image which is ran on the 6 supervised ML algorithms used in the project and displays the results along with the final prediction. The header contains a button that takes the user to the input functionality section right away.

Fig 5.5: Header Section

2. Section Explaining Cancer:

The next section after the header explains Cancer and more specifically Brain Cancer. The section displays 3 different infographics explaining the impact of Cancer around the world. The 3 infographics includes a Choropleth map which is a heat map describing the impact of brain cancer around the globe, a Line Chart describing the increase in the top 5 cancer types around the world which includes Brain Cancer, and a Bar Chart showing the impact of cancer as a disease and ranking it amongst the top 10 most fatal diseases.

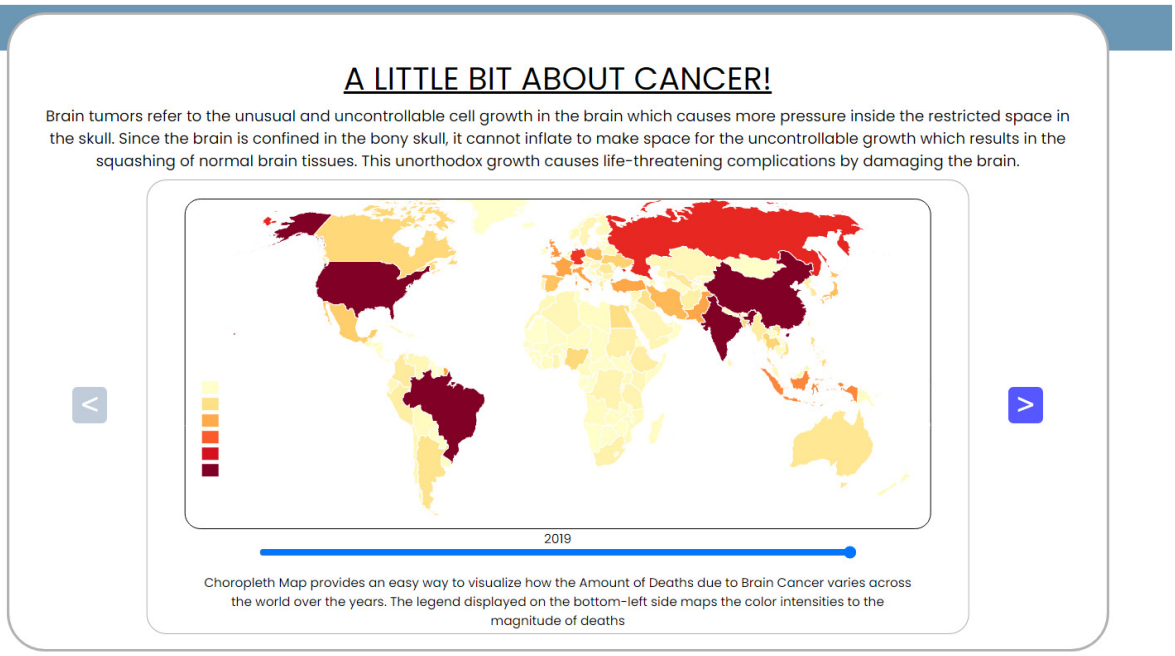


Fig 5.6: Choropleth Map in the section describing Cancer

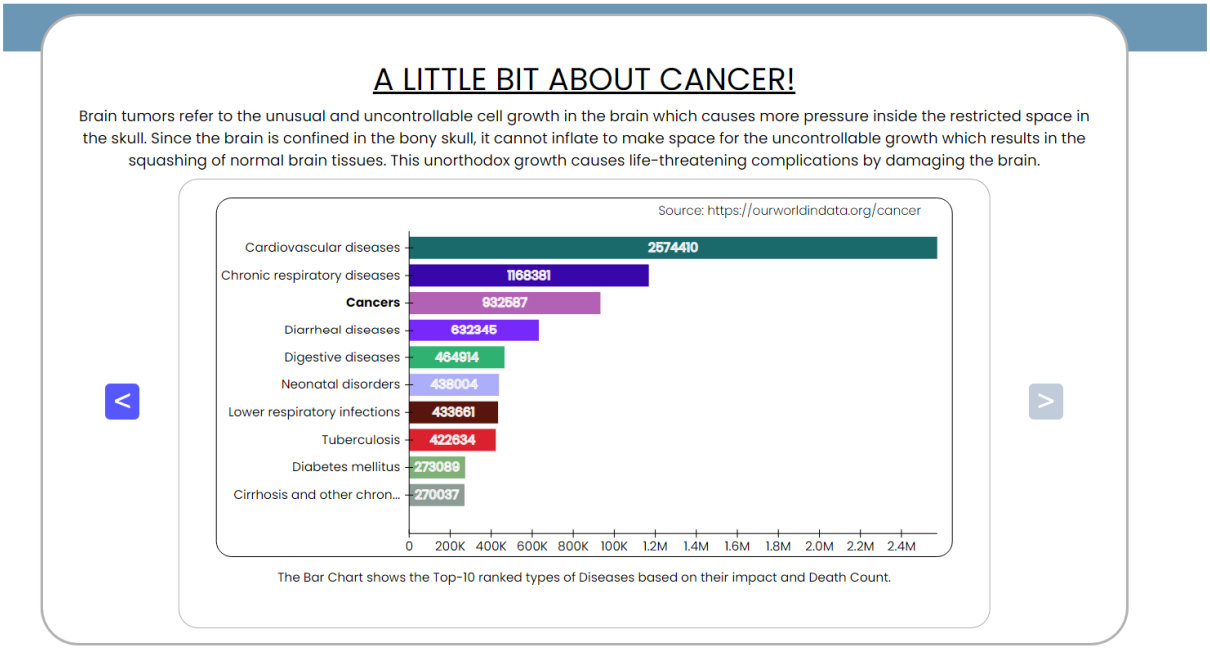


Fig 5.7: Bar Chart showing top 10 fatal diseases in the World

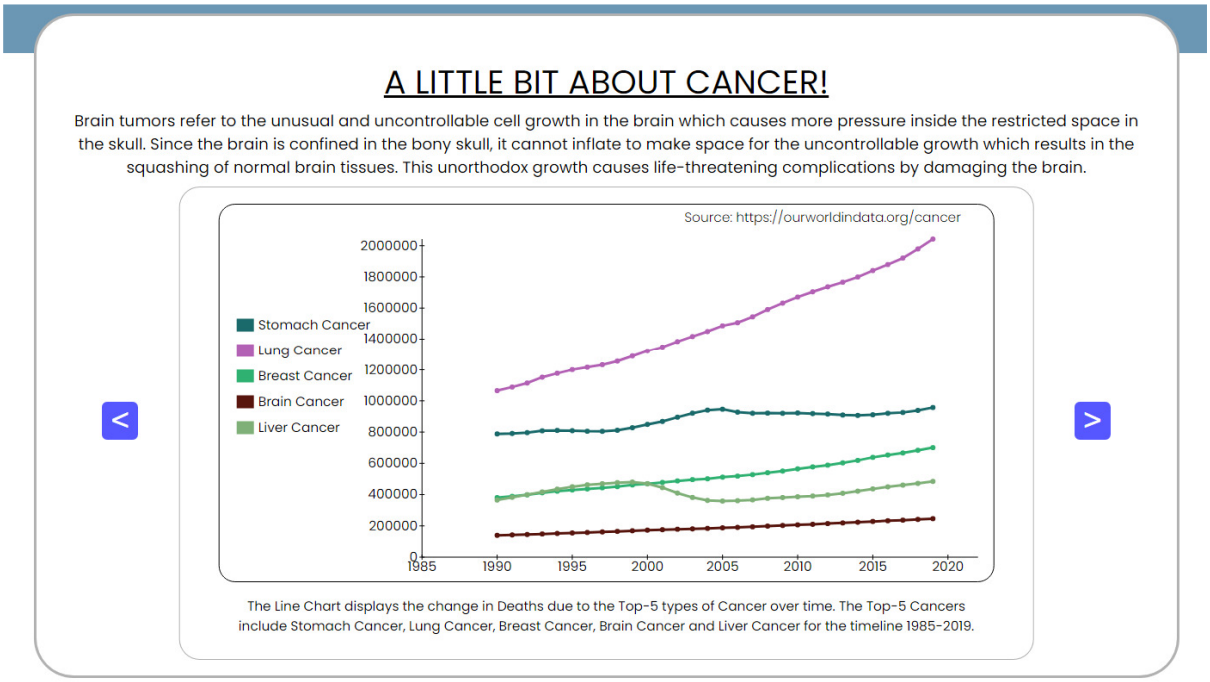


Fig 5.8: Line Chart showing increase of top 5 types of cancer over the years

3. Results:

The Results section displays the results gathered from the project implementation. It displays Bar Charts displaying the results of the 6 supervised algorithms based on the 4 factors of Accuracy, Precision, Processing Time and Recall. The section contains an interactive UI built using the React framework and D3, which is a JavaScript framework for making charts and graphs.

