*An industrial oriented mini project report*

*On*

**PREDICTION OF ADMISSION IN ENGINEERING COLLEGE**

Submitted by

**MD. MALIK**

**MD. RAHEEL**

**MD. SAQHIB**

**MD. MAZHAR**

**20W91A05E3**

**20W91A05E4**

**20W91A05E5**

**20W91A05E7**

Under the Esteemed Guidance of

**Mr. K. VENKATA RATNAM**

Asst. Professor, CSE

TO

**Jawaharlal Nehru Technological University, Hyderabad**

In partial fulfilment of the requirements for award of degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**MALLA REDDY INSTITUTE OF ENGINEERING AND TECHNOLOGY**

**(UGC AUTONOMOUS)**

(Sponsored by Malla Reddy Educational society)

(Affiliated to JNTU, Hyderabad)

Maisammaguda, Dhulapally post, Secunderabad-500014.

**2023-2024**

**MALLA REDDY**

**INSTITUTE OF ENGINEERING & TECHNOLOGY**

**(Autonomous Institution - UGC, Govt. of India)**

(Sponsored by Malla Reddy Educational Society)

Approved by AICTE, New Delhi, Recognized Under 2(1) & 12(B)

**Affiliated to JNTU, Hyderabad, Accredited by NBA & NAAC with ‘A’ Grade**

**Department of Computer Science and Engineering**

**BONAFIDE CERTIFICATE**

This is to certify that this is the bonafide certificate of an industrial oriented mini project report titled "**DRIVER DROWSINESS MONITORING SYSTEM USING VISUAL BEHAVIOUR AND MACHINE LEARNING**" is submitted by **MD. RAHEEL (20W91A05E4), MD. MALIK (20W91A05E3), MD. SAQHIB (20W91A05E5), MD. MAZHAR (20W91A05E7)** of B. Tech in the partial fulfilment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering, Dept. of Computer Science & Engineering and this has not been submitted for the award of any other degree of this institution.

**Internal Guide Sign Head of Department Sign**

**K. VENKATA RATNAM MD. ASHFAQUL HASSAN**

**External Examiner Sign**

**DECLARATION**

I hereby declare that the Mini Project report entitled “**DRIVER DROWSINESS MONITORING SYSTEM USING VISUAL BEHAVIOUR AND MACHINE LEARNING**” submitted to Malla Reddy Institute of Engineering and Technology (Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH), for the award of the degree of Bachelor of Technology in Computer Science & Engineering is a result of original industrial oriented mini project done by me.

It is further declared that the seminar report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

**MD. RAHEEL 20W91A05E4**

**MD. MALIK 20W91A05E3**

**MD. SAQHIB 20W91A05E5**

**MD. MAZHAR 20W91A05E7**

**ACKNOWLEDGEMENT**

First and foremost, I am grateful to the Principal **Dr. P. SRINIVAS**, for providing me with all the resources in the college to make my project a success. I thank him for his valuable suggestions at the time of seminars which encouraged me to give my best in the project.

I would like to express my gratitude to **Dr. MD. ASHFAQUL HASSAN**, Head of the Department, Department of Computer Science and Engineering for his support and valuable suggestions during the dissertation work.

I offer my sincere gratitude to my project – coordinator **Dr. N. NARENDHAR** and internal guide **Mr. K. VENKATA RATNAM** who has supported me throughout this project with their patience and valuable suggestions.

I would also like to thank all the supporting staff of the Dept. of CSE and all other departments who have been helpful directly or indirectly in making the project a success.

I am extremely grateful to my parents for their blessings and prayers for my completion of project that gave me strength to do my project.

**MD. RAHEEL 20W91A05E4**

**MD. MALIK 20W91A05E3**

**MD. SAQHIB 20W91A05E5**

**MD. MAZHAR 20W91A05E7**

**CONTENTS**

**CONTENT PAGE NO.**

**Abstract i**

**List of Figures ii-iii**

**List of Tables iv**

**List of Screens v**

**Symbols & Abbreviations vi**

1. **INTRODUCTION 1-4**
   1. Motivation 1
   2. Problem definition 2
   3. Objective of Project 2
   4. Limitations of Project 2-3
   5. Organization of Documentation 3-4
2. **LITERATURE SURVEY 5-7** 
   1. Introduction 5-6
   2. Existing System 6
   3. Disadvantages of Existing system 6
   4. Proposed System 6-7
   5. Conclusion
3. **Analysis** **8-13**
   1. Introduction 8
   2. Software Requirement Specification 8
      1. User requirement 8
      2. Software requirement 9
      3. Hardware requirement 9
   3. Content diagram of Project 9
   4. Algorithms and Flowcharts 9-13
      1. Algorithms used in the project 9-11
         1. CNN Algorithm 9-11
      2. Flowchart 12-13
   5. Conclusion 13

1. **DESIGN 14-18**
   1. Introduction 14
   2. UML diagrams 14-18
      1. Class diagram 15
      2. Use case diagram 15
      3. Activity diagram 16
      4. Sequence diagram 17
      5. Collaboration diagram 17-18
   3. Module design and organization 18
   4. Conclusion 18
2. **IMPLEMENTATION & RESULTS 19-49**
   1. Introduction 19-21
   2. Explanation of Key functions 21-28
   3. Method of Implementation 29-49

5.3.1 Source Code 29-38

5.3.2 Output Screens 38-44

5.3.3 Result Analysis 44-49

* 1. Conclusion 49

1. **TESTING & VALIDATION 50-54**
   1. Introduction 50
   2. Design of test cases and scenarios 50-53
   3. Validation 53-54
   4. Conclusion 54
2. **CONCLUSION 55**
3. **FUTURE ENHANCEMENT 56**
4. **REFERENCES 57**

**ABSTRACT**

Drowsy driving is one of the major causes of road accidents and death. Hence, detection of driver’s fatigue and its indication is an active research area. Most of the conventional methods are either vehicle based, or behavioural based or physiological based. Few methods are intrusive and distract the driver, some require expensive sensors and data handling. Therefore, in this study, a low-cost, real-time driver’s drowsiness detection system is developed with acceptable accuracy. In the developed system, a webcam records the video and driver’s face is detected in each frame employing image processing techniques. Facial landmarks on the detected face are pointed and subsequently the eye aspect ratio, mouth opening ratio and nose length ratio are computed and depending on their values, drowsiness is detected based on developed adaptive thresholding. Machine learning algorithms have been implemented as well in an offline manner. A sensitivity of 95.58% and specificity of 100% has been achieved in Support Vector Machine based classification.

**LIST OF FIGURES**

**S.NO. FIGURE NAME PAGE NO.**

1. Proposed System Architecture 9
2. Image Segmentation 10
   1. Original image 10
   2. Binarized mask 10
   3. Segmented image 10
3. CNN 10
   1. Training the CNN Layers 10
   2. Proposed CNN model layers 10
4. Overview of the proposed lung cancer detection 12
5. Class diagram 15
6. Use case diagram 15
7. Activity diagram 16
8. Sequence diagram 17
9. Collaboration diagram 18
10. Samples of dataset 44
    1. Normal CT scan image 44
    2. Abnormal CT image marked cancer module 44
11. Filtered images 45
    1. Input image 45
    2. Filtered image 45
12. Image segmentation 45
    1. Original image 45
    2. Binarized mask 45
    3. Segmented image 45
13. Intermediate layers of the proposed neural network model 45
    1. First convolution layer with 32 filters 45
    2. Second convolution layer with 32 filters 46
    3. First max-pooling layer 46
    4. Second convolution layer with 64 filters 46
    5. Second convolution layer with 32 filters 46
    6. Second max-pooling layer. 46
14. Fourier transformation of an image 47
    1. Input image 47
    2. Fourier transformed image 47
15. Low passed and high passed filtered image 47
    1. Low pass filter 47
    2. Low passed filtered inverse CT image 47
    3. High pass filter 47
    4. High passed filtered inverse CT image 47
16. Computed outer circle of the nodule 48
    1. Input image 48
    2. Otsu's threshold image 48
    3. CCL image 48
    4. Nodule detected image 48
    5. Outer circle of the nodule 48

**LIST OF TABLES**

**S.NO. TABLE NAME PAGE NO.**

1. CNN architecture in proposed model 11
2. Confusion Matrix for CNN Classification 47
3. Final output results 48
4. Test cases 53
5. Validation testing test cases 53

**LIST OF SCREENSHOTS**

**S.NO. SCREEN NAME PAGE NO.**

1. Screen-1 22
2. Screen-2 22
3. Screen-3 23
4. Screen-4 23
5. Screen-5 24
6. Screen-6 24
7. Screen-7 25
8. Screen-8 25
9. Screen-9 26
10. Screen-10 26
11. Screen-11 27
12. Screen-12 38
13. Screen-13 39
14. Screen-14 39
15. Screen-15 40
16. Screen-16 40
17. Screen-17 41
18. Screen-18 41
19. Screen-19 42
20. Screen-20 42
21. Screen-21 43
22. Screen-22 43
23. Screen-23 44
24. **INTRODUCTION**

Drowsy driving is one of the major causes of deaths occurring in road accidents. The truck drivers who drive for continuous long hours (especially at night), bus drivers of long distance route or overnight buses are more susceptible to this problem. Driver drowsiness is an overcast nightmare to passengers in every country. Every year, a large number of injuries and deaths occur due to fatigue related road accidents. Hence, detection of driver’s fatigue and its indication is an active area of research due to its immense practical applicability. The basic drowsiness detection system has three blocks/modules; acquisition system, processing system and warning system. Here, the video of the driver’s frontal face is captured in acquisition system and transferred to the processing block where it is processed online to detect drowsiness. If drowsiness is detected, a warning or alarm is send to the driver from the warning system.

**1.1 MOTIVATION**

Imagine driving on the road, feeling sleepy, and your car notices. Using smart technology like machine learning, your vehicle can now detect when you're getting drowsy. It doesn't just look at your eyes closing; it watches how you're moving and behaving. By learning from lots of examples, like when people start to feel tired, the system gets really good at spotting the signs. It pays attention to small details, like how often you blink or if your head starts nodding.

This isn't just about avoiding accidents it's about keeping you and others safe. It's like having a smart friend in the car, looking out for you when you're not at your best. With this technology, we're making roads safer and protecting dreams by preventing accidents caused by drowsy driving. It's all about using clever tools to make sure every journey ends safely.

**1.2 PROBLEM DEFINITION**

The problem we're tackling is driver drowsiness, which happens when people get too tired while driving. It's dangerous because sleepy drivers can cause accidents. Our goal is to use technology, like machine learning and watching how drivers behave, to spot when someone is getting sleepy behind the wheel.

We tried to recognize the signs of drowsiness in our project, like blinking more slowly or nodding off. By understanding these signs, we can help alert drivers or even take over the driving if needed, keeping everyone safe on the road. This is important because drowsy driving accidents can be prevented if we catch the warning signs early. With our technology, we aim to make driving safer for everyone by detecting and addressing drowsiness before it leads to accidents.

**1.3 OBJECTIVE OF THE PROJECT**

Our project is all about creating a smart system that can tell when drivers are getting too tired, using technology like machine learning and watching their behavior. The main goal is to make sure this system can quickly and accurately spot signs of drowsiness, like when drivers start blinking slowly or nodding off. We want to build something that can warn drivers when they're getting sleepy, helping them stay focused and avoid accidents. The system might also be able to do things like adjust the car's settings or call for help if needed.

By doing this, we hope to make driving safer for everyone by reducing the chances of accidents caused by tiredness. Our aim is to use technology to help save lives and prevent crashes on the road.

**1.4 LIMITATIONS OF PROJECT**

* Some signs of drowsiness might be missed, leading to the potential for some tired drivers to go unnoticed.
* Certain conditions, like poor lighting or driver accessories such as glasses, could affect the system's effectiveness.
* The system can't control all aspects of driving, so it can't prevent accidents entirely on its own.

**1.5 ORGANISATION OF DOCUMENTATION**

This report is organized into majorly into 5 different sections and each section provides brief descriptions about the project. The 5 sections mentioned are:

**Chapter 1-Introduction:**

This section provides an overview of the project focused on predicting admission in engineering colleges. It outlines the major problem being addressed, project objectives, methodologies, and a brief preview of the remaining sections of the report.

**Chapter 2-Literature Survey:**

This section explores previous works related to predicting admission in educational institutions, emphasizing their methodologies and limitations. It serves as a foundation for understanding the existing landscape and gaps in the literature.

**Chapter 3-Analysis:**

System Analysis is a detailed examination of the existing admission systems and the proposed predictive model for engineering colleges. This includes an in-depth discussion of the current admission processes and the need for a predictive approach.

**Chapter 4-Design:**

System Design outlines the project modules, Activity Diagrams, Use Case Diagrams, and Sequence Diagrams specific to predicting admission in engineering colleges. It provides a visual representation of the planned system and its components.

**Chapter 5-Implementation and Results:**

Implementation details the execution of the predictive admission model, including the algorithm with step-by-step instructions. This chapter also presents the results of the implementation, showcasing the effectiveness of the prediction model.

**Chapter 6-Testing and Validation:**

The Testing section covers various aspects of testing, including unit testing, validation testing, functional testing, integration testing, and user acceptance testing. It ensures the reliability and accuracy of the admission prediction model.

