A

MAJOR PROJECT

ON

AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

TARUN RAJ (177R1A05D4)

NARSING ABHITEJ (177R1A05G0)

AVINASH REDDY (177R1A0545)

JAGADISH KUMAR (167R1A05A4)

Under the Guidance of

G. VINESH SHANKAR

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2017-22

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING"** being submitted by **TARUN RAJ (177R1A05D4), NARSING ABHITEJ (177R1A05G0), AVINASH REDDY (177R1A0545) & JAGADISH KUMAR (167R1A05A4)** in partial fulfillment of the requirements for the award of the degree of B. Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of Bonafide work carried out by them under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

G. Vinesh Shankar — **Dr. A. Raji Reddy**

Internal Guide — DIRECTOR

Dr. K. Srujan Raju — **External Examiner**

Head of the Department

Submitted for Viva Voice Examination held on________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **G. VINESH SHANKAR,** Associate Professor, for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to the **Project Review Committee (PRC) Mr. J. Narasimharao, Dr. T. S. Mastan Rao, Dr. Suwarna Gothane, Mr. A. Uday Kiran, Mrs. G. Latha & Mr. A. Kiran Kumar** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri.**Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

TARUN RAJ (177R1A05D4)
NARSING ABHITEJ (177R1A05G0)
AVINASH REDDY (177R1A0545)
JAGADISH KUMAR (167R1A0A4)

ABSTRACT

Our project can be easily developed for individual heavy vehicles in a large fleet. Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption.This approach is used in conjunction with seven predictors derived from vehicle speed and road grade to produce a highly predictive neural network model for average fuel consumption in heavy vehicles.Different window sizes are evaluated and the results show that a 1 km window is able to predict fuel consumption with a 0.91 coefficient of determination and mean absolute peak-to peak percent error less than 4% for routes that include both city and highway duty cycle segments.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1. INTRODUCTION

FUEL consumption models for vehicles are of interest to manufacturers, regulators, and consumers. They are needed across all the phases of the vehicle life-cycle. we focus on modeling average fuel consumption for heavy vehicles during the operation and maintenance phase. A model that can be easily developed for individual heavy vehicles in a large fleet is proposed.Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption.These types of fleets exist in various sectors including, road transportation of goods , public transportation , construction trucks and refuse trucks. For each fleet, the methodology must apply and adapt to many different vehicle technologies (including future ones) and configurations without detailed knowledge of the vehicle's specific physical characteristics and measurement.

1.1 MOTIVATION

Trade-offs among the above techniques are primarily with respect to cost and accuracy as per the requirements of the intended application. In this paper, a model that can be easily developed for individual heavy vehicles in a large fleet is proposed. Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption. These types of fleets exist in various sectors including, road transportation of goods, public transportation, construction trucks and refuse trucks. For each fleet, the methodology must apply and adapt to many different vehicle technologies (including future ones) and configurations without detailed knowledge of the vehicle's specific physical characteristics and measurements. These requirements make machine learning the technique of choice when taking into consideration the desired accuracy versus the cost of the development and adaptation of an individualized model for each vehicle in the fleet.

2.LITERATURE SURVEY

2. LITERATURE SURVEY

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem statement.Literature survey is something when you look at a literature (publications) in a surface level, or an Aerial view. It incorporates the study of place people and productions are setting of research. It is phase where the analyst tries to know about what is all the literature related with one range of interest. Also, the relevant literature works are short-listed. Moreover, literature survey guides or helps the researcher to define/find out/identify a problem.

3.SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 PROBLEM STATEMENT

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do.

3.2 EXISTING SYSTEM

Model that can be easily developed for individual heavy vehicles in a large fleet is proposed. Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption. This approach is used in conjunction with seven predictors derived from vehicle speed and road grade to produce a highly predictive neural network model for average fuel consumption in heavy vehicles.

Different window sizes are evaluated and the results show that a 1 km window is able to predict fuel consumption with a 0.91 coefficient of determination and mean absolute peak-to peak percent error less than 4% for routes that include both city and highway duty cycle segments

3.3 SOFTWARE REQUIREMENTS

It deals with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product.

The software requirements that are required for this project are as follows:

- Operating System : Windows 10
- Technology : Machine learning
- Coding Language : Python
- UML's : Star Uml

3.4 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

The hardware requirements that are required for this project are as follows:

- Processor : Intel i5 or higher
- RAM : 8GB or more
- Hard Disk : minimum 500GB

4.PROPOSED SYSTEM

4. PROPOSED SYSTEM

As mentioned above Artificial Neural Networks (ANN) are often used to develop digital models for complex systems. The models proposed highlight some of the difficulties faced by machine learning models when the input and output have different domains. In this study, the input is aggregated in the time domain over 10 minutes intervals and the output is fuel consumption over the distance travelled during the same time period. The complex system is represented by a transfer function F(p) = o, where F(·) represents the system, 'p' refers to the input predictors and o is the response of the system or the output. The ANNs used in this paper are Feed Forward Neural Networks (FNN).

Training is an iterative process and can be performed using multiple approaches including particle swarm optimization and back propagation. Other approaches will be considered in future work in order to evaluate their ability to improve the model's predictive accuracy. Each iteration in the training selects a pair of (input, output) features from Ftr at random and updates the weights in the network. This is done by calculating the error between the actual output value and the value predicted by the model.

4.1 METHOD OF IMPLEMENTATION

The more complex the system being implemented, the more involved will be the systems analysis and design efforts required for implementation.

The implementation phase comprises several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

4.2 TECHNOLOGIES USED

The technologies that are used in the project are:

i. Machine Learning

ii. Python

iii. Python GUI – Tkinter

iv. NumPy

v. Matplotlib

vi. Pandas

vii. Sklearn

i. Machine Learning:

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Classification of Machine Learning:

Machine learning implementations are classified into three major categories, depending on the nature of the learning “signal” or “response” available to a learning system which are as follows:-

1. Supervised learning :

When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning. This approach is indeed similar to human learning under the supervision of a teacher. The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples.

2. Unsupervised learning :

When an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own comes under the category of Unsupervised Learning. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values.

They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms.As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find the web in the form of marketing automation are based on this type of learning.

3. Reinforcement learning :

When you present the algorithm with examples that lack labels, as in unsupervised learning. However, you can accompany an example with positive or negative feedback according to the solution the algorithm proposes comes under the category of Reinforcement learning, which is connected to applications for which the algorithm must make decisions (so the product is prescriptive, not just descriptive, as in unsupervised learning), and the decisions bear consequences.

In the human world, it is just like learning by trial and error. Errors help you learn because they have a penalty added (cost, loss of time, regret, pain, and so on), teaching you that a certain course of action is less likely to succeed than others.

An interesting example of reinforcement learning occurs when computers learn to play video games by themselves. In this case, an application presents the algorithm with examples of specific situations, such as having the gamer stuck in a maze while avoiding an enemy. The application lets the algorithm know the outcome of actions it takes, and learning occurs while trying to avoid what it discovers to be dangerous and to pursue survival. You can have a look at how the company Google DeepMind has created a reinforcement learning program that plays old Atari's video games. When watching the video, notice how the program is initially clumsy and unskilled but steadily improves with training until it becomes a champion.

4. Semi-supervised learning:

Where an incomplete training signal is given: a training set with some (often many) of the target outputs missing. There is a special case of this principle known as Transduction where the entire set of problem instances is known at learning time, except that part of the targets are missing.

ii. Python:

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Characteristics of Python:

Following are important characteristics of Python Programming −

- It supports functional and structured programming methods as well as OOP. It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking. It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

iii. Python GUI - Tkinter:

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a Tkinter app:

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

iv. Numpy:

NumPy is a module for Python. The name is an acronym for "Numeric Python" or "Numerical Python". It is pronounced ‘NUM-py’ or less often ‘NUM-pee’. It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of Numpy guarantee great execution speed. Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

Furthermore, NumPy enriches the programming language Python with powerful data structures, implementing multi-dimensional arrays and matrices. These data structures guarantee efficient calculations with matrices and arrays. The implementation is even aiming at huge matrices and arrays, better know under the heading of "big data". Besides that the module supplies a large library of high-level mathematical functions to operate on these matrices and arrays.

NumPy is based on two earlier Python modules dealing with arrays. One of these is Numeric. Numeric is like NumPy a Python module for high-performance, numeric computing, but it is obsolete nowadays. Another predecessor of NumPy is Numarray, which is a complete rewrite of Numeric but is deprecated as well. NumPy is a merger of those two, i.e. it is build on the code of Numeric and the features of Numarray.

v. Matplotlib:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

Matplotlib was originally written by John D. Hunter, has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc. It is a very powerful plotting library useful for those working with Python and NumPy. And for making statistical inference, it becomes very necessary to visualize our data and Matplotlib is the tool that can be very helpful for this purpose. It provides a MATLAB like interface. The only difference is that it uses Python and is open source.