Fig 5.9: Results Section

Fig 5.10: Showing interactivity of the bar charts in the result section

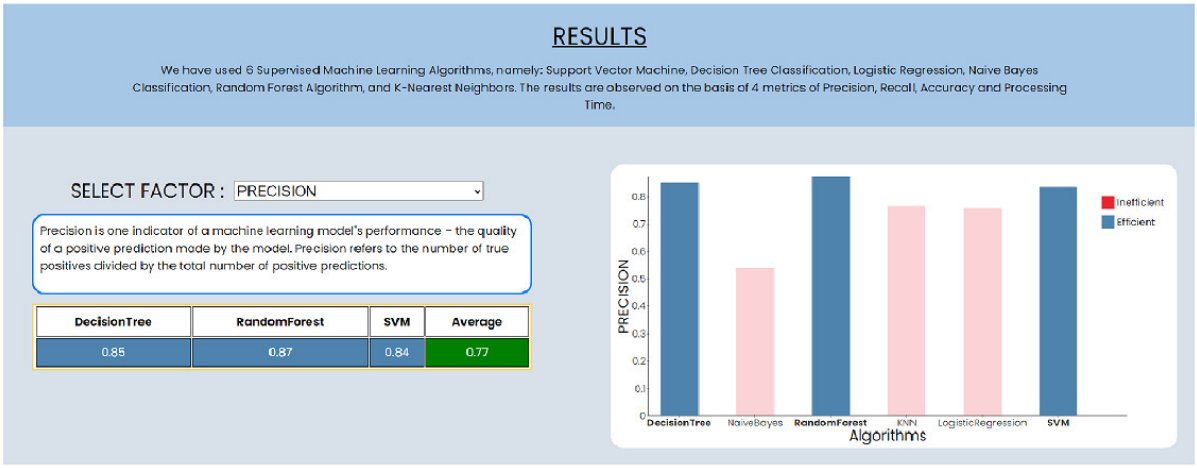


Fig 5.11: Showing filtering amongst the bar charts

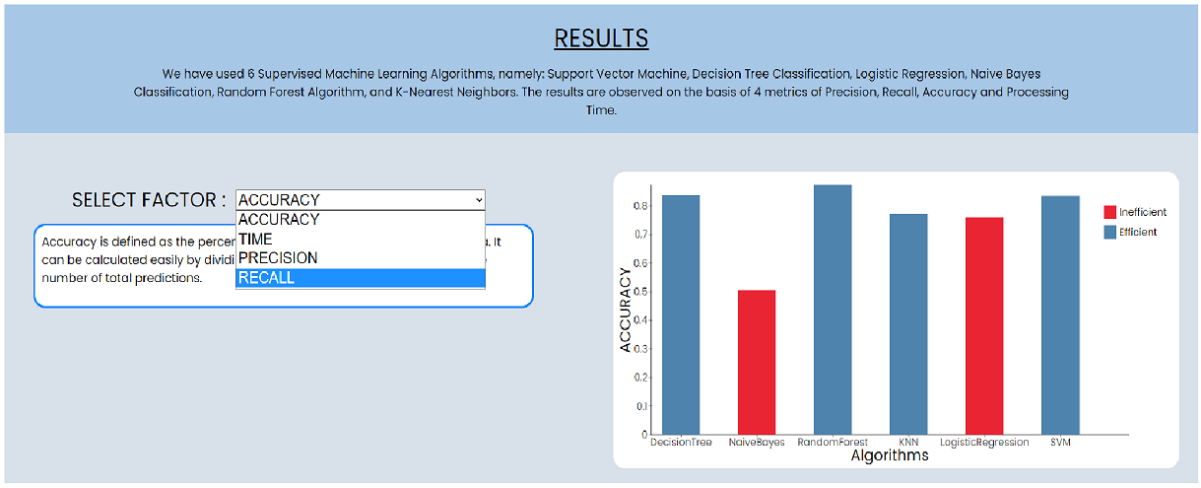
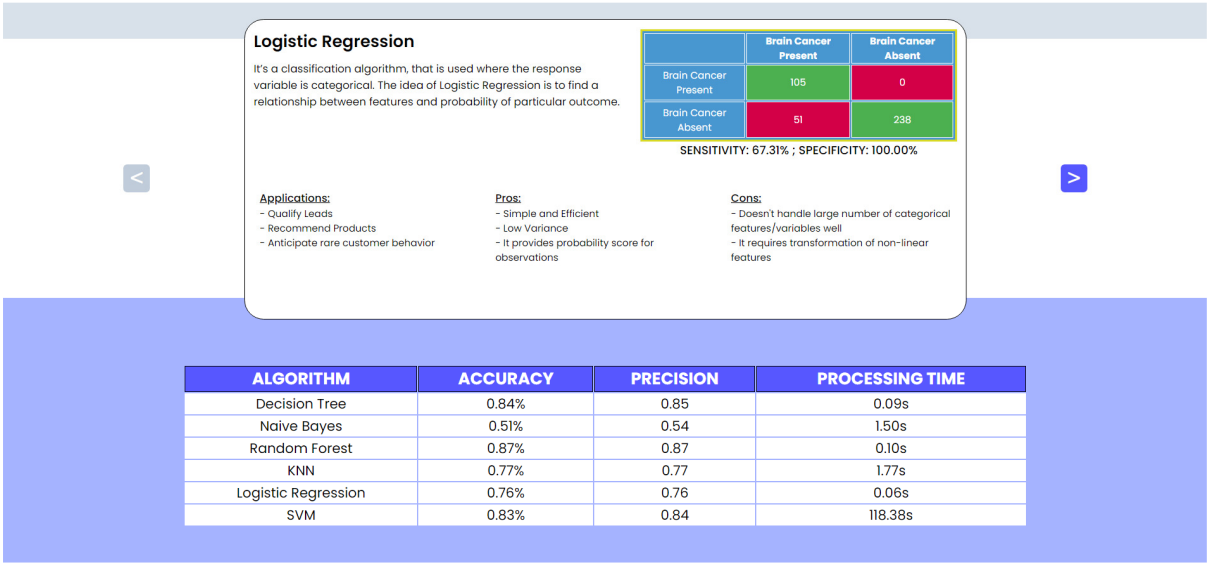


Fig 5.12: Showing options to be chosen in the results section to filter the charts based on different factors

4. Algorithms and Result Table:

This section explains the supervised learning algorithms used in the project. It displays the confusion matrix for the algorithm, its definition, their applications, pros and cons. It also displays a tabular form of the results for a general understanding of the results of the algorithms.



ALGORITHM	ACCURACY	PRECISION	PROCESSING TIME
Decision Tree	0.84%	0.85	0.09s
Naive Bayes	0.51%	0.54	1.50s
Random Forest	0.87%	0.87	0.10s
KNN	0.77%	0.77	1.77s
Logistic Regression	0.76%	0.76	0.06s
SVM	0.83%	0.84	118.38s

Fig 5.13: Section describing the algorithms used and overall comparison in tabular form

5. Upload Functionality:

The user functionality expects a user input in the form of an MRI image of the brain, which is then sent to the back-end API of the project, where all 6 algorithms process the input and return the predicted results of each algorithm for the input.