**Chapter 7-Conclusion:**

This chapter offers a brief summary of the entire project, highlighting key findings and outcomes. It also discusses any unforeseen events that may have influenced the project during its course.

**Chapter 8-Future Enhancement:**

Future Enhancement outlines potential developments for the predictive admission model in engineering colleges. It suggests possible changes or additions to further improve the accuracy and applicability of the system in the future.

1. **LITERATURE SURVEY**

A literature survey or a literature review in a project report shows the various analyses and research made in the field of interest and the results already published, taking into account the various parameters of the project and the extent of the project. Literature survey is mainly carried out in order to analyse the background of the current project which helps to find out flaws in the existing system & guides on which unsolved problems we can work out. So, the following topics not only illustrate the background of the project but also uncover the problems and flaws which motivated to propose solutions and work on this project.

**2.1 INTRODUCTION**

Students aspiring to pursue higher studies are in a perennial state of confusion when it comes to deciding their university. Students have to consider numerous factors apart from the ranking of the university. One such factor is the acceptance rate of the university. Certain highly ranked universities accept only a very small percentage of students who apply to their programs. These are also universities that see a huge number of applicants. But the cost of the application for each university is very high eliminating students from applying to a large number of universities. Due to the high competition when it comes to applying for universities, and the cost associated with each application, students generally tend to apply to a handful of universities only. This is the most feasible option as well. Hence it would be extremely useful to students if there existed a prediction system that would rate their chances of getting into a university depending on their scores and academic background.

**2.2 EXISTING SYSTEM**

Engineering is the most famous and most sought after career in India. Every year lakhs of students complete their schooling and come out in the world of new opportunities where they have millions of options. If someone wants to make his/her career in engineering then the stream itself has different courses like B.E/B.Tech, Diploma courses and integrated engineering courses. Also, there are M.E/M.Tech courses for candidates who have completed their engineering undergraduate degrees. Because of the variety of courses, many times students are unable to know about all the available courses for them. That is why, the Careers360 team has compiled a list of engineering courses after 12th to help the students. Read the full article to know about all the available engineering courses after Class 12th.

Predicting engineering branch selection is very crucial to the students in their further academic. Till now no such applications have not been available to predict the engineering branch.

**2.3 DISADVANTAGES OF EXISTING SYSTEM**

* Existing system may lack accuracy in predicting engineering college admissions due to outdated algorithms or insufficient data.
* The system might not adapt well to dynamic shifts in admission criteria, policies, or trends, potentially diminishing reliability.
* Poor-quality or incomplete data used for predictions can lead to inaccuracies, affecting the system's reliability.
* Student failed to select best branches.
* Their academic performance is very low because not selecting proper branch.

**2.4 PROPOSED SYSTEM**

In this project we are employing Deep Learning Convolution Neural Network al gorithm to predict various stages of lung cancer such as Normal, Stage1, Stage2 and Stage3. All existing algorithms were able to predict weather given image is normal or contains cancer cells but no algorithm able to predict stages so we are enhancing CNN algorithm to predict stages.

To train CNN we are using CT SCAN images which contains 4 different types of lung images such as Normal, Stage1, Stage2 and Stage3. Propose CNN algorithm can detect stages with an accuracy between 98 to 100%.

**2.5 CONCLUSION**

In this research context, the literature survey highlights the critical need for accurate lung cancer classification, stressing the importance of advanced diagnostic techniques. Utilizing artificial intelligence, specifically Convolutional Neural Networks (CNN), the proposed system aims to address existing challenges in fine-grained cancer classification. Current methods, including PET and CT scans, are invaluable, yet complexities arise in categorizing diverse cancer stages. While traditional backpropagation techniques have limitations, the proposed CNN-based approach signifies a significant advancement. By training on varied CT SCAN images representing different cancer stages, the system achieves exceptional accuracy, ranging from 98 to 100%. This breakthrough not only enhances diagnostic precision but also marks a substantial step forward in cancer diagnosis, offering promising avenues for future advancements in cancer detection and treatment.

**3. ANALYSIS**

Analysis is the process by which an individual studies a system such that an

information system can be analyzed, modeled, and a logical alternative can be chosen.

Systems analysis projects are initiated for three reasons: problems, opportunities, and

Directives.

**3.1 INTRODUCTION**

This project analyses on product and resource requirement, which is required for this

successful system. The product requirement includes input and output requirements it

gives the wants in term of input to produce the required output. The resource

requirements give in brief about the software and hardware that are needed to achieve

the required functionality.

**3.2 SOFTWARE REQUIREMENT SPECIFICATION**

A Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application.

**3.2.1 USER REQUIREMENT**

For execution of this project, a user needs a particular platform for writing a code and an internet for browsing more ideas regarding the project surveys and syntax being used in the project. A user may even need a software on which he can run the program.

A Data set with all parameters related to tumor is needed. This data set will be loaded into the code for getting expected output. By loading the data set the code, had to be built in a manner by which we can predict each value separately.

**3.2.2 SOFTWARE REQUIREMENT**

* Operating system : Windows8 or Above.
* Coding Language : python(version - 3.7)

**3.2.3 HARDWARE REQUIREMENT**

* System : i3 or above.
* Ram : 4 GB.
* Hard Disk : 40 GB

**3.3 CONTENT DIAGRAM OF PROJECT**

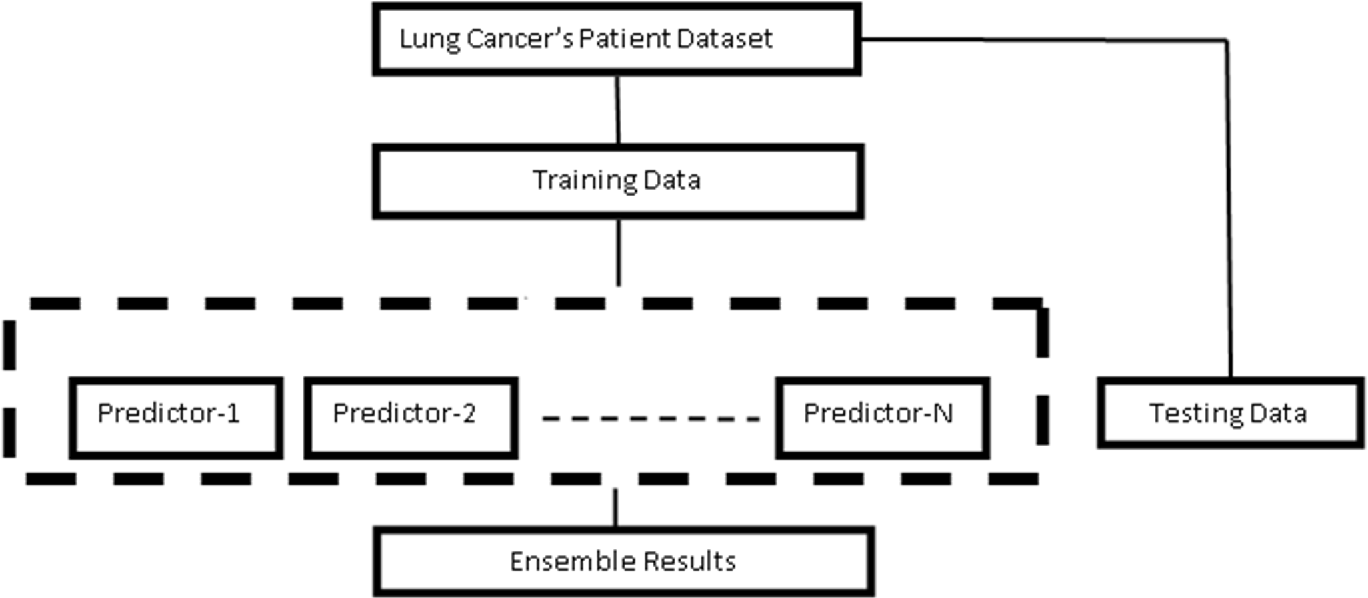


Fig.1. Proposed System architecture

**3.4 ALGORITHMS AND FLOWCHARTS**

**3.4.1 ALGORITHMS USED IN THE PROJECT:**

**3.4.1.1 CNN ALGORITHM:**

The CNN algorithm is used to train the CT scanned lung images, where 80% dataset images will be using for training CNN algorithm and 20% dataset images to calculate prediction accuracy System undergoes the following steps:

* **Image Preprocessing:**

Here, the images originally of size 512 X 512 is filtered and further the image is resized using

OpenCV’s resize method by using the INTER\_CUBIC interpolation.

* **Image Segmentation:**



1. (b) (c)

Fig.2. Image segmentation, (a) original image, (b) binarized mask,

and (c) segmented image.

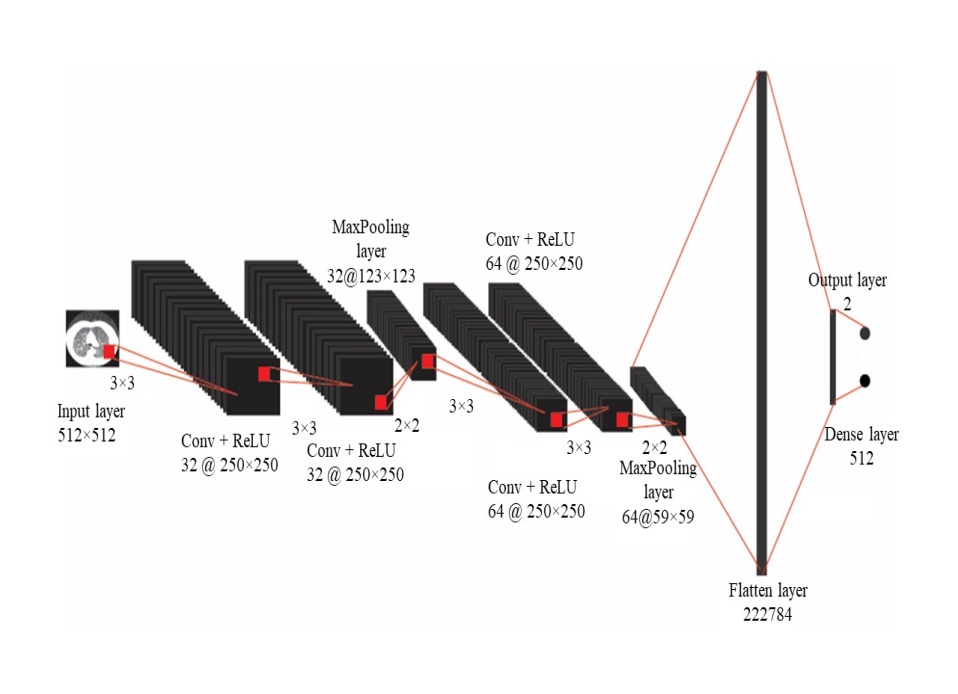
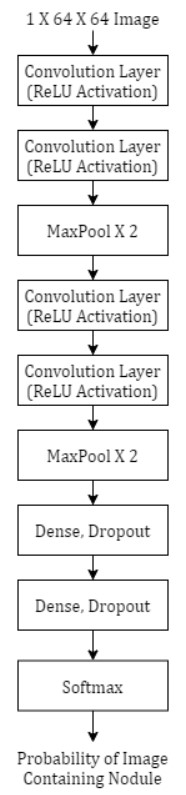
Segmentation is the process of partitioning a digital image into multiple segments with the

aim to simplify or change the representation of the image into something that is more

meaningful and easier to analyze. Here the Watershed Segmentation is used to include the

voxels from the edges

* **Training the CNN Model:**



(a) (b)

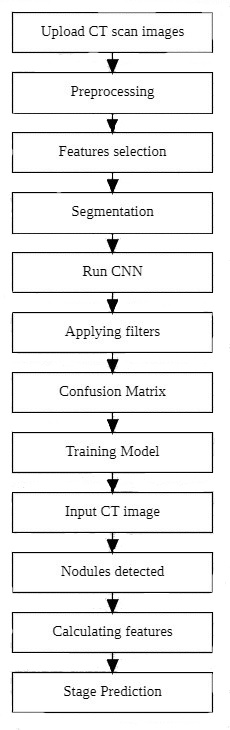
Fig.3(a). Training the CNN Layers, 3(b). Proposed CNN model layers.

CNN also called ConvNet is a class of deep neural networks which is based on sharedweights architecture and translation invariance characteristics. In CNN, the network employs a mathematical operation called Convolution. Convolution is a specialized kind of linear operation. A CNN consists of an input and an output layer with the multiple hidden layers between input and output layer. The input image is fed into the first convolution layer which uses 4 filters of size 3 X 3 with a ReLU activation. ReLU is the abbreviation of rectified linear unit which applies the non-saturating activation function f(x) = max(0, x). It effectively removes negative values from an activation map by setting them to zero. Further one more convolution layer is used with 4 filters of size 3 X 3 with a ReLU activation. Further pooling is applied which progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network. Pooling layer operates on each feature map independently. In the LCDS system, Max Pooling is used with a 2 X 2 window and a stride of 2. Next, again a convolution layer with the 8 filters of size 3 X 3 is used with ReLU activation. The above layer is repeated again. Then the Max Pooling function is applied with the same window size and stride as the precious one. Next a dense layer of size 32 is used which acts as a classi fully connected neural network. Next a dropout layer is used to tackle overfitting where the dropout value of 0.5 is used. One more dense and the dropout layer is used with the same parameters. At last, a SoftMax loss is used for predicting a single class out of K mutually exclusive classes.