Matplotlib 2.0.x supports Python versions 2.7 through 3.6. Python 3 support started with Matplotlib 1.2. Matplotlib 1.4 is the last version to support Python 2.6. Matplotlib has pledged to not support Python 2 past 2020 by signing the Python 3 Statement.

v. Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. Prior to Pandas, 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.

vi. Sklearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Origin of Scikit-Learn:

It was originally called scikits.learn and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

5.EXEPERIMENTAL ANALYSIS

5. EXPERIMENTAL ANALYSIS

5.1 UML DIAGRAMS

UML stands for Unified Modelling Language which is used in object oriented software engineering. It is a standard language for specifying, visualizing, constructing, and documenting the artifacts of the software systems. UML is different from other common programming languages like C++, Java, and COBOL etc. It is a pictorial language used to make software blueprints.

Although typically used in software engineering it is a rich language that can be used to model application structures, behavior and even business processes. There are 8 UML diagram types to help us model this behavior.

There are two types of UML modeling:

- Structural Modeling
- Behavioral Modeling

Structural Modeling:

Structural models represent the framework for the system and this framework is the place where all other components exist. Hence, the class diagram, component diagram and deployment diagrams are part of structural modelling. They all represent the elements and the mechanism to assemble them. The structural model never describes the dynamic behaviour of the system. Class diagram is the most widely used structural diagram.

Structural Modelling captures the static features of a system. They consist of the following:

i. Classes diagrams
ii. Objects diagrams
iii. Deployment diagrams
iv. Package diagrams
v. Composite structure diagrams
vi. Component diagram

Behavioral Modeling:

A Behavioral model describes the interaction in the system. It represents the interaction among the structural diagrams. Behavioral modeling shows the dynamic nature of the system. They consist of the following:

i. Activity diagrams

ii. Interaction diagrams

iii. Use case diagrams

All the above show the dynamic sequence of flow in a system.

5.1.1 USE CASE DIAGRAM:

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. The "actors" are people or entities operating under defined roles within the system.

As the most known diagram type of the behavioral UML diagrams, use-case diagrams give a graphic overview of the characters involved in a system, different functions needed by those characters and how these different functions are interacted.

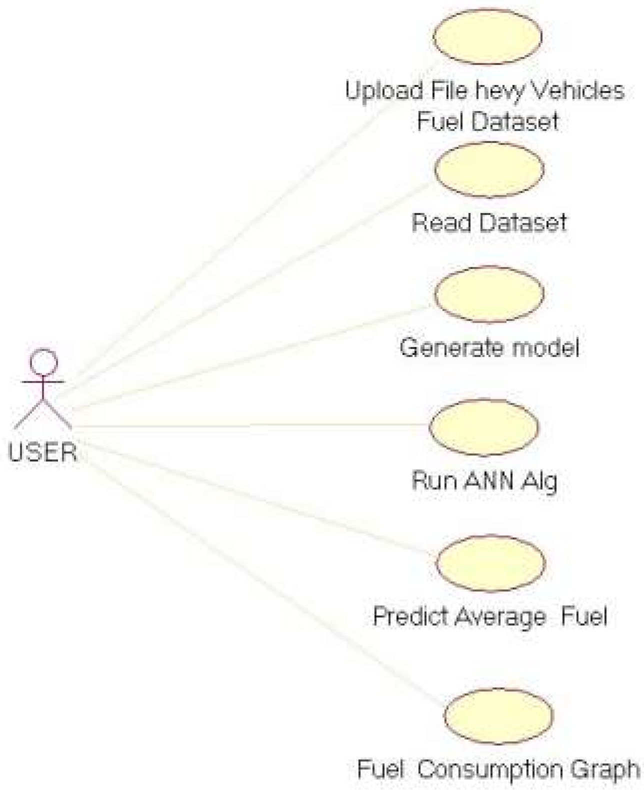

Figure 5.1.1: Use case diagram

5.1.2 CLASS DIAGRAM

Class diagrams are the main building blocks of every object oriented methods. It is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

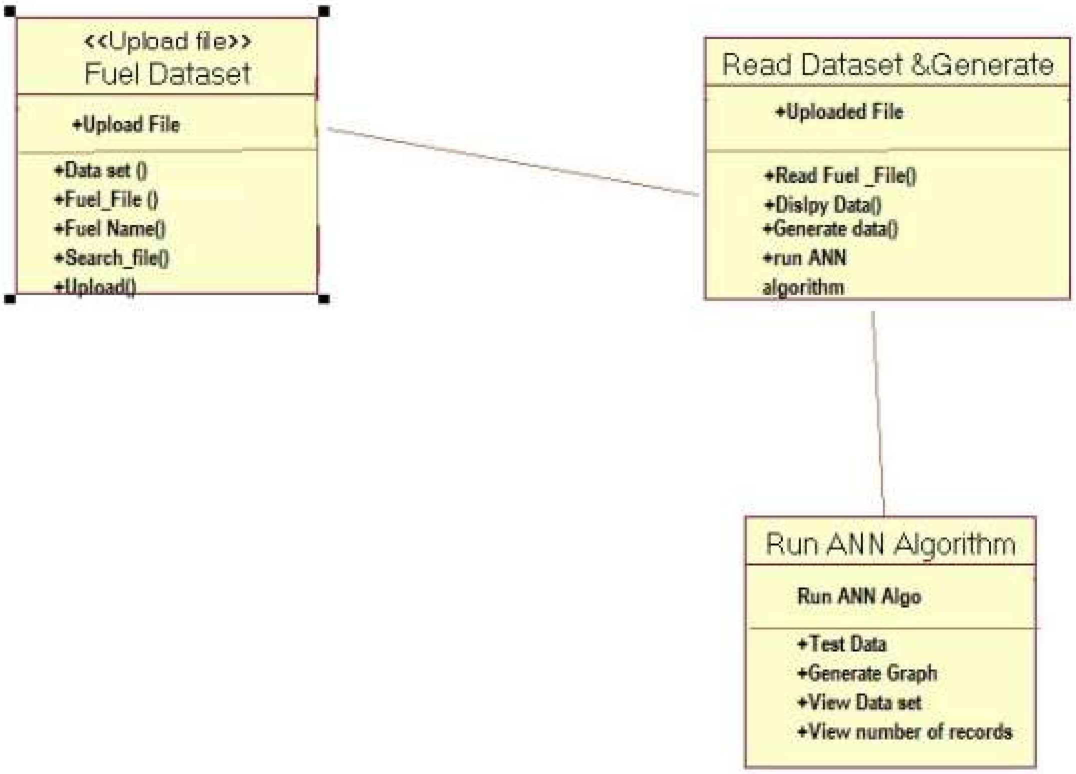

Figure 5.1.2: Class diagram

5.1.3 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Sequence diagrams emphasize on time sequence of messages and are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