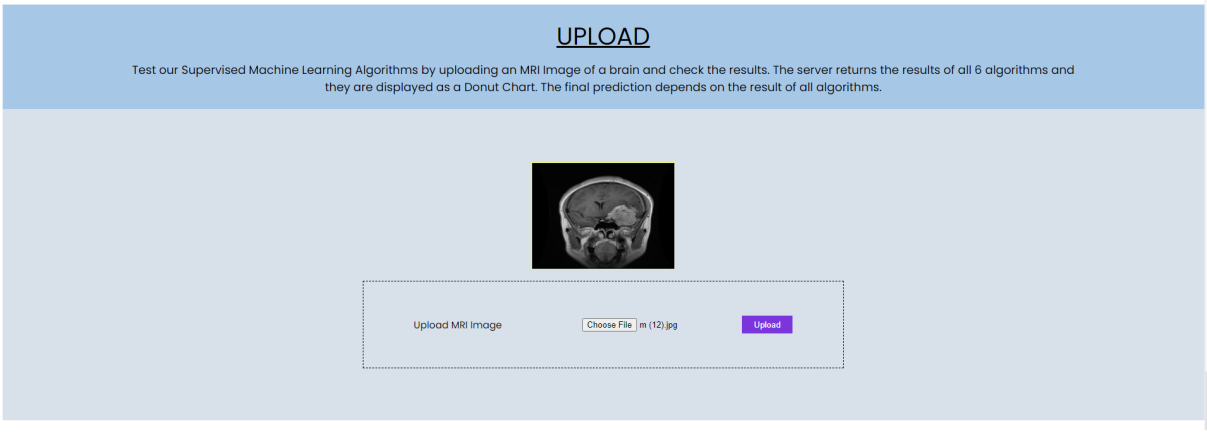


Fig 5.14: Upload Section on the website

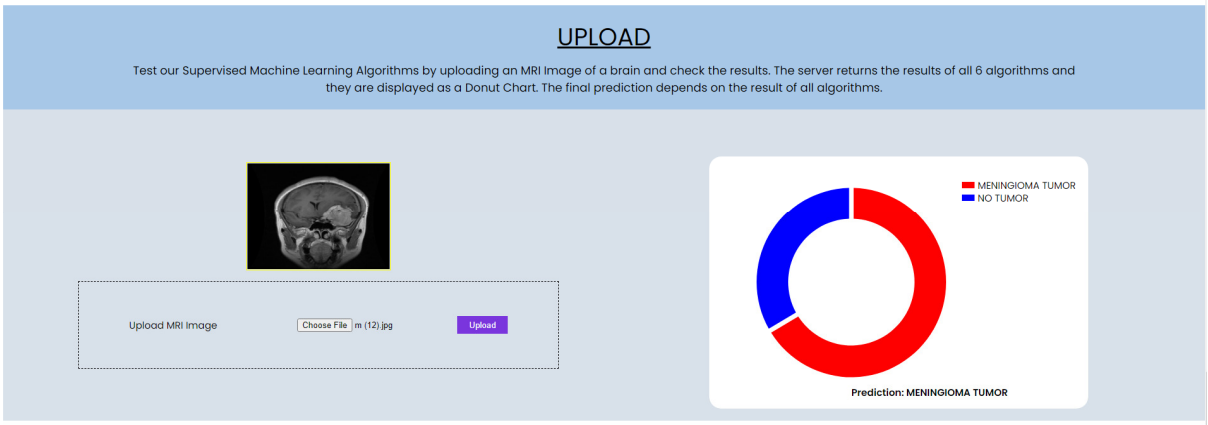


Fig 5.15: Showing image upload functionality

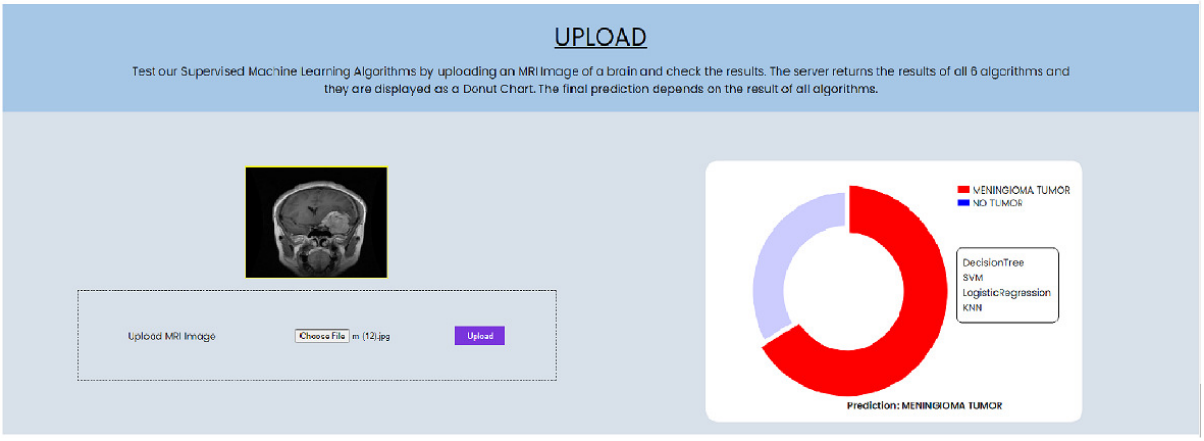


Fig 5.16: Showing interactivity of the upload functionality

6. Footer:

This is the last section of the web page and describes the motivation behind the project and why our team chose it. It also gives links to the Github repository where the project is hosted as well as links to the various sources of data that is being rendered on the page. It also contains LinkedIn links of all the team members. The link to download the mobile app of the project is also given here.