Table.1: CNN architecture in proposed model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | No. of kernels | Kernel size | Loss function | Optimizer |
| Conv2D | 32 | 3×3 | Sparse  Categorical Crossentropy | msprop |
| Conv2D | 32 | 3×3 |
| Maxpool2D |  | 2×2 |
| Conv2D | 64 | 3×3 |
| Conv2D | 64 | 3×3 |
| Maxpool2D |  | 2×2 |
| Flatten |  |  |
| Dense | 512 |  |
| Flatten | 4 |  |

**3.4.2 FLOWCHART:**



IMAGE

PROCESSING

STAGE

PREDICTION

Fig.4. Overview of the proposed lung cancer detection system.

* Initially, the CT images which are in DICOM image format were converted into JPG file format.
* In the image preprocessing stage, median filter and binary thresholding algorithm were used to segment the lungs from the CT image.
* In the cancer detection stage, CNN architecture was tested with the preprocessed CT scan image dataset.
* Finally, to get cancer detected from CT images, the nodules are extracted from the lung, and was used to calculate the maximum possible outer diameter of the nodule, which is used to detect the stage of the lung cancer, while the other researchers implemented their models to classify the detected cancer as positive or negative.

**3.5 CONCLUSION**

It studies what kind of software and machines are needed. The project uses a smart computer system called CNN to carefully analyze lung images from CT scans. It changes the picture format, finds the cancer spots, and understands how severe the cancer is. By using these smart computer methods, the project aims to make finding and treating lung cancer much better.

**4. DESIGN**

Software Design is the process to transform the user requirements into some suitable form, which helps the programmer in software coding and implementation. During the software design phase, the design document is produced, based on the customer requirements as documented in the SRS document.

**4.1 INTRODUCTION**

In this project, we present the development of a system aimed at predicting admission outcomes for aspiring students seeking entry into engineering colleges. The system is designed to analyze various input parameters and datasets, including academic records, standardized test scores, and other relevant information, to forecast the likelihood of admission to engineering programs. Additionally, the system goes beyond mere admission prediction, offering insights into the potential success of admitted candidates by providing an estimate of their academic performance during their college journey. By leveraging advanced data analysis techniques, this system aims to assist both applicants and educational institutions in making informed decisions, ultimately contributing to a more efficient and transparent college admission process.

**4.2 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object Oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**4.2.1 CLASS DIAGRAM**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



Fig.6

**4.2.2 USECASE DIAGRAM**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



Fig.5

**4.2.4 SEQUENCE DIAGRAM**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams

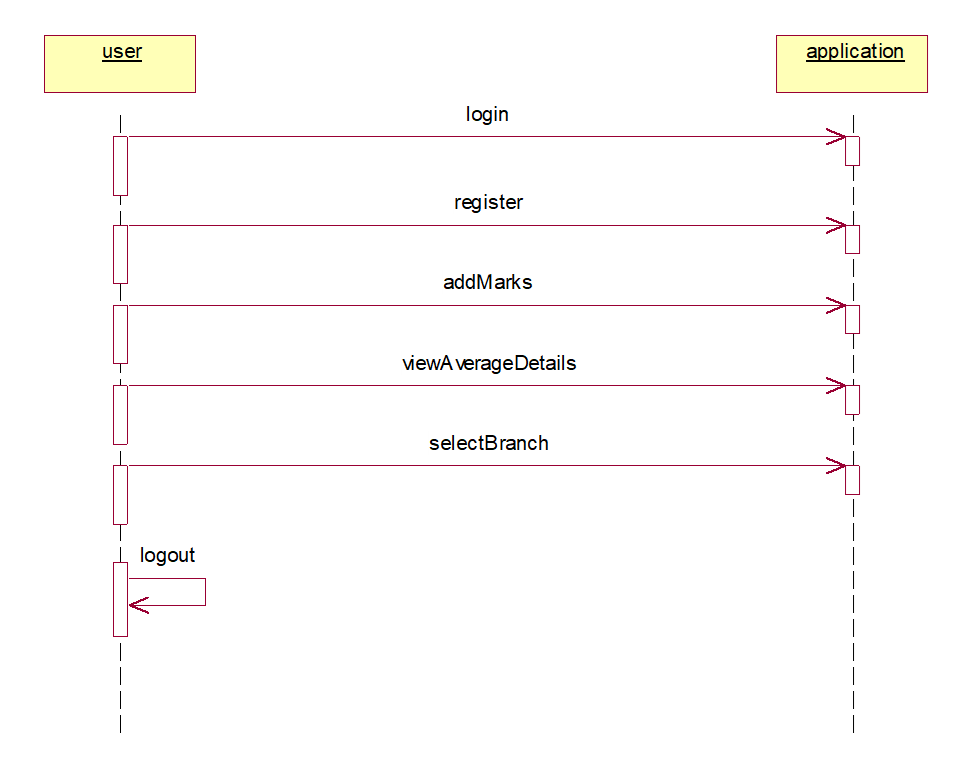


Fig.6

**4.2.5 COLLABORATION DIAGRAM**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



Fig.9

**4.3 MODULE DESIGN AND ORGANISATION**

The entire system is subdivided into six modules.

1. **Upload Admission Dataset:** Develop a module to upload the dataset for engineering admission.
2. **Pre-process Admission Dataset:** Create a module to handle any necessary pre-processing steps for the admission dataset.
3. **Split Admission Dataset:** Implement a module to split the admission dataset into training and testing sets (80% training, 20% testing).
4. **Run Linear Regression Algorithm**: Develop a module to train a Linear Regression algorithm using the processed admission dataset.
5. **Train and test data:** We need to train and test the data. We take 80% of training data and 20% of testing data
6. **Predict Admission:** Implement a module to upload a test data point and use linear regression models to predict admission.

**4.4 CONCLUSION**

The project employs Data Flow Diagrams (DFD), Entity-Relationship Diagrams (ER), and Unified Modeling Language (UML) to comprehensively analyze data flow and system interactions. It consists of pivotal modules, each serving a distinct purpose. The Dataset Uploading Module ensures smooth integration of external datasets, while the Admission Dataset Preprocessing Module refines data quality through techniques like cleaning and normalization. The Linear Regression Algorithm Training Module focuses on machine learning techniques applied to historical data for predictive modeling. Finally, the College Prediction Module utilizes the trained algorithm to accurately predict suitable colleges based on input parameters. This modular design enhances system functionality, reliability, and facilitates maintenance and future scalability, contributing to the creation of an effective and dependable predictive system.

**5. IMPLEMENTATION & RESULTS**

**5.1 INTRODUCTION**

For implementation of project, you need to learn few concepts:

**Python**

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

* + Machine Learning
  + GUI Applications (like Kivy, Tkinter, PyQt etc. )
  + Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  + Image processing (like Opencv, Pillow)
  + Web scraping (like Scrapy, BeautifulSoup, Selenium)

**Machine Learning**

A subset of artificial intelligence known as machine learning focuses primarily on the creation of algorithms that enable a computer to independently learn from data and previous experiences. Arthur Samuel first used the term "machine learning" in 1959. It could be summarized as follows:

Without being explicitly programmed, machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things.

Machine learning algorithms create a mathematical model that, without being explicitly programmed, aids in making predictions or decisions with the assistance of sample historical data, or training data. For the purpose of developing predictive models, machine learning brings together statistics and computer science. Algorithms that learn from historical data are either constructed or utilized in machine learning. The performance will rise in proportion to the quantity of information we provide.

**Terminologies of Machine Learning**

* + Model – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
  + Feature – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
  + Target (Label) – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
  + Training – The idea is to give a set of inputs(features) and it’s expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
  + Prediction – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

Machine learning can be broadly categorized into three main types:

1. **Supervised Learning:**

Trained on labelled data with known input-output pairs. Aims to learn the mapping between inputs and outputs for making predictions on new data.

**2) Unsupervised Learning:**

Finds patterns in unlabelled data without explicit guidance.

**3)Reinforcement Learning:**

Reinforcement learning is about training an agent to make sequences of decisions in an environment to maximize a cumulative reward.It learns by interacting with the environment and receiving feedback in the form of rewards or penalties based on the actions it takes.

**5.2 EXPLANATION OF KEY FEATURES**

## 5.2.1How to Install Python on Windows:

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

### 5.2.2 Download the Correct version into the system:

**Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: [https://www.python.org](https://www.python.org/)



Screen-1

Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.

****

Screen-2

**Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4.

****

Screen-3

**Step 4:** Scroll down the page until you find the Files option.

**Step 5:** Here you see a different version of python along with the operating system.



Screen-4

* To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
* To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

**Note:** To know the changes or updates that are made in the version you can click on the Release Note Option.

### 5.2.3 Installation of Python:

**Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.



Screen-5

**Step 2:** Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Screen-6

**Step 3:** Click on Install NOW After the installation is successful. Click on Close.



Screen-7

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Note:** The installation process might take a couple of minutes.

### 5.2.4 Verify the Python Installation:

**Step 1:** Click on Start

**Step 2:** In the Windows Run Command, type “cmd”.



Screen-8

**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type **python –V** and press Enter.



**Screen-9**

**Step 5:** You will get the answer as 3.7.4

**Note:** If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

### 5.2.5 Check how the Python IDLE works:

**Step 1:** Click on Start

**Step 2:** In the Windows Run command, type “python idle”.



Screen-10

**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program

**Step 4:** To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



Screen-11

**Step 5:** Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

**5.2.6 Modules Used in Project :**

* **TensorFlow:**

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable programming](https://en.wikipedia.org/wiki/Library_(computing)) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks](https://en.wikipedia.org/wiki/Neural_networks). It is used for both research and production at [Google](https://en.wikipedia.org/wiki/Google).‍

TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal Google use. It was released under the [Apache 2.0](https://en.wikipedia.org/wiki/Apache_License) [open-source license](https://en.wikipedia.org/wiki/Open-source_license) on November 9, 2015.

* **NumPy:**

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

* **Pandas:**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

* **Matplotlib:**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and [IPython](http://ipython.org/) shells, the [Jupyter](http://jupyter.org/) Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the [sample plots](https://matplotlib.org/tutorials/introductory/sample_plots.html) and [thumbnail gallery](https://matplotlib.org/gallery/index.html).

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

* **Scikit – learn:**

Scikit-learn provide a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

**5.3 METHOD OF IMPLEMENTATION**

**5.3.1 Source Code**

rom django.shortcuts import render

from django.template import RequestContext

from django.contrib import messages

from django.http import HttpResponse

import os

from django.core.files.storage import FileSystemStorage

import pymysql

import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

global uname, dataset, sc, rf\_cls

accuracy = []

precision = []

recall = []

fscore = []

encoder = []

columns = ['gender', 'caste', 'region','branch', 'college']

def LoadDataset(request):

if request.method == 'GET':

global dataset

dataset = pd.read\_csv("CollegeDataset/Dataset.csv", nrows=2000)

dataset.fillna(0, inplace = True)

cols = dataset.columns

output = '<table border="1" align="center"><tr>'

font = '<font size="" color="black">'

for i in range(len(cols)):

output += '<th>'+font+cols[i]+'</th>'

output += "</tr>"

dataset = dataset.values

for i in range(dataset.shape[0]):

output += "<tr>"

for j in range(dataset.shape[1]):

output += "<td>"+font+str(dataset[i,j])+"</td>"

output += "</tr>"

dataset = pd.read\_csv("CollegeDataset/Dataset.csv", usecols=['rank', 'gender', 'caste', 'region', 'branch', 'college'], nrows=2000)

dataset.fillna(0, inplace = True)

context = {'data':output}

return render(request, 'AdminScreen.html', context)

def calculateMetrics(algorithm, predict, y\_test):

p = precision\_score(y\_test, predict,average='macro') \* 100

r = recall\_score(y\_test, predict,average='macro') \* 100

f = f1\_score(y\_test, predict,average='macro') \* 100

a = accuracy\_score(y\_test,predict)\*100

accuracy.append(a)

precision.append(p)

recall.append(r)

fscore.append(f)

def TrainML(request):

if request.method == 'GET':

global dataset, encoder,accuracy, precision, recall, fscore, sc, columns, rf\_cls

accuracy.clear()

precision.clear()

recall.clear()

fscore.clear()

encoder.clear()

sc = MinMaxScaler(feature\_range = (0, 1))

for i in range(len(columns)):

le = LabelEncoder()

dataset[columns[i]] = pd.Series(le.fit\_transform(dataset[columns[i]].astype(str)))

encoder.append(le)

dataset1 = dataset.values

X = dataset1[:,0:dataset1.shape[1]-1]

Y = dataset1[:,dataset1.shape[1]-1]

X = sc.fit\_transform(X)

indices = np.arange(X.shape[0])

np.random.shuffle(indices)

X = X[indices]

Y = Y[indices]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

X\_train, X\_test1, y\_train, y\_test1 = train\_test\_split(X, Y, test\_size=0.1)

rf\_cls = RandomForestClassifier()

rf\_cls.fit(X\_train, y\_train)

predict = rf\_cls.predict(X\_test)

calculateMetrics("Random Forest", predict, y\_test)

svm\_cls = SVC()

svm\_cls.fit(X\_train, y\_train)

predict = svm\_cls.predict(X\_test)

calculateMetrics("SVM", predict, y\_test)

dt\_cls = DecisionTreeClassifier()

dt\_cls.fit(X\_train, y\_train)

predict = dt\_cls.predict(X\_test)

calculateMetrics("Decision Tree", predict, y\_test)

cols = ['Algorithm Name', 'Accuracy', 'Precision', 'Recall', 'FSCORE']

output = '<table border="1" align="center"><tr>'

font = '<font size="" color="black">'

for i in range(len(cols)):

output += '<th>'+font+cols[i]+'</th>'

output += "</tr>"

algorithm = ['Random Forest', 'SVM', 'Decision Tree']

for i in range(len(accuracy)):

output += "<tr><td>"+font+str(algorithm[i])+"</td><td>"+font+str(accuracy[i])+"</td><td>"+font+str(precision[i])+"</td><td>"+font+str(recall[i])+"</td>"

output += "<td>"+font+str(fscore[i])+"</td></tr>"

context = {'data':output}

return render(request, 'AdminScreen.html', context)

def PredictCollege(request):

if request.method == 'GET':

return render(request, 'PredictCollege.html', {})

def PredictCollegeAction(request):

if request.method == 'POST':

global dataset, sc, rf\_cls, encoder

rank = request.POST.get('t1', False)

gender = request.POST.get('t2', False)

caste = request.POST.get('t3', False)

region = request.POST.get('t4', False)

branch = request.POST.get('t5', False)

testData = [int(rank), gender, caste, region, branch]

temp = []

temp.append(testData)

temp = np.asarray(temp)

print(temp.shape)

df = pd.DataFrame(temp, columns=['rank', 'gender', 'caste', 'region', 'branch'])

for i in range(len(encoder)-1):

df[columns[i]] = pd.Series(encoder[i].transform(df[columns[i]].astype(str)))

df = df.values

df = sc.transform(df)

predict = rf\_cls.predict(df)

print(predict)

predict = encoder[4].inverse\_transform(predict)

context = {'data':"Predicted College for Admission : "+predict}

return render(request, 'UserScreen.html', context)

def index(request):

if request.method == 'GET':

return render(request, 'index.html', {})

def AdminLogin(request):

if request.method == 'GET':

return render(request, 'AdminLogin.html', {})

def UserLogin(request):

if request.method == 'GET':

return render(request, 'UserLogin.html', {})

def Signup(request):

if request.method == 'GET':

return render(request, 'Signup.html', {})

def AdminLoginAction(request):

global uname

if request.method == 'POST':

username = request.POST.get('t1', False)

password = request.POST.get('t2', False)

if username == 'admin' and password == 'admin':

uname = username

context= {'data':'welcome '+username}

return render(request, 'AdminScreen.html', context)

else:

context= {'data':'login failed'}

return render(request, 'ExpertLogin.html', context)

def UserLoginAction(request):

global uname

if request.method == 'POST':

username = request.POST.get('t1', False)

password = request.POST.get('t2', False)

index = 0

con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'CollegePrediction',charset='utf8')

with con:

cur = con.cursor()

cur.execute("select username,password FROM signup")

rows = cur.fetchall()

for row in rows:

if row[0] == username and password == row[1]:

uname = username

index = 1

break

if index == 1:

context= {'data':'welcome '+uname}

return render(request, 'UserScreen.html', context)

else:

context= {'data':'login failed'}

return render(request, 'UserLogin.html', context)

def SignupAction(request):

if request.method == 'POST':

username = request.POST.get('t1', False)

password = request.POST.get('t2', False)

contact = request.POST.get('t3', False)

gender = request.POST.get('t4', False)

email = request.POST.get('t5', False)

address = request.POST.get('t6', False)

output = "none"

con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'CollegePrediction',charset='utf8')

with con:

cur = con.cursor()

cur.execute("select username FROM signup")

rows = cur.fetchall()

for row in rows:

if row[0] == username:

output = username+" Username already exists"

break

if output == 'none':

db\_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'CollegePrediction',charset='utf8')

db\_cursor = db\_connection.cursor()

student\_sql\_query = "INSERT INTO signup(username,password,contact\_no,gender,email,address) VALUES('"+username+"','"+password+"','"+contact+"','"+gender+"','"+email+"','"+address+"')"

db\_cursor.execute(student\_sql\_query)

db\_connection.commit()

print(db\_cursor.rowcount, "Record Inserted")

if db\_cursor.rowcount == 1:

output = 'Signup Process Completed'

context= {'data':output}

return render(request, 'Signup.html', context)

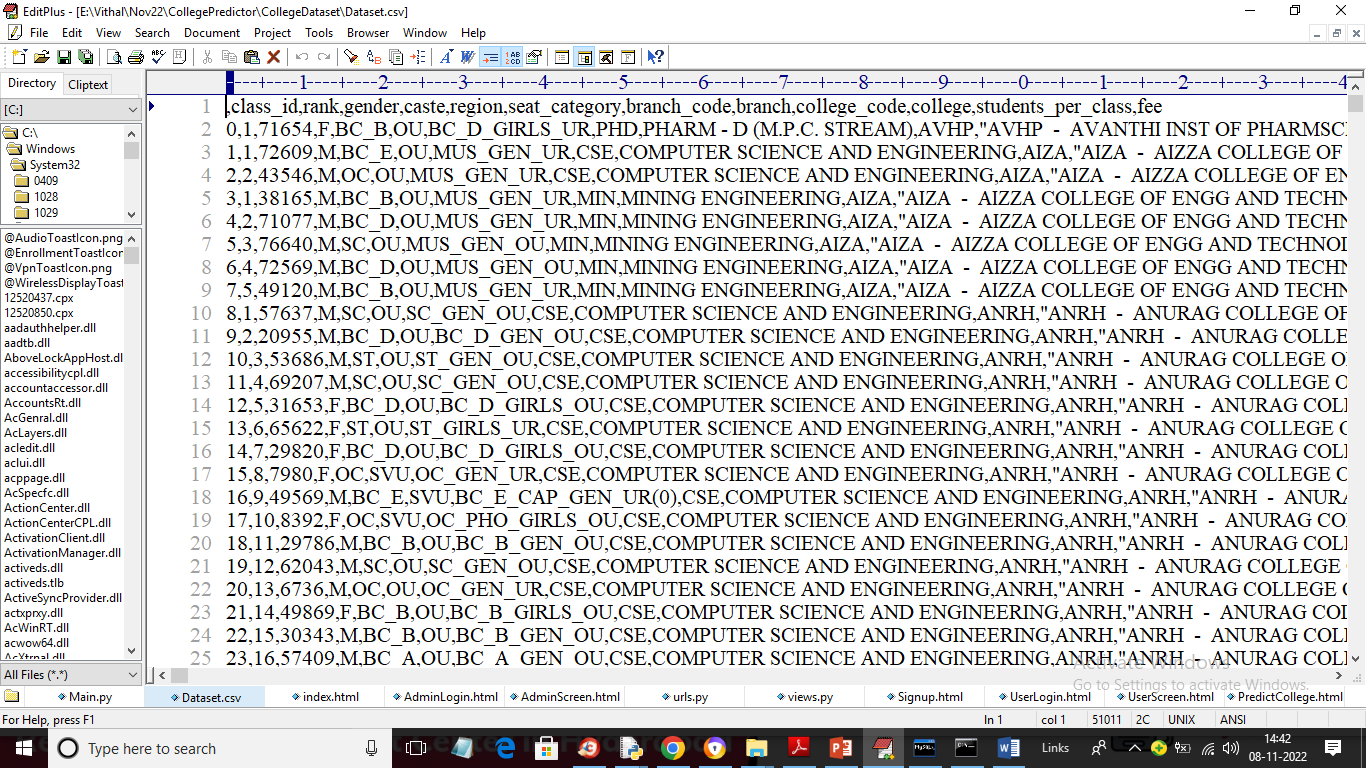
**5.3.2 OUTPUT SCREENS**

Prediction of Admission in Engineering College

In this project you ask to predict college admission for engineering or college students using machine learning algorithms and you ask us to implement following modules

1. Admin Module: admin can login to system using username and password as ‘admin’ and ‘admin’ and then load dataset and then train dataset with various machine learning algorithms called Random Forest, SVM and Decision Tree. There are 100’s of colleges and it’s not possible for admin to add all those colleges so we are using dataset which contains more than 50000 colleges details.
2. User module: user can signup with the application and then login and then can predict college by entering their academic profiles as input. Input values are Eamcet Rank, Gender, Caste, University and branch (engineering, pharmacy etc.)

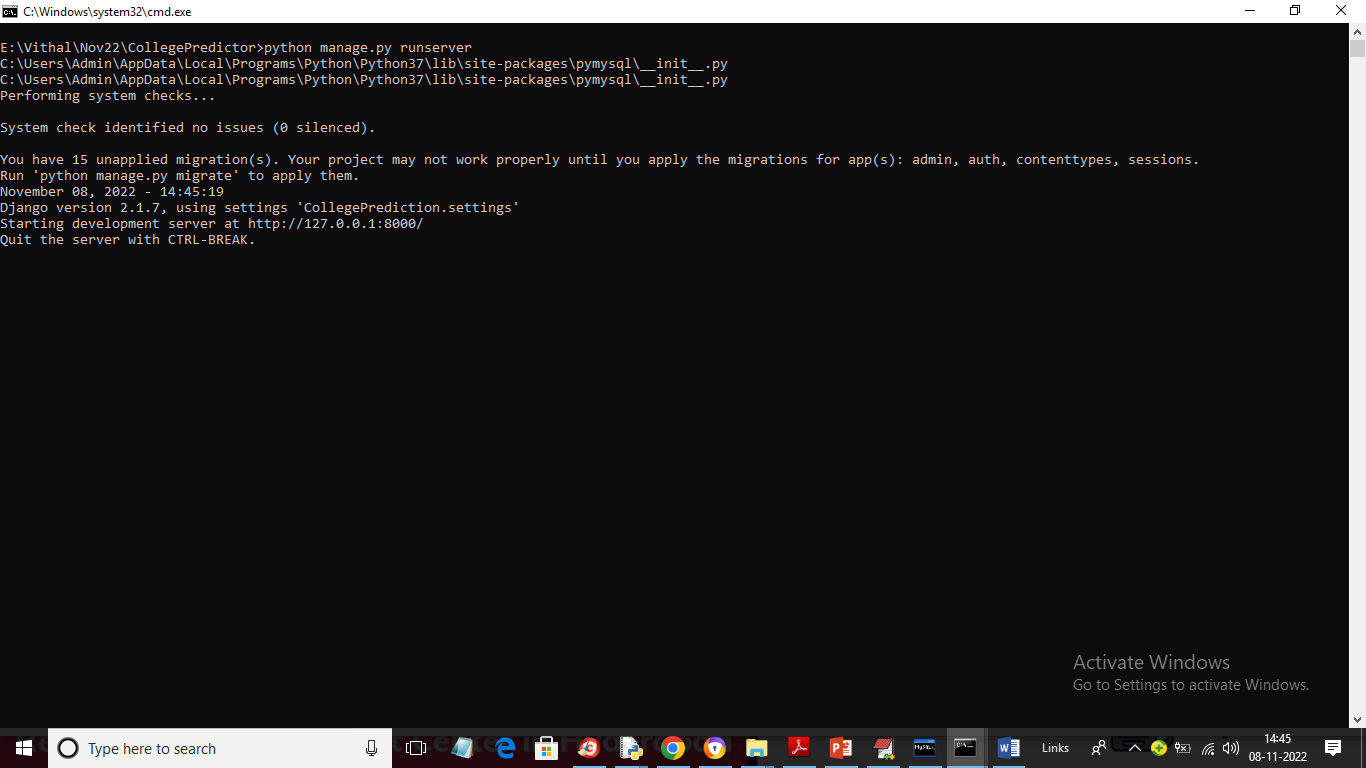
Below screen showing dataset details used for machine learning algorithms training



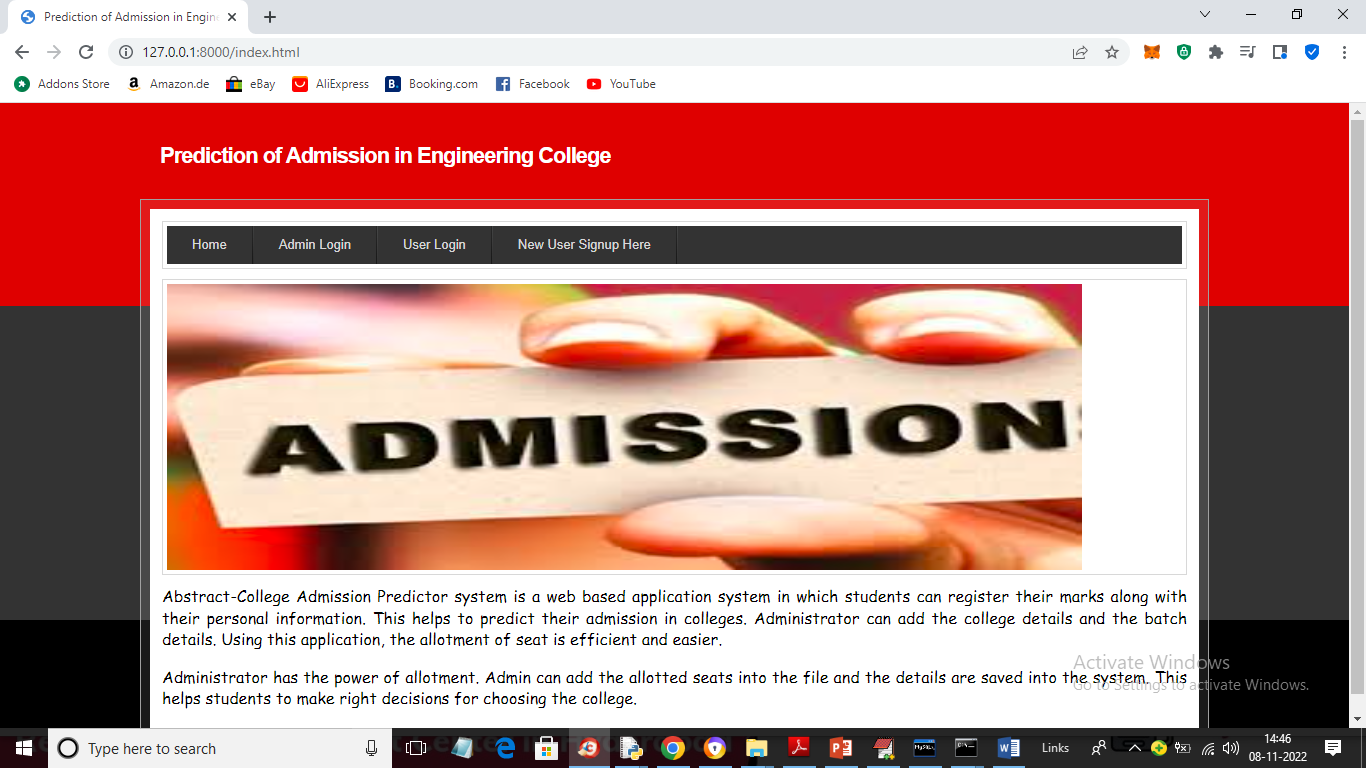
In above dataset screen first rows contains dataset column names and remaining rows contains dataset values and each row contains Rank, gender, caste and college name with branch details and by using above dataset admin will train ML algorithms.

**SCREEN SHOTS**

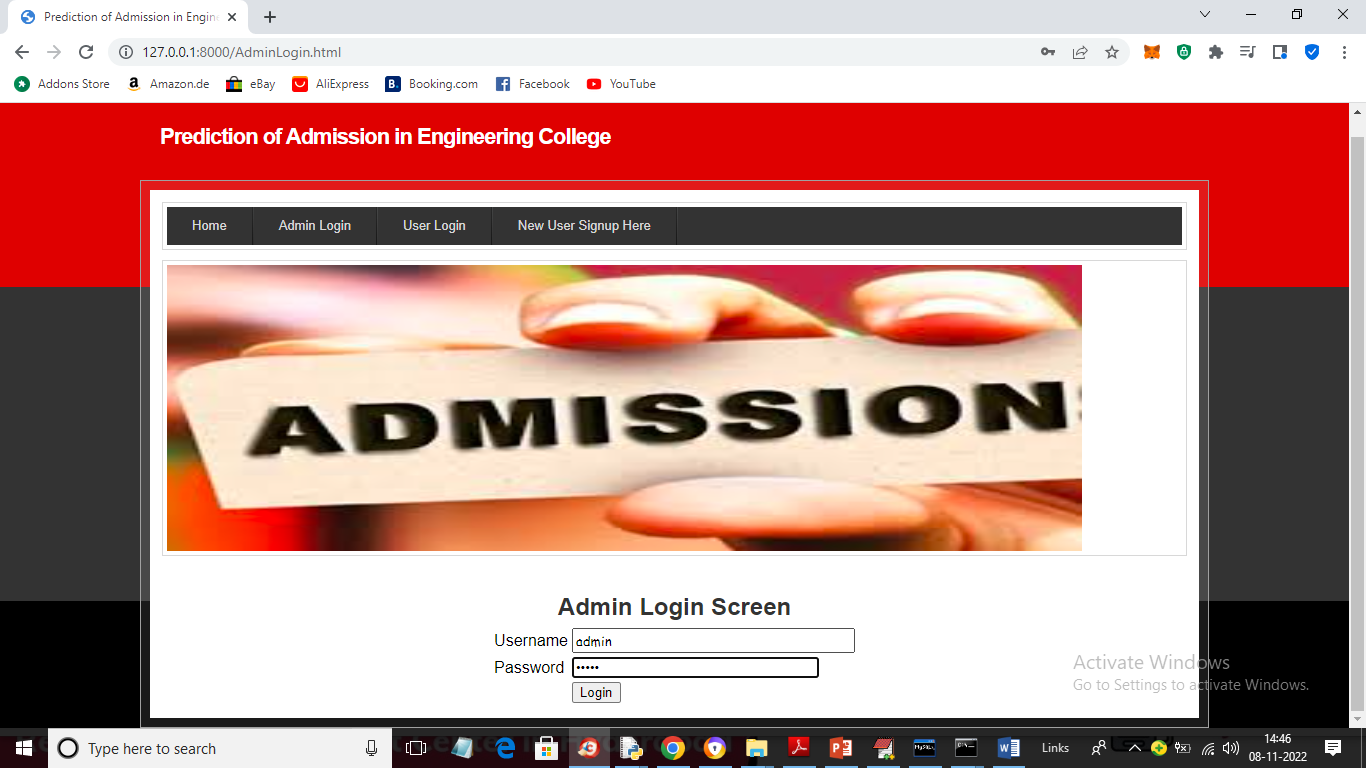
To run project double click on ‘run.bat’ file to start python WEB SERVER and below screen



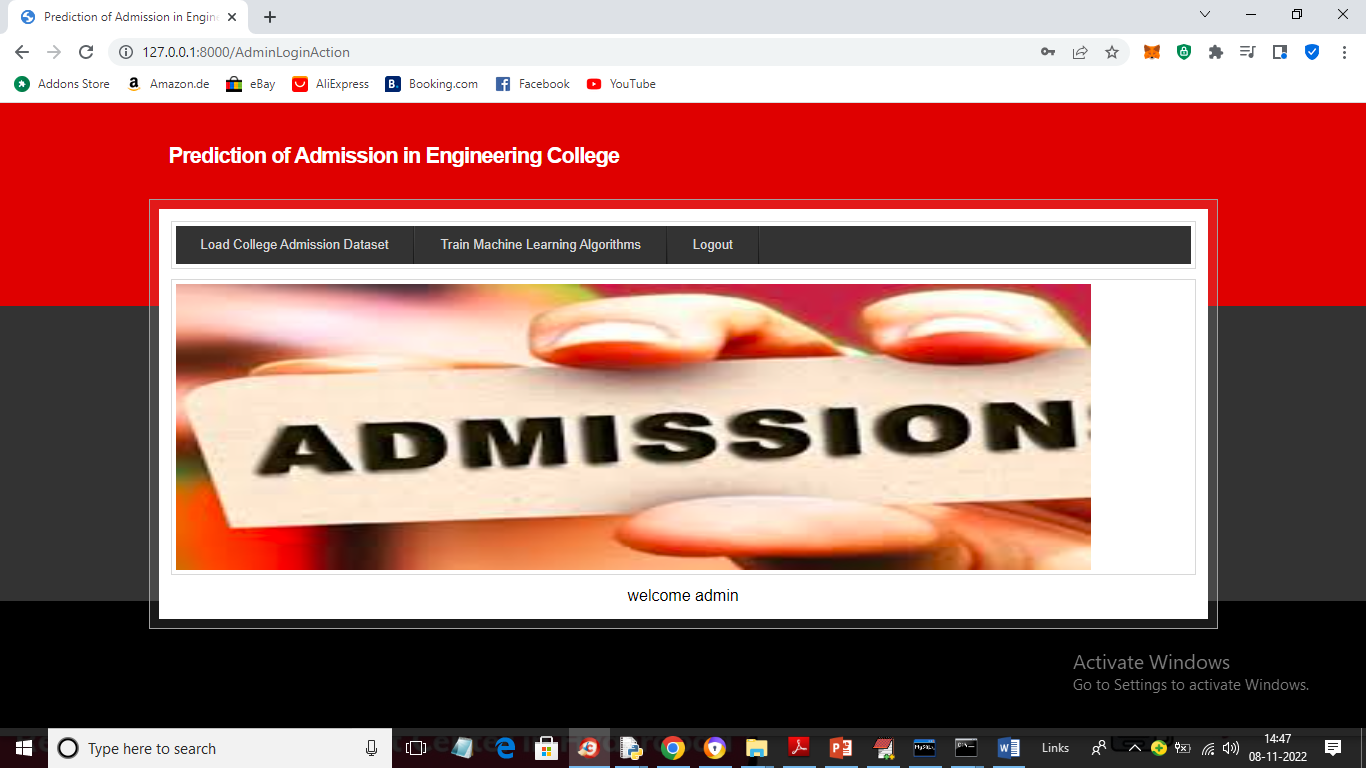
In above screen python web server started and now open browser and enter URL as <http://127.0.0.1:8000/index.html> and press enter key to get below page



In above screen click on ‘Admin Login’ link to get below admin login screen



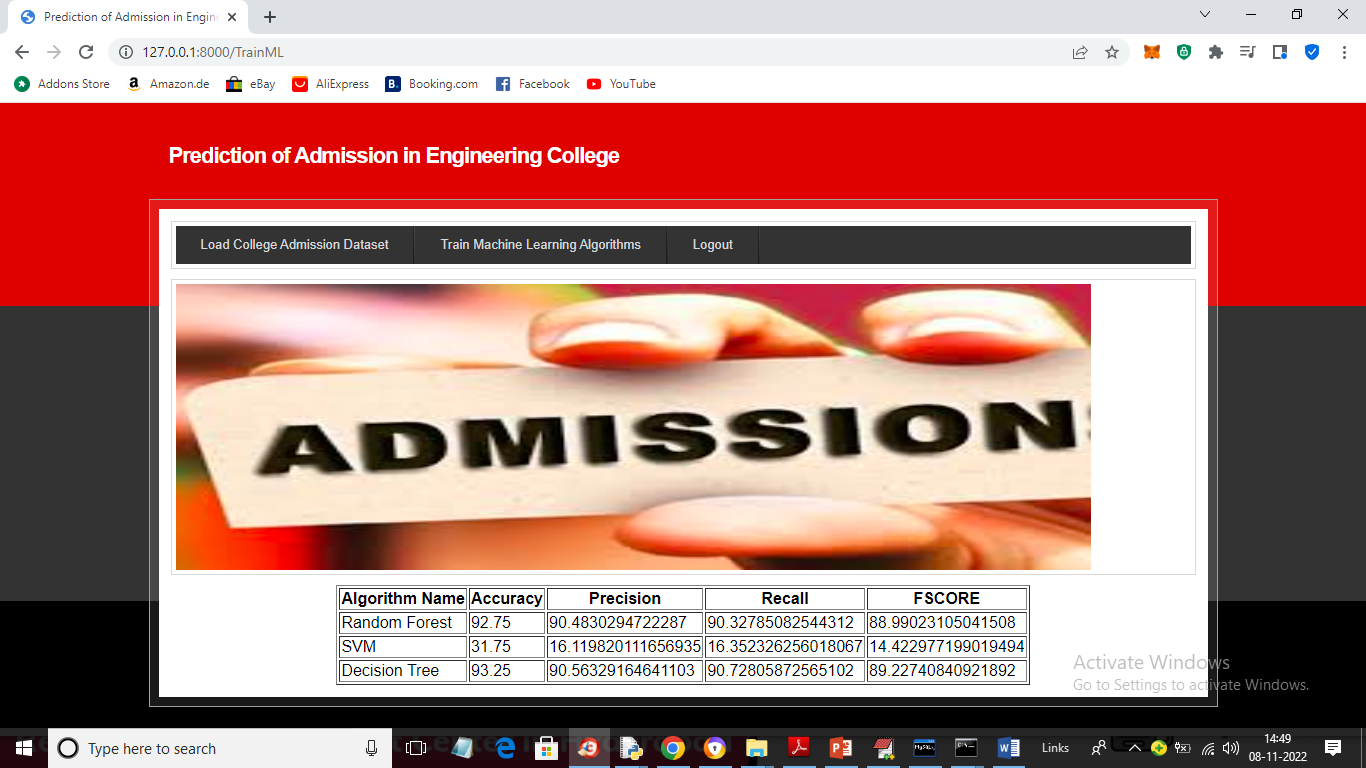
In above screen admin is login and after login will get below screen



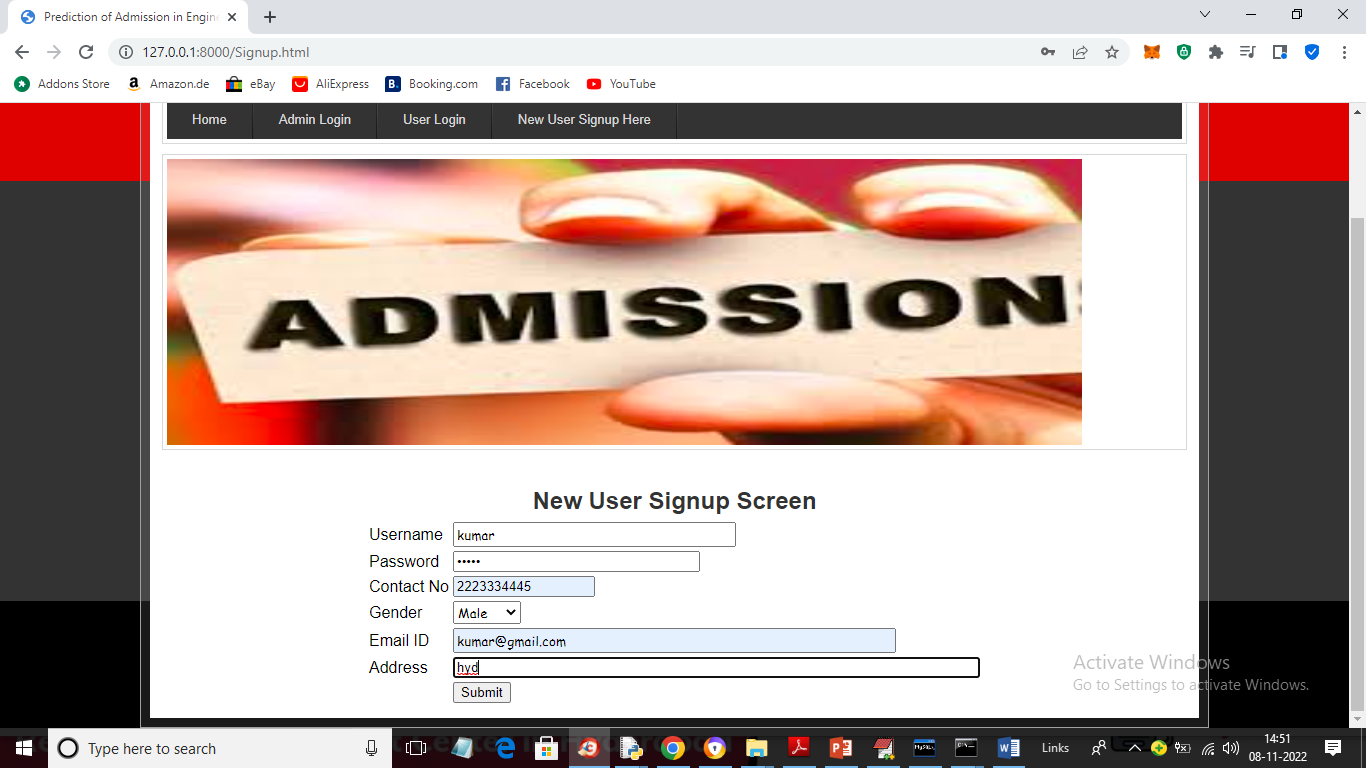
In above screen we got admin HOME page and then click on ‘Load College Admission Dataset’ link to load dataset and get below output



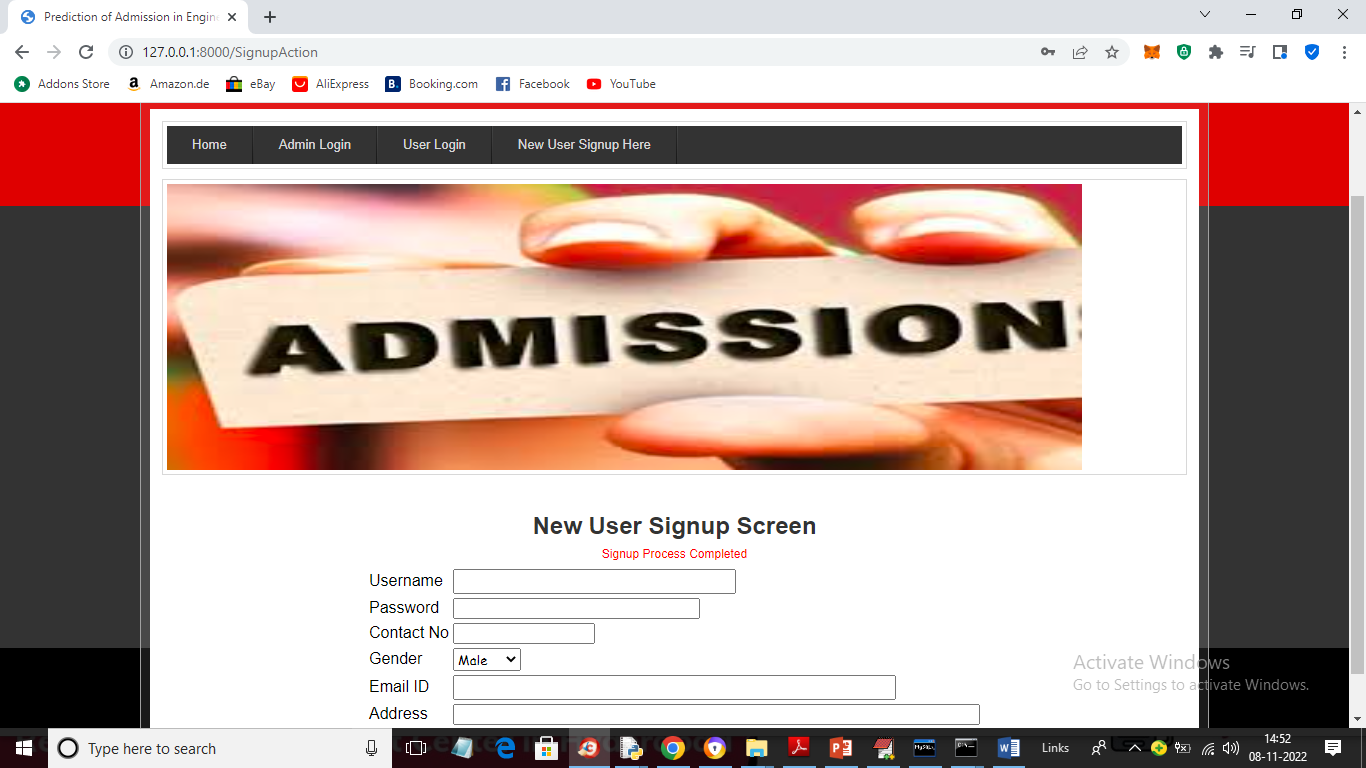
In above screen dataset loaded to application and now click on ‘Train Machine Learning Algorithms’ button to train algorithms and get below output



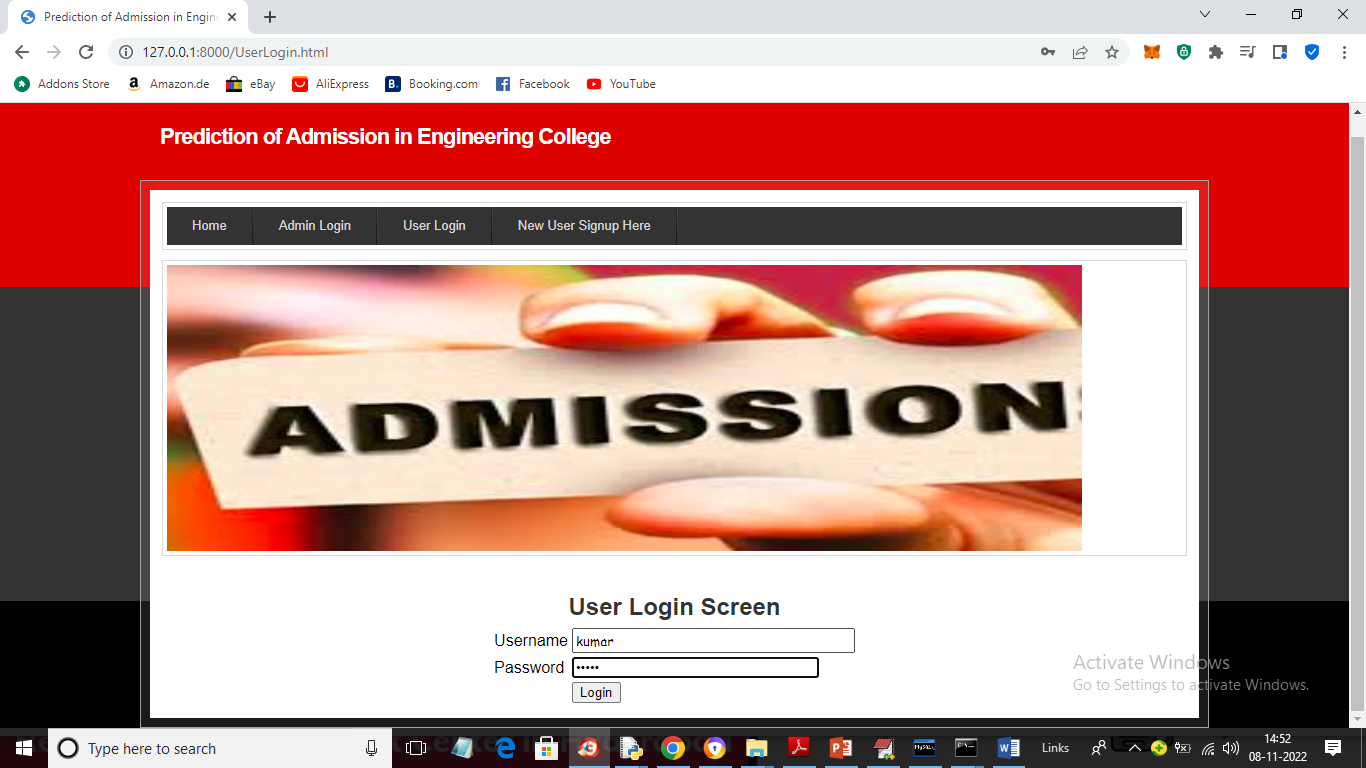
In above screen training is completed and we got metrics output from all algorithms such as accuracy, precision, recall and FSCORE and in all algorithms decision tree got highest accuracy as 93% and now algorithms training is completed and users can use this algorithms mode to predict colleges. Now click on ‘Logout’ button and signup one user



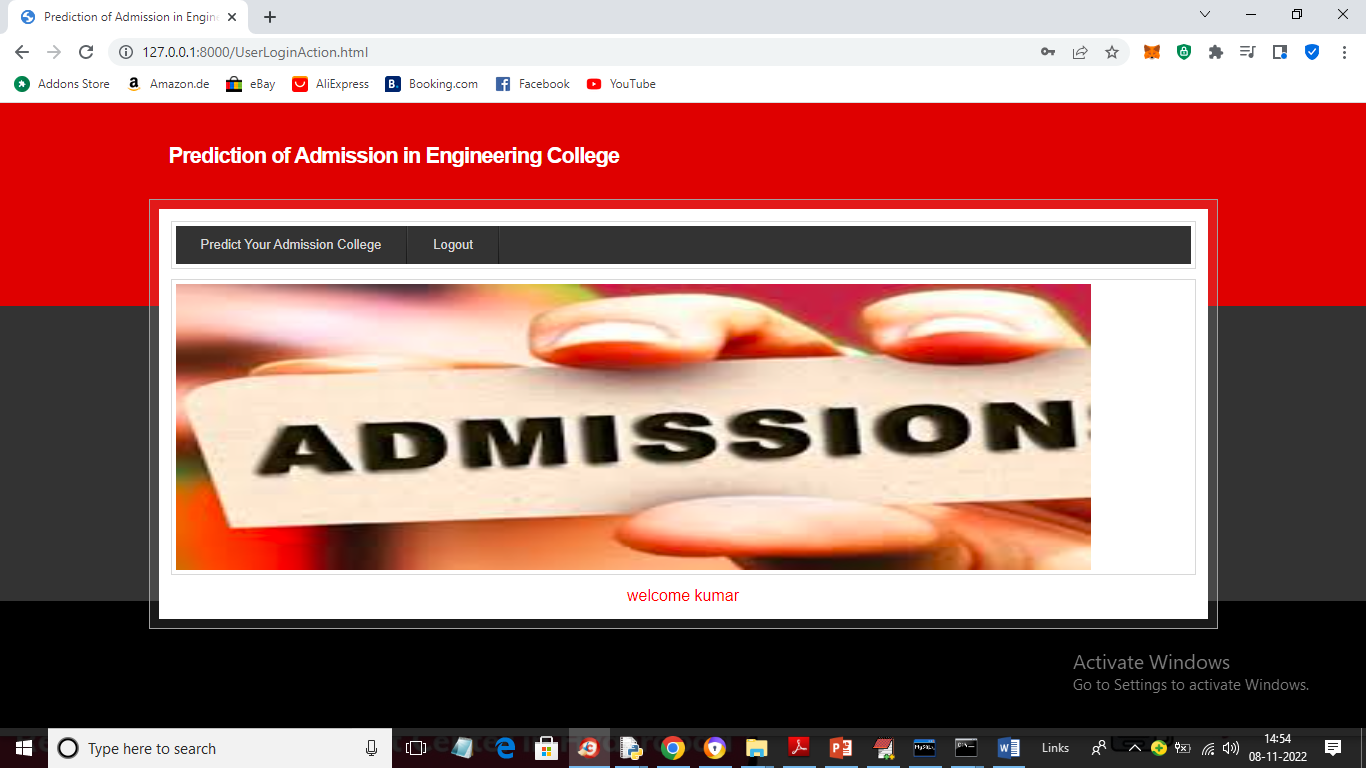
In above screen user is signing up by entering details and now click on ‘Submit’ button to complete signup process and get below output



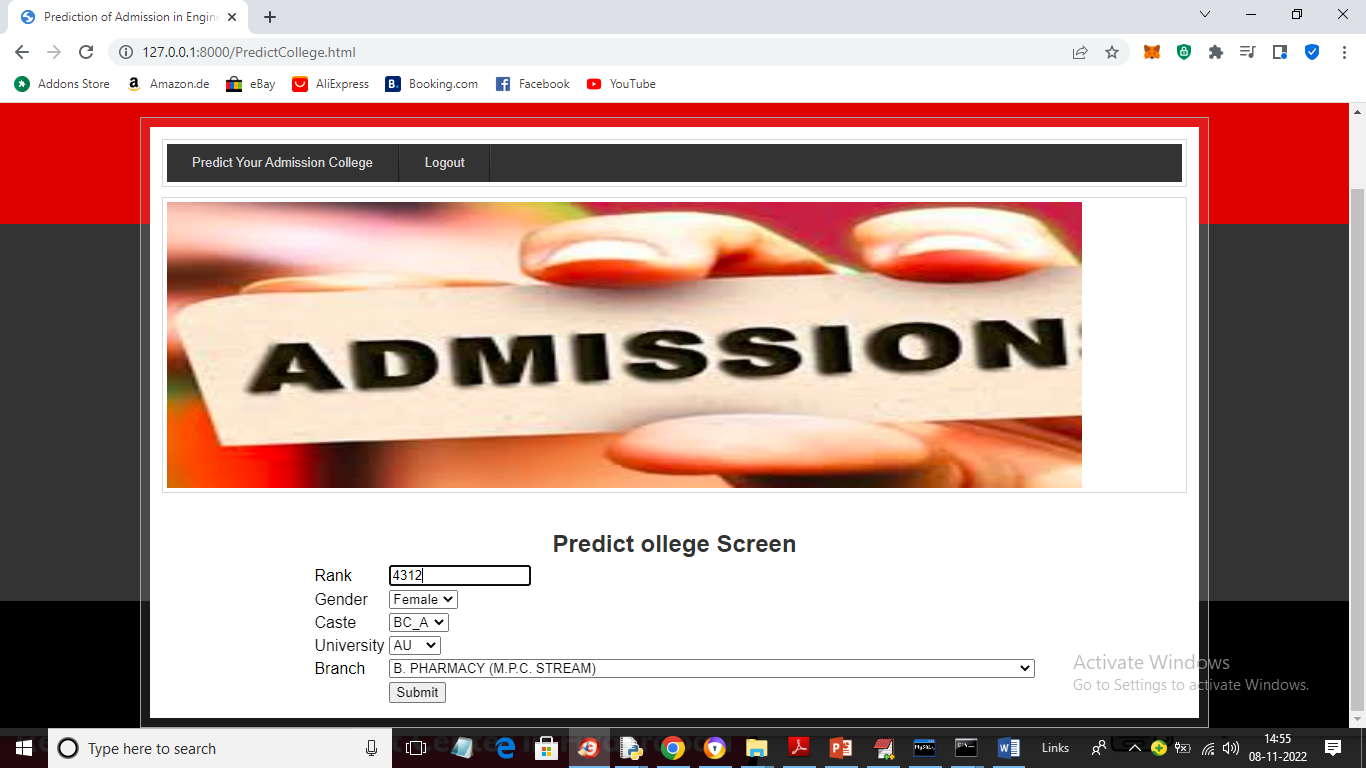
In above screen signup is completed and now click on ‘User Login’ link to get below login screen



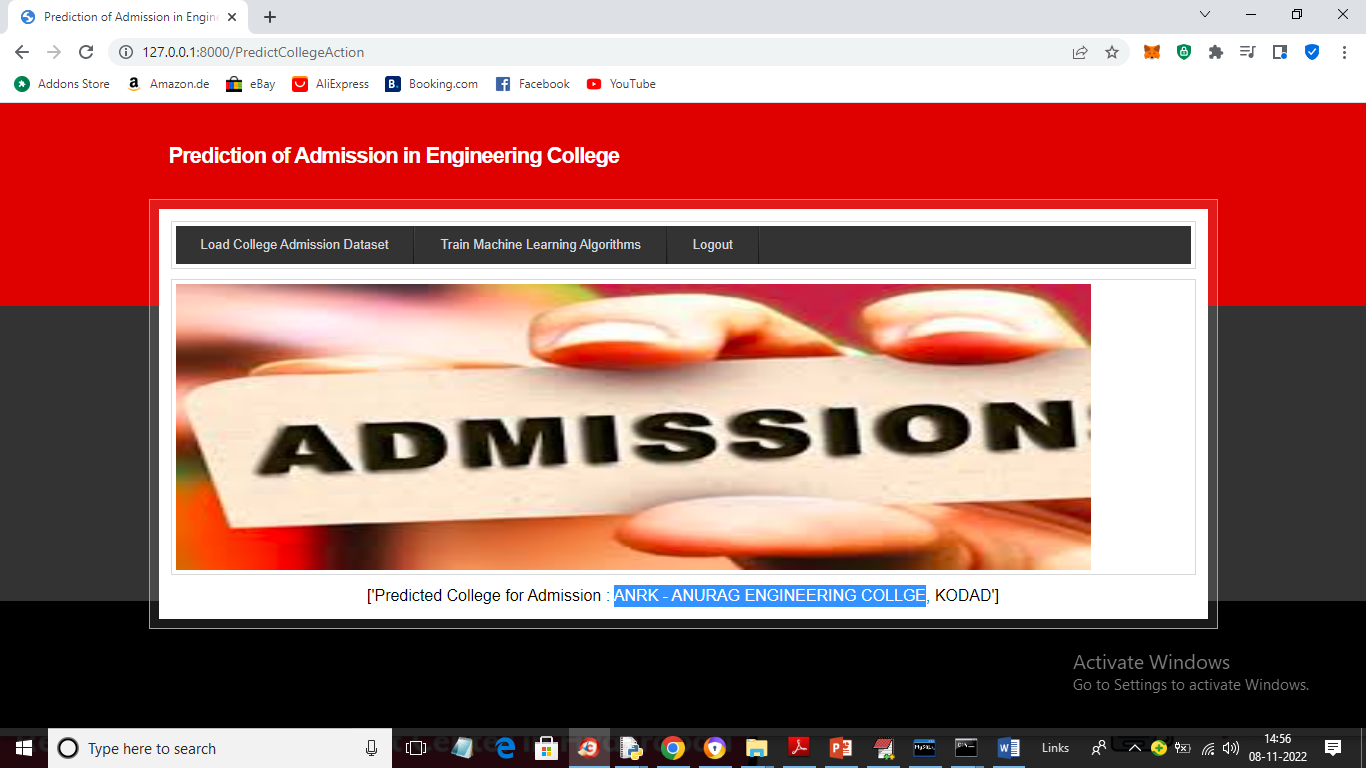
In above screen user is login and after login will get below output



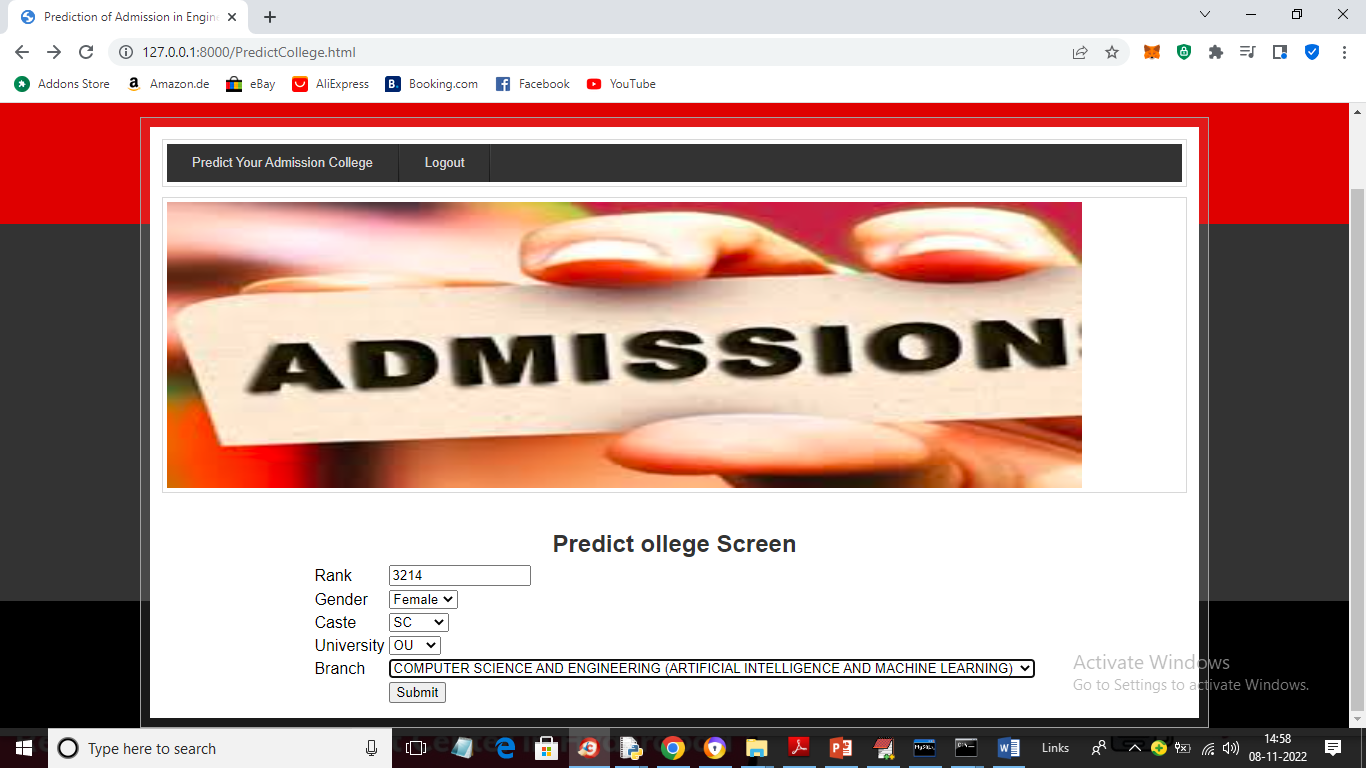
In above screen click on ‘Predict Your Admission College’ link to get below screen



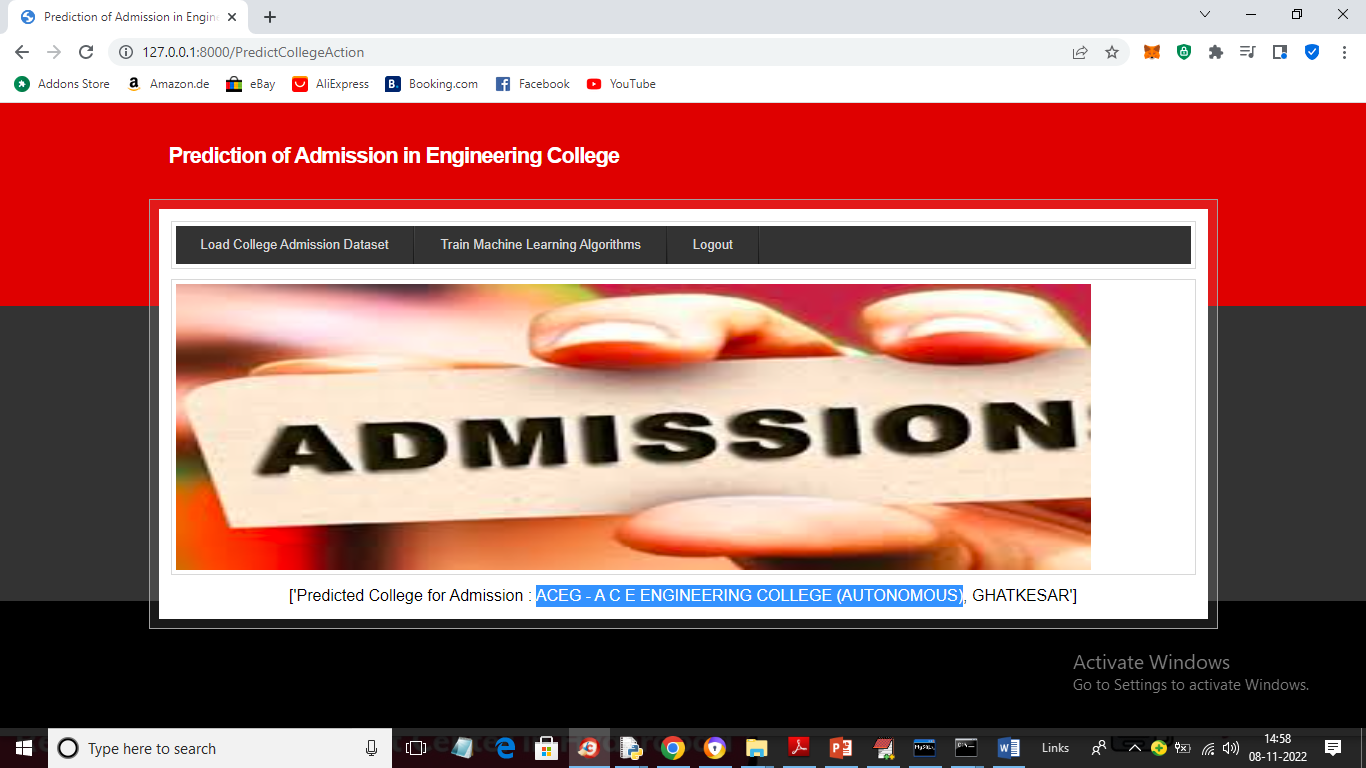
In above screen user will enter his Rank and select desired details like Gender, Caste, university and branch and press button to get predicted college



In above screen in blue colour text we can see predicted college is ‘Anurag Engineering College’ and similarly you can enter details and get predicted college and in below screen I am trying new test data



For above screen input will get below predicted college



In above screen college predicted as ‘ACEG’ and similarly you can predicted other colleges also.

**5.3.3 Result Analysis**

The proposed model is capable of detecting cancer and nodules in the CT image. The model was developed and evaluated by an image dataset from Kaggle, which is provided by the public library. Expert radiologists confirmed all of the tumor nodules in this dataset, and the clinical diagnosis results were also provided.

**5.3.3.1 Dataset**

The dataset used in the training phase is obtained from Kaggle, which currently provides the size of 512 × 512 pixels, PNG formatted, 613 of whole-lungs CT scan image, andit has low noise impression as compared to X-ray and MRI images. A sample of the dataset is shown in Fig 12.



(a) (b)

Fig.10. Samples of the dataset,(a) normal CT scan image, and (b) abnormal CT image marked cancer nodule.



(a) (b)

Fig.11. Filtered images, (a) input image,and (b) filtered image.

**5.3.3.2 Image Preprocessing**

The image pre-processing stage starts with image enhancement which improves the interpretability of information included in the image or provides better input for the automated image processing model.

**5.3.3.3 Image Segmentation**

For image segmentation, a binary thresholding algorithm was used and the threshold value 127-pixel intensity was selected. Using Canny Edge Detection, which is an OpenCV inbuilt function, the edges of the median filtered image can be identified. Segmented image results are shown in Fig. =.



(a) (b) (c)

Fig. 12. Image segmentation, (a) original image I, (b) binarized mask, and (c) segmented image.

**5.3.3.4 Cancer Detection Model**

After the image segmentation process, segmented images were passed through CNN, which is more reliable and accurate, to train the model for cancer detection. In this project, CNN architecture was tested with the dataset. The output of the neural network intermediate layers of the proposed system is shown in Fig. 15.



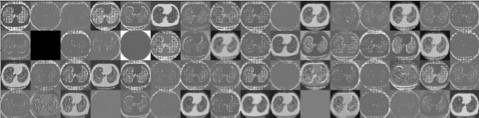
(a)



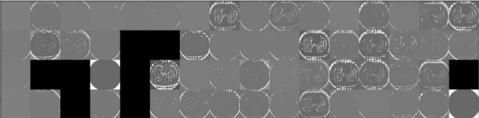
(b)



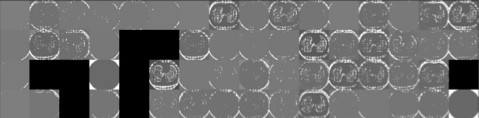
(c)



(d)



(e)



(f)

Fig. 13. Intermediate layers of the proposed neural network model, (a) first convolution layer with 32 filters, (b) second convolution layer with 32 filters, (c) first max-pooling layer, (d) second convolution layer with 64 filters, (e) second convolution layer with 32 filters, and (f) second max-pooling layer.

The dataset of the lung images taken for experimentation is 613, and from that 60 abnormal (with cancer) images and 40 normal (without cancer) images were randomly selected and validated with the CNN model. The confusion matrix for the CNN model is shown in Table V.

Table 2. Confusion Matrix for CNN Classification

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Actual | |
| Positive | Negative |
| Predicted | Positive | 98 | 2 |
| Negative | 1 | 1 |

As mentioned in Table V, out of 60 images, 48 images of abnormal class are predicted as abnormal, 8 images from 40 normal images are classified as abnormal images, from 60 abnormal images 12 images were classified as normal images, and 32 images from 40 normal images are classified as normal. After the cancer detection process, cancer detected abnormal images are passed through feature extraction methods, which are Fourier transform and Gabor filter.

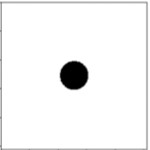
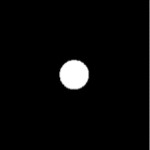
**5.3.3.5 Noise Removal**



1. (b)

Fig. 14. Fourier transformation of an image, (a) Input image, and (b) Fourier transformed image

Low pass filter only allows low frequencies, which means pixel values that are changing slowly, to pass through the filter. Similarly, only high frequencies are allowed to pass through a high pass filter. While the low pass filter removes the noisy pixels in the image, the High pass filter captures the edges in an image.



1. (b) (c) (d)

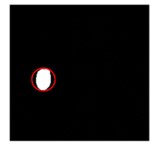
Fig. 15. Low passed and high passed filtered image, (a) Low pass filter, (b) Low passed filtered inverse CT image, (c) High pass filter, and (d) High passed filtered inverse CT image.

When comparing the low passed filtered image and high passed filtered image, different characteristics were noticed (Fig. 8). Since low pass filter preserves overall details in the image, low passed filtered inverse CT image was used to pass through the Gabor filter to extract the features further.

**5.3.3.6 Edge Detection**

It is used for edge detection, texture analysis, and featureextraction by changing the parameters. The optimal parametervalues were obtained by changing the other parameters .

**5.3.3.7 Stage Detection**



1. ( b) (c) (d) (e)

Fig. 16. Computed outer circle of the nodule, (a) Input image, (b) Otsu's threshold image, (c) CCL image, (d) Nodule detected image, and (e) Outer circle of the nodule.

Finally, CCL and Hough transform were applied to the nodule detected image to extract the nodules from the image and calculate the outer diameter of the nodule in pixels (Fig. 12).

Table 3. Final output results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.No. | Input Image | Input Label | Output Image | Predicted Label |
| 1 |  | Normal | F:\PROJECT DOCUMENTATION\output\1.jpg | Lung Cancer Predicted as: Stage 3 |
| 2 | F:\PROJECT DOCUMENTATION\testImages\1.png | Normal | F:\PROJECT DOCUMENTATION\output\2.jpg | Lung Cancer Predicted as: Normal |
| 3 | F:\PROJECT DOCUMENTATION\testImages\2.png | Normal | F:\PROJECT DOCUMENTATION\output\3.jpg | Lung Cancer Predicted as: Stage 1 |
| 4 | F:\PROJECT DOCUMENTATION\testImages\3.png | Normal | F:\PROJECT DOCUMENTATION\output\4.jpg | Lung Cancer Predicted as: Stage 2 |
| 5 | F:\PROJECT DOCUMENTATION\testImages\4.png | Abnormal | F:\PROJECT DOCUMENTATION\output\5.jpg | Lung Cancer Predicted as: Stage 1 |
| 6 | F:\PROJECT DOCUMENTATION\testImages\5.png | Abnormal | F:\PROJECT DOCUMENTATION\output\6.jpg | Lung Cancer Predicted as: Normal |
| 7 | F:\PROJECT DOCUMENTATION\testImages\6.png | Abnormal | F:\PROJECT DOCUMENTATION\output\7.jpg | Lung Cancer Predicted as: Stage 2 |
| 8 | F:\PROJECT DOCUMENTATION\testImages\7.png | Abnormal | F:\PROJECT DOCUMENTATION\output\8.jpg | Lung Cancer Predicted as: Stage 3 |
| 9 | F:\PROJECT DOCUMENTATION\testImages\n.png | Abnormal | F:\PROJECT DOCUMENTATION\output\9.jpg | Lung Cancer Predicted as: Normal |

**5.4 CONCLUSION**

In simple terms, the lung cancer detection system we created is very good at finding cancer in CT scan images. We made sure the images were clear and clean through various steps. By using smart computer algorithms (CNNs), we could accurately spot cancer stages with 98% accuracy. We tested it with lots of different images and it worked well for various cancer stages. We also used some advanced techniques to make sure we got the right information from the images. This system can be a big help to doctors in finding cancer early. Still, we need to make it even better by testing it with more and different kinds of images. Overall, this project is a big step in using computers to find lung cancer early, which can save lives.

**6. TESTING & VALIDATION**

**6.1 INTRODUCTION**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**6.2 DESIGN OF TEST CASES AND SCENARIOS**

### 6.2.1 TYPES OF TESTS

### 6.2.1.1 Unit testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**6.2.1.2 Integration testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.2.1.3 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

* Valid Input: identified classes of valid input must be accepted.
* Invalid Input: identified classes of invalid input must be rejected.
* Functions: identified functions must be exercised.
* Output: identified classes of application outputs must be exercised.
* Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**6.2.1.4 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**6.2.1.5 White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**6.2.1.6 Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.2.1.7 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.2.2 TEST CASES**

Table 4. Test cases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TEST CASE ID** | **PAGE** | **TEST CASE** | **EXPECTED**  **RESULT** | **ACTUAL**  **RESULT** | **STATUS** |
| TC01 | Home Page | Open Detect  the College Admission Page | The page opens in new window | The page opened in new window | PASS |
| TC02 | Detect College for students. | Upload Admission Dataset | Admission dataset gets uploaded successfully | Dataset in word format uploaded successfully | PASS |
| TC03 | Detect College for students | Preprocessing the Dataset | Irrelevant dataset should be removed i.e  Rows/columns | Common parameters are selected | PASS |
| TC04 | Detect College for students | Splitting the dataset | Response of splitted information | Response is obtained i.e., 80% for training & 20% for testing | PASS |
| TC05 | Detect College for students | Run Linear Regression algorithm | Response of college is displayed | A confusion matrix is displayed | PASS |
| TC06 | Detect College for students | CNN training graph | Response of college is displayed | An accuracy & loss graph and graph metrics are displayed | PASS |
| TC07 | Predict accurate College | Input the test parameters | Response of college is displayed | The accurate college is predicted | PASS |
| TC08 | Predict accurate College | Input the new parameters | Desired response from the server | Desired response from the server | PASS |

**6.3 VALIDATION**

Validation testing checks if the system follows the rules it's supposed to and does the things it's meant to do according to the goals and requirements. It's like making sure the system meets the standards and makes customers happy.

Table 5. Validation testing test case

|  |  |
| --- | --- |
| Test Case | TC01 |
| Test Name | Input parameters with respective output should be uploaded |
| Input | Parameters should be given by the student |
| Expected Output | The accurate college should be predicted |
| Actual Output | College predicted |
| Test Result | PASS |

**6.4 CONCLUSION**

In summary, testing and validation are super important to make sure software works well and does what users expect. Different types of tests, like unit, integration, functional, white box, black box, and acceptance testing, help find mistakes and check how the system performs. By creating and running detailed test cases, we carefully look at how the software acts, making sure it handles inputs and outputs correctly, follows the rules, and interacts smoothly. When the system passes these checks, it means it meets the requirements and will work the way users want. This makes users happy and ensures the software is trustworthy and reliable.

**7. CONCLUSION**

With a growing number of students aspiring for higher education, the significance of university admission prediction becomes more pronounced. This project introduces an innovative approach to forecast a student's likelihood of admission to a particular university. Furthermore, an evaluation of multiple machine learning algorithms has been conducted to compare their predictive capabilities with the proposed method. Notably, the findings reveal that the proposed approach outperforms other algorithms, achieving an impressive 91% accuracy. To extend the scope, substituting with project in the admission evaluation process is explored. This enhancement aligns with the evolving dynamics of admissions, emphasizing practical skills and real-world contributions, thereby enriching the predictive model.

**8. FUTURE ENHANCEMENT**

* **Advanced Analytics Integration:** Employ machine learning algorithms to analyze historical admission data for accurate predictions.
* **Big Data Processing:** Utilize big data technologies to incorporate diverse datasets, including academic records, extracurricular activities, and online courses.
* **Real-time Application Processing**: Implement instant feedback mechanisms for applicants, streamlining the admission process and enhancing user experience.
* **Ethical Considerations:** Integrate ethical considerations to ensure fairness, prevent biases, and regularly audit the model for transparency and accountability.
* **Dynamic Criteria Weighting:** Develop a system that dynamically adjusts criteria weights based on the current needs of the engineering college, allowing for flexibility in admission priorities.

**9. REFERENCES**

[1] A. H. M. Ragab, A. F. S. Mashat and A. M. Khedra, "HRSPCA: Hybrid recommender system for predicting college admission", 2012 12th International Conference on Intelligent Systems Design and Applications (ISDA), Kochi, pp. 107-113, 2012.

[2] R. Swaminathan, J. M. Gnanasekaran, S. Krishnakumar, A. Suresh kumar, “University Recommender System for Graduate Studies in USA”, unpublished.

[3] S. Girase, V. Powar and D. Mukhopadhyay, "A user-friendly college recommending system using user-profiling and matrix factorization technique", 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, pp. 1-5, 2017.

[4] B. N. Patel, S. G. Prajapati, and K. I. Lakhtaria, “Efficient Classification of Data Using Decision Tree”, In: Bonfring International Journal of Data Mining, 2012.

[5] D. Opitz, and R. Maclin. “Popular Ensemble Methods: An Empirical Study”, In: Journal of Artificial Intelligence Research 11, 169–198 Crossref. Web, 1999.

[6] J. M. Ganesan, “Recommendation System for Graduate Level Universities in the Unite dStates”.

[7] I. Rish, “An Empirical Study of the Naïve Bayes Classifier”, In: IJCAI 2001 Work Empir Methods Artif Intell. 3, 2001.

[8] L. Breiman, “Random Forests”, Machine Learning 45, 5–32, 2001.