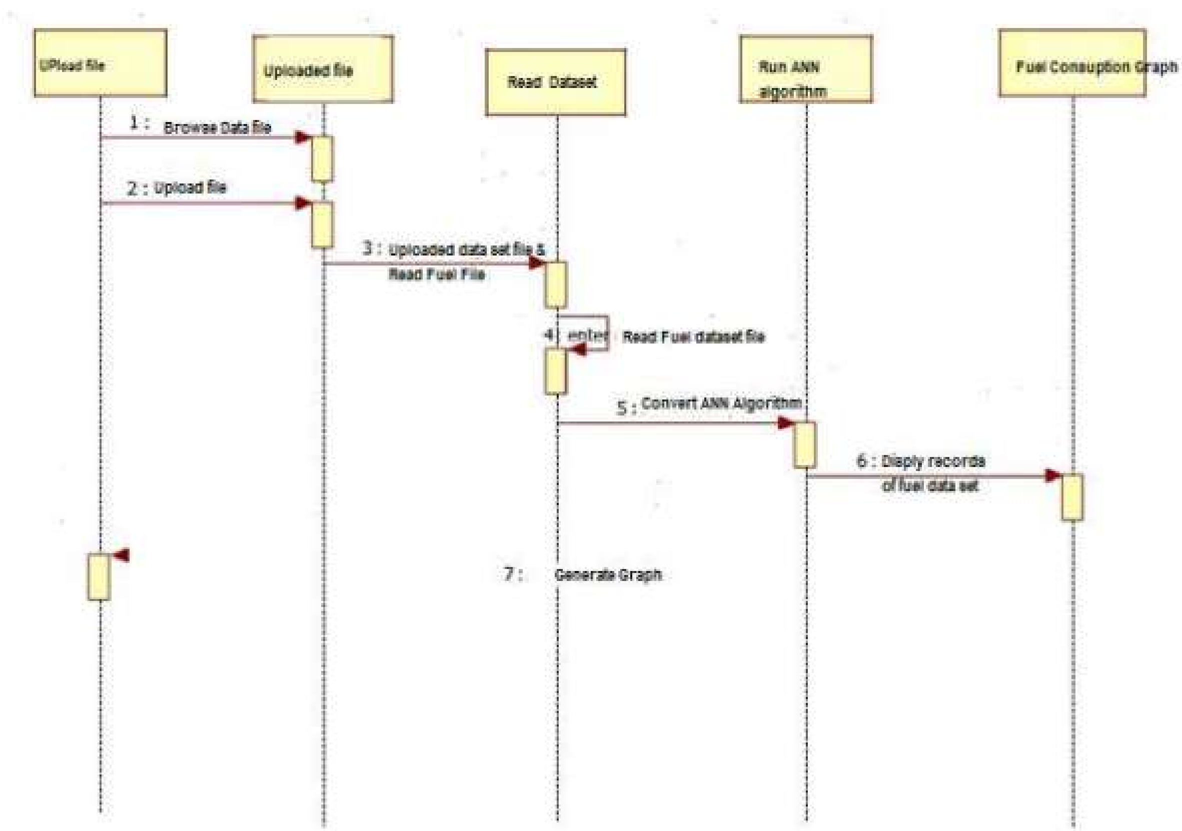

Figure 5.1.3: Sequence diagram

5.1.4 ACTIVITY 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.

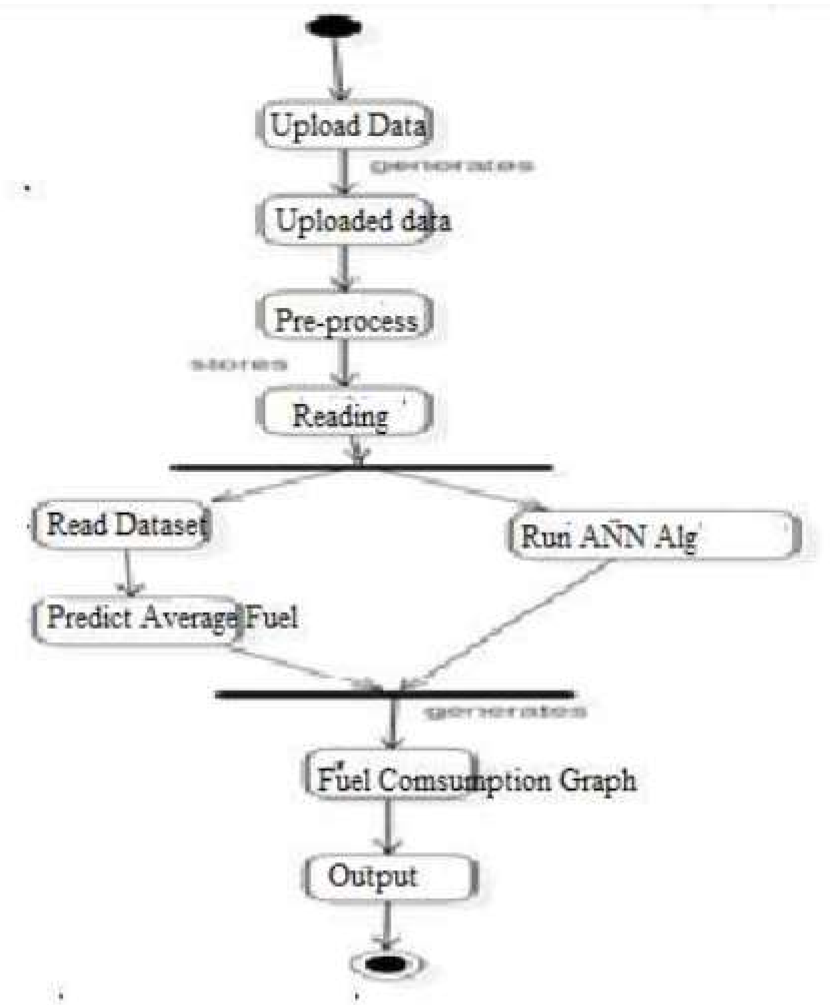

Figure 5.1.4: Activity Diagram

5.1.5 COMPONENT DIAGRAM :-

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment.

A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.

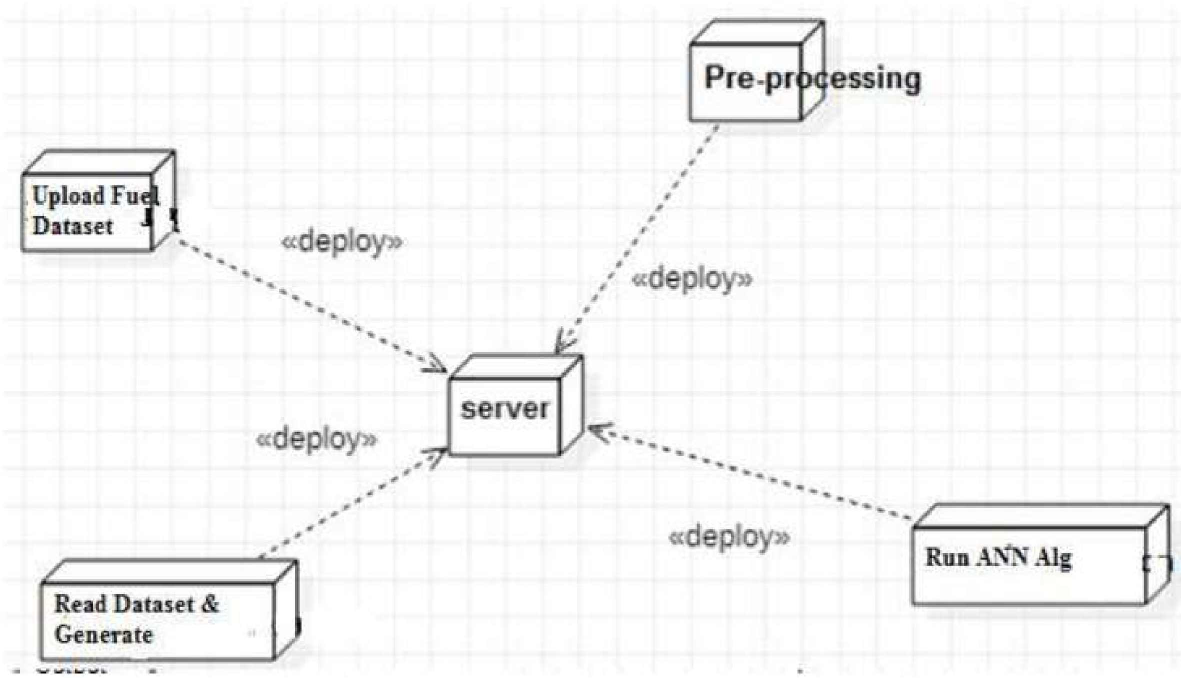

Figure 5.1.5 : Component Diagram

5.2 Data Flowchart Diagram :-

A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes. Typically a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows.

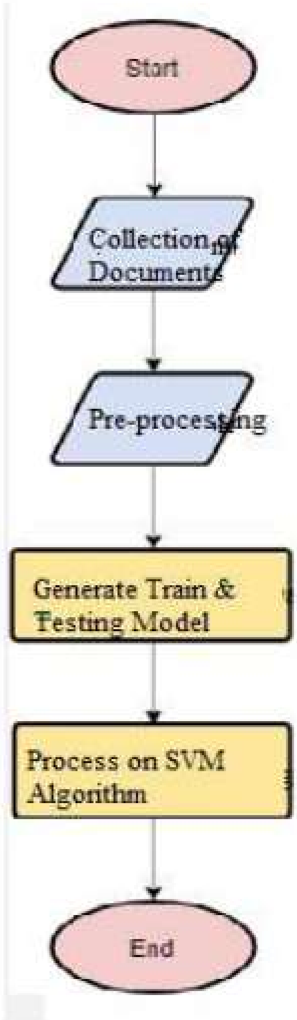

Figure 5.2: Flow Chart Diagram

5.3 ARCHITECTURE:

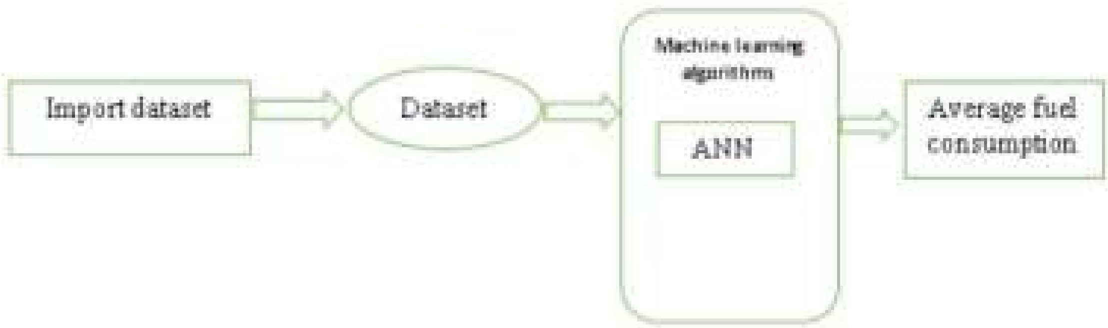

Figure 5.3 : Architecture

6.TESTING

6. TESTING

6.1 INTRODUCTION

Software Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps or missing requirements in contrast to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing. In simple terms, Software Testing means Verification of Application Under Test (AUT). Software testing is a critical element of software quality and assurance and represents ultimate review of specifications, design and coding. Testing is an exposure of the system to trial input to see whether it produces correct output. The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability. It mainly aims at measuring specification, functionality and performance of a software program or application.

6.2 TESTING STRATEGY :

Software Testing can be broadly classified into two types:

i. Manual Testing

ii. Automated Testing

i. Manual Testing:

Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective. It ensures whether the application is working, as mentioned in the requirement document or not. Test cases are planned and implemented to complete almost 100 percent of the software application. Test case reports are also generated manually. Manual Testing is one of the most fundamental testing processes as it can find both visible and hidden defects of the software. The difference between expected output and output, given by the software, is defined.

Types of Manual Testing:

There are various methods used for manual testing. Each technique is used according to its testing criteria. Types of manual testing are given below:

- White Box Testing
- Black Box Testing

White Box Testing

White –box testing, sometimes called glass-box testing is a test, case designed method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.Basis path testing is a white-box testing technique. The basis path method enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set are guaranteed to exercise every statement in the program at least one time during testing.

Black Box Testing

Black-box testing ,also called behavioral testing, focuses on the functional requirements of the s/w. Black-box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements of a program. It is a complementary approach that is likely to uncover a different class of errors that white-box methods could not. Black-box testing purposely disregards control structure; attention is focused on information domain. By applying black-box techniques, we derive a set of cases that satisfies the criteria test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specified.

ii. Automation Testing:

Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly. Apart from regression testing, automation testing is also used to test the application from load, performance, and stress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparison to manual testing.

6.3 TESTING ACTIVITIES:

Software level testing can be majorly classified into 4 levels:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Acceptance Testing

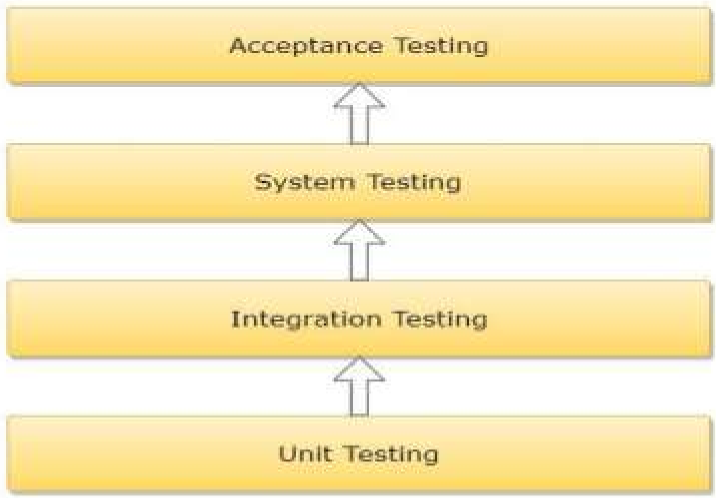

Fig 6.3 Levels of Testing

i. Unit Testing:

Unit Testing is a software testing technique by means of which individual units of software i.e. group of computer program modules, usage procedures and operating procedures are tested to determine whether they are suitable for use or not. It is a testing method using which every independent module is tested to determine if there are any issue by the developer himself. It is correlated with functional correctness of the independent modules.

Unit Testing is defined as a type of software testing where individual components of a software are tested. Unit Testing of software product is carried out during the development of an application. An individual component may be either an individual function or a procedure. Unit Testing is typically performed by the developer.

ii. Integration Testing:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units. Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

In integration testing, testers test the interfaces between the different modules. These modules combine together to form a bigger component or the system. Hence, it becomes very crucial to validate their behavior when they work together. Apart from the interfaces, they also test the integrated components. Integration testing is the next level of testing after unit testing. Testers do it after completion of the unit testing phase. Integration testing techniques can be a white box or black box depending on the project requirements.

iii. System Testing:

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In other words, System Testing means testing the system as a whole. All the modules/components are integrated in order to verify if the system works as expected or not. System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product. The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.

iv. Acceptance Testing:

Acceptance Testing is a method of software testing where a system is tested for acceptability. The major aim of this test is to evaluate the compliance of the system with the business requirements and assess whether it is acceptable for delivery or not. The standard definition of Acceptance testing is given as, "It is a formal testing according to user needs, requirements and business processes conducted to determine whether a system satisfies the acceptance criteria or not and to enable the users, customers or other authorized entities to determine whether to accept the system or not." Acceptance Testing is the last phase of software testing performed after System Testing and before making the system available for actual use.

6.4 VALIDATION

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability. It mainly aims at measuring specification, functionality and performance of a software program or application.

1. Verification: It refers to the set of tasks that ensure that software correctly implements a specific function.
2. Validation: It refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Verification: "Are we building the product right?"

Validation: "Are we building the right product?"

7.DISCUSSION OF RESULTS

7. DISCUSSION OF RESULTS

7.1 SAMPLE CODE

Source Code:

```
from tkinter import messagebox

from tkinter import *

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

import numpy as np

from tkinter.filedialog import askopenfilename

import pandas as pd

from sklearn import *

from sklearn.model_selection import train_test_split

from keras.models import Sequential

from keras.layers.core import Dense,Activation,Dropout

from keras.callbacks import EarlyStopping

from sklearn.preprocessing import OneHotEncoder  from keras.optimizers import Adam
import os

main = tkinter.Tk()

main.title("Average Fuel Consumption") #designing main screen

main.geometry("1300x1200")

global filename

global train_x, test_x, train_y, test_y

global balance_data

global model

global ann_acc
```

```
global testdata
global predictdata
def importdata():
 global balance_data
 balance_data = pd.read_csv(filename)
 balance_data = balance_data.abs()
 return balance_data
def splitdataset(balance_data):
 global train_x, test_x, train_y, test_y
 X = balance_data.values[:, 0:7]
 y_ = balance_data.values[:, 7]
 print(y_)
 y_ = y_.reshape(-1, 1)
 encoder = OneHotEncoder(sparse=False)
 Y = encoder.fit_transform(y_)
 print(Y)
train_x, test_x, train_y, test_y = train_test_split(X, Y, test_size=0.2)   text.insert(END,"Dataset
Length : "+str(len(X))+"\n");
return train_x, test_x, train_y, test_y
def upload(): #function to upload tweeter profile
global filename
filename    =    filedialog.askopenfilename(initialdir="dataset")
text.delete('1.0', END)
text.insert(END,filename+" loaded\n\n");
def generateModel():
global train_x, test_x, train_y, test_y
data = importdata()
```

```
text.insert(END,"Splitted Training Length : "+str(len(train_x))+"\n");
text.insert(END,"Splitted Test Length : "+str(len(test_x))+"\n"); def
ann():
global model
global ann_acc
model = Sequential()
model.add(Dense(200, input_shape=(7,), activation='relu', name='fc1'))
model.add(Dense(200, activation='relu', name='fc2'))
model.add(Dense(19, activation='softmax', name='output'))
optimizer = Adam(lr=0.001)
model.compile(optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
print('CNN Neural Network Model Summary: ')
print(model.summary())
model.fit(train_x, train_y, verbose=2, batch_size=5, epochs=200)
results = model.evaluate(test_x, test_y)
text.insert(END,"ANN Accuracy for dataset "+filename+"\n");
text.insert(END,"Accuracy Score : "+str(results[1]*100)+"\n\n")
ann_acc = results[1] * 100
def predictFuel():
global testdata
global predictdata
text.delete('1.0', END)
filename = filedialog.askopenfilename(initialdir="dataset")
testdata = pd.read_csv(filename)
testdata = testdata.values[:, 0:7]
predictdata = model.predict_classes(testdata)
```

```
for i in range(len(testdata)):

text.insert(END,str(testdata[i])+"    Average    Fuel    Consumption    :
"+str(predictdata[i])+"\n");

def graph():

x = []

y = []

for i in range(len(testdata)):

x.append(i)

y.append(predictdata[i])

plt.plot(x, y)

plt.xlabel('Vehicle ID')

plt.ylabel('Fuel Consumption/10KM')

plt.title('Average Fuel Consumption Graph')

plt.show()

font = ('times', 16, 'bold')
title = Label(main, text='A Machine Learning Model for Average Fuel Consumption in
Heavy Vehicles')

title.config(bg='greenyellow', fg='dodger blue')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

font1 = ('times', 14, 'bold')
```

```
uploadButton = Button(main, text="Upload Heavy Vehicles Fuel Dataset",
command=upload)
uploadButton.place(x=50,y=550)
uploadButton.config(font=font1)
modelButton = Button(main, text="Read Dataset & Generate Model",
command=generateModel)
modelButton.place(x=420,y=550)
modelButton.config(font=font1)
annButton = Button(main, text="Run ANN Algorithm", command=ann)
annButton.place(x=760,y=550)
annButton.config(font=font1)
predictButton = Button(main, text="Predict Average Fuel Consumption",
command=predictFuel)
predictButton.place(x=50,y=600)
predictButton.config(font=font1)

graphButton = Button(main, text="Fuel Consumption Graph", command=graph)
graphButton.place(x=420,y=600)
graphButton.config(font=font1)
exitButton = Button(main, text="Exit", command=exit)
exitButton.place(x=760,y=600)
exitButton.config(font=font1)
main.config(bg='LightSkyBlue')
main.mainloop()
```