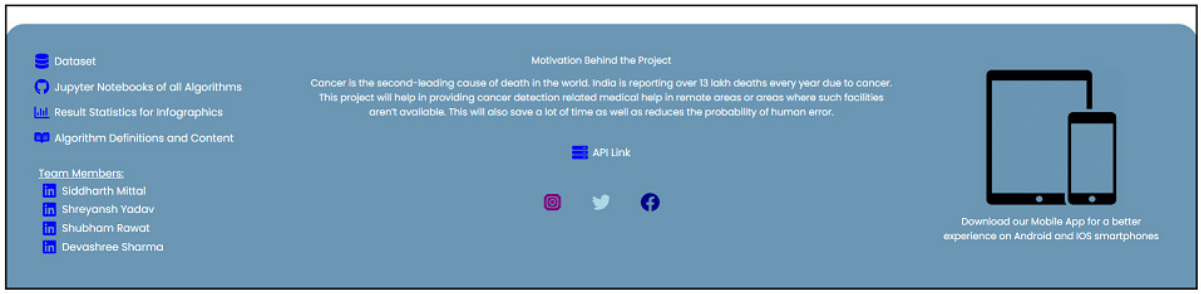


Fig 5.17: Footer Section showing links to the source code, datasets and team member's LinkedIn accounts

5.3 MOBILE APPLICATION RESULTS:

1. On-Boarding Screen: The first screen for a new user is the on-boarding screen which helps the user understand the application as well as the user functionality provided. It contains 3 tiles displaying information about Cancer, our motivation behind the project, and a little bit about the user functionality.

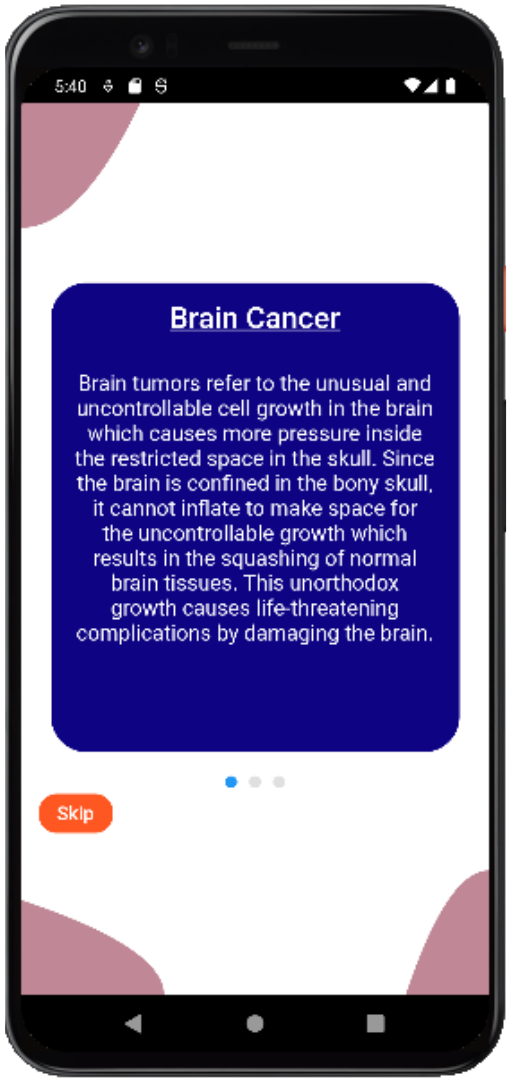


Fig 5.18: On-Boarding Screen

2. Home Screen:

The home screen displays the different sections of the application. It contains the app bar common to all the screens, the header showcasing the title of the project. It also contains a section which displays the summary of the different algorithms used. This is an interactive UI, by clicking on the tiles, the user is taken to a different screen displaying all the information about the algorithm. The results tile follows, which is also interactive, and when clicked takes to a new screen displaying the results as infographics. The next section contains the upload feature and the footer follows.

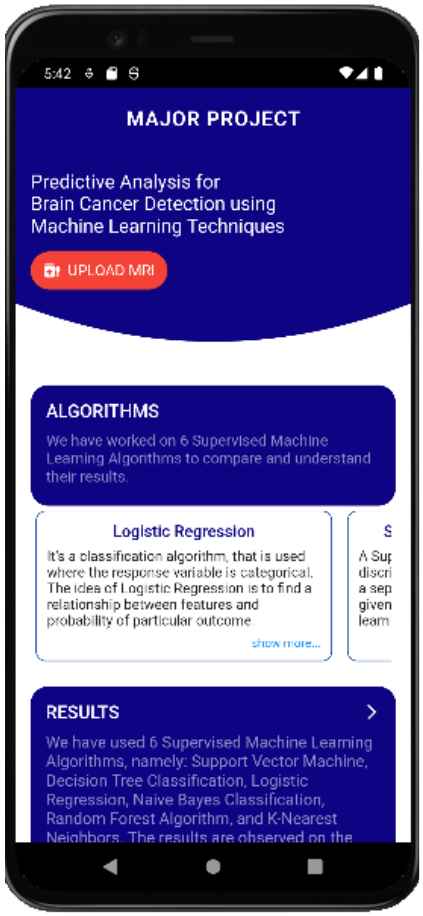


Fig 5.19: Home Screen

3. Algorithms Screen: This screen displays all the information about the algorithm which was clicked by the user on the home screen. It displays the confusion matrix for the algorithm, its definition, their applications, pros and cons.

