ROAD SURFACE ANALYSIS AND CLASSIFICATION

A partial Report submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

CH VARUN SAI	**221910309003**
ABHINAY KOTLA	**221910309015**
MARAM ASHWITHA	**221910309023**
MARUPAKA VISHISHTA	**221910309024**

Under the esteemed guidance of

Dr. Sitharamulu

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE&ENGINEERING

GITAM

(Deemed to be University)

HYDERABAD

April - 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

We hereby declare that the project entitled **"Road Surface Analysis and Classification"** is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfilment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 30-03-2023

Name	Registration No.	Signature(s)
CH VARUN SAI	**221910309003**	
ABHINAY KOTLA	**221910309015**	
MARAM ASHWITHA	**221910309023**	
MARUPAKA VISHISHTA	**221910309024**	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled **"Road Surface Analysis and Classification"** is a bonafide record of work carried out by students and submitted **Ch. Varun Sai (221910309003), Abhinay Kotla (221910309015), M. Ashwitha (221910309023), M. Vishishta (221910309024)** in partial fulfilment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide	Project Coordinator	Head of the Department
Dr. Sitharamulu	**Dr. S. Aparna**	**Dr. K. S. Sudeep**
Assistant Professor **Dept. of CSE**	**Assistant Professor** **Dept. of CSE**	**Associate Professor & HOD** **Dept. of CSE**

ACKNOWLEDGEMENT

Our project report would not have been successful without the help of several people. We would like to thank the personalities who were part of our seminar in numerous ways, those who gave us outstanding support from the birth of the seminar.

We are extremely thankful to our honourable Pro-Vice-Chancellor, **Prof. D. Sambasiva Rao**, for providing the necessary infrastructure and resources for the accomplishment of our seminar. We are highly indebted to **Prof. N. Seetharamaiah**, Associate Director, School of Technology, for his support during the tenure of the seminar.

We are very much obliged to our beloved **Dr. K.S. Sudeep**, Head of the Department of Computer Science & Engineering, for providing the opportunity to undertake this seminar and encouragement in the completion of this seminar.

We hereby wish to express our deep sense of gratitude to **Dr. S.Aparna**, Project Coordinator, Department of Computer Science and Engineering, School of Technology and to our guide, **Dr. Sitharamulu**, Assistant Professor, Department of Computer Science and Engineering, School of Technology for the esteemed guidance, moral support and invaluable advice provided by them for the success of the project report.

We are also thankful to all the Computer Science and Engineering department staff members who have cooperated in making our seminar a success. We would like to thank all our parents and friends who extended their help, encouragement, and moral support directly or indirectly in our seminar work.

Sincerely,

Sincerely,

221910309003-CH. VARUNSAI

221910309015-K. ABHINAY

221910309023-M. ASHWITHA

221910309024-M. VISHISHTA

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

A smooth road surface leads to Happy Journey where a rough road surface leads to frustration of the driver and wastage of valuable travel time and fuel. Proper road maintenance measures like maintaining the needed men and machinery leads to a smooth surface. Other factors like road curvature and road type (like Ghat road) effects the road surface in long run. This project aims to use probabilistic machine learning models trained on dataset containing factors like climatic conditions, materials used, curvature of road, road length, road width, banking information, maintenance factors like people at work etc, to predict the health of the road. This can be used in predicting roads lifespan with a specific material set and features. Trained machine learning model can be used in applications like google maps to predict navigation path by considering the road conditions.

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

A smooth road surface leads to Happy Journey where a rough road surface leads to frustration of the driver and wastage of valuable travel time. Proper road maintenance measures like maintaining the needed men and machinery leads to a smooth surface. Other factors like road curvature and road type (like Ghat road) effects the road surface in long run. This project considers a dataset comprising of the above-mentioned maintenance factors and road features and analyses the dataset using Bernoulli Naive Bayes classification and Gaussian Naive bayes classification. The current roadway monitoring is expensive and not systematic. This paper proposes a new system able to evaluate the pavement quality of road infrastructure. The embedded system records and processes the acoustic data of the wheel-road interaction and classifies in real-time roadways' health thanks to integrated AI solutions.

1.2 OBJECTIVE

The project comprises of four modules. The first module deals with storing the road Maintenance factors to the database. The second module comprises of adding the Road Features to the Database. The third module deals with running Gaussian Naive Bayes Classification on the Road Maintenance dataset. The fourth module deals with the Execution of Bernoulli Naive Bayes Classification on the same dataset for comparison purpose.

1.3 LIMITATIONS

1.3.1 In the existing system, measures are taken to smoothen the road surface after the surface got damaged considerably. The project predicts about the smoothness of the road surface by using machine learning so that the surface can be repaired before the complete damage of the road happens.

1.3.2 Though this project is cost effective, there is still an uncertainty prevailing while comparing many attributes with various other application models.

1.3.3 To store certain information about a road surface, larger numbers are transferred into the database for analysis purposes, which is also continuous in nature. The outcome project would require faster database technologies for faster analysis of roads.

1.4 OUTCOMES

This project can predict the road condition considering all the necessary factors which are effective in analysing a road surface.

1.5 APPLICATIONS

A major application where this project can be useful is in road construction centres where they can refer to various attributes necessary in making a better surfaced road which are stored in the database. A website or an application can be derived from this project where the live analysis can be done to while travelling. The user only must enter the place at which he is traveling. Google maps can be useful in giving the accurate location to the application in order to analyse the road surface lively.

CHAPTER 2

LITERATURE REVIEW

2.1Introduction

2.1.1 ALESSIO GAGLIARDI (et.al,) The current roadway monitoring is expensive and not systematic. This paper proposes a new system able to evaluate the pavement quality of road infrastructure. The embedded system records and processes the acoustic data of the wheel-road interaction and classifies in real-time roadways' health thanks to integrated AI solutions. The measurements to produce the dataset to train a convolutional neural network (CNN) were collected using a vehicle operating at different cruise speeds in Pisa. The dataset is composed by acoustic data belonging to several typologies of roads: dirty or grass roads, high roughness surfaces and roads with cracks or potholes. The raw audio signals were split, labelled, and converted into images by calculating the Mel spectrogram. Finally, the authors designed a tiny CNN with a size equal to 18 kB able to classify between four different classes: good quality road, ruined road, silence and unknown. The CNN architecture achieves an accuracy of about 93% on the original model and 90% on the quantized one. Quantization permits to convert the final architecture into a suitable form to be deployed on a low-complex embedded system integrated in the tyre cavity.[1]

Road surface is an essential component of roadways. The main requirements a roadway should meet are evenness, tyre road friction, carrying capacity and low noise level. However, this infrastructure is subject to permanent stress and needs to be repaired or renewed to ensure the substance and utility value. Maintaining a good road surface quality is a major challenge for governments around the world. In fact, ruined surfaces are responsible for car accidents, poor driving quality as well as environmental noise. In addition, traffic noise and noise caused by motor vehicles are even considered as a serious health problem today. Nowadays most steps of the evaluation are done manually by an inspector who drives along the road, collects raw data, identifies the type of defects and their location, and calculates a specific index for road surface condition (International Roughness Index - IRI).

In the last years, researchers have presented several methods to address the problem of road anomalies detection. The number of devices designed for this task is increasing and mainly includes road profilometers, cameras and

inertial sensors. There are also few works based on acoustic sensors that represent a new research field. Table 1 shows a brief comparison of different sensors used in the literature for road quality detection. Each system can be combined with machine learning or physical model providing different outputs, e.g., evaluation index or defect type. The technologies focused on automated road surface monitoring can be divided into different groups based on the output [2] as reported below. • Presence: it answers the question whether a defect exists in the given data or not. • Detection: it identifies the exact position of the defect within the street. • Measurement: it provides the spatial measurements of the defect, e.g., width and depth of pothole. Road profilometers are devices used to calculate parameters, which describe longitudinal and transverse evenness and the cross fall of a road surface. An example of this application is proposed by Sjogren et al. in [3] where profilometers are applied to measure rut depth. Unfortunately, these systems are very expensive (over 3000 $) and require to be mounted on special vehicles involving additional costs.

2.1.2Young Song and Vladimir Shin, A fundamental step of any classifier is feature extraction. Principal component analysis (PCA) is regarded as the main feature extraction method of compressing high-dimensional data and can reduce/remove redundant variables from the original input data. PCA feature extraction has found application in many areas, including image indexing, noise reduction, pattern recognition, regression estimation, and so on [4], [5]. It computes a compact and optimal description of the data set. To reduce the classification of high-dimensional inputs, the score vectors of the PCA are classified instead of the raw data. The PCA is applied to process and product monitoring [6]. The on-line data must be explained by the PCA for all of the score vectors-based methods of classification.

The frequently used method for detecting outliers is using the sample variance. A data is declared outlier if its distance to the sample mean is greater than twice the sample variance. However, in practice, the sample variance can easily be inflated by outliers, causing the masking problem and the method fails [7]. The robust alternative method is based on replacing the sample mean by the sample median and the sample variance by the median absolute deviation (MAD). In this work, we apply the MAD algorithm proposed in [8].

2.1.3M.V. Medvedev, V.I. Pavlov, Road surface is an upper layer of road way, which is under direct load of transport vehicles and exposure of nature factors [9]. Road surface quality is an estimation of acceptability of road surface for further exploitation. According to technical standards and descriptions is possible to determine two classes for good and bad road surface. If road surface has cracks, holes, track pits and road marking is missing, this road surface area is damaged and needs repairing ("bad" road surface). If road surface has smooth surface without defects and readily visible road marking, this road surface does not need repairing ("good" road surface). The major problem of road surface quality estimation using images obtained from dashboard cameras lies in fact that even human cannot always determine the road surface quality correctly. It is connected with the presence of other different elements of road environment (transport vehicles, traffic signs, pedestrians etc.). To reduce errors from automated system it is possible to consider the front area beside transport vehicle. According to technical standards width of the pathway is determined from 2.75 m to 3.75 m that is why the width of area beside transport vehicle not greater than 4 m should be considered [10]. Recommended distance between cars in cities lies between 3 m and 5 m that is why the length of area beside transport vehicle not greater than 5 m should be considered.

In early research most approaches of road surface quality estimation were based on road cracks detection. For example, in paper [11] four methods of threshold value for cracks detection estimation were considered: Otsu method, histogram regression-based method, relaxation method and Kitler method. Authors mentioned that histogram regression-based method gives best results for road surface cracks detection. Authors of paper [12] calculated the difference between modified Otsu method value and the half of standard derivation of the brightness of all image pixels as threshold value to divide pixels on two classes: potential cracks and nocracks. In work [13] pixels with minimal aggregated brightness were considered only in 4 directions (0°, 45°, 90°, 135°). Texture, form of cracks and pixel brightness were used as features for defects detection accuracy improvement.

2.2 Classification techniques

2.2.1 Convolutional neural network

In a preliminary phase, two convolutional neural networks were designed and compared: the first has a lightweight architecture and is referred to with the term Tiny; the other, characterized by an architecture more complex, is inspired by cnn-trad-fpool3. In this paper, we will refer to the second architecture with the term Conv for the sake

of brevity. The Conv model is composed by two convolutional layers with 64 filters each, one maxpooling layer and one fully connected layer. The presence of the two convolutional and the maxpooling layers assures very

good achievements and contributes to the regularization of the model and to greater efficiency in training and evaluation time. Both models accept as input a 49 × 40 image corresponding to the size of the spectrogram discussed above. Unfortunately, this involves that the Conv model has a size of ∼ 30 MB, being infeasible the deployment on limited memory embedded devices. Conversely, the so-called tiny model has only one level of convolution and is less deep than the Conv model. This is why it occupies a memory size of ∼ 18 kB making it a perfect candidate to be used in conjunction with microcontrollers. The reduced size is due to the fact that there is just one convolutional layer and the maxpooling layer is missing. Moreover, the size of the kernel is reduced, and the stride step is increased.

2.2.2 Support Vector Machine

One of the most well-liked supervised learning algorithms, Support Vector Machine, or SVM, is used to solve Classification and Regression problems. However, it is primarily employed in Machine Learning Classification issues. The SVM algorithm's objective is to establish the best line or decision boundary that can divide n-dimensional space into classes, allowing us to quickly classify new data points in the future. A hyperplane is the name given to this optimal decision boundary. SVM selects the extreme vectors and points that aid in the creation of the hyperplane. Support vectors, which are used to represent these extreme cases, are the basis for the SVM algorithm.

2.3 Hardware approach

2.3.1 Embedded systems

As regards the hardware components, a proper electronic board was developed. A block diagram of the components is depicted in Figure 8. The core of the board is the ESP32-WROOM-32D module [29], selected because it supports TensorFlow Lite and most of the libraries used for data processing. The internal microcontroller is an Xtensa dual core microprocessors 32-bit LX6 ultra-low power at 40 nm technology equipped

with 520 kB of SRAM, 4 MB of SPI FLASH memory and onboard antenna. These characteristics make the ESP32 module particularly suitable for IoT and AI applications, such as this. The microphone inside the tyre is connected to the board via an SMA connector. The acoustic signal from the microphone first passes through an active second-order Sallen-Key bandpass filter with a frequency range of 1 Hz to 16 kHz. Two single-package operational

amplifiers are employed for this purpose using a Texas Instruments TL072IDR [30]. Then, the filtered signal is sampled by the 12-bit SAR ADC internal to the ESP32 chip at 16 kHz sampling rate. Three chips are responsible for power management. The first is the MAX77757 from Maxim Integrated [31], which enables battery charging and control.

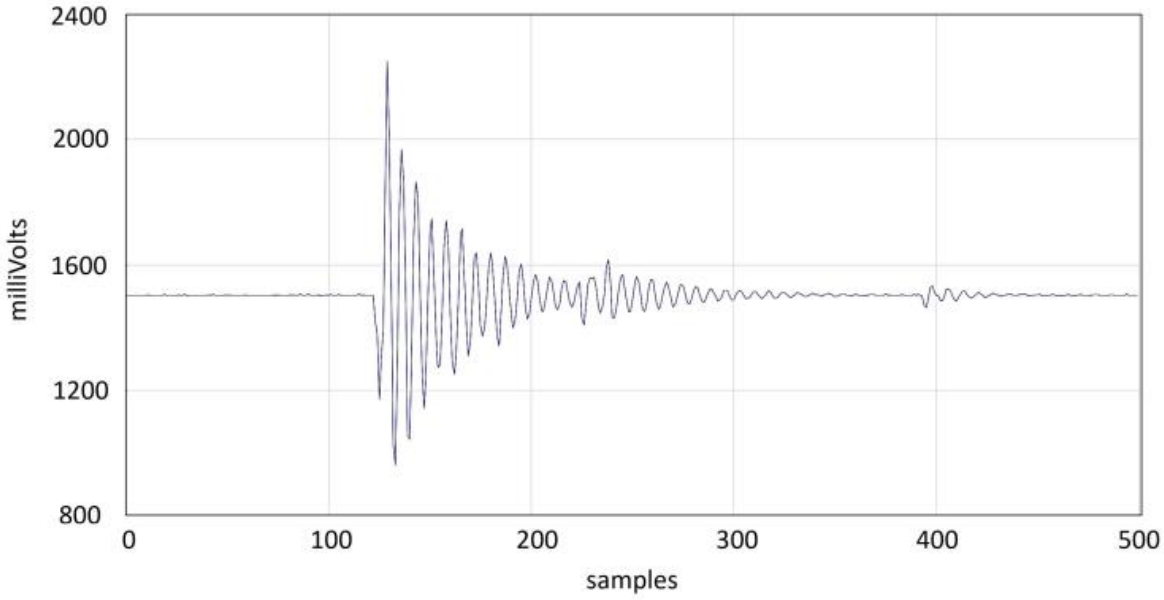

Fig 2.1: Acoustic output behaviour

CHAPTER 3

PROBLEM ANALYSIS

3.1 Problem Statement

The project aims at developing a tool for Road Surface Analysis and classification, with all the above-mentioned advantages. This tool is developed by using Python along with its layout toolkit PyQt, PyUIC and sklearn modules. There are four modules in the project. In the first module, the road maintenance factors are stored in the database. The second module consists of updating the database with the Road Features. Running Gaussian Naive Bayes Classification on the Road Maintenance dataset is the subject of the third module. The execution of Bernoulli Naive Bayes Classification on the same dataset for comparison purposes is the subject of the fourth module.

3.2 Existing System

The current roadway monitoring is expensive and not systematic. It proposes a new system able to evaluate the pavement quality of road infrastructure. The embedded system records and processes the acoustic data of the wheel-road interaction and classifies in real-time roadways' health thanks to integrated AI solutions. The measurements to produce the dataset to train a convolutional neural network (CNN) were collected using a vehicle operating at different cruise speeds in the area of Pisa. The dataset is composed by acoustic data belonging to several typologies of roads: dirty or grass roads, high roughness surfaces and roads with cracks or potholes. The raw audio signals were split, labelled, and converted into images by calculating the Mel spectrogram.

3.3 Advantages

1.The project is useful for the roads and buildings department to recognize bad roads so that they can initiate steps to repair those roads.

2.The project is useful for the tourists to avoid travel on bad road surfaces by planning their tour through alternate roads.

3.The project finally leads to improve the traveller's comfort.

3.4 Proposed System

A smooth road surface leads to Happy Journey where a rough road surface leads to frustration of the driver and wastage of valuable travel time. Proper road maintenance measures like maintaining the needed men and machinery leads to a smooth surface. Other factors like road curvature and road type (like Ghat road) effects the road surface in long run. This project considers a dataset comprising of the above-mentioned maintenance factors and road features and analyses the dataset using Bernoulli Naive Bayes classification and Gaussian Naive bayes classification.

3.5 Functional Requirements

3.5.1 It requires a 64-bit windows Operating System.

3.5.2 Python Qt Designer for designing user interface.

3.5.3 SQLite3 for storing database Entities.

3.5.4 Pyuic for converting the layout designed user interface (UI) to python code.

3.5.5 Python language for coding.

3.5.6 sklearn tool.

3.6 Non-functional requirements

3.6.1 It requires a minimum of 2.16 GHz processor.

3.6.2 It requires a minimum of 4 GB RAM.

3.6.3 It requires 64-bit architecture.

3.6.4 It requires a minimum storage of 500GB.

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

4.1.1 Architecture Diagram

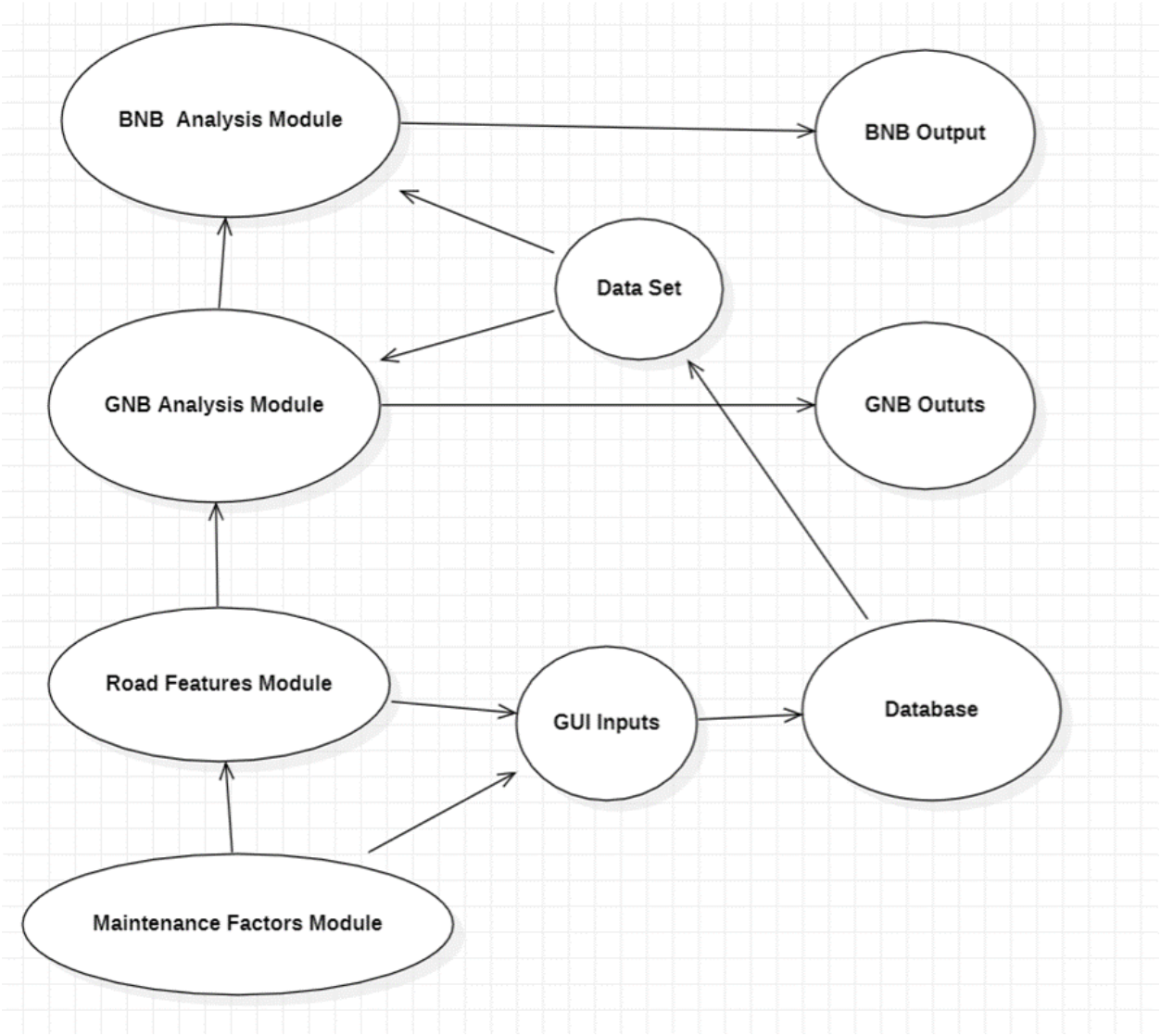

Fig 4.1: Architecture diagram

The project comprises of four modules. The first module deals with storing the road Maintenance factors to the database. The second module comprises of adding the Road Features to the Database. The third module deals with running Gaussian Naive Bayes Classification on the Road Maintenance dataset. The fourth module deals with the execution of Bernoulli naïve bayes classification on the same data set.

4.2 UML diagrams

A general-purpose modelling language is the Unified Modelling Language (UML). The primary goal of UML is to establish a uniform method for visualising a system's design process. It resembles blueprints used in other engineering disciplines quite a bit.

The best way to comprehend any complex system is to create some type of diagram or image. These illustrations have a stronger effect on our comprehension. If we take a closer look, we will see that the use of diagrams is widespread and takes many different shapes across a variety of industries.

To better and more clearly comprehend the system, we create UML diagrams. It would be impossible to depict every facet of the system in a single diagram. To cover the majority of a system's components, UML specifies a variety of diagram types.

UML was issued as a recognised standard by the International Organization for Standardization (ISO) in 2005. UML has undergone numerous revisions over time and is regularly examined.

4.2.1 Data Flow diagram

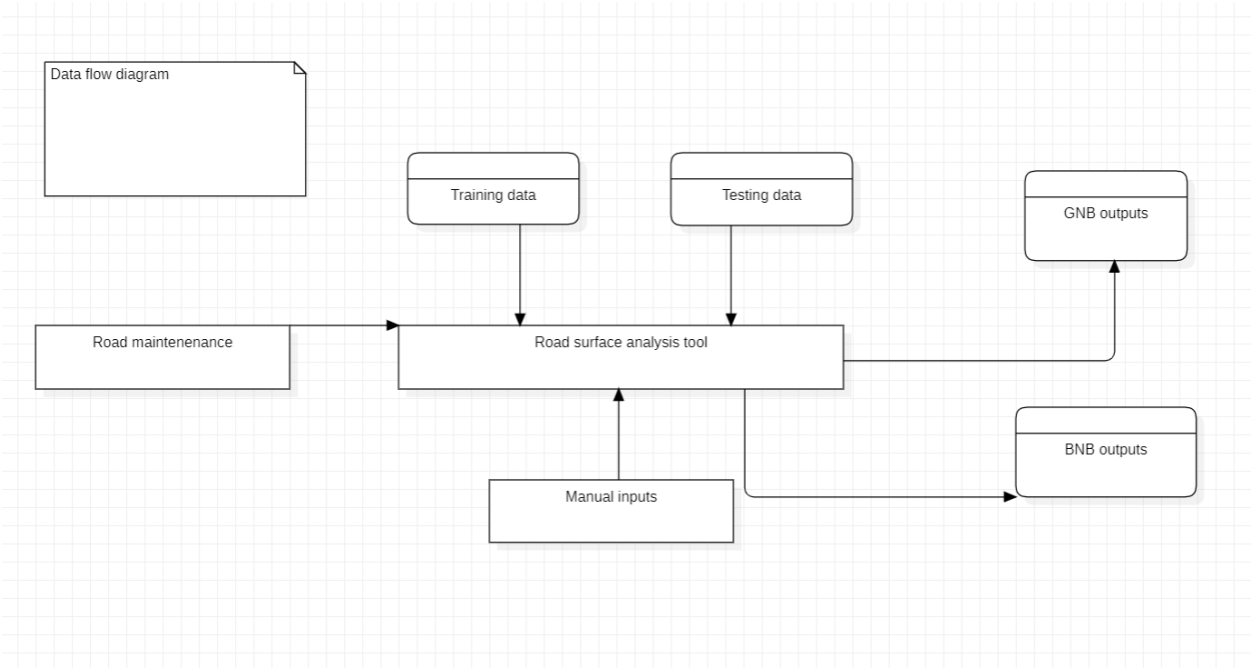

Fig 4.2: Data flow diagram

Description of Data Flow Diagram

Road Features and Maintenance Factors are to be provided as Input to the system, using the corresponding user interfaces. The system then calculates, BNB Accuracy & Predictions, GNB Accuracy & Predictions. Also, it generates a graphical plot showing the Training and testing Dataset sizes.

4.2.2 Use Case Diagram

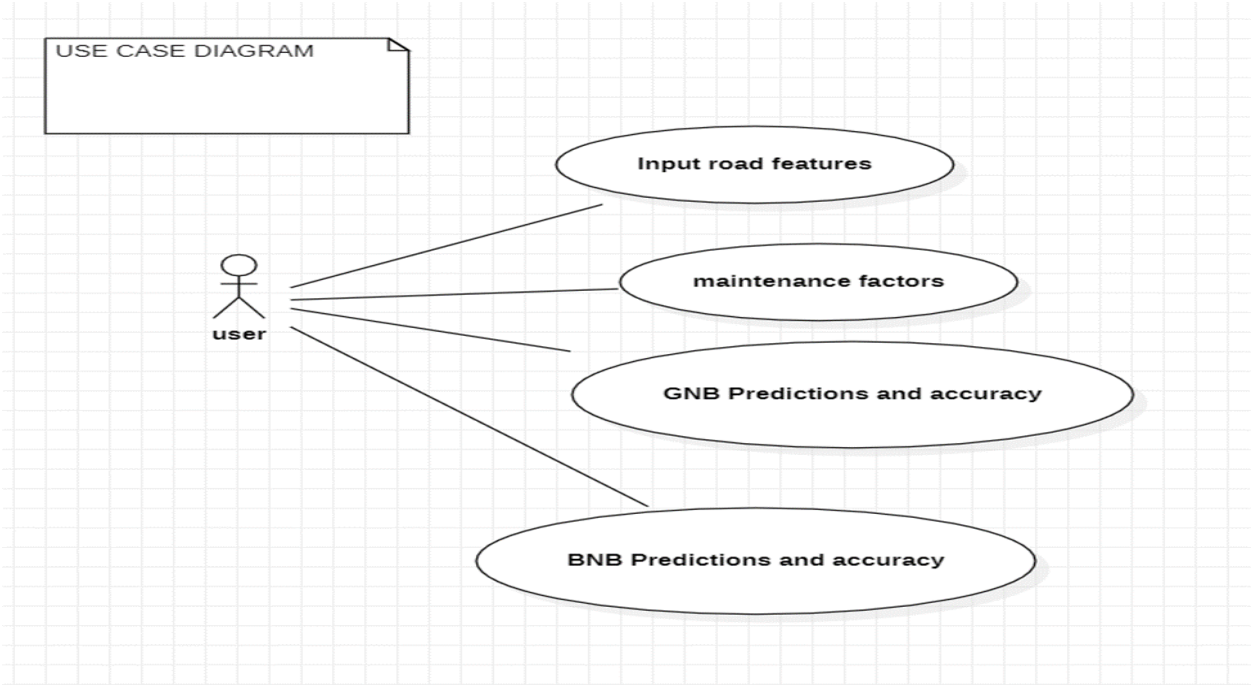

Fig4.3: Use Case Diagram

Description of Use case diagram

Use case diagrams are usually referred to as behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. As we can see the user is interacting with system by a UI through which the customer can perform above mentioned operations like providing Road Features, maintenance Factors, and then calculating the BNB & GNB.

4.2.3 Sequence Diagram

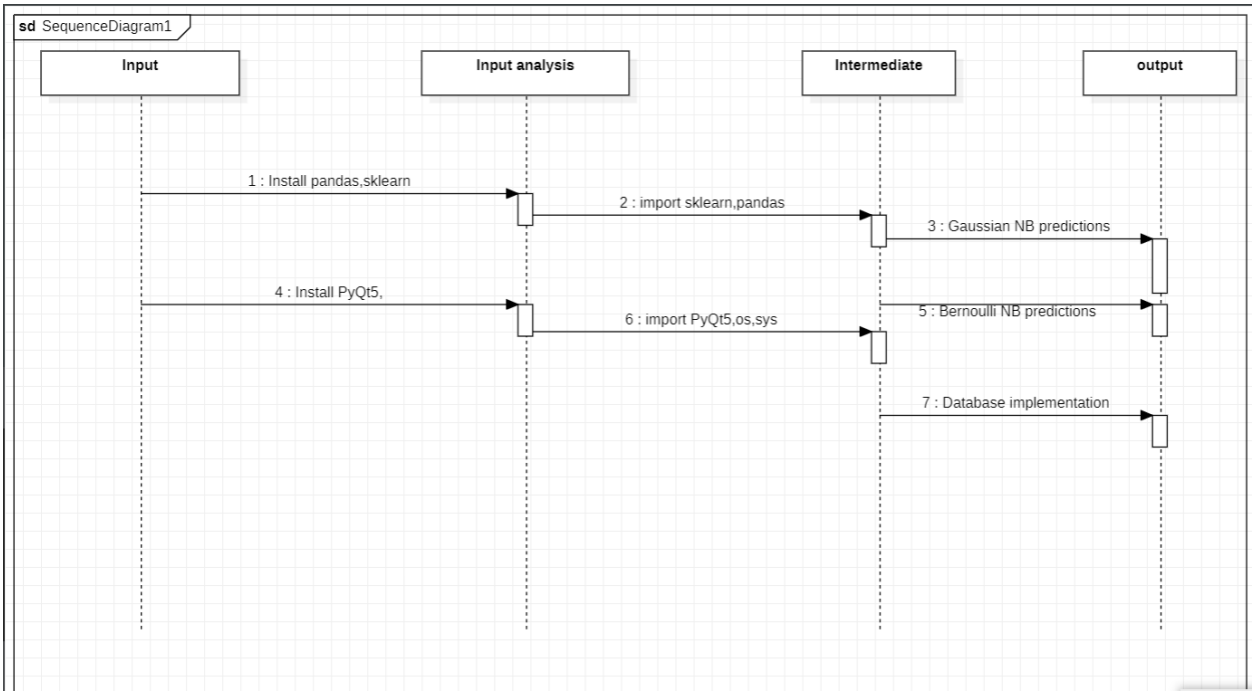

Fig 4.4 Sequence Diagram

Description of Sequence Diagram

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence, from above mentioned sequence diagram we have to go in sequence: Enter the needed details as shown in the above figure, Provide the Training Data, Testing Data and then calculate the accuracies of BNB, GNB along with the generation of the data comparison plot.

CHAPTER 5

IMPLEMENTATION

5.1 Overview of Technologies

5.1.1 Python

Python is a widely used high level programming language for general purpose programming, created by Guido Van Rossum and first released in 1991. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale. Python features a dynamic type of system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. Python, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of its variant implementations. Python is managed by the non-profit Python Software Foundation. Python was conceived in the late 1980s, and its implementation began in December 1989 by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL) capable of exception handling and interfacing with the operating system Amoeba. Van Rossum is Python's principal author, and his continuing central role in deciding the direction of Python is reflected in the title given to him by the Python community, Benevolent Dictator for Life (BDFL). About the origin of Python, Van Rossum wrote in 1996.

Python 2.0 was released on 16 October 2000 and had many major new features, including a cycle-detecting garbage collector and support for Unicode. With this release the development process was changed and became more transparent and community backed. Python 3.0 (initially described as Python 3000 or py3k), is a major, backward-incompatible release that was released after a long period of testing on 3 December 2008. Many of its major features have been back ported to the backwards-compatible Python 2.6.x and 2.7.x version series.

The End-of-Life date (EOL, sunset date) for Python 2.7 was initially set at 2015, then postponed to 2020 out of concern that a large body of existing code cannot easily be forward ported to Python 3. In January 2017, Google announced work on a Python 2.7 to Go trans compiler, which The Register speculated was in response to Python 2.7's planned end-of life but Google cited performance under concurrent workloads as their only motivation.

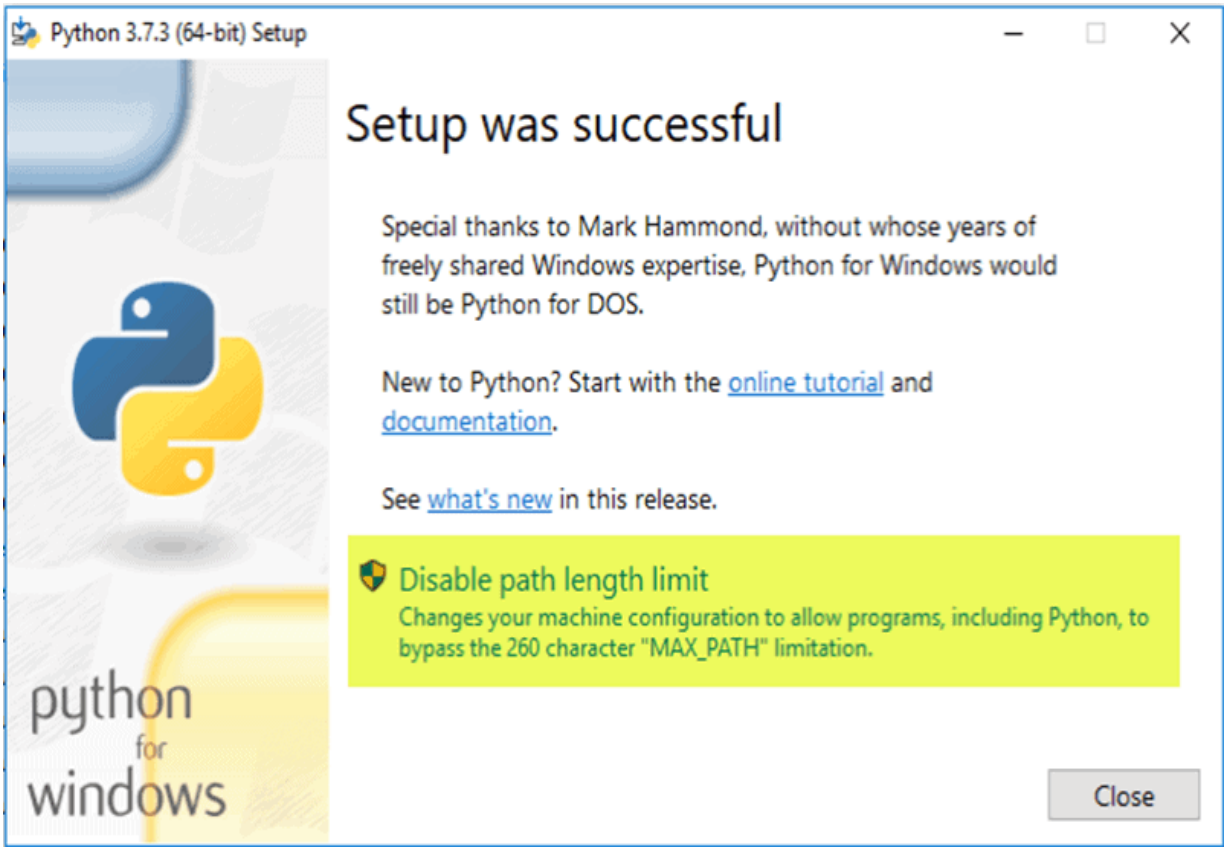

Fig 5.1: Python installation successful

5.1.2 VS Code

This is a user-friendly text editor which contains various number of functions which are useful for the user. We can run most of the programming languages in this open-source text editor. Python experience is very fast and clean in VS code and is one of the most used text editors in the world.

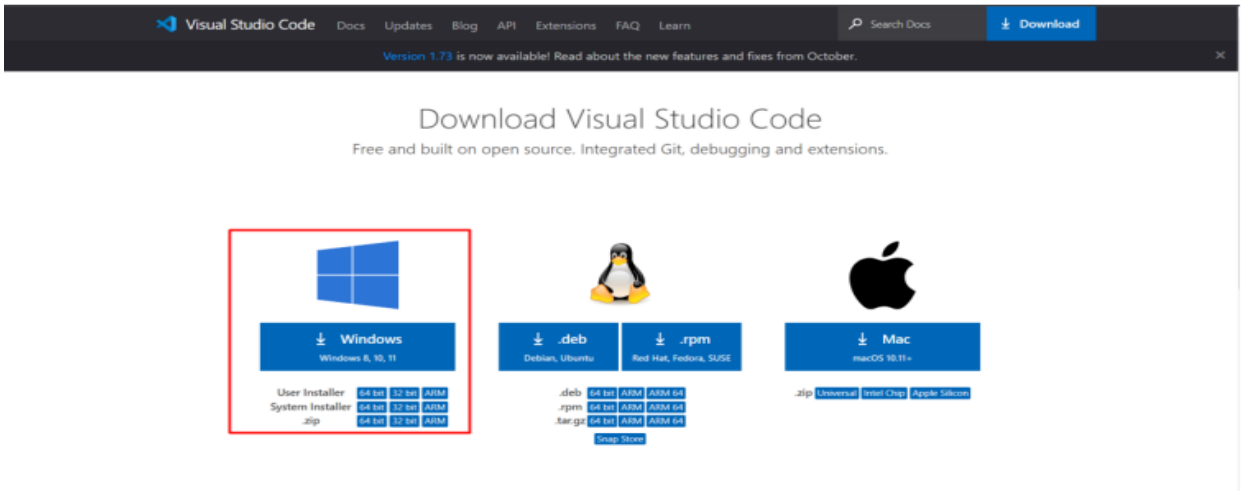

Fig 5.2: VS Code Installer

5.1.3 Qt Designer

Qt Designer is the tool used to create the Graphical user interfaces in this project. The tool can be found in the Library/Bin folder of Anaconda tool as shown in the following screen shot. As this designer tool is used again and again, in the project, it's better to create a short cut on the desktop, by using a right click on designer.exe, as shown in the following figure.

1. Once the designer tool is opened, it results in a screen as shown below.

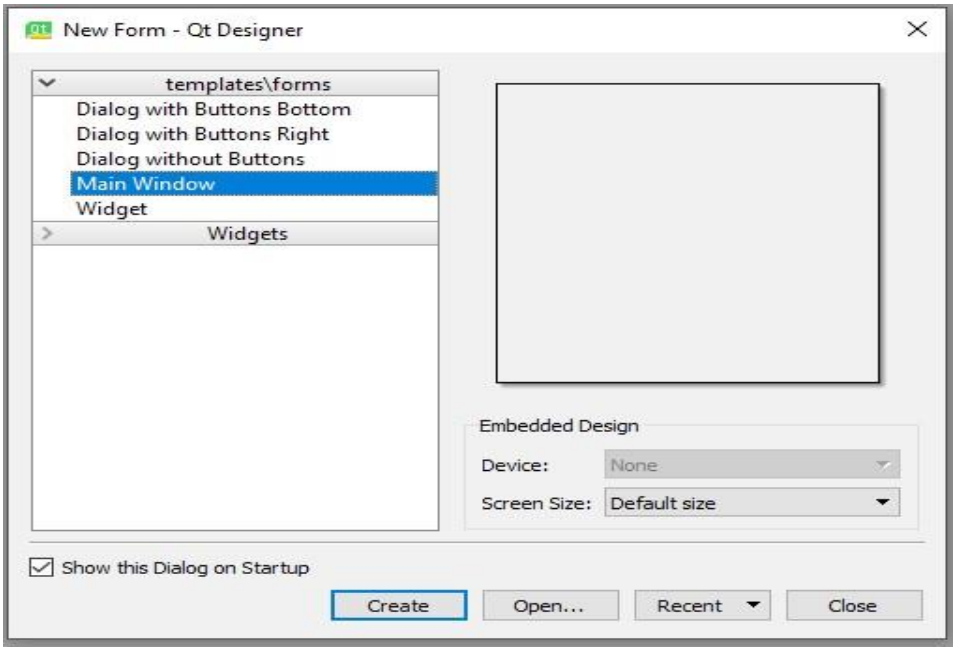

Fig 5.3: Designer Screen

2.The main window can be created by clicking on the Create button, after highlighting the main window (shown in blue color, in the above figure).

The main window can be given a title, by using the properties window, that is in the right-hand side middle portion, as shown below.

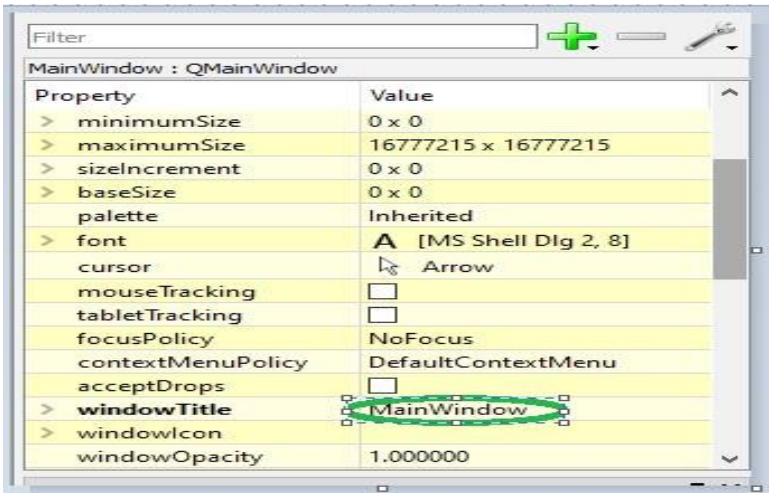

Fig 5.4: Creating a main window

3.Clicking on the Add font option, will result in the following screen.

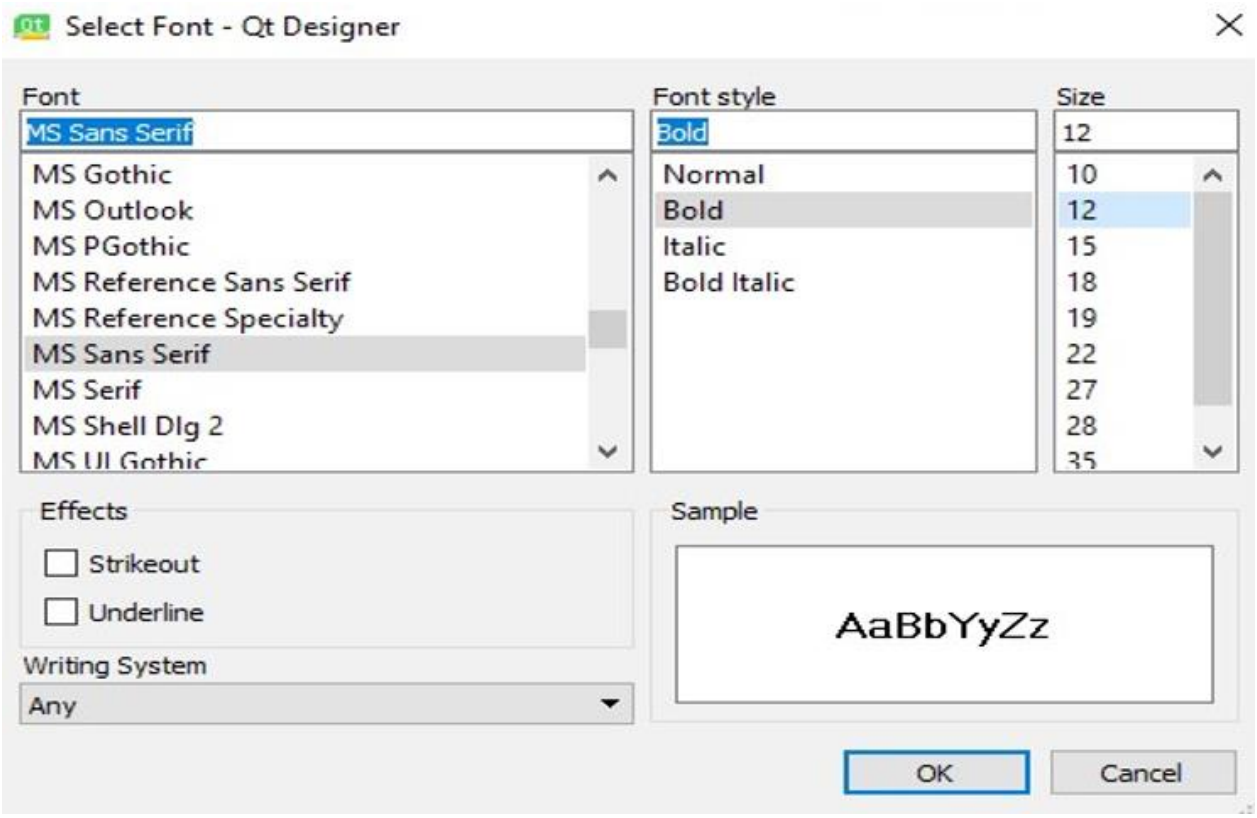

Fig 5.5: Adding Font

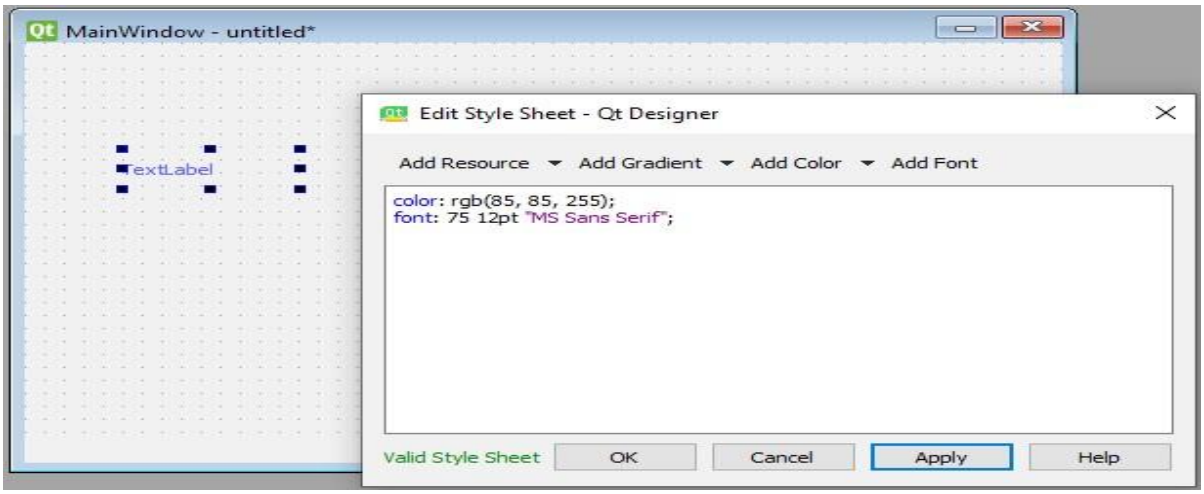

Fig 5.6 Adding color to font

5.1.4 Star UML

Star UML is an open-source software where we can design various number of UML diagrams for any product development. It helps us analyse the product or software diagrammatically which helps the user understand the product easily and quickly.

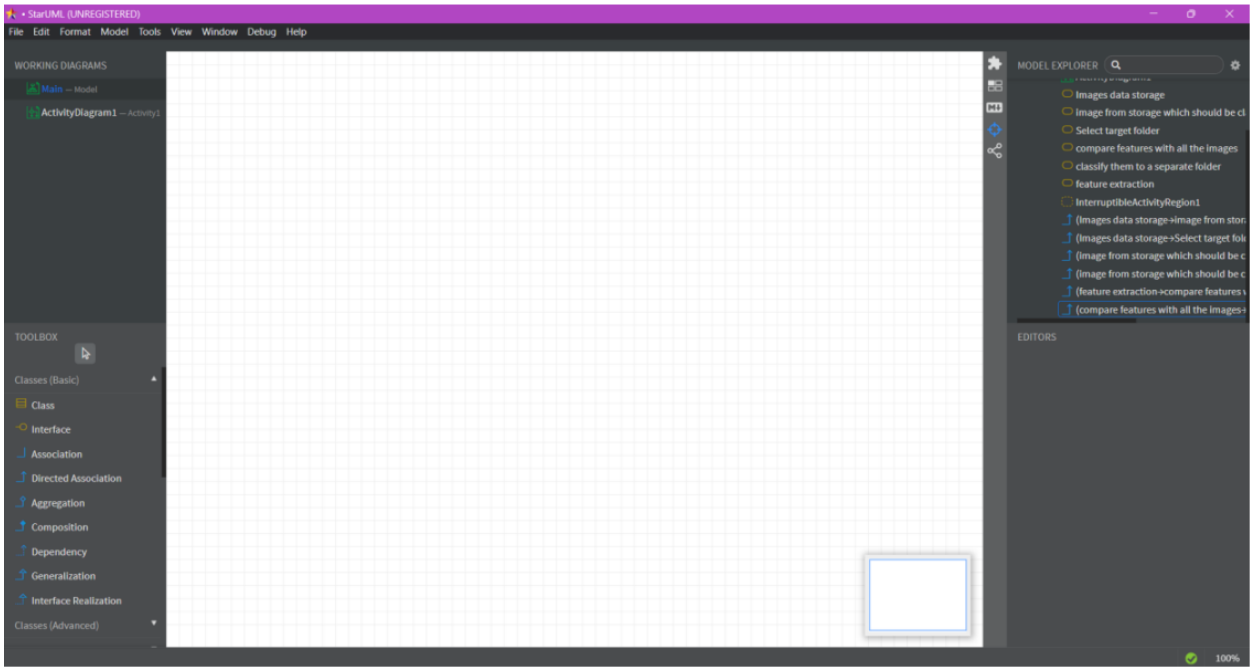

Fig 5.7: Star UML

5.2 Installations

5.2.1 pyqt5

Execute the command from command prompt: pip3 install pyqt5. You should get the pyqt5 successfully, installed as shown in the following two figures.

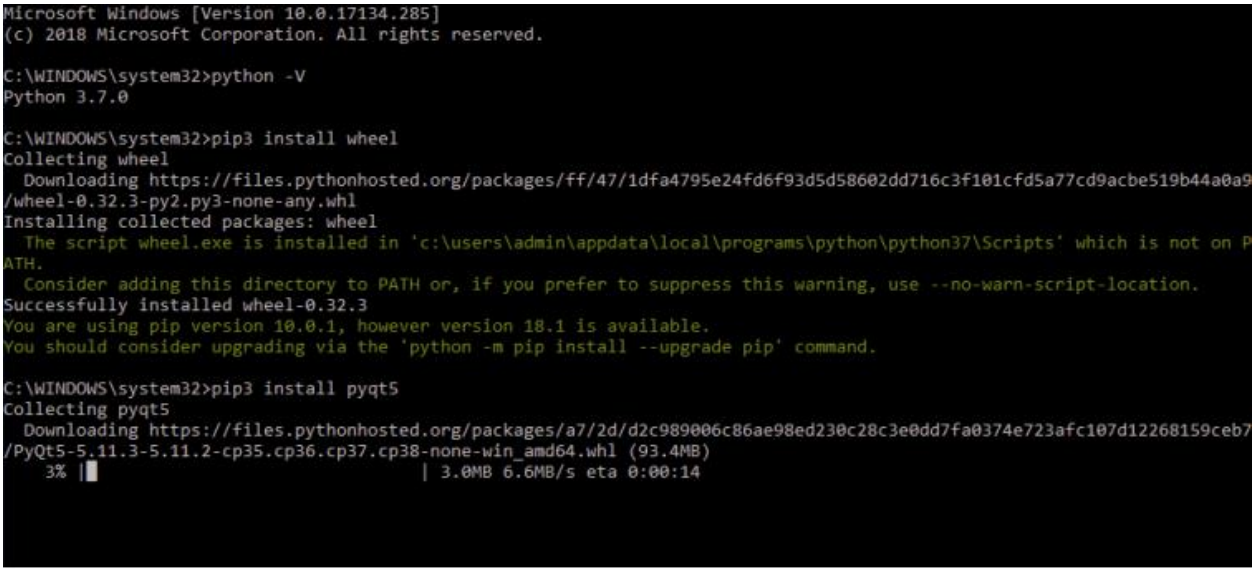

Fig 5.8: pyqt5 installation

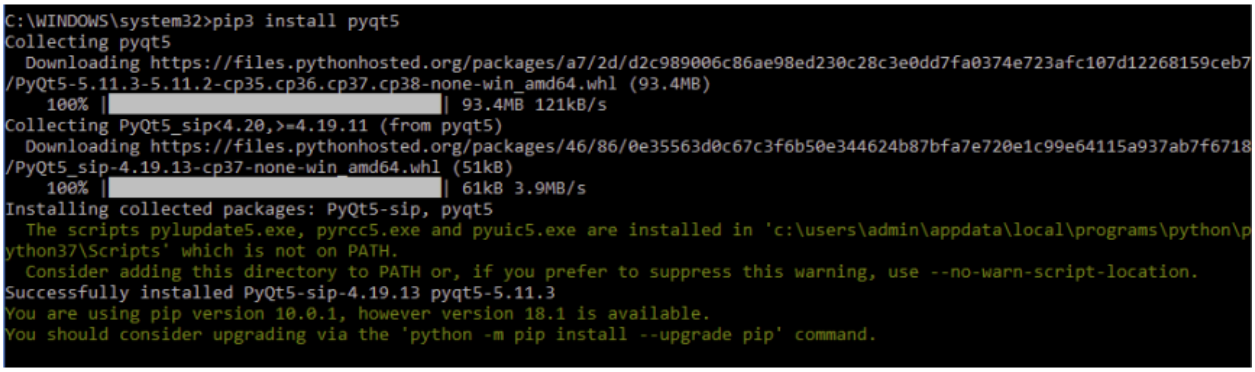

Fig 5.9: pyqt5 installation

5.2.2 pillow

Execute the command from command prompt: pip3 install pillow. You should get the pillow successfully, installed as shown in the following figure.

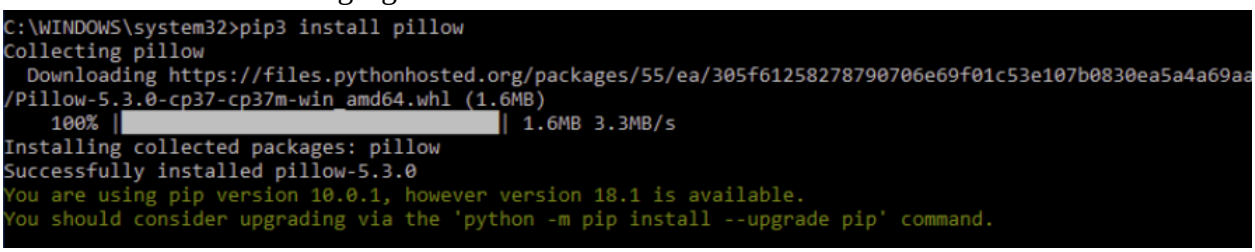

Fig 5.10: Pillow installation

5.2.3 Open-cv

Execute the command from command prompt: pip3 install OpenCV-python. You should get the OpenCV-python successfully, installed as shown in the following figure.

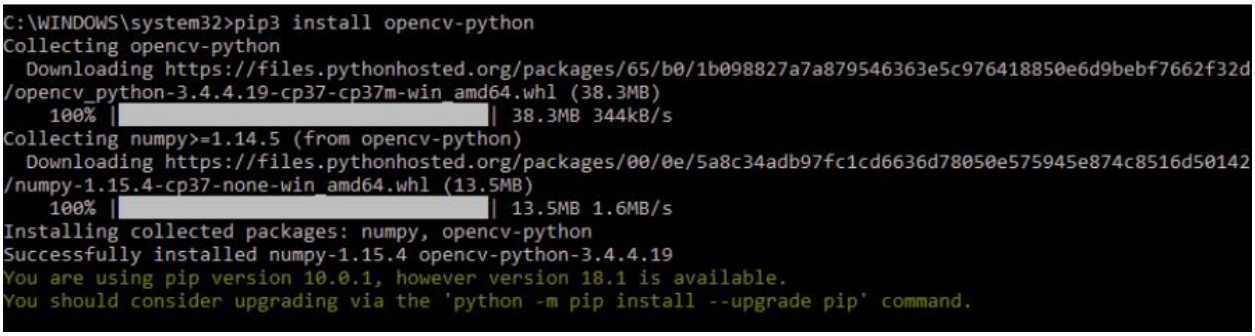

Fig 5.11: OpenCV installation

5.2.4 SQLite

Install SQLite by downloading it from the following site: https://www.sqlite.org/download.html. Open this site, go to the bottom, and click on "sqlite-tools-win32-x86-32--------.zip", as shown in the following figure.

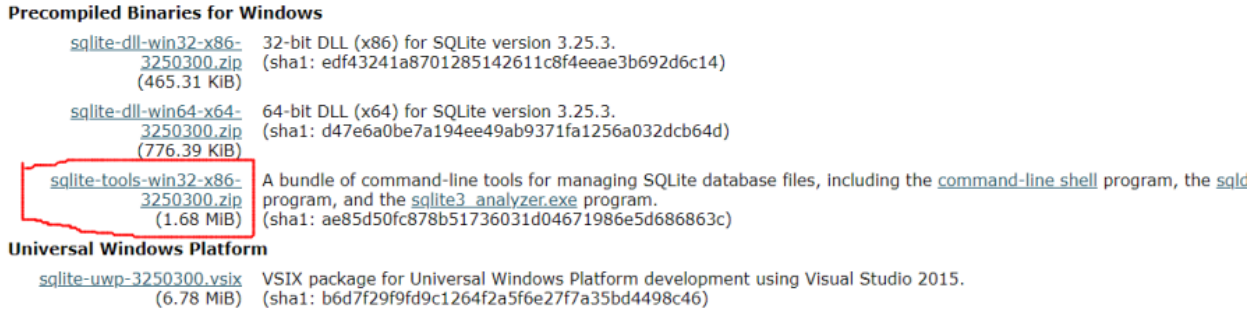

Fig 5.12: SQLite installation

5.3 Libraries Imported

5.3.1 OS

In Python, the OS module provides functions for interacting with the operating system. Python's standard utility modules include OS. This module allows you to use operating system-specific functionality on the go. Many functions for interacting with the file system are included in the os and os. path modules.

```
os.system("python roadftr1.py")
```

OS functions:

1) Executing a shell command
os.system()

2) Get the users environment
os.environ()
3) Returns the current working directory.
os.getcwd()

4) Return the real group id of the current process.
os.getgid()

5) Return the current process's user id.
os.getuid()

6) Returns the real process ID of the current process.
os.getpid()

7)Set the current numeric umask and return the previous umask.
os.umask(mask)

8)Return information identifying the current operating system.
os.uname()

9) Change the root directory of the current process to path.
os.chroot(path)

10)Return a list of the entries in the directory given by path.
os.listdir(path)

11) Create a directory named path with numeric mode mode.
os.mkdir(path)

12) Recursive directory creation function.
os.makedirs(path)

13) Remove (delete) the file path.
os.remove(path)

5.3.2 Pyqt5

PyQt5 is a cross-platform GUI toolkit that includes Python bindings for the Qt v5 framework. Because of the tools and simplicity provided by this library, it is possible to create an interactive desktop application with great ease. A GUI application is made up of two parts: the front end and the back end. PyQt5 includes a tool called

'QtDesigner' that allows you to design the front-end using a drag-and-drop method, allowing you to spend more time on back-end stuff.

```
self.label = QtWidgets.QLabel(self.centralwidget)
self.label.setGeometry(QtCore.QRect(20, 10, 261, 41))
self.label.setMaximumSize(QtCore.QSize(291, 16777215))
```

5.3.3 Pandas

Pandas is an open-source library designed primarily for working with relational or labelled data in an easy and intuitive manner. It offers several data structures and operations for manipulating numerical data and time series. This library is based on the NumPy library. Pandas is quick, with high performance and productivity for users.

```
df = pd.read_csv('rmdataset.csv')
test = pd.DataFrame(testSet)
```

5.3.4 Sys

The Python sys module offers a number of methods and variables that can be used to modify various elements of the Python execution environment. Due to its accessibility to variables as well as functions that have a close relationship with the interpreter, it enables working on the interpreter.

Lots of scripts need access to the arguments passed to the script when the script was started.

argvargv (or to be precise sys.argv) is a list, which contains the command-line arguments passed to the script. The first item of this list contains the name of the script itself. The arguments follow the script name.

Every serious user of a UNIX or Linux operating system knows standard streams, i.e. input, standard output and standard error. They are known as pipes. They are commonly abbreviated as stdin, stdout, stderr. The standard input (stdin) is normally connected to the keyboard, while the standard error and standard output go to the terminal (or window) in which you are working. These data streams can be accessed from Python via the objects of the sys module with the same names, i.e. sys.stdin, sys.stdout and sys.stderr. The standard output (stdout) can be redirected e.g. into a file, so that we can process this file later with another program. The same is possible with the standard error stream, we can redirect it into a file as well. We can redirect both stderr and stdout into the same file or into separate files.

```
sys.exit(app.exec_())
```

5.3.5 Scikit-learn.

An open-source Python toolkit called scikit-learn uses a unified interface to perform a variety of machine learning, pre-processing, cross-validity, and visualisation methods.

Scikit-learn' s salient characteristics are:

Tools for data extraction and data analysis that are easy to use and effective. Support vector machines, random forests, gradient boosting, k-means, and other classification, regression, and grouping methods are included. Everyone-friendly and useful in different situations. Built on top of Matplotlib, SciPy, and NumPy. BSD licence; open source; useful in commerce.

```
from sklearn.naive_bayes import GaussianNB, BernoulliNB
```

5.3.6 sqlite3

Python integrates with the SQLite database using the SQLite3 library. It offers a clear and user-friendly interface for working with SQLite datasets as part of the standardised Python DBI API 2.0.

```
con = sqlite3.connect('rsurface1')
```

we need to create a new database and open a database connection to allow sqlite3 to work with it. Call sqlite3.connect() to create a connection to the database tutorial.db in the current working directory, implicitly creating it if it does not exist:

```
import sqlite3
con = sqlite3.connect("tutorial.db")
```

5.4 Algorithms and concept

5.4.1 Naïve bayes algorithm

A probabilistic machine learning approach called Naive Bayes can be applied to a variety of categorization applications. Naive Bayes is frequently used for document classification, spam filtering, prediction, and other tasks. This method takes its name from Thomas Bayes' discoveries, on which it is based.

The method combines features into its model that are independent of one another, hence the term "Nave." Any changes to the value of one feature of the algorithm have no direct effect on the value of any other feature. The

Naive Bayes algorithm's key benefit is that it is an easy-to-use yet effective method. It is built on a probabilistic model, making predictions fast and in real-time with an easily codable algorithm.

5.4.2 The bayes rule

By using the training dataset to determine P (X | Y), we can then use the Bayes algorithm to find P (Y | X). All you have to do to accomplish this is swap out A and B in the formulas above for X and Y. X would be the known variable for observations, while Y would be the unknown variable. You must determine the likelihood of Y given that X has previously happened for each row in the dataset.

What transpires, though, if Y contains more than two categories? To determine the successful Y class, we must compute the probability of each Y class.

We proceed from P (X | Y) to P (Y | X) using the Bayes rule.

From training data, we know that P (X | Y) = P (X Y) / P(Y)

(Evidence | Result)

Unknown - to be anticipated from test results: P (X Y) / P (Y | X)(X)

(Result | Evidence)

Bayes Rule: P(Y | X) = P(X | Y) = P(Y | X) * P(Y) / P (X)

5.4.3 Bernoulli Naïve Bayes Classifier

Bernoulli Naive Bayes is a machine learning variant of the Naive Bayes algorithm. It is extremely useful when the dataset has a binary distribution with the output label present or absent. The main advantage of this algorithm is that it only accepts binary values for features such as: Is it true or false? Yes, no, 0 or 1.

Here are some additional benefits to using this algorithm for binary classification: When compared to other classification algorithms, it is extremely fast. Machine learning algorithms do not always perform well when the dataset is small, but this is not the case with this algorithm, which produces more accurate results than other classification algorithms.

Formula:

$$p(x) = P[X = x] = \begin{cases} q = 1 - p & x = 0 \\ p & x = 1 \end{cases}$$

$$X = \begin{cases} 1 & \text{Bernoulli trial} = \mathbf{S} \\ 0 & \text{Bernoulli trial} = \mathbf{F} \end{cases}$$

5.4.4 Gaussian Naïve Bayes Classifier

A classification method used in machine learning (ML) called Gaussian Naive Bayes (GNB) is based on the probabilistic method and Gaussian distribution. Gaussian Naive Bayes makes the assumptions that each parameter—also known as a feature or a predictor—has the ability to predict the output variable on its own. The final prediction, which provides a probability that the dependant variable will be classified into each group, is the culmination of all previous predictions. The group with the highest likelihood receives the final designation. If we believe that Xs have a Gaussian or normal distribution, we must replace it with the normal distribution's probability density and call it the Gaussian Naive Bayes model. You must the X mean and variance in order to calculate this formula. Sigma and mu are the continuous variable X's variance and mean as calculated for the specified class c of Y in the formulas above. Through a frequency, the aforementioned formula determined the probability for input values for each class. For the full distribution, we can get the mean and standard deviation of x's for each class.

Formula:

$$P(X|Y = c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{\frac{-(x-\mu_c)^2}{2\sigma_c^2}}$$

As a result, we must also record the mean and standard deviation for each input variable for each class in addition to the probabilities for each class.

mean(x) = 1/n * total(x)

Sqrt(1/n * sum(xi-mean(x)2)

is the formula for standard deviation (x).

In this case, where n is the number of instances, sum() is the sum function, sqrt() is the square root function, and xi is a particular x value, the square root of the average of differences of each x and the mean of x is determined.

CHAPTER 6

TESTING & CODING

6.1 Testing

6.1.1 Accuracy Testing

Bernoulli naïve bayes classification

Given the circumstances, Bernoulli naïve bayes classifier gives an approximate accuracy of 61%.

```
 def bnb(self):
      df = pd.read_csv('rmdataset.csv')
      df["RoadSurface"] = df["RoadSurface"].map({'Poor':0 ,'Avg':1 ,'Good':2})
      data =
df[["IDMachines","PeopleAtwork","StreetLights","Accidents","DamagedMovers","StRoadLength","RoadCurv
ature","HPBends","RoadType","RoadWidth","AvgSpeed","RoadSurface"]].to_numpy()
      inputs = data[:,:-1]
      outputs = data[:, -1]
      training_inputs = inputs[:2500]
      training_outputs = outputs[:2500]
      testing_inputs = inputs[2500:]
      testing_outputs = outputs[2500:]
      classifier = BernoulliNB()
      classifier.fit(training_inputs, training_outputs)
      predictions = classifier.predict(testing_inputs)
      accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
      print("The accuracy of BNB Classifier on testing data is: " + str(accuracy))
```

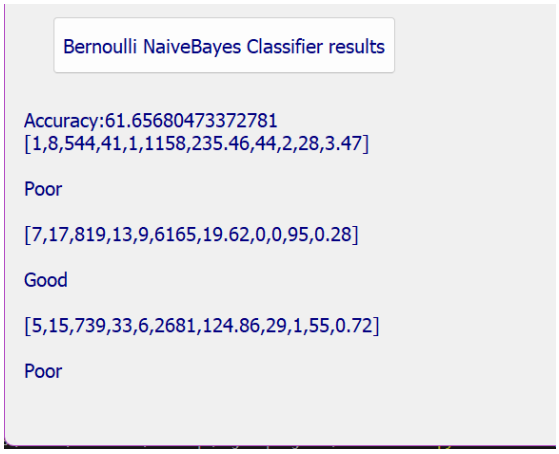

Fig 6.1: Bernoulli naïve bayes accuracy

Gaussian naïve bayes classification

Gaussian naïve bayes classifier, being a perfect classifier for continuous data, gives an accuracy rate approximate to 90%.

```
 def gnb(self):
      df = pd.read_csv('rmdataset.csv')
      df["RoadSurface"] = df["RoadSurface"].map({'Poor':0 ,'Avg':1 ,'Good':2})
      data =
df[["IDMachines","PeopleAtwork","StreetLights","Accidents","DamagedMovers","StRoadLength","RoadCurv
ature","HPBends","RoadType","RoadWidth","AvgSpeed","RoadSurface"]].to_numpy()
      inputs = data[:,:-1]
      outputs = data[:, -1]
      training_inputs = inputs[:2500]
      training_outputs = outputs[:2500]
      testing_inputs = inputs[2500:]
      testing_outputs = outputs[2500:]
      classifier = GaussianNB()
      classifier.fit(training_inputs, training_outputs)
      predictions = classifier.predict(testing_inputs)
      accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
      print("The accuracy of GNB Classifier on testing data is: " + str(accuracy))
```

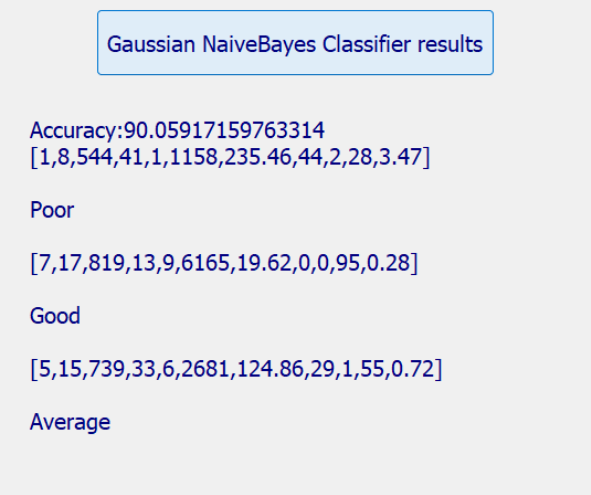

Fig 6.2: Gaussian naïve bayes accuracy

6.2 Sample Code

6.2.1 Gui and classifiers

This file contains the main code of the Gui and the classifier results.

```
from PyQt5 import QtCore, QtGui, QtWidgets
import sys
import os
from sklearn.naive_bayes import GaussianNB, BernoulliNB
import pandas as pd
from sklearn.metrics import accuracy_score

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(1134, 872)
        MainWindow.setStyleSheet("color: rgb(0, 0, 127);\n"
"font: 75 12pt \"MS Shell Dlg 2\";")
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(50, 230, 291, 71))
        self.pushButton.setObjectName("pushButton")
        self.pushButton_3 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_3.setGeometry(QtCore.QRect(670, 420, 361, 61))
        self.pushButton_3.setObjectName("pushButton_3")
        self.pushButton_4 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_4.setGeometry(QtCore.QRect(50, 420, 361, 61))
```

```
  def gnb(self):
        df = pd.read_csv('rmdataset.csv')
        df["RoadSurface"] = df["RoadSurface"].map({'Poor':0 ,'Avg':1 ,'Good':2})
        data =
df[["IDMachines","PeopleAtwork","StreetLights","Accidents","DamagedMovers","StRoadLength","RoadCurv
ature","HPBends","RoadType","RoadWidth","AvgSpeed","RoadSurface"]].to_numpy()
        inputs = data[:,:-1]
        outputs = data[:, -1]
        training_inputs = inputs[:2500]
        training_outputs = outputs[:2500]
        testing_inputs = inputs[2500:]
        testing_outputs = outputs[2500:]
        classifier = GaussianNB()
        classifier.fit(training_inputs, training_outputs)
        predictions = classifier.predict(testing_inputs)
        accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
```

```
            print("The accuracy of GNB Classifier on testing data is: " + str(accuracy))
        testSet = [[1,8,544,41,1,1158,235.46,44,2,28,3.47]]
        test = pd.DataFrame(testSet)
        predictions = classifier.predict(test)
        self.ui.label_3.setText(str(accuracy))
        text = str(accuracy) + "\n" + "[1,8,544,41,1,1158,235.46,44,2,28,3.47]"
        self.ui.label_3.setText(text)
```

6.2.2 accuracy display and external Gui

```
from PyQt5 import QtCore, QtGui, QtWidgets
from sklearn.naive_bayes import BernoulliNB,GaussianNB
from sklearn import *
import pandas as pd
from sklearn.metrics import accuracy_score
import os

class Ui_Dialog(object):
    def setupUi(self, Dialog):
        Dialog.setObjectName("Dialog")
        Dialog.resize(1064, 840)
        self.pushButton = QtWidgets.QPushButton(Dialog)
        self.pushButton.setGeometry(QtCore.QRect(40, 580, 131, 41))
        font = QtGui.QFont()
        font.setFamily("Ubuntu")
        font.setPointSize(12)
        self.pushButton.setFont(font)
        self.pushButton.setObjectName("pushButton")
```

```
        data =
df[["IDMachines","PeopleAtwork","StreetLights","Accidents","DamagedMovers","StRoadLength","RoadCurv
ature","HPBends","RoadType","RoadWidth","AvgSpeed","RoadSurface"]].to_numpy()
        inputs = data[:,:-1]
        outputs = data[:, -1]
        training_inputs = inputs[:2500]
        training_outputs = outputs[:2500]
        testing_inputs = inputs[2500:]
        testing_outputs = outputs[2500:]
        classifier = BernoulliNB()
        classifier.fit(training_inputs, training_outputs)
        predictions = classifier.predict(testing_inputs)
        accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
        print("The accuracy of BNB Classifier on testing data is: " + str(accuracy))
        testSet =
[self.textEdit.toPlainText(),self.textEdit_2.toPlainText(),self.textEdit_3.toPlainText(),self.textEdit_4.toPl
ainText(),self.textEdit_5.toPlainText(),self.textEdit_6.toPlainText(),self.textEdit_7.toPlainText(),
```

```
self.textEdit_8.toPlainText(),self.textEdit_9.toPlainText(),self.textEdit_10.toPlainText(),self.textEdit_11.toPlain
Text()]
        testSet_2=[]
        for i in testSet:
            if i.strip().isdigit() or i.replace('.', '', 1).isdigit():
                testSet_2.append(i)
            else:
                self.label_12.setText("Invalid Input")
                x=1
                break
        print(x)
        if(x==0):
            test = pd.DataFrame([testSet_2])
            predictions = classifier.predict(test)
            if predictions ==0:
                self.label_12.setText("The road surface is Poor")
```

6.2.3 Storing maintenance factors in the database using SQLite

```
import sys
from mfactors import *
from PyQt5 import QtWidgets, QtGui, QtCore

import sqlite3
con = sqlite3.connect('rsurface1')

#import MySQLdb as mdb
#con = mdb.connect('localhost', 'team1', 'test623', 'drless1');

class MyForm(QtWidgets.QMainWindow):
  def __init__(self,parent=None):
     QtWidgets.QWidget.__init__(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.insertvalues)
```

```
  def insertvalues(self):
    with con:
      cur = con.cursor()
      id = str(self.ui.lineEdit_9.text())
      s1 = str(self.ui.lineEdit_4.text())
      s2 = str(self.ui.lineEdit_5.text())
      s3 = str(self.ui.lineEdit_6.text())
      s4 = str(self.ui.lineEdit_2.text())
      s5 = str(self.ui.lineEdit_11.text())
```

```
    s6 = str(self.ui.lineEdit_7.text())
    location = str(self.ui.lineEdit_3.text())
    cur.execute('INSERT INTO mfactors VALUES(?,?,?,?,?,?,?,?)',(id,location,s1,s2,s3,s4,s5,s6))
    con.commit()
```

6.2.4 Storing road factors in the database using SQLite

```
import sys
from roadftr import *
from PyQt5 import QtWidgets, QtGui, QtCore

import sqlite3
con = sqlite3.connect('rsurface1')

#import MySQLdb as mdb
#con = mdb.connect('localhost', 'team1', 'test623', 'drless1');

class MyForm(QtWidgets.QMainWindow):
 def __init__(self,parent=None):
   QtWidgets.QWidget.__init__(self,parent)
   self.ui = Ui_MainWindow()
   self.ui.setupUi(self)
   self.ui.pushButton.clicked.connect(self.insertvalues)
   #self.ui.pushButton_2.clicked.connect(self.testdetails)

 def insertvalues(self):
  with con:
    cur = con.cursor()
    id = str(self.ui.lineEdit_9.text())
    location = str(self.ui.lineEdit_3.text())
    s1 = str(self.ui.lineEdit_4.text())
    s2 = str(self.ui.lineEdit_5.text())
    s3 = str(self.ui.lineEdit_6.text())
    s4 = str(self.ui.lineEdit_2.text())
    s5 = str(self.ui.lineEdit_11.text())
    cur.execute('INSERT INTO rftr VALUES(?,?,?,?,?,?,?)',(id,location,s1,s2,s3,s4,s5))
    con.commit()
```

CHAPTER 7

RESULT ANALYSIS

7.1 Output samples

The objective of this stage is to create the most basic model that can adequately and quickly formulate a target value. Through model adjustment, a data scientist can accomplish this objective. To attain an algorithm's optimal performance, model parameters are optimised in this manner.

1. Main user interface opens through main.exe

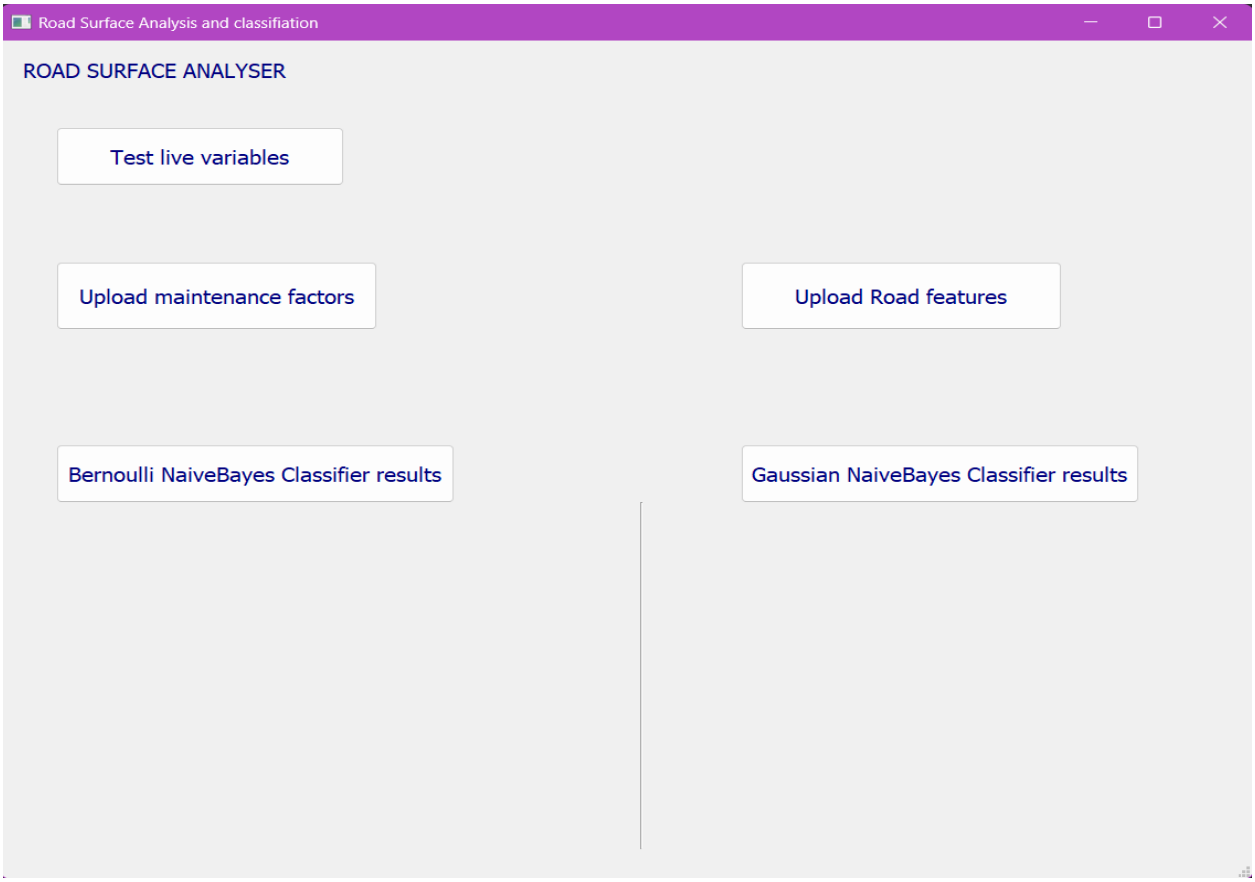

Fig 7.1: Main interface

2. Live variables button will open another interface through which live road analysis is done.

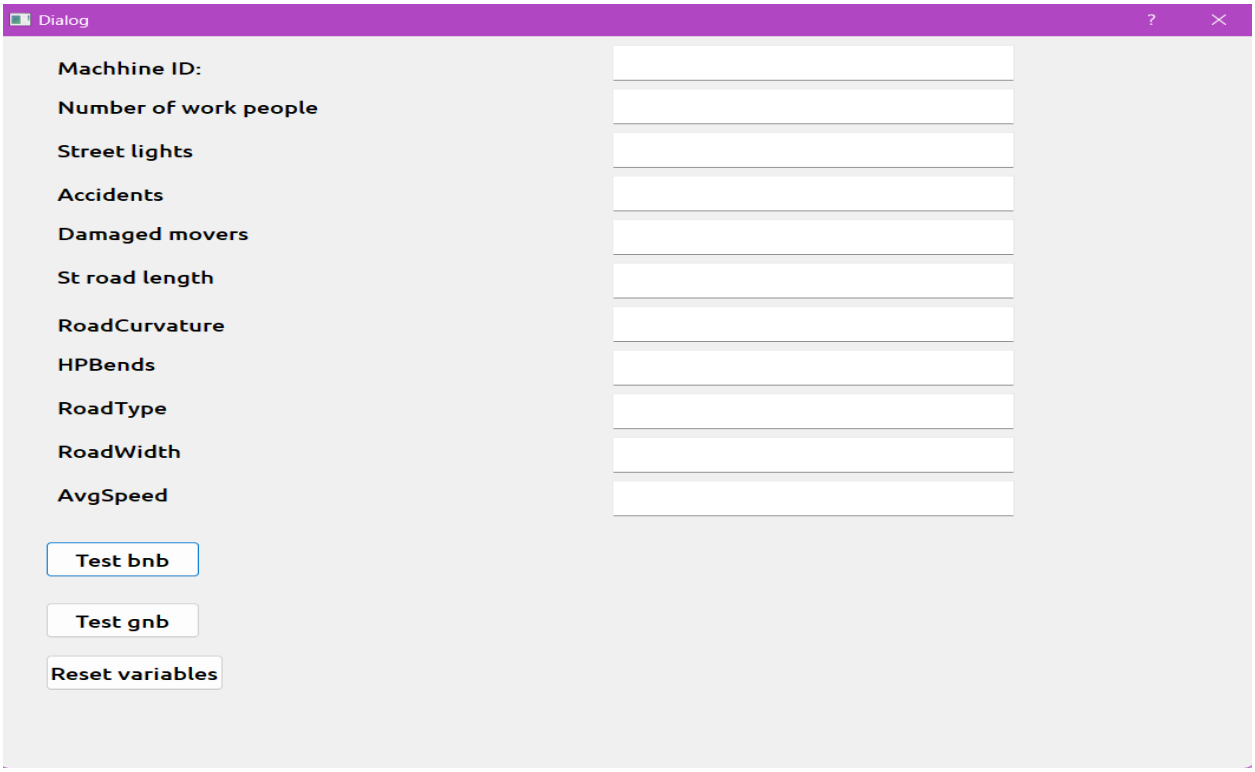

Fig 7.2: Live variables interface

3. Bernoulli naïve bayes prediction to given attribute values.

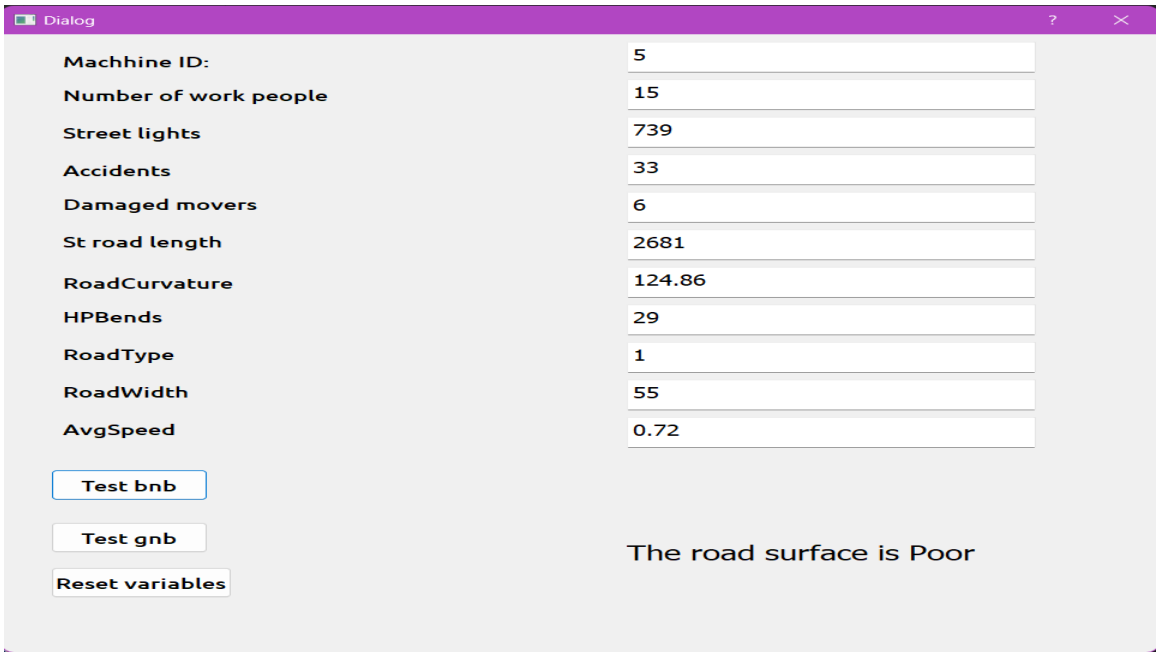

Fig 7.3: Bernoulli naïve bayes prediction

4. Gaussian naïve bayes prediction to given attribute values.

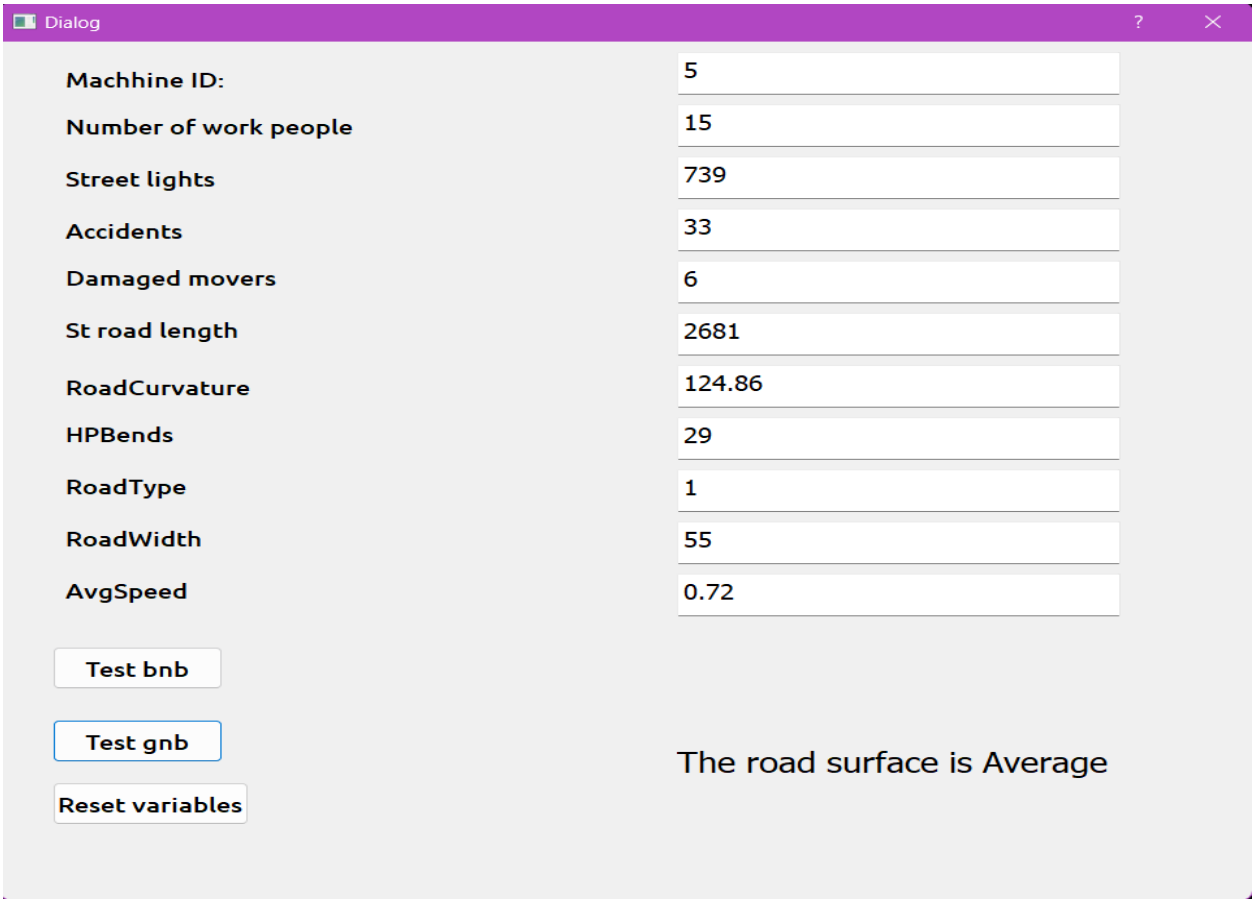

Fig 7.4: Gaussian naïve bayes prediction

6. Reset button will reset the variables to blank.

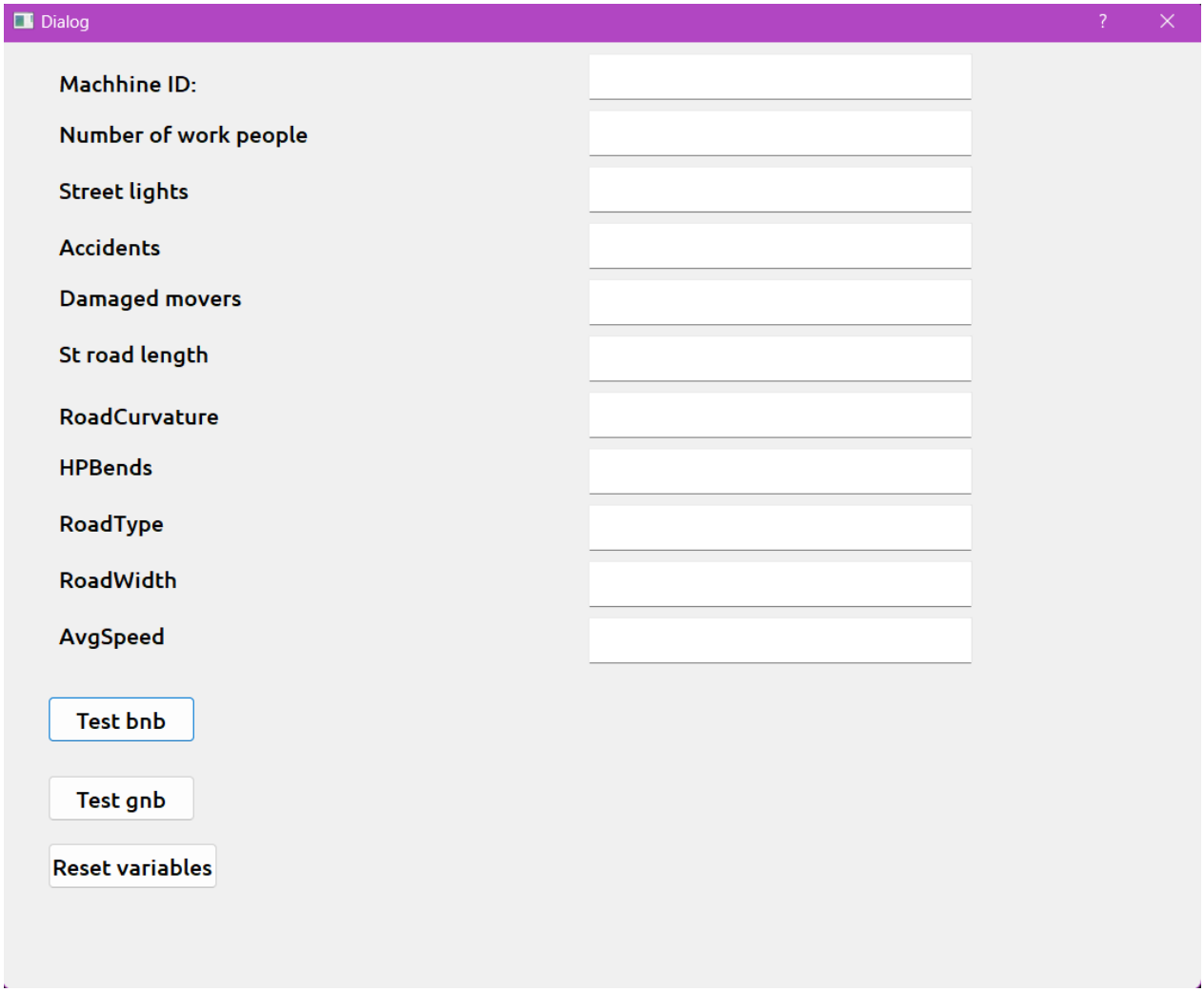

Fig 7.5: Reset

5. Upload maintenance factors button would store the road maintenance factors into a database.

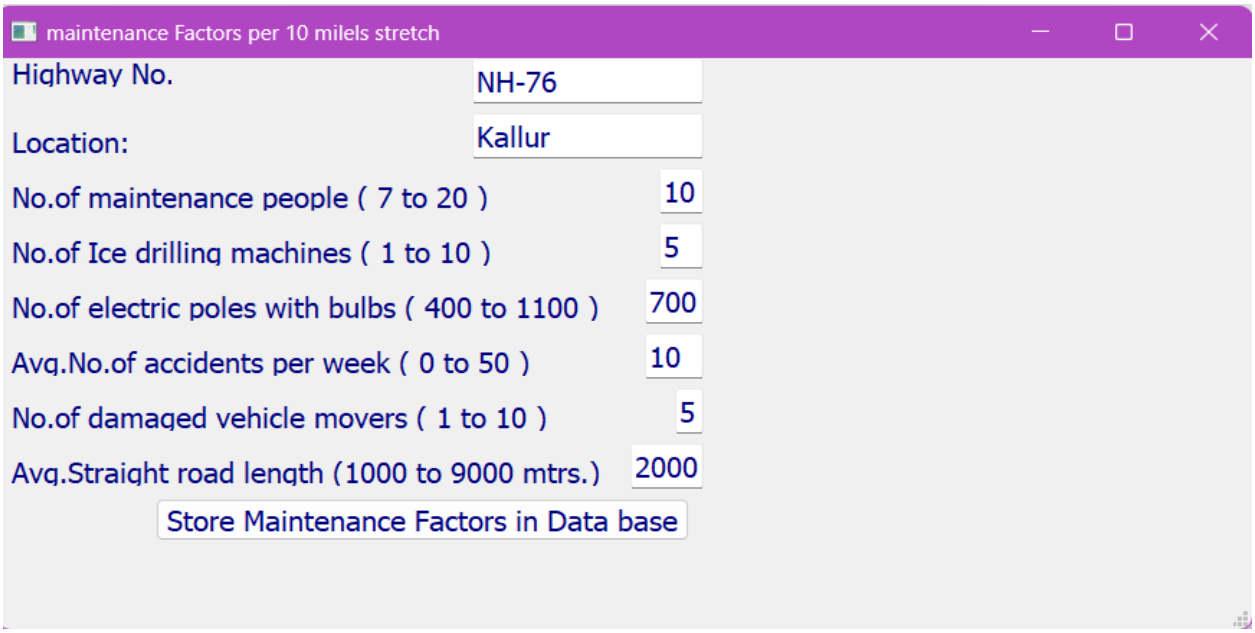

Fig 7.6: Uploading Maintenance factors into a database

6. mfactors table in database using SQLite.

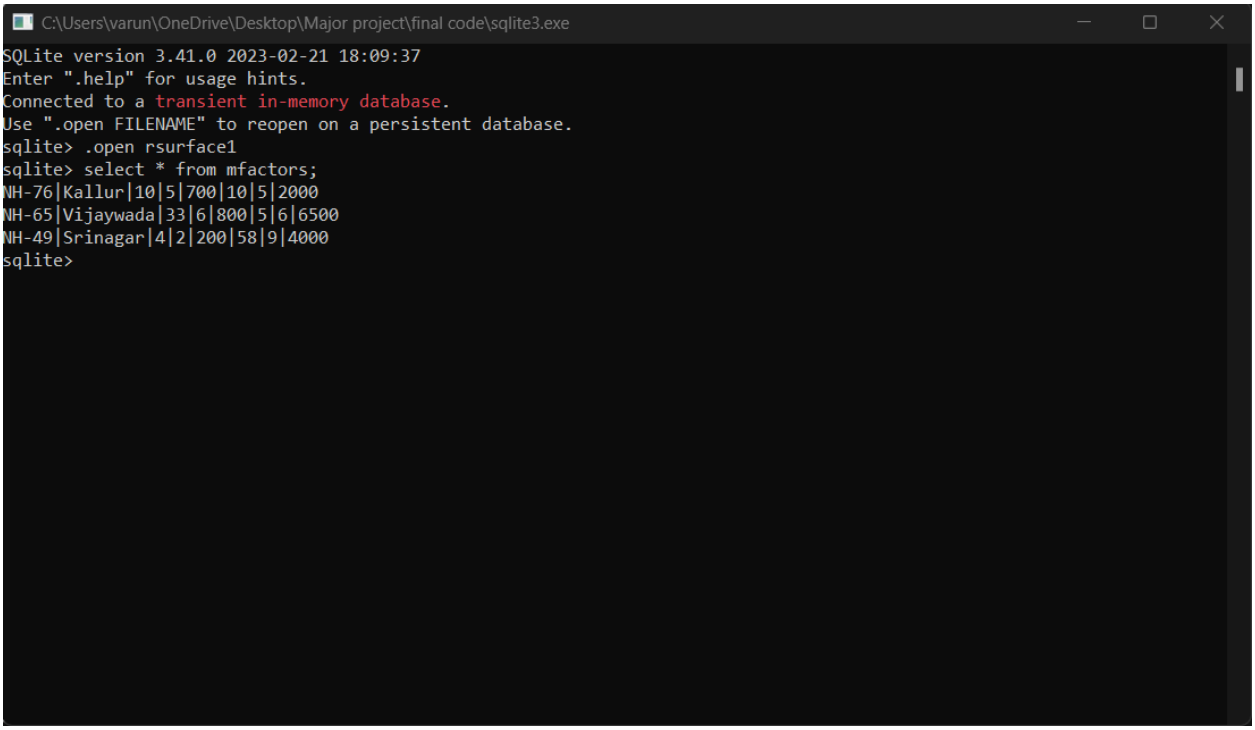

Fig 7.7: mfactors table

7) Upload Road features button would store the road features into a database.

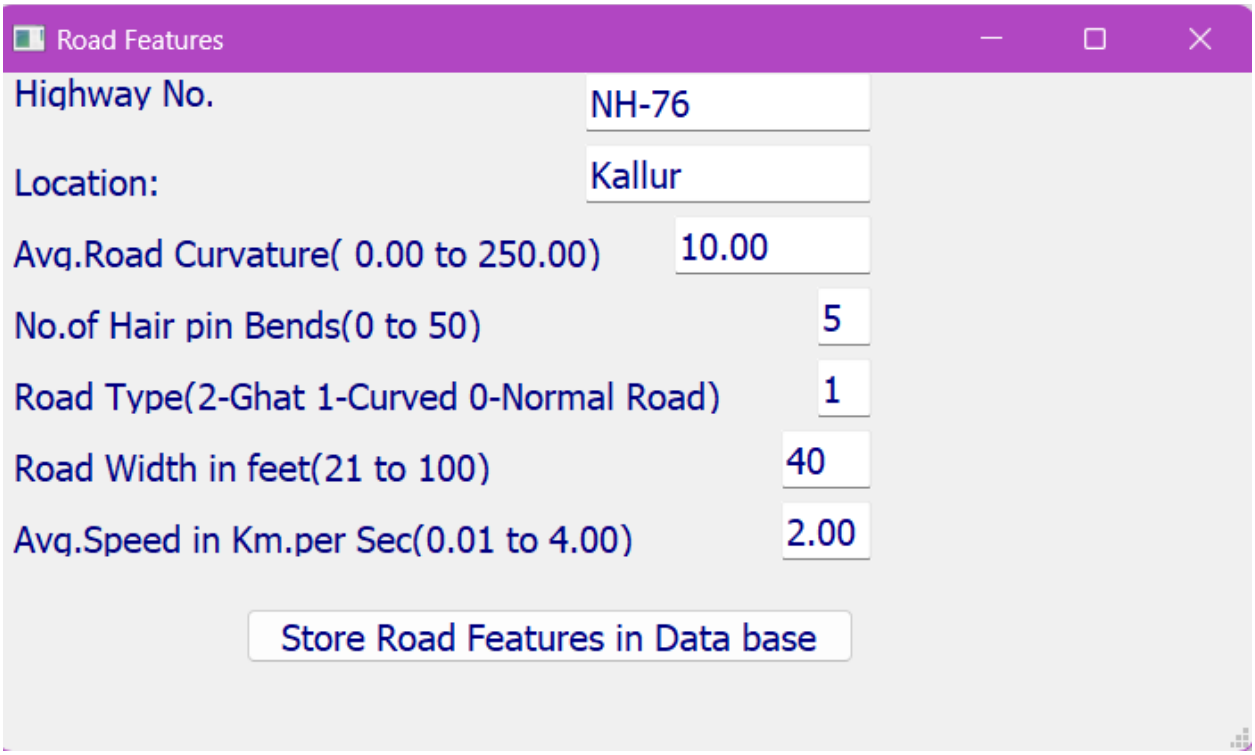

Fig 7.8: Uploading Road features into the database

8) rftr table in database using SQLite.

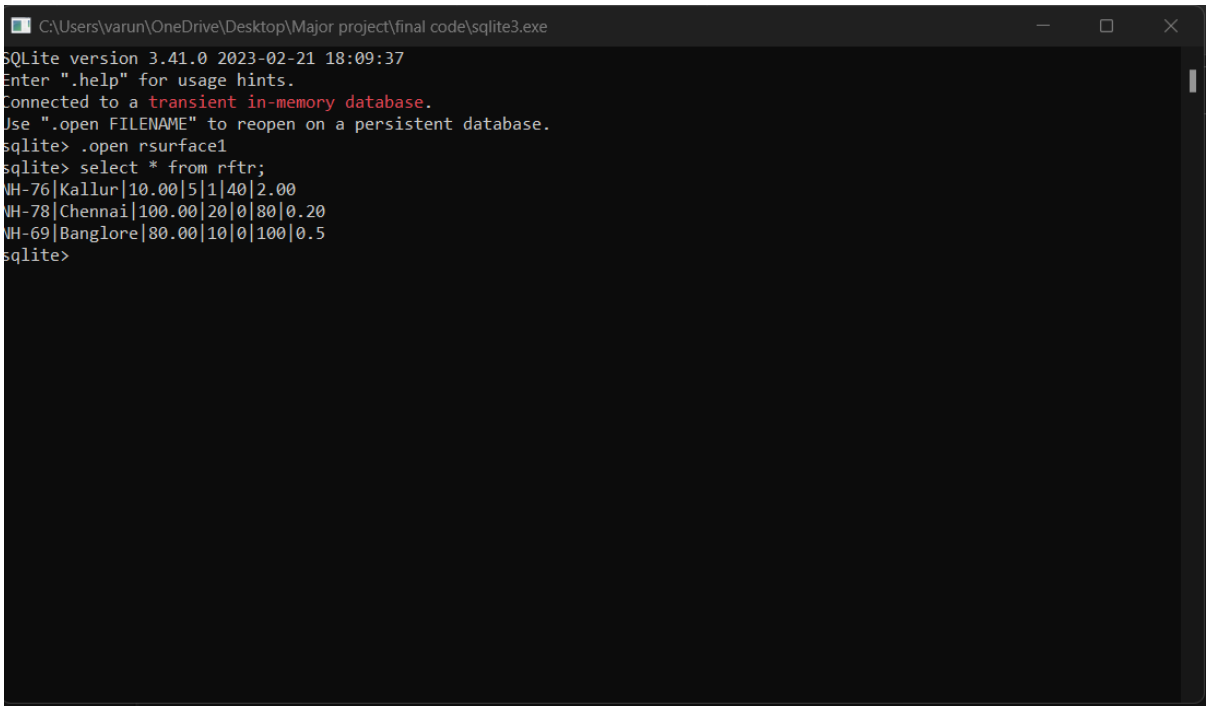

Fig 7.9: rftr table

9) Accuracy test for Bernoulli naïve bayes classifier.

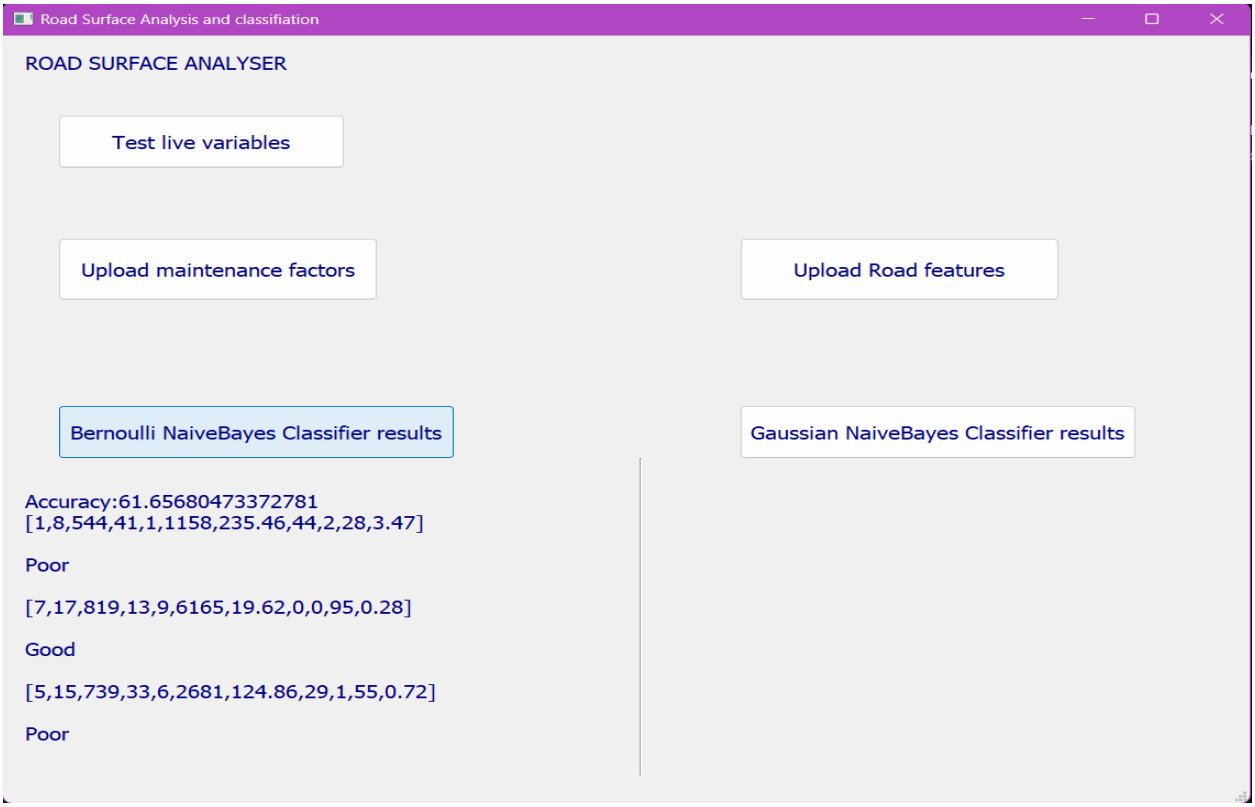

Fig 7.10: Accuracy test of Bernoulli naïve bayes classifier

10) Accuracy test for Gaussian naïve bayes classifier.

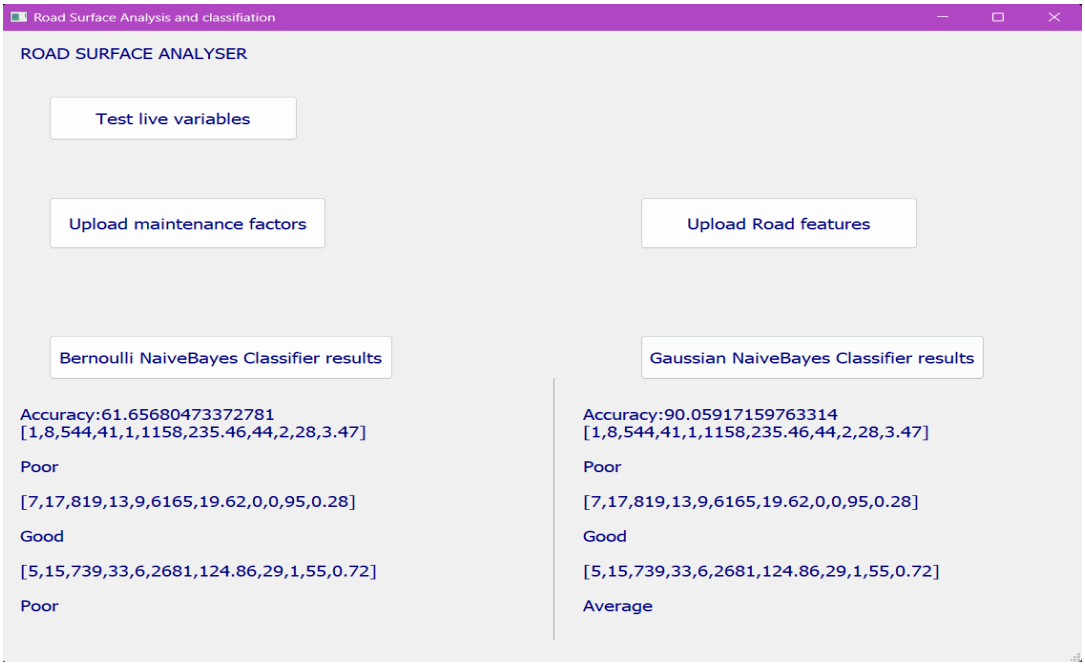

Fig 7.11: Accuracy test of Bernoulli naïve bayes classifier

7.2 Existing system method

Some audio files were extracted from the test set, converted into quantized vectors and used as inputs of the model. By observing the classifier prediction on mocked data input, it was possible to verify the reliability of the final model on the real hardware. As was foreseeable, the board processes the data and simultaneously detects the correct class of the received input vectors. In addition, we checked the functionality of BLE. As expected, the BLE application operates as required. Tiny CNN confusion matrix on test set: accuracy of: 91%. TABLE 7.2. Performance metrics on test set. for pairing request from a client and then, it starts sending the detected labels to the external apparatus.

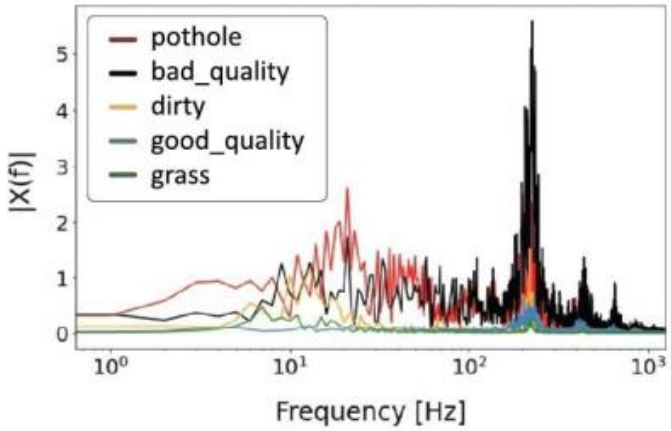

Fig 7.12: Spectrum of each considered road surface

7.2.1 Tabular performance metrics on test set

	Precision	**Recall**	**F1- score**
Silence	1.000	1.000	1.000
Unknown	0.917	0.917	0.917
Good quality	0.923	1.000	0.960
Ruined	0.909	0.833	0.870
Accuracy	0.937		

Table 7.1: Conv CNN model

	Precision	Recall	F1- score
Silence	1.000	1.000	1.000
Unknown	0.875	0.875	0.875
Good quality	0.880	0.917	0.898
Ruined	0.870	0.833	0.851
Accuracy	0.916		

Table 7.2: Tiny CNN model

7.3 Proposed system method

Numerical information is extracted from the test set and set to analysis, on which necessary probabilities are applied by the classifiers to get accurate results. Considering all the attributes which effect the road surface quality, this project takes all of them into consideration to give out better results than the existing system, as that system is only taking in a few attributes. These attributes were captured lively such as through vision sensors and acoustic sensors, in which other internal information cannot be considered while the analysis.

This project takes in the overall attributes and analyses every numerical information and gives us the overall accuracy and prediction about the road surface quality. Gaussian naïve bayes classifier on test set: 90% accurate.

7.3.1 Tabular comparison of results

	Precision	Recall
Good	1.000	1.000
Average	0.799	0.899
Bad	0.741	0.652
Accuracy	0.616	

Table 7.3: Bernoulli naïve bayes model

	Precision	Recall
Good	1.000	1.000
Average	0.899	0.899
Bad	0.652	0.652
Accuracy	0.905	

Table 7.4: Gaussian naïve bayes model

CHAPTER 8

SYSTEM TESTING

8.1 Software Testing

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

8.1.1 Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

8.1.2Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

8.1.3Basics of software testing

There are two basics of software testing: Black box testing and white box testing.

1. Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

2. White box Testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

3. Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

4. Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

5. Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

6. System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

7. Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

CHAPTER 9

CONCLUSION

This project entitled **"Road Surface Analysis and classification."** is useful for the roads and buildings department to recognize bad roads so that they can initiate steps to repair those roads. The project is useful for the tourists to avoid travel on bad road surfaces by planning their tour through alternate roads and this project finally leads to the traveller's comfort.

Existing system:

In this work, they have presented an application for real-time road surface monitoring based on AI tools. The algorithm is designed to be implemented on a microcontroller board equipped with a microphone that captures sounds inside the cavity of a tyre. Preliminary experimental results show that the device is capable of detecting the quality of the asphalt with an accuracy of 91% on the test set. This demonstrates the suitability of the proposed Tiny architecture for this application and of the Mel inspired spectrogram as input to detect road health. The presented approach takes advantage of innovative techniques. In fact, the deployment of AI system settled on embedded system is a cutting-edge technology, focus of many current research. The fact of having a light, low-power and low-cost device is a great advantage respect to the current state of the art and commercial technologies. Moreover, the methodology adopted in the present paper is not susceptible to light condition or environmental noise.

Proposed system:

In this work, we have represented a GUI based application for all-round performance to predict the road surface quality. The algorithm takes in all the necessary attributes which are used to define the quality of a road surface, analyses the data and gives a favourable outcome which results in the prediction. Experimental results shows that the algorithm is capable of pursuing 90% accuracy undertaking all the factors which effect the definition of a road surface.

FUTURE SCOPE

As of now, the project is implemented by using Gaussian Naive Bayes and Bernoulli Naive Bayes classifiers. Application of other classifiers like Voting Classifier, needs to be further explored.

The preliminary obtained results are very encouraging. However, new acquisition campaigns are necessary to enrich the dataset and make the convolutional neural network more robust. The new dataset will consist of audio tracks acquired directly from the device developed and presented in this work. As already said, in this study, it was employed a Raspberry Pi 3 for data acquisition, and it is worth noting that its analogue front-end is the same as that used in our custom board.

The accuracy of the system will be evaluated by collecting the online prediction achieved on the vehicle in motion. Finally, a distinct smartphone application will be developed with the idea of acquiring GPS coordinates as well as road quality. In this way, it will be possible to create a web platform containing the map of roads and their quality.

CHAPTER 10

REFERENCES

[1] ALESSIO GAGLIARDI, V. STADERINI AND S. SAPONARA: An Embedded System for Acoustic Data Processing and AI-Based Real-Time Classification for Road Surface Analysis, IEEE Access.

[2] S. C. Radopoulou and I. Brilakis, ‘‘Automated detection of multiple pavement defects,’’ J. Comput. Civil Eng., vol. 31, no. 2, Mar. 2017, Art. no. 04016057.

[3] L. Sjögren and T. Lundberg, ‘‘Design an up to date rut depth monitoring profilometer, requirements and limitations,’’ Statens väg-och Transportforskningsinstitut, Linköping, Sweden, Tech. Rep., 2005.

[4] G. L. Shevlyakov and N. O. Vilchevski, Robustness in Data Analysis: criteria and methods, VSP, Utrecht. 2002.

[5] N. V. Nguyen, G. L. Shevlyakov and V. Shin, “MAD robust fusion with non-Gausssian channel noise,” IEICE Trans. on Fundamentals of Electronics, vol. E92-A, no. 5, pp. 1293–3000, May, 2009.

[6] R. R. Wilcox, Fundamentals of Modern Statistical Methods: Substantially Improving Power and Accuracy, 1st Ed. Springer, March 2001.

[7] R. R. Wilcox, Fundamentals of Modern Statistical Methods: Substantially Improving Power and Accuracy, 1st Ed. Springer, March 2001.

[8] N. V. Nguyen, G. Shevlyakov and V. Shin, “Robust state estimation fusion in power system,” International Journal of Innovative Computing, Information and Control, vol. 7, (to appear), January 2011.

[9] GOST 32825-2014, Automobile roads for general use. Road pavements. Methods for measuring the geometric dimensions of damage, Mezhgos. sovet po standartizatsii, metrologii i sertifikatsiiicheskikh razmerov povrezhdeniy, [Interstate council for standardization, metrology and certification of damage dimensions], Moskva, Standartinform Publ., 2014, pp. 1-2 (In Russian).

[10] GOST 52399-2005, Geometric elements of highways, Mezhgos. sovet po standartizatsii, metrologii i sertifikatsiiicheskikh razmerov povrezhdeniy [Interstate council for standardization, metrology and certification of damage dimensions], Moskva, Standartinform Publ., 2014. pp. 3-4 (In Russian).

[11] H.N. Koutsopoulos, A.B. Downey, "Primitive-Based Classification of Pavement Cracking Images", Journal of Transportation Engineering, vol. 3, i. 119, pp. 402-418.

[12] H. Oliveira, P.L. Correia, "Automatic road crack segmentation using entropy and image dynamic thresholding", Proc. Eur. Signal Process Conf. Glasgow, 2009, pp. 622-626.

[13] T.S. Nguyen, S. Begot, F. Duculty, M. Avila, "Free-Form Anysotropie: A new method for crack detection on pavement surface images", 18th IEEE International Conference on Image Processing, Brussels, 2011, pp. 1069-1072.