7.2 RESULTS

Figure 7.2.1: Home page

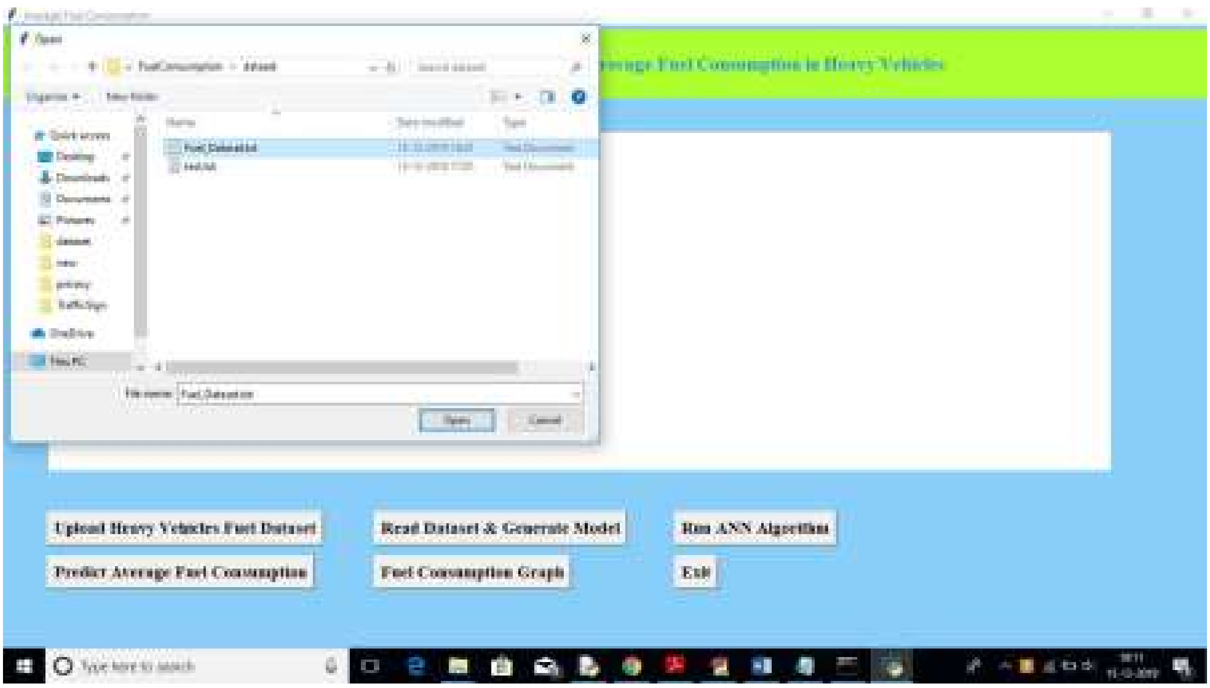

Figure 7.2.2: Uploading train dataset

Figure 7.2.3: uploaded dataset

Figure 7.2.4: number of records of dataset,train,test

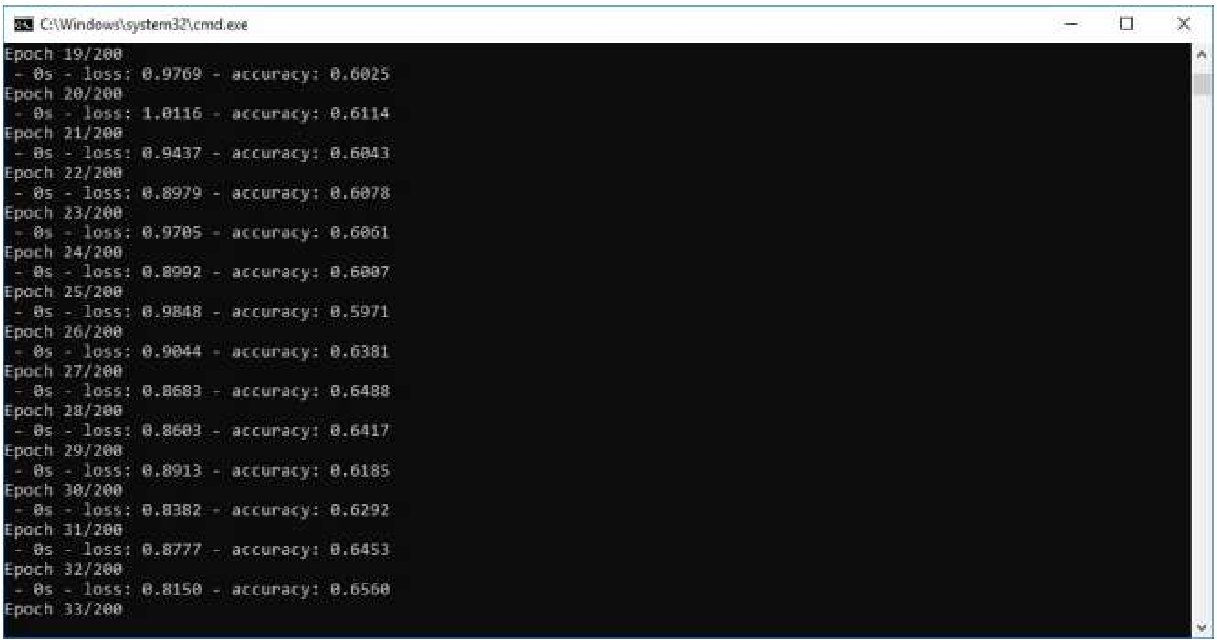

Figure 7.2.5: ANN processing details

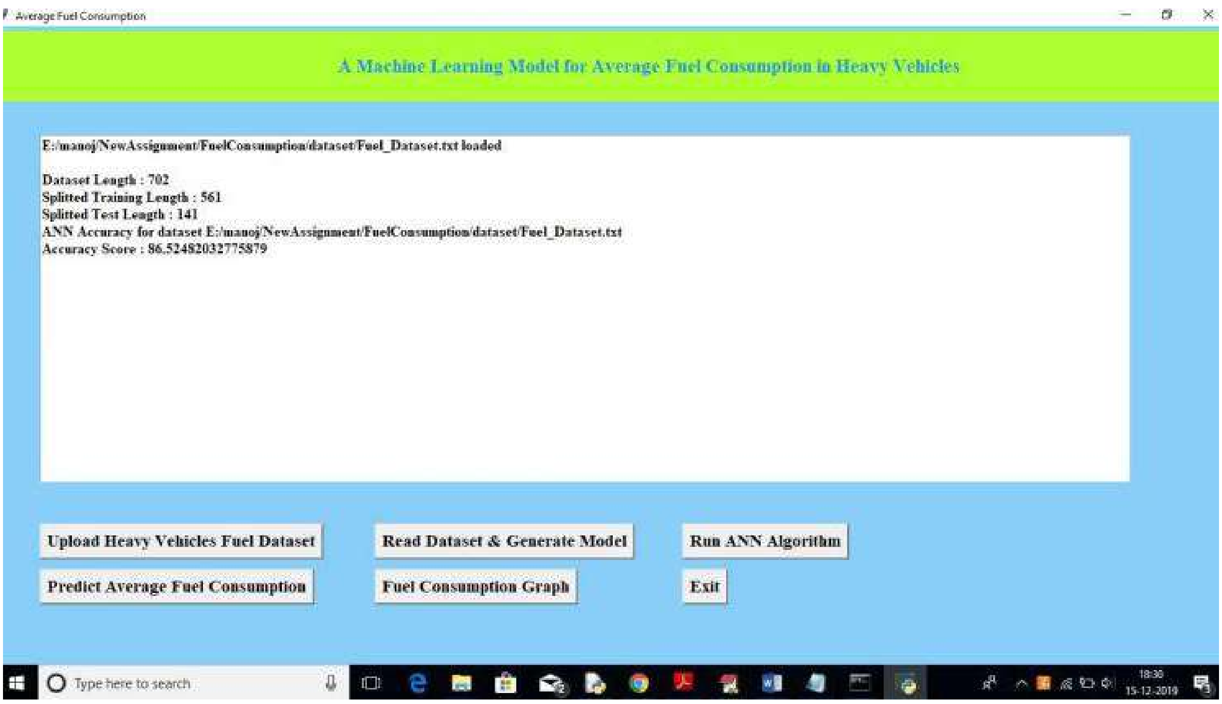

Figure 7.2.6: ANN prediction accuracy

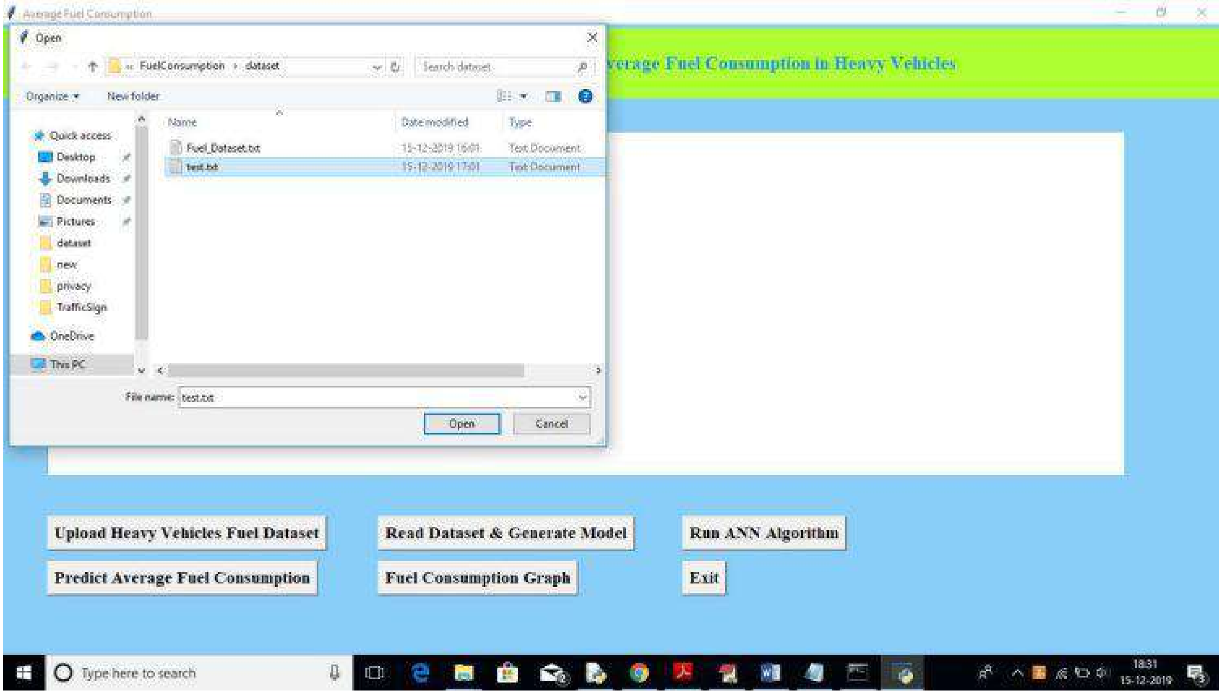

Figure 7.2.7: Uploading test data

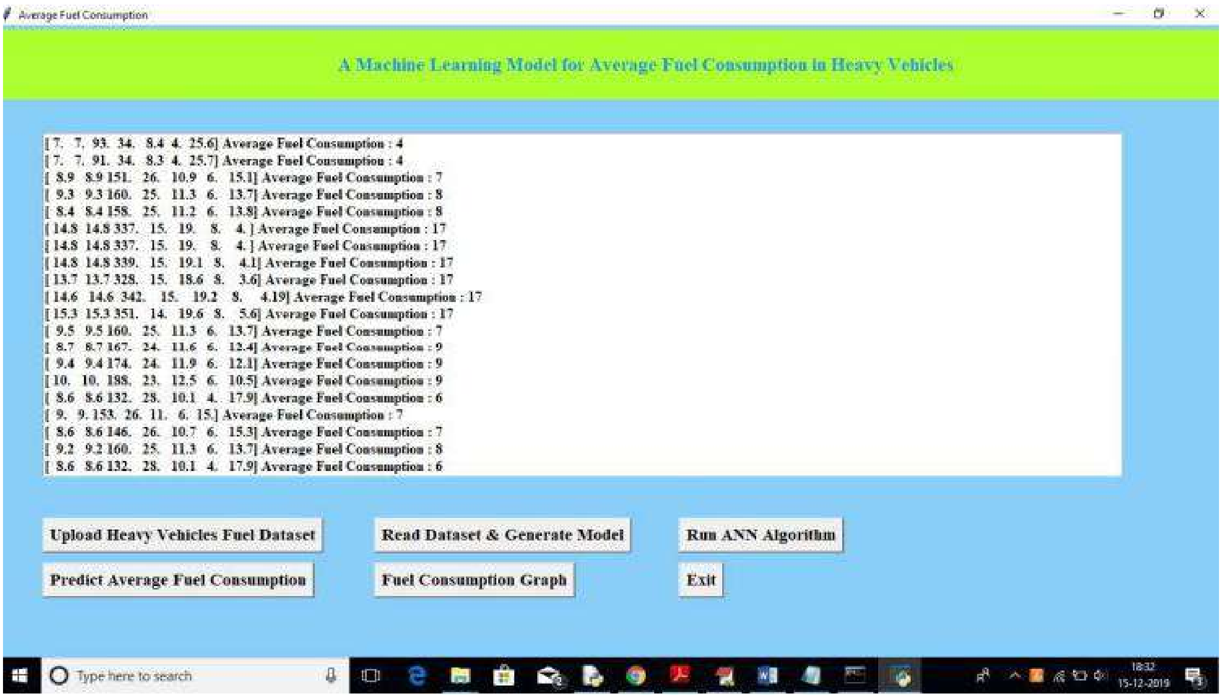

Figure 7.2.8: Average fuel consumption of each record per 100km

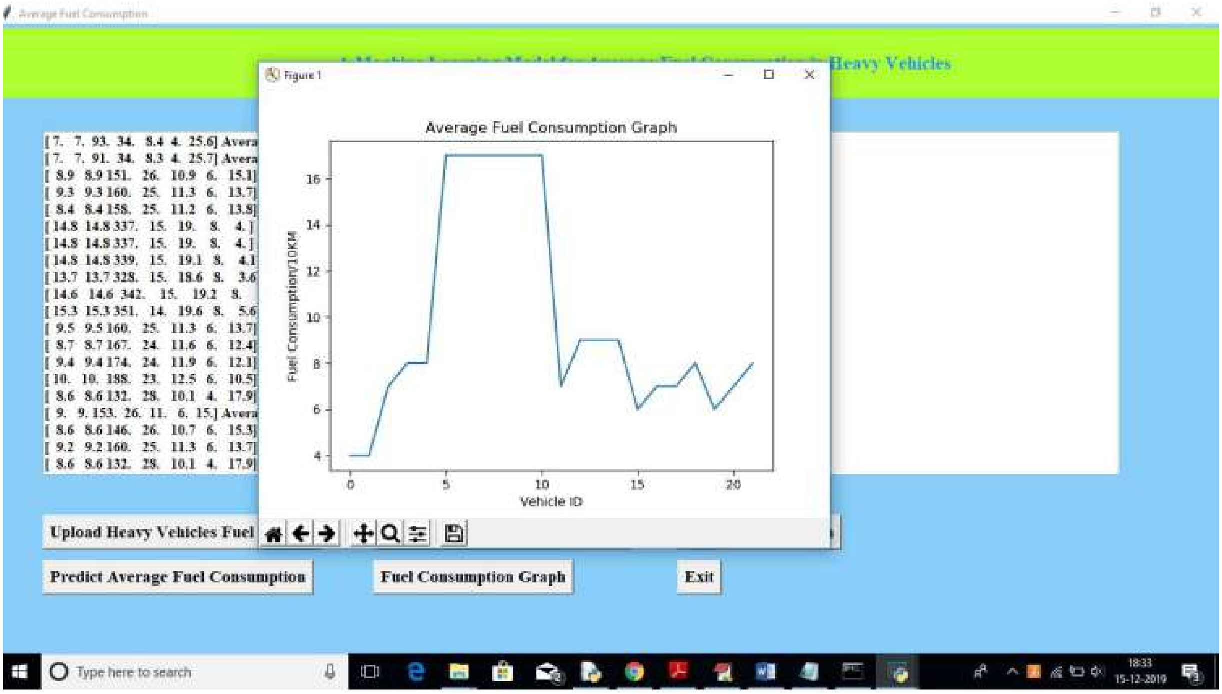

Figure 7.2.9: Graph represents fuel consumption of that record

8.CONCLUSION & FUTURE ENHANCEMENT

8. CONCLUSION & FUTURE ENHANCEMENT

8.1 CONCLUSION:

Machine learning model that can be conveniently developed for each heavy vehicle in a fleet. The model relies on seven predictors: number of stops, stop time, average moving speed, characteristic acceleration, aerodynamic speed squared, change in kinetic energy and change in potential energy.The last two predictors are introduced in this paper to help capture the average dynamic behavior of the vehicle. All of the predictors of the model are derived from vehicle speed and road grade.

These variables are readily available from telematics devices that are becoming an integral part of connected vehicles. Moreover, the predictors can be easily computed on-board from these two variables.

8.2 FUTURE ENHANCEMENT:

In this paper the author is describing a concept to predict average fuel consumption in heavy vehicles using Machine Learning Algorithms such as ANN (Artificial Neural Networks). To predict fuel consumption author has extracted 7 predictor features from heavy vehicle dataset

1. Future work also includes investigating the minimum distance required for training each model and analyzing how often does a model need to be synchronized with the physical system in operation by using online training in order to maintain the prediction accuracy of the model.
2. Predict average fuel consumption in heavy vehicles using Machine Learning Algorithm- such as ANN (Artificial Neural Networks). To predict fuel consumption author has extracted 7 predictor features from heavy vehicle dataset such as:

   num_stops,time_stopped,average_moving_speed,
   characteristic_acceleration,aerodynamic_speed_squared,change_
   in_kinetic_energy, change_in_potential_energy, class

9.REFERENCES

9. REFERENCES:

9.1 TEXT BOOKS

- Machine Learning with Python: Design and Develop Machine Learning and Deep Learning Technique using real world code examples, Abhishek Vijayvargia, 1st Edition, 2019.
- Python GUI Programming - A Complete Reference Guide: Develop responsive and powerful GUI applications with PyQt and Tkinter, Alan D. Moore, B. M. Harwani, 2019.
- Machine Learning Advances in Payment Card Fraud Detection, Nick Ryman Tubb, Paul Krause, 2019.
- Machine Learning for Beginners: The Definitive Guide to Neural Networks, Random Forests, and Decision Trees, Jennifer Grange, 2017.

9.2 WEBSITES

- www.google.com
- www.tutorialspoint.com
- www.geeksforgeeks.org
- ieeexplore.ieee.org
- en.wikipedia.org/wiki/Machine_learning
- www.slideshare.net

9.3 PAPER REFERENCES:

1. B. Lee, L. Quinones, and J. Sanchez, “Development of greenhouse gas emissions model for 2014-2017 heavy-and medium-duty vehicle compliance,” SAE Technical Paper, Tech. Rep., 2011.
2. G. Fontaras, R. Luz, K. Anagnostopoulus, D. Savvidis, S. Hausberger, and M. Rexeis, “Monitoring co2 emissions from hdv in europe-an experimental proof of concept of the proposed methodolgical approach,” in 20th International Transport and Air Pollution Conference, 2014.
3. S. Wickramanayake and H. D. Bandara, “Fuel consumption prediction of fleet vehicles using machine learning: A comparative study,” in Moratuwa Engineering Research Conference (MERCon), 2016. IEEE, 2016, pp. 90–95.
4. L. Wang, A. Duran, J. Gonder, and K. Kelly, “Modeling heavy/mediumduty fuel consumption based on drive cycle properties,” SAE Technical Paper, Tech. Rep., 2015.
5. Fuel Economy and Greenhouse gas exhaust emissions of motor vehicles Subpart B - Fuel Economy and Carbon-Related Exhaust Emission Test Procedures, Code of Federal Regulations Std. 600.111-08, Apr 2014.
6. SAE International Surface Vehicle Recommended Practice, Fuel Consumption Test Procedure - Type II, Society of Automotive Engineers Std., 2012.
7. F. Perrotta, T. Parry, and L. C. Neves, “Application of machine learning for fuel consumption modelling of trucks,” in Big Data (Big Data), 2017 IEEE International Conference on. IEEE, 2017, pp. 3810–3815.
8. S. F. Haggis, T. A. Hansen, K. D. Hicks, R. G. Richards, and R. Marx, “In-use evaluation of fuel economy and emissions from coal haul trucks using modified sae j1321 procedures and pems,” SAE International Journal of Commercial Vehicles, vol. 1, no. 2008-01-1302, pp. 210–221, 2008.
9. A. Ivanco, R. Johri, and Z. Filipi, “Assessing the regeneration potential for a refuse truck over a real-world duty cycle,” SAE International Journal of Commercial Vehicles, vol. 5, no. 2012-01-1030, pp. 364–370, 2012.
10. A. A. Zaidi, B. Kulcsr, and H. Wymeersch, “Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks,” IEEE Transactions on Intelligent

9.4 GITHUB LINK

https://github.com/Tarunraj808/Average-fuel-consumption-in-heavy-vechiles-using-machine-learning

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:27/June-2022 Impact Factor- 6.752 www.irjmets.com

AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING

Vinesh Shankar*[1], Tarun Raj *[2], Narsing Abhitej

*1 Dept. Of CSE, CMR Technical Campus,

Kandlakoya(V), Hyderabad, Telangana, India.

1 vineshshankar@cmrtc.ac.in
2 177r1a05d4@cmrtc.ac.in
3 177r1a05go@cmrtc.ac.in

ABSTRACT

Our project can be easily developed for individual heavy vehicles in a large fleet. Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption. This approach is used in conjunction with seven predictors derived from vehicle speed and road grade to produce a highly predictive neural network model for average fuel consumption in heavy vehicles. Different window sizes are evaluated and the results show that a 1 km window is able to predict fuel consumption with a 0.91 coefficient of determination and mean absolute peak-to peak percent error less than 4% for routes that include both city and highway duty cycle segments.

Keywords: DDOS, CUSUM-Based Entropy Detection, Time-Based Entropy Detection, PCAP Dataset.

I. INTRODUCTION

FUEL consumption models for vehicles are of interest to manufacturers, regulators, and consumers. They are needed across all the phases of the vehicle life-cycle. we focus on modeling average fuel consumption for heavy vehicles during the operation and maintenance phase. A model that can be easily developed for individual heavy vehicles in a large fleet is proposed.Relying on accurate models of all of the vehicles in a fleet, a fleet manager can optimize the route planning for all of the vehicles based on each unique vehicle predicted fuel consumption thereby ensuring the route assignments are aligned to minimize overall fleet fuel consumption.These types of fleets exist in various sectors including, road transportation of goods , public transportation , construction trucks and refuse trucks. For each fleet, the methodology must apply and adapt to many different vehicle technologies (including future ones) and configurations without detailed knowledge of the vehicle's specific physical characteristics and measurement.

II. LITERATURE SURVEY

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem statement.Literature survey is something when you look at a literature (publications) in a surface level, or an Aerial view. It incorporates the study of place people and productions are setting of research. It is phase where the analyst tries to know about what is all the literature related with one range of interest. Also, the relevant literature works are short-listed. Moreover, literature survey guides or helps the researcher to define/find out/identify a problem.

III. PROPOSED SYSTEM

As mentioned above Artificial Neural Networks (ANN) are often used to develop digital models for complex systems. The models proposed highlight some of the difficulties faced by machine learning models when the input and output have different domains. In this study, the input is aggregated in the time domain over 10 minutes intervals and the output is fuel consumption over the distance travelled during the same time period. The complex system is represented by a transfer function F(p) = o, where F(·) represents the system, 'p' refers to the input predictors and o is the response of the system or the output. The ANNs used in this paper are Feed Forward Neural Networks (FNN).

III. ARCHITECTURE

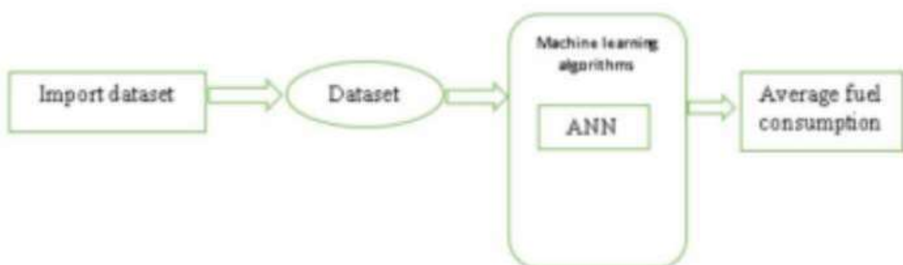

Figure 3.1: Architecture of the Model.

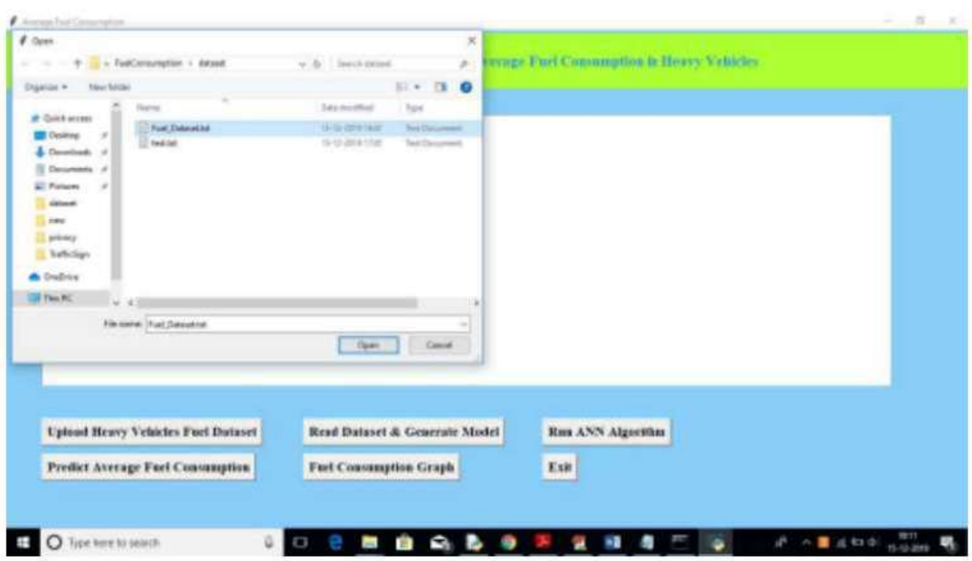

Figure 3.2 Architecture of the Model.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:27/June-2022 Impact Factor- 6.752 www.irjmets.com

Figure 3.3 number of train,dataset,test

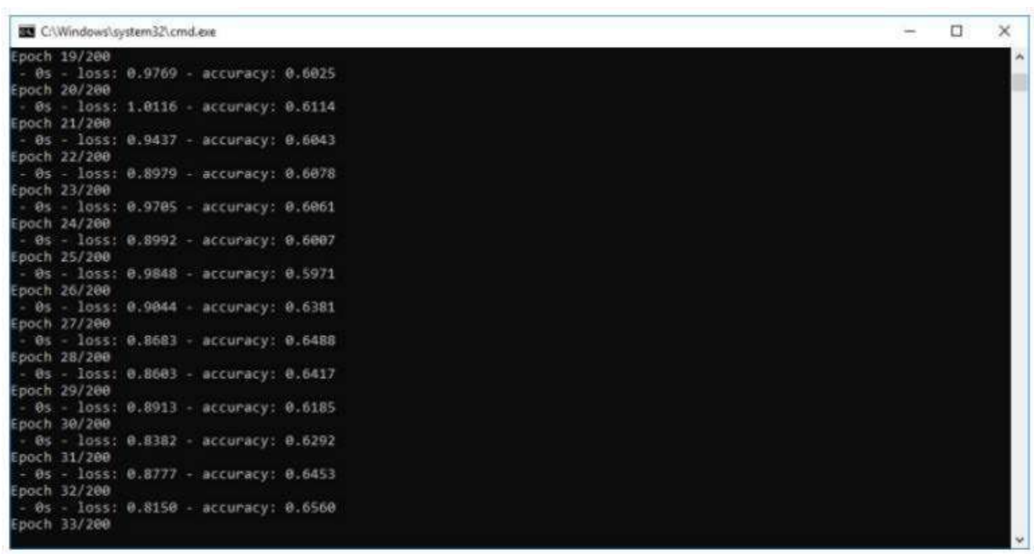

Figure 3.4 ANN predication details

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:27/June-2022 Impact Factor- 6.752 www.irjmets.com

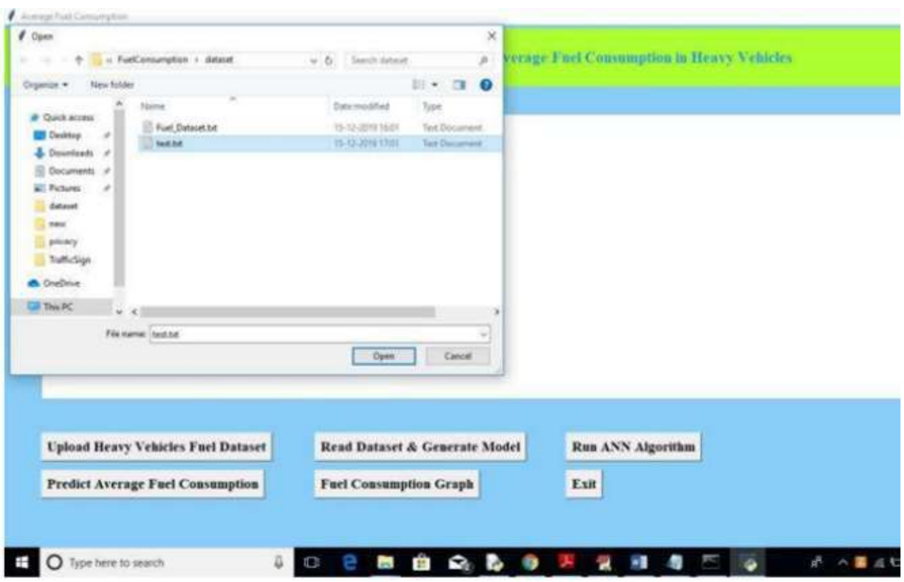

Figure 3.6 uploading testdata

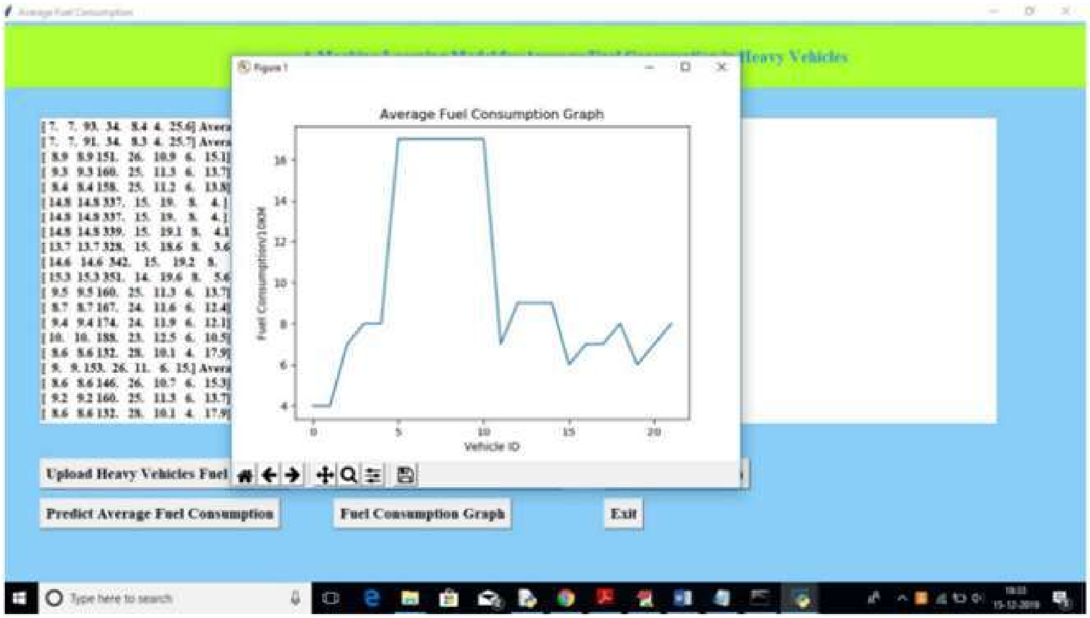

Figure 3.6 Graph represents fuel consumption of that record

V. CONCLUSION

Machine learning model that can be conveniently developed for each heavy vehicle in a fleet. The model relies on seven predictors: number of stops, stop time, average moving speed, characteristic acceleration, aerodynamic speed squared, change in kinetic energy and change in potential energy.The last two predictors are introduced in this paper to help capture the average dynamic behavior of the vehicle. All of the predictors of the model are derived from vehicle speed and road grade.

These variables are readily available from telematics devices that are becoming an integral part of connected vehicles. Moreover, the predictors can be easily computed on-board from these two variables.

VI. REFERENCES

[1] B. Lee, L. Quinones, and J. Sanchez, "Development of greenhouse gas emissions model for 2014-2017 heavy-and medium-duty vehicle compliance," SAE Technical Paper, Tech. Rep., 2011.

[2] G. Fontaras, R. Luz, K. Anagnostopoulus, D. Savvidis, S. Hausberger, and M. Rexeis, "Monitoring co2 emissions from hdv in europe-an experimental proof of concept of the proposed methodolgical approach," in 20th International Transport and Air Pollution Conference, 2014.

[3] S. Wickramanayake and H. D. Bandara, "Fuel consumption prediction of fleet vehicles using machine learning: A comparative study," in Moratuwa Engineering Research Conference (MERCon), 2016. IEEE, 2016, pp. 90–95.

[4] L. Wang, A. Duran, J. Gonder, and K. Kelly, "Modeling heavy/mediumduty fuel consumption based on drive cycle properties," SAE Technical Paper, Tech. Rep., 2015.

[5] Fuel Economy and Greenhouse gas exhaust emissions of motor vehicles Subpart B - Fuel Economy and Carbon-Related Exhaust Emission Test Procedures, Code of Federal Regulations Std. 600.111-08, Apr 2014.

[6] SAE International Surface Vehicle Recommended Practice, Fuel Consumption Test Procedure - Type II, Society of Automotive Engineers Std., 2012.

[7] F. Perrotta, T. Parry, and L. C. Neves, "Application of machine learning for fuel consumption modelling of trucks," in Big Data (Big Data), 2017 IEEE International Conference on. IEEE, 2017, pp. 3810–3815.

[8] S. F. Haggis, T. A. Hansen, K. D. Hicks, R. G. Richards, and R. Marx, "In-use evaluation of fuel economy and emissions from coal haul trucks using modified sae j1321 procedures and pems," SAE International Journal of Commercial Vehicles, vol. 1, no. 2008-01-1302, pp. 210–221, 2008.

[9] A. Ivanco, R. Johri, and Z. Filipi, "Assessing the regeneration potential for a refuse truck over a real-world duty cycle," SAE International Journal of Commercial Vehicles, vol. 5, no. 2012-01-1030, pp. 364–370, 2012.

[10] A. A. Zaidi, B. Kulcsr, and H. Wymeersch, "Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks," IEEE Transactions on Intelligent .

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 4/Issue 06/40600074129

Date: 27/06/2022

Certificate of Publication

*This is to certify that author “**Vinesh Shankar** ” with paper ID “**IRJMETS4060007 4129**” has published a paper entitled “AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING” **in International Research Journal Of Mo dernization In Engineering Technology And Science (IRJMETS), Volume 4, Issue 27, June 2022***

Editor in Chief

IRJMETS Impact Factor 6.752

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
e-ISSN: 2582-5208
Ref: IRJMETS/Certificate/Volume 4/Issue 06/40600074129
Date: 27/06/2022

Certificate of Publication

*This is to certify that author "**Tarun Raj** " with paper ID "**IRJMETS4060007 4129**" has published a paper entitled "AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 4, Issue 27, June 2022***

Editor in Chief

IRJMETS Impact Factor 6.752

We Wish For Your Better Future
www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 4/Issue 06/40600074129 *Date: 27/06/2022*

Certificate of Publication

*This is to certify that author "**Narsing Abhitej**" with paper ID "**IRJMETS4060007 4129**" has published a paper entitled "AVERAGE FUEL CONSUMPTION IN HEAVY VEHICLES USING MACHINE LEARNING" **in International Research Journal Of Mo dernization In Engineering Technology And Science (IRJMETS), Volume 4, Issue 27, June 2022***

Editor in Chief

IRJMETS Impact Factor 6.752

We Wish For Your Better Future

www.irjmets.com