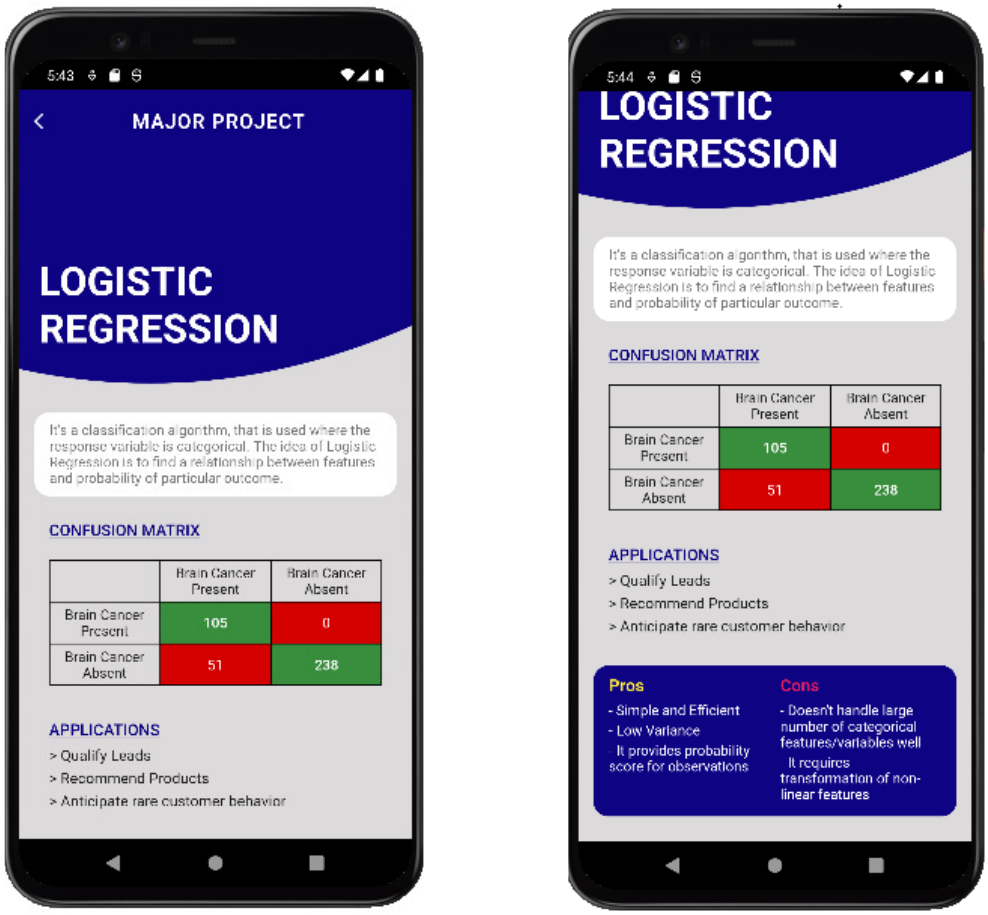


Fig 5.20: Algorithm Screens

4. Results Screen:

The Results screen displays the infographics similar to the website. It displays bar charts for the results of the project based on the 4 factors. It also displays a tabular form of the results for a general understanding of the results of the algorithms.

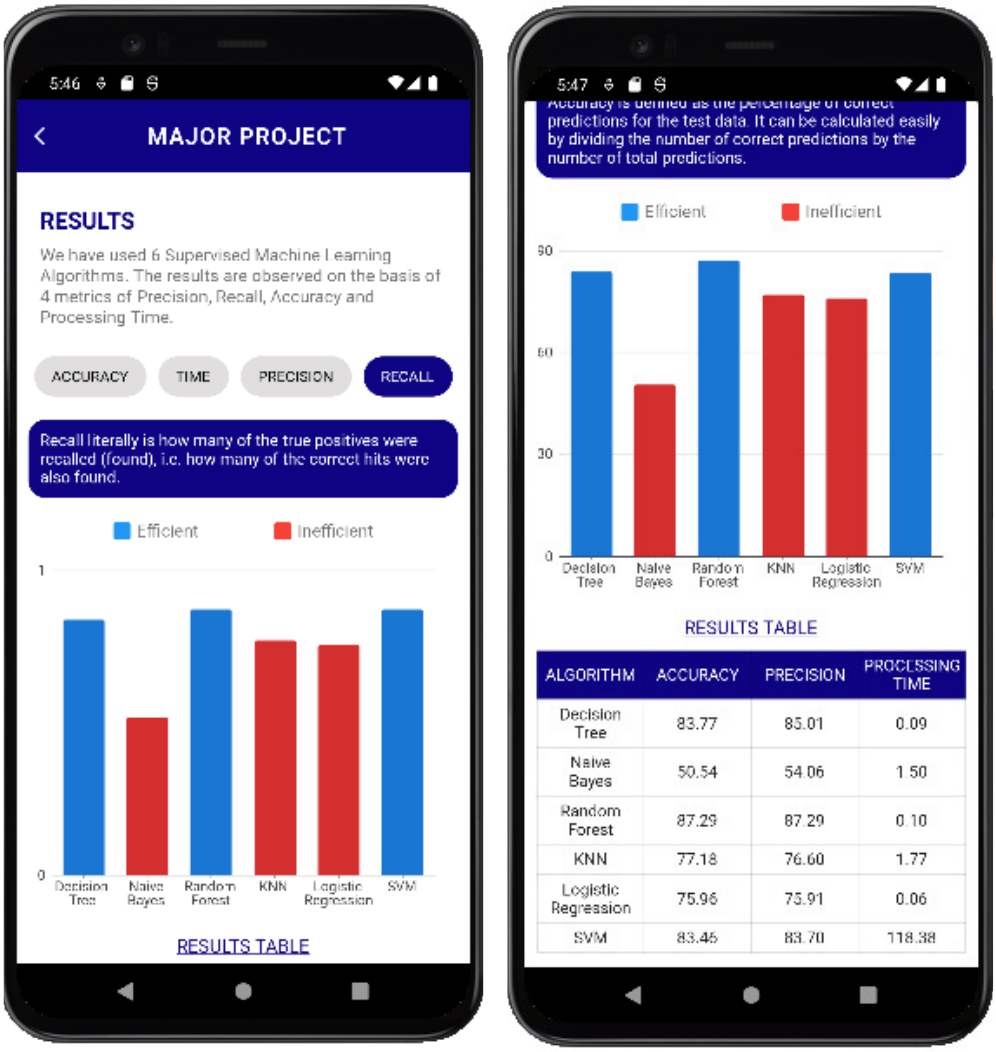


Fig 5.21: Results Screen

5. Upload Section:

The Upload section contains the form to get an MRI Image of the brain as input from the user and send it to the back-end API. The results retrieved from the API are then displayed to the user along with the final prediction.

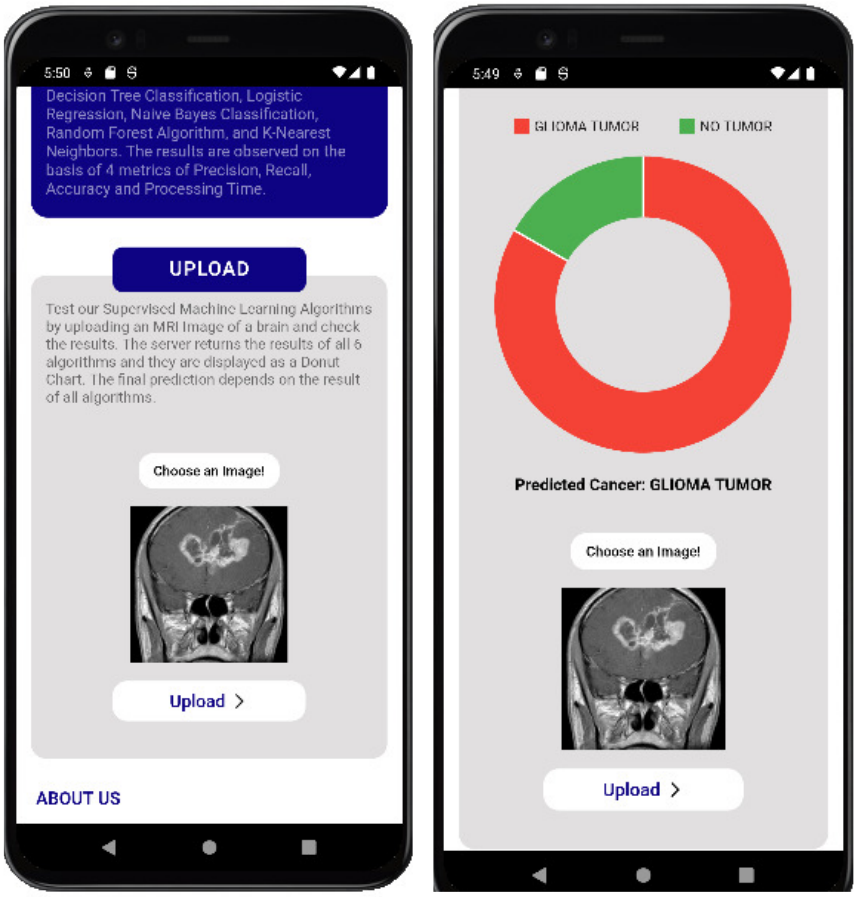


Fig 5.22: Upload Section

6. Footer Section:

The footer section displays the motivation behind the project as well as links to the LinkedIn accounts of all the team members.

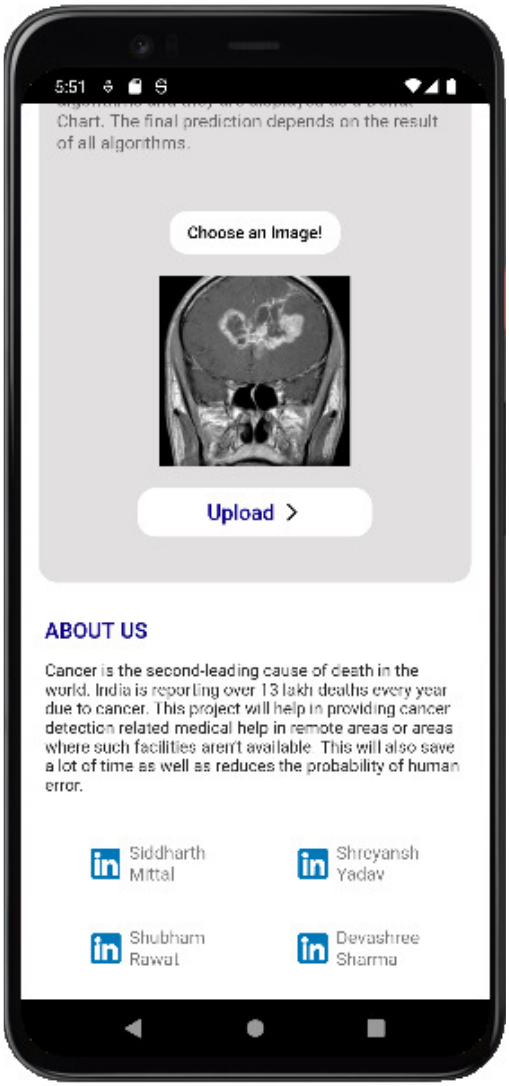


Fig 5.23: Footer Section

6. CONCLUSION

The 6 supervised learning algorithms are reviewed on the chosen dataset of MRI images and compared to show the contrasting differences for brain tumour detection. This juxtaposition is based on experimental analysis of the techniques on certified collections of MRI images of brain. The implementation of all the algorithms follows a generalized pipeline of phases. The pipeline comprises of stages like pre-processing of the dataset so that it can be aptly used with the algorithm, tumour segmentation, feature extraction, and then classification. The simple algorithms like Logistic Regression, Naive Bayes, and KNN are not able to keep up with the much complex and advanced supervised methods like Random Forest algorithm and Decision Tree Classification in terms of accuracy, precision and recall. The traditional SVM algorithm keeps up with the recent algorithms in terms of accuracy and precision, but its high processing time makes it obsolete during the current time of technology advancement. These algorithms, however, are not at par with the much more complex deep learning algorithms like the CNN model and the transfer learning models. They come on top with higher accuracy and lower loss and are also much more efficient in terms of time. All these pros make them the better choice when it comes to real-world applications and deployment.

With developments happening by the second, the medical applications of machine learning for brain cancer detection are far from complete. Even with the rapid advancements in technology now a days, understanding the medical background of the how and why of tumours and their intricate details will give the much-needed boost to assemble better medical procedures to deal with them which can be further transferred to machine learning algorithms. Medical studies relating to brain cancer depict that only early detection of such ailment can help the patient's survival rate. Early detection leads to a better diagnosis and treatment of the patient at an early stage which can be made possible using machine learning models that help discard human error and help the patients.

7. FUTURE SCOPE

With the computation speeds of computer systems getting quicker by the minute, the usage of ML algorithms in early detection of life-threatening diseases like cancers will help in better prediction of tumors and weed out the incorrect false positives and negatives because of human error. With the many challenges like handling the complex cancers precisely obstructing the path, further Research is needed to help solve these challenges and provide help not only in metropolitan areas but also in remote areas where hospitals don't have the capacity to hold the expensive equipment or employ doctors specializing in these areas.

The presented work can be further investigated by chaining more than one classifier or more than one feature selection techniques to examine their effect on the accuracy and precision of the already presented models.

The work can also be extended for other types of cancers.

In the future, studies such as improving accuracy with a low rate of error utilizing various classifier algorithms will be undertaken.

8. REFERENCES

[1] Tanzila Saba, Recent advancement in cancer detection using machine learning: Systematic survey of decades, comparisons and challenges, Journal of Infection and Public Health, Volume 13, Issue 9, 2020

[2] Zhengyu Yu, Qinghu He, Jichang Yang, and Min Luo: A Supervised ML Applied Classification Model for Brain Tumours MRI, Front Pharmacol, 2022

[3] Amin, J., Sharif, M., Haldorai, A. et al. Brain tumour detection and classification using machine learning: a comprehensive survey. Complex Intell. Syst. (2021)

[4] Komal Sharma, Akwinder Kaur, and Shruti Gujral: Brain Tumour Detection based on Machine Learning Algorithms, International Journal of Computer Applications, Volume 103, October 2014

[5] React Documentation by Facebook: https://reactjs.org/docs/getting-started.html

[6] The Data Driven Documentation and online gallery: https://d3js.org/

[7] The Flutter Documentation by Google: https://docs.flutter.dev/

[8] The freeCodeCamp website and YouTube channel: https://www.freecodecamp.org/

DrillBit

The Report is Generated by DrillBit Plagiarism Detection Software

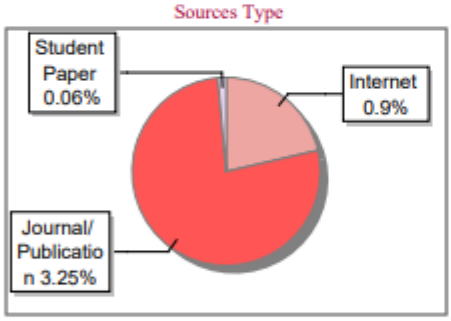
Submission Information

Author Name	Siddharth
Title	Siddharth
Paper/Submission ID	510523
Submission Date	2022-05-04 16:42:57
Total Pages	47
Document type	Project Work

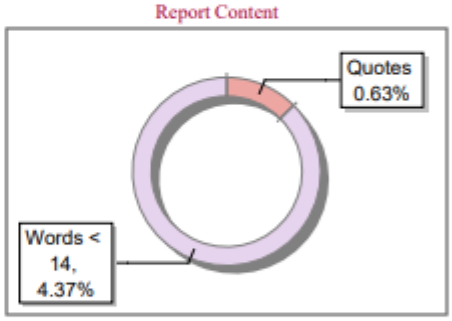
Result Information

Similarity **6 %**

Sources Type



Report Content



Exclude Information

Quotes	Not Excluded
References/Bibliography	Not Excluded
Sources: Less than 14 Words Similarity	Not Excluded
Excluded Source	**0 %**
Excluded Phrases	Not Excluded

A Unique QR Code use to View/Download/Share Pdf File

DrillBit

DrillBit Similarity Report

6	38	A	A-Satisfactory (0-10%) B-Upgrade (11-40%) C-Poor (41-60%) D-Unacceptable (61-100%)
SIMILARITY %	MATCHED SOURCES	GRADE	

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	09bd5b2db5.testurl.ws	<1	Internet Data
2	Polarization singularities of optical fields caused by structural dislocations i by Savaryn-2013	<1	Publication
3	en.wikibooks.org	<1	Internet Data
4	Machine Learning-Based Predictive Modeling of Complications of Chronic Diabetes by Derevitskii-2020	<1	Publication
5	GITAM UNIVERSITY Thesis Published in - www.inflibnet.ac.in	<1	Publication
6	Anomaly detection in Skin Model Shapes using machine learning classifiers by Yacob-2019	<1	Publication
7	Benchmark-Based Reference Model for Evaluating Botnet Detection Tools Driven by by Huancay-2020	<1	Publication
8	Adaptation and learning over complex networks From the Guest Editors by Sayed-2013	<1	Publication
9	A new algorithm for the compression of ECG signals based on mother wav by Abo-Zahhad-2013	<1	Publication
10	Laser induced fluorescence measurements of dissolved oxygen concentration fields by Roy-2000	<1	Publication
11	www.programmableweb.com	<1	Internet Data

COMMENT BY EXTERNAL EXAMINER

Name and Signature of External Examiner: