Predicting Environment Variables Using Accumulated Sensor Data

Classifying Road Roughness Profiles

Master's thesis in Master Programme Complex Adaptive System

ARAVIND INBASEKARAN

DEPARTMENT OF PHYSICS

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2021
www.chalmers.se

Master's thesis 2021

Predicting Environment Variables Using Accumulated Sensor Data

Classifying Road Roughness Profiles

Aravind Inbasekaran

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Chalmers University of Technology
Gothenburg, Sweden 2021

Predicting Environment Variables Using Accumulated Sensor Data

Supervisor: Owais Mahmudi, Scania AB
Examiner: Mats Granath, Chalmers University of Technology

Master's Thesis 2021
Department of Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: An image of Scania truck .

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2021

Abstract

Environmental factors such as road roughness play a crucial role in operation of heavy vehicles. In order to recommend appropriate specifications to customers, manufacturers need to know which kind of road profile a vehicle will be operated on. By recommending correct specifications, manufacturers not only improve life of a vehicle but also trust of their customers.

Over the years, many methods have been proposed to find the roughness profile of the roads. Our work focuses on how the road roughness can be classified using the data obtained from the on-board sensors fitted in the vehicles and with the help of HERE API which is a third party API that returns the roughness value information. The data that we have used are the accelerometer readings, Global Positioning System (GPS) and the roughness index values from HERE API for the classification. These data are collected over different periods for different vehicles and stored in a database. The main aim of this thesis is to build a Machine Learning model which will be able to classify a specific route/road between two GPS points into three main roughness categories - Good, Fair and Poor. This has been approached with the Support Vector Machine model and Multi-Layer Perceptron model.

Keywords: Machine Learning, accelerometer, GPS, on-board sensors, HERE API, Road roughness, Neural Network, MLP, SVM.

Acknowledgement

I would like to thank Mats Granath, supervisor and examiner at Chalmers University of Technology for his valuable inputs and guidance throughout the thesis work.

I would like to sincerely thank my supervisor, Owais Mahmudi at Scania AB. Whenever I ran into a problem or got stuck with something, he helped me to proceed in the right path. I would like to thank Julia Liden, Kuo-Yun Liang and Richard Hedlund at Scania AB for their technical inputs. I would like to express my gratitude to my manager Katarina Prytz at Scania AB for providing me the opportunity.

I would like to express my gratitude and appreciation to my brother Arivoli, and my friends Aqshay Chandramouli, Deepak Guru Ganesan and Prashant Thangavel for their technical input and help throughout my Masters program. Finally, I would like to thank my family for their continuous support and encouragement throughout my years of study.

Aravind Inbasekaran, Gothenburg, June 2021

Contents

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**
1.1 Problem Formulation . . . 1
1.2 Purpose . . . 2
1.2.1 Objectives . . . 2
1.2.2 Goals . . . 3
1.2.3 Limitations . . . 3
1.3 Literature Study . . . 3

2 Theory **5**
2.1 International Roughness Index (IRI) . . . 5
2.2 Machine Learning . . . 6
2.2.1 Supervised Learning . . . 7
2.2.1.1 Support Vector Machine . . . 8
2.2.1.1.1 Parameters . . . 10
2.2.1.1.2 Kernels . . . 10
2.2.1.2 Neural Networks . . . 11
2.2.1.2.1 Multi-Layer Preceptrons . . . 12

3 Methods **15**
3.1 Platform . . . 15
3.2 Workflow . . . 15
3.3 Literature review summary . . . 16
3.4 Dataset . . . 16
3.5 Data pre-processing . . . 17
3.5.1 Data extraction and mapping . . . 17
3.5.1.1 Calculating Pseudo-damage . . . 18
3.5.1.2 Extracting the labels . . . 19
3.5.2 Data distribution . . . 19
3.6 Multi-layer perceptron . . . 21
3.6.1 Network architecture . . . 21
3.6.2 Experimental Setup . . . 22
3.6.3 Evaluation . . . 24
3.7 Support Vector Machines . . . 24

3.7.1 Experiment 25
3.7.2 Evaluation 25

4 Results **27**
4.1 Multi-layer Perceptron 27
4.1.1 Network with 2 hidden layers 27
4.1.2 Multi-layer perceptron with 3 hidden layers 29
4.2 Support Vector Machines 31

5 Discussion and Conclusion **35**
5.1 Discussion 35
5.2 Future Work 36
5.3 Conclusion 36

Bibliography **37**

List of Figures

1.1 Key steps in the thesis 2

2.1 Illustration of quarter car model.[11] 5
2.2 IRI Scale.[18] 6
2.3 Image showing different categories of machine learning. 7
2.4 Image showing subcategories of supervised learning. 8
2.5 Image showing hyper plane with maximum margin.[12] 9
2.6 2-D to higher dimensional mapping using kernel trick.[13] 9
2.7 Low regularization parameter 10
2.8 High regularization parameter 10
2.9 A simple perceptron model which has 3 inputs. 11
2.10 A Neural Network model with one input, hidden and an output layer. 12
2.11 A Multi-layer perceptron with two hidden layers. 13

3.1 Workflow overview. 16
3.2 List of variables in each database. 17
3.3 Example representation of table after mapping for one of the unique identifier 1. 18
3.4 Distribution of good, fair and rough roads in the dataset. 20
3.5 Road roughness in Austria, Russia and Italy 20
3.6 Summary of the architecture. 22
3.7 Summary of the architecture. 22
3.8 AUC-ROC curve. 26

4.1 Accuracy and Loss on training set and validation set 28
4.2 Confusion matrix of the model. 28
4.3 Accuracy and Loss on training set and validation set 30
4.4 Confusion matrix of the model. 30
4.5 Confusion matrix of the model. 32
4.6 Decision boundaries drawn by the SVM. 33
4.7 ROC-AUC curve. 33

List of Tables

3.1 Road roughness conditions of other countries in the dataset. 21
3.2 Accuracies for different activation functions in the hidden layer 23
3.3 Parameters for the multi-layer perceptron model 24
3.4 Accuracies of different kernels with the default paremeter 25
3.5 Parameters used in Grid Search 25

4.1 The parameters for the network 27
4.2 Classification report for the network 29
4.3 The parameters for the network 29
4.4 Classification report for the network 31
4.5 Metrics for two and three hidden layer network 31
4.6 Parameters for SVM 31
4.7 Classification report for SVM 32

List of abbreviations

AUC	Area Under Curve
API	Application Programming Interface
CNN	Convolutional Neural Networks
FMDB	Fleet Management Database
FN	False Negative
FP	False Positive
GPS	Global Positioning System
IRI	International Roughness Index
LSTM	Long Short Term Memory
MDB	Mapping Database
MLP	Multi-Layer Perceptron
ODDP	Operational Data Database
RBF	Radial Basis Function
CAN	Control Area Network
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristics
RNN	Recurrent Neural Networks
SVM	Support Vector Machines
TP	True Positive
TN	True Negative

1

Introduction

Road infrastructure has become an important part of the modern society. Large number of vehicles, be it trucks, buses, two wheelers, cars use roads everyday for travel. When large number of vehicles are using road as transport, their condition is an important factor to be considered. Poor road conditions leads to risk of vehicle damage and might lead to accidents in worst case scenario. When moving on to the commercial part, operating wrong type of vehicle on wrong surface might lead to failure of parts or the entire vehicle. Trucks are operated for different commercial purposes across the world such as transport, mining etc. Thus, vehicles are to be configured based on the condition of the road/surface where it will be operated. For example, when it comes to mining, the surface where the vehicle will be operated will be rough with many distress such as bumps, cracks, potholes etc. Operating a vehicle that is suitable for a smooth road on such surface will lead to mechanical failure and leads to high repair cost. With trucks being operated night and day, longer downtime may lead to loss in business as well. Therefore, the condition where a vehicle will be operated has to be evaluated first and then used.

There are many ways by which the road quality can be evaluated. Scania uses traditional method of collecting data from the on-board sensors fitted in the truck. Many methods has been proposed in literature so far to classify the road roughness. Among them, Machine learning is being widely used for classification while neural networks are being used more recently. This thesis focuses on using various Machine Learning techniques for multi-class classification to classify the roads into good, fair and poor. The study on how various methods are used to accumulate the data and how various methods are used to calculated the roughness has been reviewed.

1.1 Problem Formulation

The focus of this thesis is to build a machine learning model to classify the roughness of the surface. The accelerometer data for the model are being recorded and stored periodically.

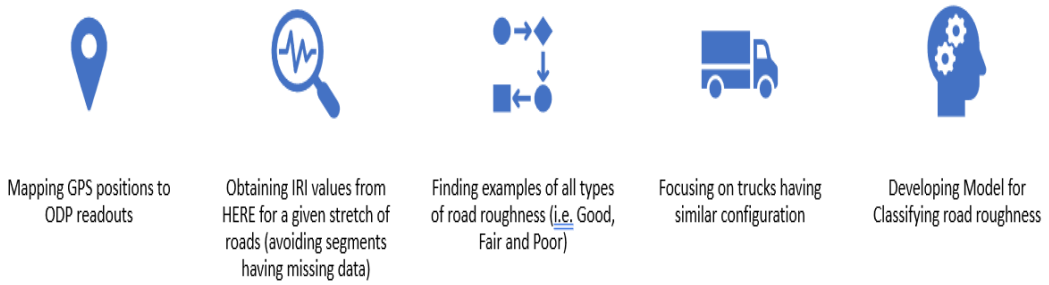

Figure 1.1: Key steps in the thesis

Figure 1.1 shows the step by step approach towards the problem

- Mapping the accelerometer readings with GPS positions
- Calculate pseudo damage from the accelerometer readings
- Pass the GPS positions to the HERE API and get the roughness categories for the link
- Collect the data for all the available vehicles with similar configuration
- Develop a machine learning model for classification

1.2 Purpose

Scania has trucks and buses running all around the globe. They sell customized trucks as per the requirement of the customer as well. These trucks will be operated at different surface conditions such as highways, urban areas, forests and in mining conditions. Thus information about the surface conditions of the regions where the trucks will be operated are required to build a truck. This thesis focuses on building a classification model that classifies the surface conditions into three categories based on the sensor data. This classification of road roughness type will be helpful for the company to choose/build a customized vehicle based on the road condition of the place where the vehicle will be operated. This helps in reducing the mechanical failure of parts and reduces the cost to a great amount.

1.2.1 Objectives

- One of the main objectives is to create a dataset for the model. Map the accelerometer readings with the GPS readings since they are not mapped.
- Find a suitable model for the problem through literature study.
- Build a model and evaluate their performance with the data set.

- Improve the top performing model.
- Compare performance of the best model.
- Deploy the model on Amazon Web Services.

1.2.2 Goals

- Create a structured and clean data set for training the model
- Finalize a model for implementation
- Build a basic Support Vector Machine and Neural Network model and train them with the data
- Improve the performance of the models

1.2.3 Limitations

The data for training the model has to be extracted from different sources. Since the data in each of the sources are of large size, cleaning them as per the use. This could consume more additional time than expected. The sensors might have picked some noisy GPS positions. These GPS positions have to be filtered before sending them to the HERE API. Due to time constraints, not every machine learning model will be explored. Rather two machine learning models have been selected and will be evaluated.

1.3 Literature Study

Over the years, there are many ways by which the road roughness is being calculated. Paper [1] explores the use of a system called "Pothole Patrol" where a system that uses a three-axis acceleration sensor with GPS are deployed in vehicles to find the road conditions. There are other simple methods that are being followed where mobile phones are used to collect the data using an android app which uses the phone's GPS system. [2] A mobile phone is used as a cheap solution to gather data for calculating the road roughness. In addition to these, there are various apps in which the users can take photos of the roads with potholes, cracks, etc and send it to the server where it is then classified using image processing techniques.

The most commonly used method in calculating road roughness is by finding the International Roughness Index(IRI). Research paper[11] by Yuchuan Du, Chenglong Liu, Difei Wu and Shengchuan Jiang uses Z-axis from the accelerometer readings to find the IRI. A research paper[4] by Lepine et al proposes the use of Machine Learning in finding the shocks in the acceleration data. They trained the Machine Learning algorithm on artificially generated data which are similar to the shock produced by the vehicles. Paper[5] by Azza Allouch et al explores a Machine Learning algorithm to classify the road condition into "smooth" or "pot-holed". They used Decision Tree, SVM and Naive Bayes for building the model. Christian Gorges et al[6] explored how various machine learning algorithms such as Logistic Regression, Decision Tree, Discriminant Analysis, Naive Bayes, SVM, etc works on road roughness classification. In addition to using sensors or smartphone applications to

calculate road roughness, image processing techniques have been used as well. Arun Mohan and Sumathi Poobal[7] explores the usage of Image processing techniques in detecting the cracks on the surface. Research[8] by Teron Nguyen et all uses different methods such as Model-based methods(Kalman Filter and Sliding mode observer), Machine Learning techniques(SVM and Neural network) and Transfer function.

2

Theory

This section explains the theory behind the various concepts that are related to this thesis. The concepts of International Roughness Index, Machine Learning, Support Vector Machines and Neural Networks are explained below.

2.1 International Roughness Index (IRI)

IRI is the measure of the condition of the surface in the direction in which the vehicle is traveling. The World Bank worked on converting the raw data obtained from different countries and developed International Roughness Index as a method of measuring the surface roughness in the 1980s. They simulated a model called the quarter car system to get the response of the vehicle. This quarter car system is simulated at a speed of 80km/h. This quarter model has two parts: sprung mass which comprises a body of the vehicle and unsprung mass which comprises of a tire and the suspension. The figure below shows the quarter car model2.1.

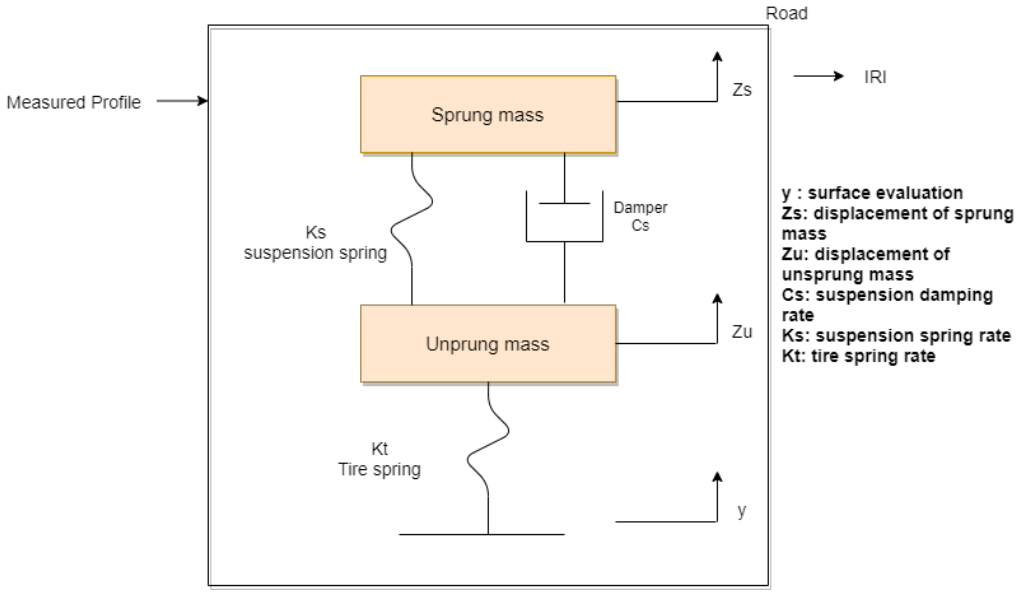

Figure 2.1: Illustration of quarter car model.[11]

The roughness of the surface causes vibration in the model which results in vertical acceleration [11]. The following formula was used for calculating the IRI where L is

the length of the section, Z is the acceleration.

$$\mathrm{IRI} = \frac{1}{L}\int_0^L |Z_s - Z_u|\, dx \tag{2.1}$$

The IRI threshold limit is not constant all over the world. It varies from country to country. The IRI threshold depends upon the length of the road, speed limit, surface type, etc. The below figure 2.2 shows different surfaces/conditions and their IRI[10]. From the figure, it is easy to interpret that among all the factors, speed is the most important factor that affects the IRI. There are many ways by which the measurements have been done over the years. At first, the rod and level survey technique has been used. Then dipstick profiler has been used. As technology grew, measurement has been done using image analysis techniques, sensors are being fitted into the vehicles to measure the road profiles. At Scania, sensors are fitted in the trucks to accumulate the readings.

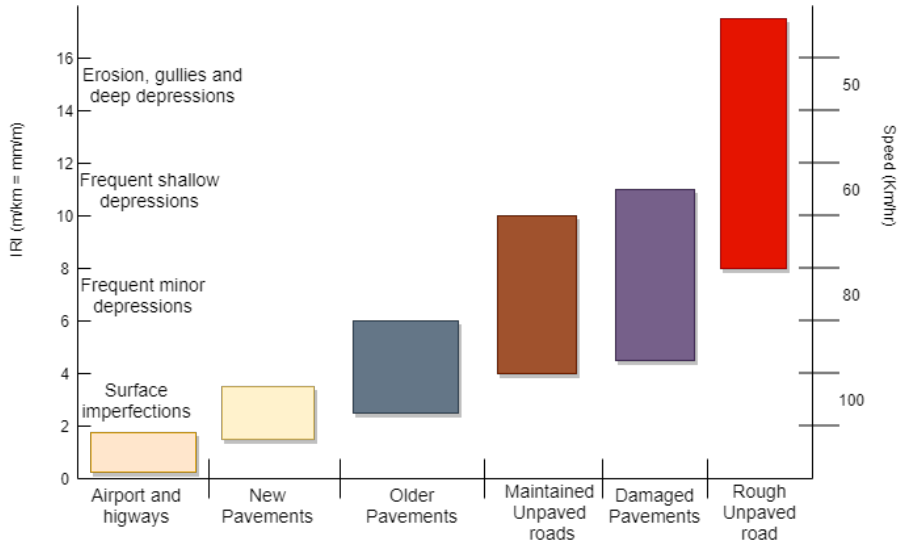

Figure 2.2: IRI Scale.[18]

2.2 Machine Learning

Machine Learning is a sub-field of Artificial intelligence. In simple terms, machine learning could be explained as the ability of the computers to learn and work with the data given to them, without any intervention from humans. The working of machine learning can be comprised in 5 steps - preparing the data set and split them into a training set and a test set, creating a machine learning model, training the model with the created training set, test the performance of the model on the test data set and finally improve the performance of the model. Of all, creating a proper data set for the model is the most important step in machine learning. Talking about the data, the data can be of many forms - images, text, numbers, etc. Building a machine learning model depends upon the data at hand and the goal to be achieved. Machine Learning is being used all around the world in many industries like Retail, Finance, Health Care, etc. Most of these industries collect real-time

data periodically and improve their machine learning model. Any company which has access to large data and a good machine learning model has an edge over their competitors.

Machine learning in turn is really broad. This is further divided into three categories- Supervised Learning, Unsupervised Learning and Reinforcement Learning. Supervised learning models are trained with labels. Unsupervised learning models are used when the labels aren't available for the data. Unsupervised machine learning methods try to decode the structure behind the features and tries to extract labels from them. Unsupervised learning is more about detecting a pattern from the features. These kinds of algorithms are largely used in spam detection, fraud detection, etc. Reinforcement learning is a method of training the model to achieve goal based on the reward and punishment concept. In reinforcement learning, training data isn't used, rather the model learns by trial and error. A reward is given to the model whenever it makes the right move such that the model learns which is the best move. This is repeated until the end goal is reached with the maximum possible reward points. This thesis works involves data with labels available. In this project supervised learning has been used.

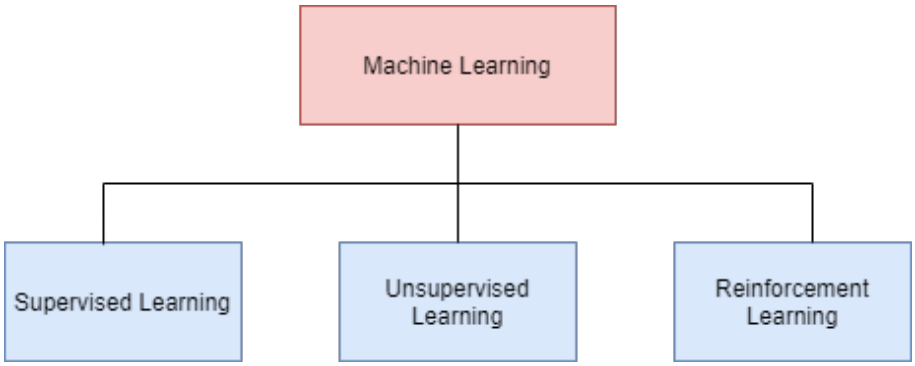

Figure 2.3: Image showing different categories of machine learning.

2.2.1 Supervised Learning

Supervised learning is a subcategory of Artificial Intelligence in which the model is trained with labels to predict or classify the outcomes. The model will try to find the relation between the features and the labels and then it will predict the outcomes for the new features which the model has never seen before. Supervised learning is categorized into classification and regression. Classification problems are used to predict which class the data belongs to. The classification problem could be binary or multi-class classification. Regression problems predict the outcome as a real value. In regression problems model tries to find a relation between independent and dependent variables. Since this thesis work focuses on road roughness classification, it is categorized under a multi-class classification problem. The methods that have been explored in the thesis work are Support Vector Machine and Neural Networks.

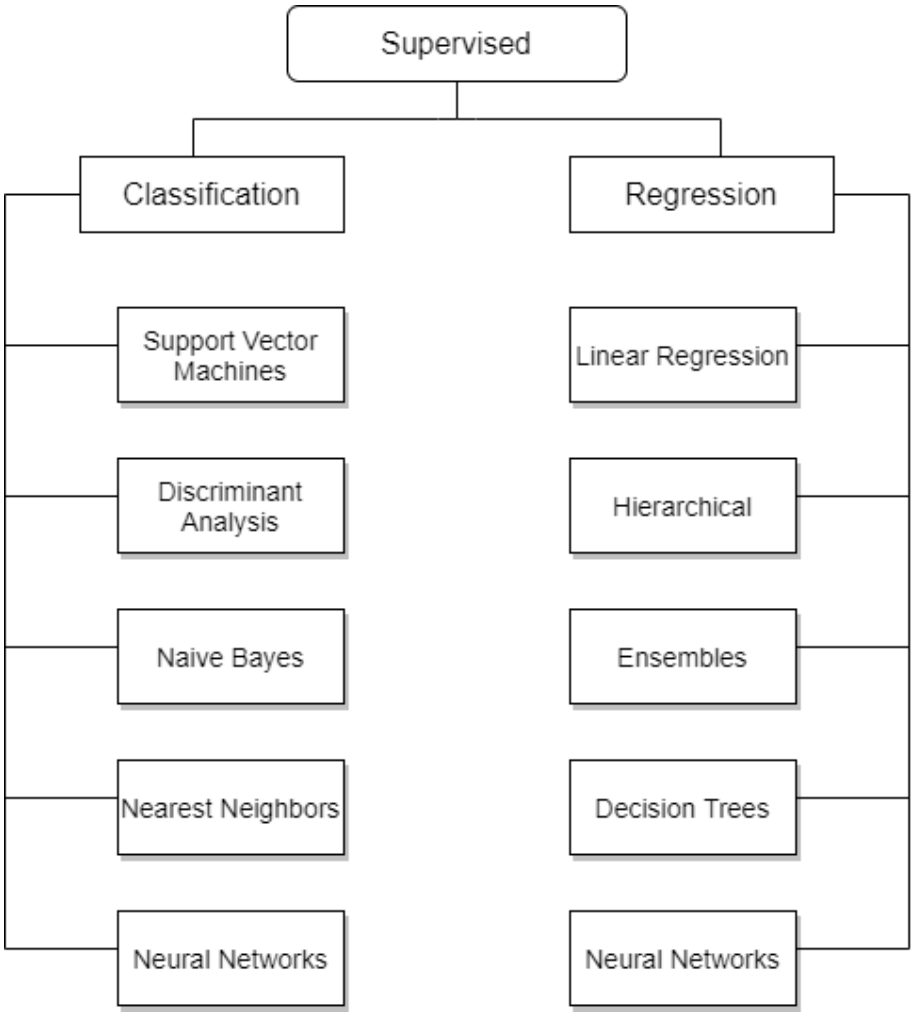

Figure 2.4: Image showing subcategories of supervised learning.

2.2.1.1 Support Vector Machine

Support Vector Machine is one of the widely used supervised machine learning algorithms [16]. Support vector machines can be used for both classification and regression problems. In simple terms, linearly separable data is data that can be separated into their two classes (binary classification) with a line. There will be cases when it isn't possible to separate the data into their classes. Such problems are called non-linearly separable problems. In such cases, the Support vector machine is a really useful algorithm. Support vector machine moves the data points into a higher dimensional plane. Support vector machine works by finding a hyperplane that separates the data points.

A proper hyperplane separates both the classes with a maximum distance between the closes data points of both the classes (binary in this case). In binary classification, the hyperplane is just a simple line. When dealing with more than two features, the hyperplane becomes a two-dimensional plane in 3d by projecting them to a higher dimension. The distance between the data points and the hyperplane is called a margin. In the figure below, a hyperplane could be seen that separates

both the data points with a maximum margin [2.5].

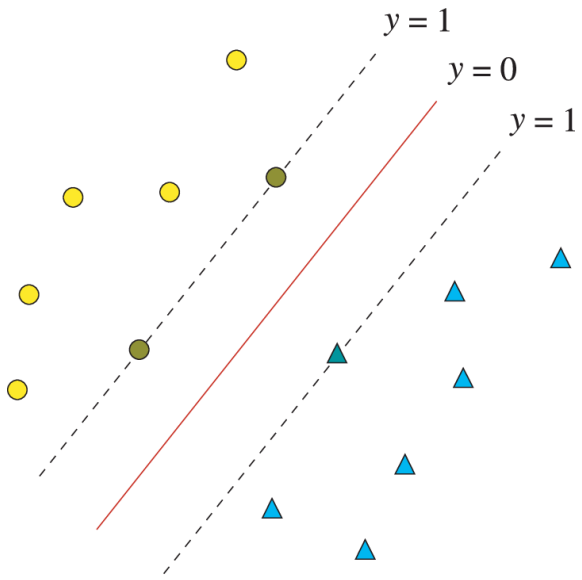

Figure 2.5: Image showing hyper plane with maximum margin.[12]

A linearly separable problem with a support vector machine looks like the image above. In the case of non-linearly separable problems, the data points are moved to higher dimensional space to separate them. This can be achieved with the help of the kernels. Kernel's function is to map the non-linearly separable data into a higher dimension so that they can be separated with a plane. This is called the kernel trick. The figure below [2.6] shows the kernel trick. In two dimensional plane, the data is cannot be separated by a line but projecting them to a higher dimension, a plane can be used now to separate them.

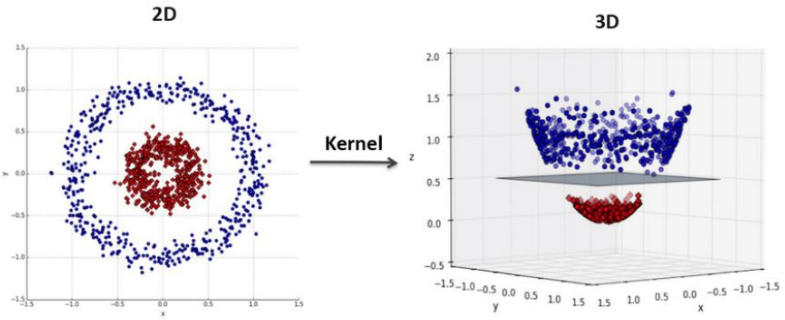

Figure 2.6: 2-D to higher dimensional mapping using kernel trick.[13]

Considering a two-dimensional problem, this can be linearly separated by a line if the data are linearly separable. The line is given by a function

$$y = ax + b \tag{2.2}$$

Replacing x with x1 and x2

$$ax1 - x2 + b = 0 \tag{2.3}$$

Further defining x1 and x2 as x and w as (a,-1) from the equation 2.3 will give the equation for the hyperplane 2.4.

$$wx + b = 0 \tag{2.4}$$

With the hyperplane, the hypothesis function is defined as

$$h(xi) = \begin{cases} +1 & \text{if wx+b} \geq 0 \\ -1 & \text{if wx+b} < 0 \end{cases} \tag{2.5}$$

The point the lies on or above the hyperplane will be classified as +1 and anything that lies beloew the hyperplane will be classifier as -1.

2.2.1.1.1 Parameters : There will be cases where data overlaps each other, in that case, even projecting them to a higher dimensional space would still be difficult to separate them. Drawing a straight plane might lead to misclassification of two or three points. In such case, regularization parameter comes into play.

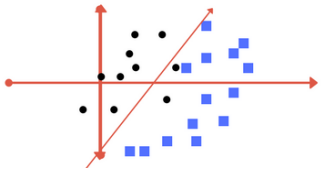

Figure 2.7: Low regularization parameter

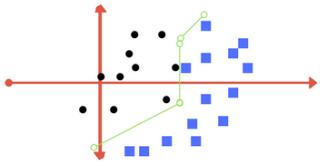

Figure 2.8: High regularization parameter

Setting regularization parameter to a smaller value makes the hyper plane margin large. This large margin is prone to outliers [2.7]. On setting the regularization parameter to a large value, the hyper plane margin is small and it works better in classifying all the data points correctly [2.8]. But changing the regularization parameter to higher value to classify all the data points in the training set would be an expensive operation.

Another parameter in SVM is gamma which gives weightage to the curvature of the decision boundary. Setting gamma to a lower value, even the distant data points from the hyper plane are considered and with higher gamma value, only the closest data points to the hyper plane are considered.

2.2.1.1.2 Kernels : In SVM, kernels are set of mathematical functions which takes in the data as the input and transforms them as required. There are many kernels within the SVM but the most commonly used kernels are Polynomial kernel, Gaussian kernel, Gaussian radial basis function, Sigmoid kernel and Linear kernel.

Linear kernel is one of the common kernel in SVM. Linear kernel is used when the data is linearly separable and when there are many number of features. And linear kernel is the fastest kernel of all. Here X_i is the input data point and X_j is the support vector and c is the constant which is optional.

$$F(X_i, X_j) = (X_i^T X_j) + c \tag{2.6}$$

Polynomial Kernel is used for linearly separable data as well but it is not as accurate as Linear kernel. Polynomial Kernel is expressed as

$$F(X_i, X_j) = (X_i^T X_j + C)^d \tag{2.7}$$

Gaussian kernel or radial basis function kernel is for non-linearly separable data and when there isn't enough knowledge on the data. This works well on separating non-linear data.

$$F(X_i, X_j) = exp(-\frac{||X_i - X_j||^2}{2\sigma^2}) \tag{2.8}$$

σ here is the variance, X_i and X_j are the euclidean distance between the data points. Another equivalent function to the above equation is given as

$$F(X_i, X_j) = exp(-\gamma||(X_i - X_j||^2) \tag{2.9}$$

Here, γ is equal to $\frac{1}{(2\ \sigma\ )^2}$.

Sigmoid kernel is another type of kernel which has originated from the neural networks. It is given by the formula

$$F(X_i, X_j) = tanh(\gamma(X_i^T X_j) + C) \tag{2.10}$$

Higher values of gamma and C gives more precise results but they are computationally expensive.The different parameters are tuned and kernels are used as per the problem and the data set.

2.2.1.2 Neural Networks

Neural Networks are a subgroup of Machine Learning and they are modeled and named after the neural structure of the human brain [17]. They try to mimic the neurons in the human brain, how they send signals to each other. First, an artificial neuron called Perceptron has been designed. Perceptron works by taking in several inputs and produces a single binary output like 0 or 1. The output of the perceptron is decided based on the weighted sum. Each input has a weight and the output depends on the sum of input and the weight. A threshold is set and if the weighted sum is greater than the threshold, it gives one of the outputs and if the sum is less than the threshold, it gives another output.

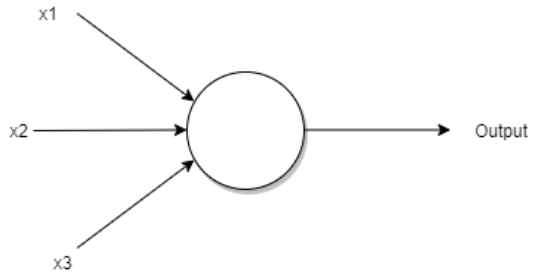

Figure 2.9: A simple perceptron model which has 3 inputs.

Neural networks consists of layers of these perceptrons. A basic neural network would contain an input layer, a hidden layer and an output layer. Each of these nodes connects with each other node in the subsequent layer and has weights and threshold associated with them.

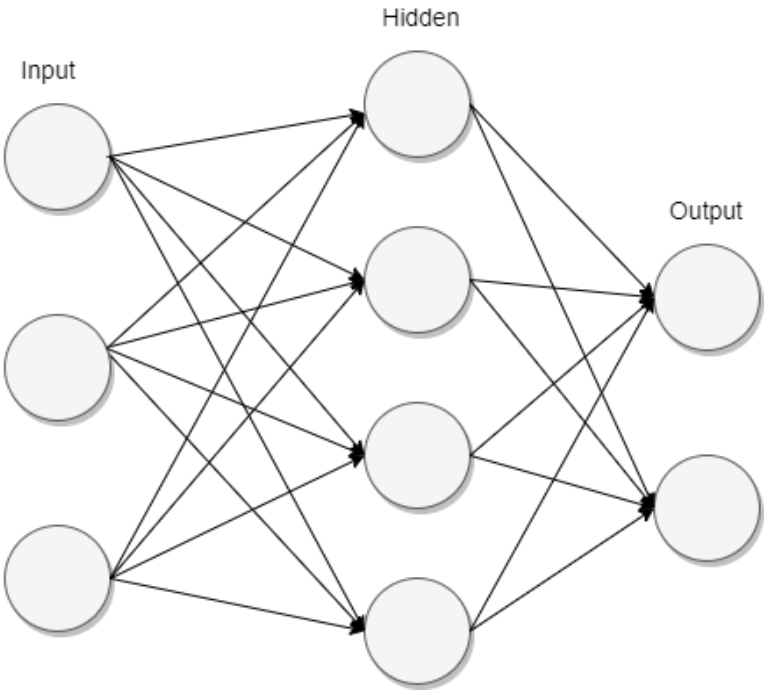

Figure 2.10: A Neural Network model with one input, hidden and an output layer.

The most used neural network algorithm is the one with backpropagation. In a neural network, when training the model, the weights are assigned randomly at first. Then the network uses the training data with the initial weights and produces the output. The error at the output is propagated back into the model and the weights are tweaked. This is continued until the error is minimum. The functioning of neural networks could be explained by the formula

$$f = f(W^T X + b) \tag{2.11}$$

Here, W is the weight, b is the bias, and f is the activation function.

There are many different types of neural networks like Multi-layer Perceptron, Convolutional neural networks, recurrent neural networks, Generative Adversarial networks, etc. Multi-layer perceptron is bi-directional. It has both forward and backward propagation. The weights are adjusted due to backpropagation and the error could be reduced. CNN contains additional layers called convolutional layers. CNN is widely used for image data. Recurrent Neural networks are another type of neural network in which a layer sends back the output of the layer to make the prediction better. RNN is used for text data, music generation, etc.

2.2.1.2.1 Multi-Layer Preceptrons : Multi-layer perceptron is a deep neural network. A multi-layer perceptron is a stack of multiple perceptrons which forms

a layer. A basic multi-layer perceptron consists of an input layer, a hidden layer and an output layer. Multi-layer perceptrons are mostly used for supervised learning problems. Multi-layer perceptron can be used for both binary and multi-class classification.

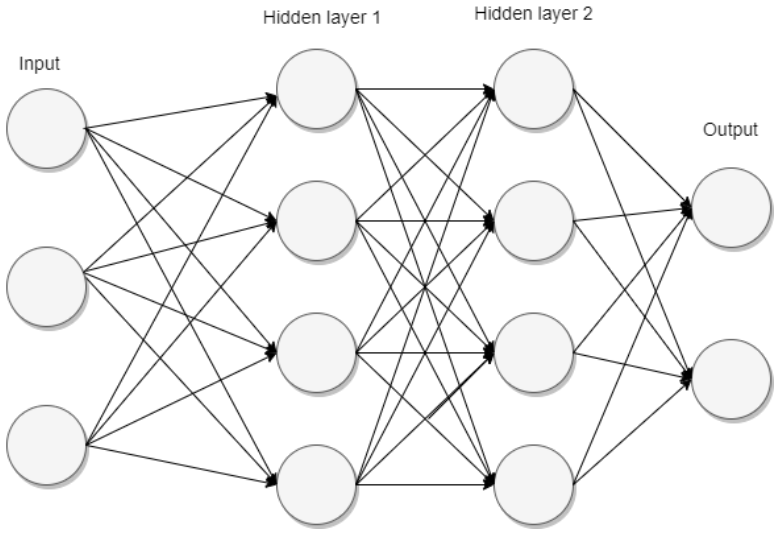

Figure 2.11: A Multi-layer perceptron with two hidden layers.

In a multi-layer perceptron, the inputs are sent through the layers by multiplying the weights between the layers. Then the output is calculated with the activation function associated with the hidden layers and it passed on to the next hidden layer if any and the same steps are repeated and finally sent to the output layer. The activation function calculates the weighted sum and adds it to the bias and determines the output. Many activation functions are used for the layer. The most commonly used are Rectified Linear Activation, Sigmoid activation and Hyperbolic Tangent activation.

Rectified Linear Activation function is one of the most basic and widely used activation function. ReLU functions returns the input value as the output if the value is positive or returns zero if the value is negative. ReLU is given by the formula,

$$f(x) = max(0, x) \tag{2.12}$$

Sigmoid activation function is also called as logistic function. It outputs the value in between the range of 0 and 1. Higher the value, the output is close to 1 and lower the value, the output is close to 0. Sigmoid function is given by,

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.13}$$

Tanh activation function is similar to sigmoid activation function, but the output range is in between -1 and 1.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.14}$$

In the output layer, backpropagation is carried out. Backpropogation is a technique to optimize the weight to reduce the loss. Since the weights are randomly assigned, the output for the initial random weight would be different from the expected weight. This error rate is reduced through backpropagation. This is the working of a basic multi-layer perceptron.

3
Methods

This chapter explores the roughness profiles of selected vehicles. It also describes the machine learning model chosen for this purpose and its architecture. The pre-processing and how the data set is created will be explained in detail. It further explores a detailed explanation of the machine learning models used and the hyper-parameter tuning to improve the model performance etc.

3.1 Platform

The programming language used for the thesis is Python. We chose Python as the main programming language since it provides extensive support for implementation of machine learning algorithms. Hive has been used to extract and transform data on hadoop based datalake. PySpark has been used to load the data into the data lake server from the data warehouse. Sci-kit learning and Keras libraries have been used to build the machine learning model. The code was run on the JupyterLab environment on Scania's data lake.

3.2 Workflow

The main goal is to build a machine learning model for multi-class classification. The workflow of the thesis is given below. At first, a literature study is done to explore the related work done regarding the road roughness classification which is explained in the section 1.3. The next step is to summarize the result from the literature study to decide on the way of approach and which model to choose for classification. Exploring the data deals with exploring the different data that are required for this thesis work and devise methods for cleaning them. Then the large unstructured data is cleaned into clean data which would be used for the model. The next step is to create a script to send the position data to the HERE API to get data. The data obtained from the HERE API is used as labels for the dataset. The features are extracted and passed on to the models like Support Vector Machine and Multi-layer perceptron for training and then finally tested the model performance. Then the performance of the models are compared and the results are provided. A detailed explanation of the workflow is presented in the later sections.

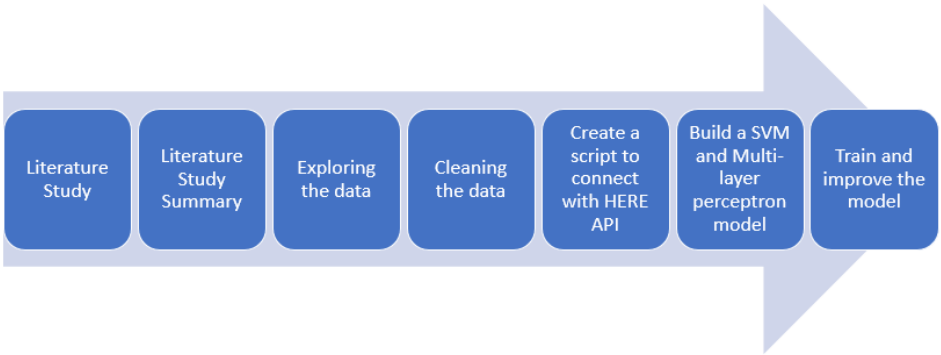

Figure 3.1: Workflow overview.

3.3 Literature review summary

A number of methods over the years have been suggested and used to classify road roughness. The way the data is collected also differs a lot from study to study. For the current problem, the data is collected through the onboard sensors. Since the current problem focuses on multi-class classification, we approach the problem by using machine learning. Various machine learning algorithms have been studied. After extensive review, Support Vector Machine has been selected as one of the models. Neural Network hasn't been explored much because of the requirement of a large proper dataset. Since we have sufficient data, Multi-layer perceptron has been chosen as the other model. As of considering the dataset, the accelerometer on the z-axis is taken.

3.4 Dataset

This section gives a detailed insight into the dataset that has been used for training the model. The below picture 3.2 shows the main dataset that has been used. A table from the Operational Data Database (ODDP) consists of the accelerometer readings, speed, chassis number where all the accelerometer readings are updated periodically and their respective speed. The Fleet Management Database (FMDB) contains the GPS positions of the vehicles and external equipment reference numbers. The Mapping Database (MDB) contains an unique identifiers 1 and 2 and chassis number. The accelerometer readings are to be mapped with the GPS positions. Hence a table inside MDB is used for mapping. The GPS positions are frequently updated whereas the accelerometer readings are updated based on an interval. The accelerometer readings are stored in the form of a rain flow matrix.

Figure 3.2: List of variables in each database.

3.5 Data pre-processing

Data pre-processing is one of the major part with this thesis. Since the data wasn't readily available or can't be extracted from a single source, significant amount of processing has to be done to make the data available for analysis. This section explores detailed pre-processing steps that has been performed in the thesis.

3.5.1 Data extraction and mapping

The values table from the ODDB database, contains not only the accelerometer readings and speed data, but it also contains every other readings that are sent from the sensors. The first step is to extract, accelerometer readings on Z-axis and speed data from this table. Hive queries have been used for all the data extraction processes. The next step is to extract the GPS positions, time and date from the 'position' table in FMDB database. The unique identifiers 1 and 2 are extracted from another data source . This table is used for mapping the accelerometer readings with the GPS positions.

Mapping accelerometer readings with positions, no common columns are present in both the table which could be used. Therefore, a table from another data source has to be used for mapping. The table from MDB has columns 'unique identifier 1' and 'unique identifier 2'. Two different unique identifiers from two different data sources were then mapped. Since the positions are updated frequently, it has to be pre-processed so that it can be mapped with accelerometer readings which are updated on some time interval. The accelerometer import data is mapped with the snapshot import date and the dates which match with each other are taken and mapped

together. GPS positions on dates between two accelerometer reading's import date have been taken as well to get a description of the route. These GPS position's respective accelerometer readings are calculated by finding the difference between the two rain flow matrices. The speed values are calculated in the similar way, for these new accelerometer readings, the average speed is taken. There are some noise in the speed recorded by the sensors, those values are removed and realistic values that are less than 120km/hr are stored.

Unique identifier 1	Unique identifier 2	Accelerometer readings (4x34 rainflow matrix)	Date	Speed	GPS Positions
Xxxx	yyyy	Matrix 1	21-05-2021	72	Position 1
Xxxx	yyyy	Matrix 2	25-05-2021	68	Position 2
Xxxx	yyyy	Matrix 3	28-05-2021	78	Position 3
Xxxx	yyyy	Matrix 2 - Matrix 1	22-05-2021 to 24-05-2021	64	Position 4
Xxxx	yyyy	Matrix 3 – Matrix 2	26-05-2021 to 27-05-2021	100	Position 5

Figure 3.3: Example representation of table after mapping for one of the unique identifier 1.

The table 3.3 above shows the example representation of how the data looks after the mapping. In this example, the first three rows contain the accelerometer readings and the last two rows contain the difference of the accelerometer readings. Since the ODDP table contains all the variables recorded by the on-board sensors, navigating through them to get the accelerometer readings and speed based on few conditions using queries took a significant amount of time.

3.5.1.1 Calculating Pseudo-damage

The next step is calculating the pseudo-damage values from the accelerometer readings. Pseudo-damage is the damage that a vehicle accumulates on the travel through the surface. It is calculated per distance driven to the road stretches. The pseudo-damage values differ by a great number between different roads and different vehicles. The pseudo-damage is calculated as

$$d = \sum n_q * a_q^{\beta} \tag{3.1}$$

Here, a is the sum of all the rows in the accelerometer rainflow matrix. β is the fatigue exponent which is a constant for all the accelerometer readings. n is the range of values for each column. This pseudo-damage is calculated for every accelerometer values and stored in the dataframe. This value is used as one of the features for the model.

3.5.1.2 Extracting the labels

Now that the two features speed and pseudo-damage which will be used for training the model have been mapped and added to a dataframe, the next step is to get the labels for the dataset. The label for the data is the roughness values for the road stretch. The pseudo-damage corresponds to the condition of the road. Higher pseudo-damage corresponds to relatively poor road and lower pseudo-damage corresponds to the relatively good road.

HERE API is used as the source for extracting the labels for the data. The positions for each accelerometer readings are passed onto the HERE API and respective roughness values are obtained. The GPS positions and other parameters are passed to the HERE API and it returns the roughness values as a result. The parameters that have been used has been stated below

```
apikey = "APIKEY"
mode = "fastest; truck; traffic:disabled"
attributes = "ROAD ROUGHNESS FCn(*)"
waypoints 0 to N
```

Apart from the API key, the mode can be set accordingly. Since the mode of transport is truck, the truck has been passed with other values like disabled traffic and fastest route. The road roughness function attribute returns all the values regarding the roughness like roughness value, linkID, IRI, etc. Waypoints take in starting waypoint, destination waypoint and set of waypoints between them to get the values of the entire route.

The URL request can handle only up to 150 positions in a single request. Therefore the positions are sent in a batch of 150 positions. The sensors picks up some noisy positions as well which are irrelevant to the region. These positions are removed one by one and are request is sent back again until all the noisy positions are removed. Then the final set is passed into the URL and the HERE API returns the roughness values of the road stretch. The roughness values returned from the HERE API are taken and appended to a list and the average is taken. This average roughness is mapped to the respective accelerometer reading. Now that the roughness values are added to the data, it has to be categorized into three. The IRI index is not consistent all over the world. It changes with country to country. For the thesis, values between 0-2.5 are categorized as good road or 1. values between 2.5-4 are categorized as fair road or 2. Any values equal to and above 4 are categorized as poor road or 3.

3.5.2 Data distribution

The dataset isn't balanced. There are very few values with roughness category 3. One of the reasons could be that most of the dataset has EU countries where quality of the roads are not poor in general. And the other reasons could be that lately, lot of rough roads are reconstructed into good roads and since the stretch of a road

typically spans over several days, average roughness is taken for the stretch and it would fall under fair category. The distribution of three roughness categories is given in the figure below 3.4.

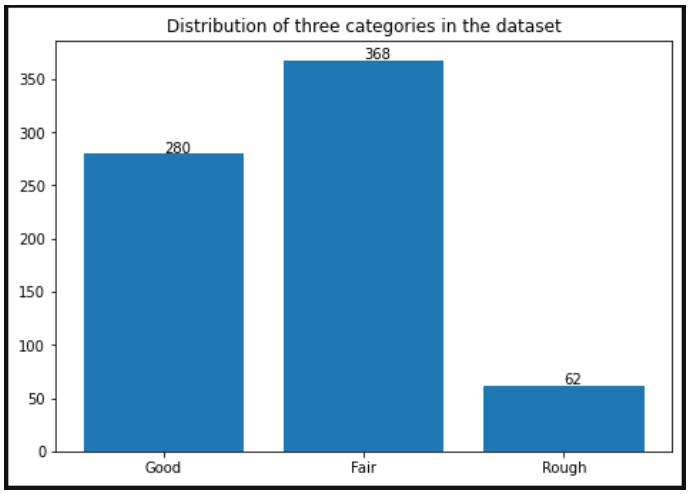

Figure 3.4: Distribution of good, fair and rough roads in the dataset.

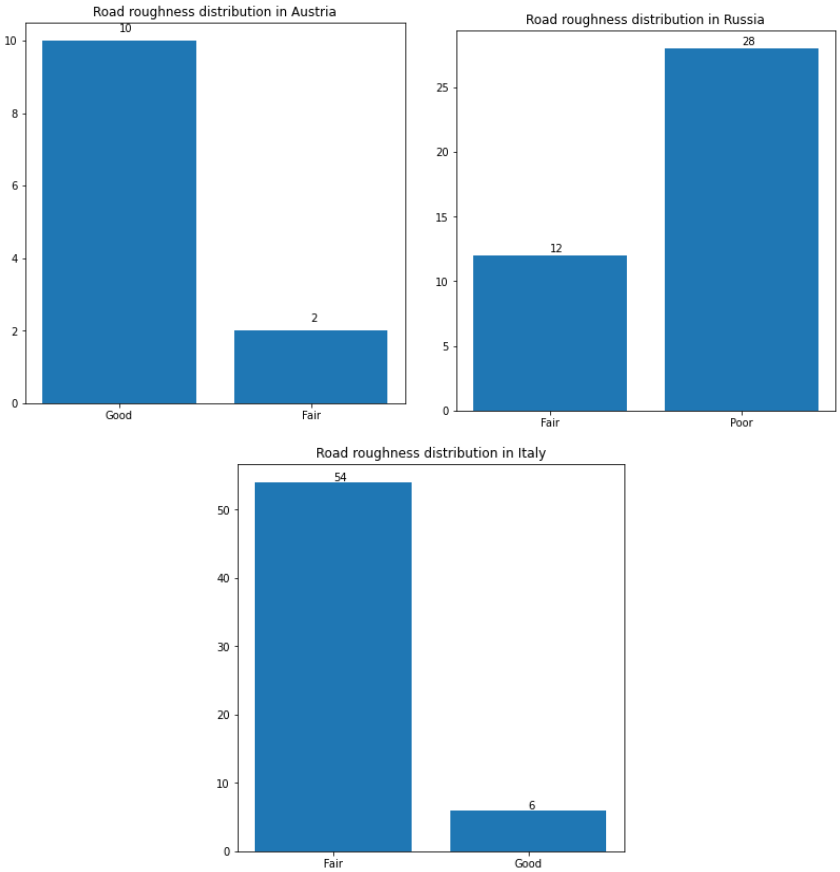

Figure 3.5: Road roughness in Austria, Russia and Italy

The above set of figures (3.5) contains the road roughness in 3 different countries,

namely, Austria, Italy and Russia. From the plots, in Austria most of the road stretches are good road and very few are fair road and none with bad roads. In Italy, most of them are fair and in Russia, most of the roads falls under rough category and few in fair category and none in the good category.

Table 3.1: Road roughness conditions of other countries in the dataset.

Country	Good	Fair	Bad
Brazil	0	74	3
Germany	47	12	0
Bosnia and Herzegovina	3	0	0
New Zealand	0	9	0
Spain	55	2	0
Malaysia	0	15	6
Taiwan	0	11	2
Czech	24	8	0
Hungary	14	0	0
United Kingdom	1	83	7
Sweden	23	0	0
Netherlands	0	7	0
France	19	2	0
Chile	0	13	0
Finland	19	0	0
Poland	19	17	8
Belgium	0	7	0

The table 3.1 above shows the roughness values of the road stretches of the countries in the dataset. There are only few countries with bad road conditions while the rest falls under either good or fair road.

3.6 Multi-layer perceptron

3.6.1 Network architecture

The network for the problem used is a multi-layer perceptron model. Keras is used for building a neural network model. A simple neural network model with one input layer, two hidden layers and an output layer is built. The layers are built sequentially which will be mentioned when creating the network. The dimension to the input layer is given as 2 since there are 2 features. The first hidden layer has 16 nodes, the second hidden layer has 12 nodes. Third layer is the output layer with dimension 3 since there are three classes. The activation function, optimizer selected has been explained in detail in the experiment section. The network summary of the model is given below 3.6.

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_59 (Dense)             (None, 16)                48
_________________________________________________________________
dense_60 (Dense)             (None, 12)                204
_________________________________________________________________
dense_61 (Dense)             (None, 3)                 39
=================================================================
Total params: 291
Trainable params: 291
Non-trainable params: 0
```

Figure 3.6: Summary of the architecture.

Another architecture that has been used is a MLP with three hidden layers. The first hidden layer with 32 nodes, second hidden layer with 16 nodes and third hidden layer with 12 nodes. The network summary of this architecture is given below 3.7

```
Model: "sequential_96"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_326 (Dense)            (None, 32)                96
_________________________________________________________________
dense_327 (Dense)            (None, 16)                528
_________________________________________________________________
dense_328 (Dense)            (None, 12)                204
_________________________________________________________________
dense_329 (Dense)            (None, 3)                 39
=================================================================
Total params: 867
Trainable params: 867
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.7: Summary of the architecture.

3.6.2 Experimental Setup

This section explores the experimental setup for multi-layer perceptron. The speed and pseudo-damage values has been taken as features and the categories based on the IRI values has been taken as the label. The features and labels has to be transformed first before sending them into the model. The features are normalized first. Two different normalization methods are used and the best one is selected. First, Min-Max scaler transformation is used. Min-Max scaler transforms the values

into the range of 0 and 1. The Min-Max scaler transformation is given by

$$X.std = \frac{X - X.min}{X.max - X.min} \tag{3.2}$$

Standard scaler is the another transformation that has been used. Standard scaler transforms the data such that it has mean value 0 and standard deviation 1. On using both the transformations, standard scaler produces better result and standard scaler has been chosen for normalizing the data. The target labels are 1-good, 2-fair and 3-poor. These values are one-hot encoded before passing them into the model. One-hot encoding converts the categorical values into binary vectors. The labels are encoded as follows

1 - 1 0 0
2 - 0 1 0
3 - 0 0 1

The data is split into training and test set. 20% of the data is used as test set and validation set. Stratify is used in test train split because this keeps the same percentage of three classes in the test set as in the training set. This is to avoid further class imbalance. Now the network architecture is experimented with different activation functions in each of the layers. The activation functions in the hidden layers are experimented with Leaky ReLU, Tanh and ReLU whose accuracy is given in the table 3.2.

Activation functions	Accuracy
ReLU	68%
Tanh	64%
Leaky ReLU	63%

Table 3.2: Accuracies for different activation functions in the hidden layer

The ReLU activation function produces better accuracy when compared to the other activation functions. With ReLU activation function in the 2 hidden layer, the output layer's activation function is experimented with Softmax and Sigmoid activation function. Softmax works better than the sigmoid activation function. Sigmoid works well for binary classification but since the problem deals with multi-class classification, softmax works better. The categorical cross-entropy loss function is used since it works better for multi-class classification problems. It is also called as softmax loss. Stochastic Gradient Descent and Adam optimizer are tried with the network architecture. Finally, Adam optimizer is used which has the combination of best features from RMSProp and AdaGrad and provides optimization to handle the sparse gradients.

The final set of parameters that has been chosen is given in the table 3.3. The data is trained on the model with these parameters.

Hidden layer 1	ReLU
Hidden layer 2	ReLU
Output layer	Softmax
Loss function	Categorical cross-entropy
Optimizer	Adam

Table 3.3: Parameters for the multi-layer perceptron model

3.6.3 Evaluation

For the classifier above, it is evaluated with few evaluation metrics like confusion matrix, accuracy, recall, precision, F1 score and comparing the loss. These evaluation techniques helps to get to a conclusion about the performance of the model.

Confusion matrix is an evaluation metric for classification problems. It compares the actual target labels with the predicted labels of the model. Confusion matrix has 4 main terminology namely, 'True Positive' which means that the model has predicted positive and it is true, 'True Negative' in which the model has predicted negative and is true, 'False Positive' in which the model has predicted false and it isn't true and 'False Negative' in which the model has predicted negative and it is false. The accuracy, precision, recall and F1 score is given in the equations (3.3),(3.4),(3.7) and (3.6).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.3}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.4}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.5}$$

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.6}$$

Since the problem deals with class imbalance and multiple classes, Cohen's Kappa score has been chosen as one of the evaluation metrics. The `P_o` is the observed agreement and `P_e` is the expected agreement.

$$K = \frac{P_o - P_e}{1 - P_e} \tag{3.7}$$

3.7 Support Vector Machines

The simple SVM supports only binary classification. In order to use it for multi-class classification, One vs One strategy is to be followed. The One vs One splits

the classification into one binary classification for each pair of classes. In such case, the classifier uses classifier of number $\frac{N(N-1)}{2}$.

3.7.1 Experiment

The input features are transformed first. As in the MLP, standard scaler transformation has been used. The 80% of the data as the training set and rest as the test set and stratification has been used. The performance of different SVM kernels has been evaluated. Linear kernel, RBF kernel, Polynomial kernel and sigmoid kernel has been used. The data is trained on the all four kernels with their default parameters and the accuracy is obtained as in the table 3.4.

Kernel	Accuracy
Linear	50.34%
Polynomial	51.03%
RBF	60.68%
Sigmoid	46.89%

Table 3.4: Accuracies of different kernels with the default paremeter

From experimenting with different kernels, RBF kernels produces better accuracy when compared to the other kernels. Now with RBF kernels, the Gamma and C values are to be tuned to find the best fit. Grid search has been used to find the best gamma and C values. The list of parameters that has been used in the grid search is given below 3.5. From this, C = 1000 and gamma = 5 are found to be the best parameters. Increasing the values of gamma and C will give more precise classification but the chances of overfitting is high.

C	1000, 10000, 100000, 5000, 50000
Gamma	0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1, 1, 5, 10

Table 3.5: Parameters used in Grid Search

3.7.2 Evaluation

The evaluation metrics used for SVM is similar to the one used for multi-layer perceptron. In addition to that, Area Under Curve(AUC) - Receiver Operating Characteristics (ROC) curve. This is used to estimate how much the model is able to distinguish between the classes at different thresholds. Higher the AUC curve, the model is better at predicting. The figure3.8 shows the AUC-ROC curve. TPR is the True Positive Rate and FPR is the False Positive Rate.

$$TruePositiveRate = \frac{TP}{TP + FP} \tag{3.8}$$

$$FalsePositiveRate = \frac{FP}{TN + FP} \tag{3.9}$$

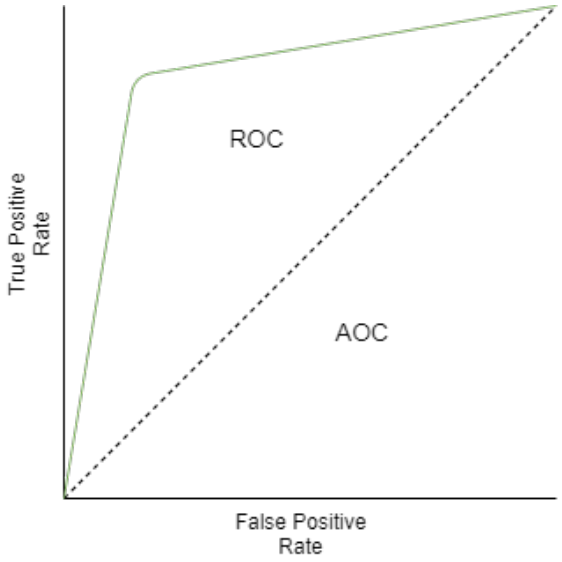

Figure 3.8: AUC-ROC curve.

For multi-class classification, AUC-ROC curve uses one vs all methodology. Label 1 is classified vs label 2 and 3, label 2 is classified vs label 3 and 1 and label 3 against 1 and 2.

The detailed results of the multi-layer perceptron model and SVM model with the above mentioned parameters and the performance evaluation of them with the metrics mentioned has been explained in the next section.

4

Results

In this section, the results obtained for multi-layer perceptrons and SVM are analyzed. The evaluation metrics mentioned in the previous section are used to evaluate the performance of the model.

4.1 Multi-layer Perceptron

The multi-layer perceptron is experimented with two different architectures. First with a network with two hidden layer and then a network with three hidden layers. The different evaluation metrics are used for both the network architecture and the results are produced in this section.

4.1.1 Network with 2 hidden layers

As stated in the experiment section, ReLU activation function for hidden layer and softmax activation function for output layer works better than the other activation functions with categorical cross-entropy loss and adam optimizer. The hyper parameters for this model is given in the table 4.1.

Parameters	Values
Hidden layers	2
Activation function-hidden layers	ReLU
Activation function-output layer	softmax
Nodes at first hidden layer	16
Nodes at second hidden layer	8
Number of epochs	100
Batch size	16

Table 4.1: The parameters for the network

For the architecture with the above mentioned parameters, the accuracy and loss over the 100 epochs is plotted. The accuracy and loss is based on the model's performance on the validation set.

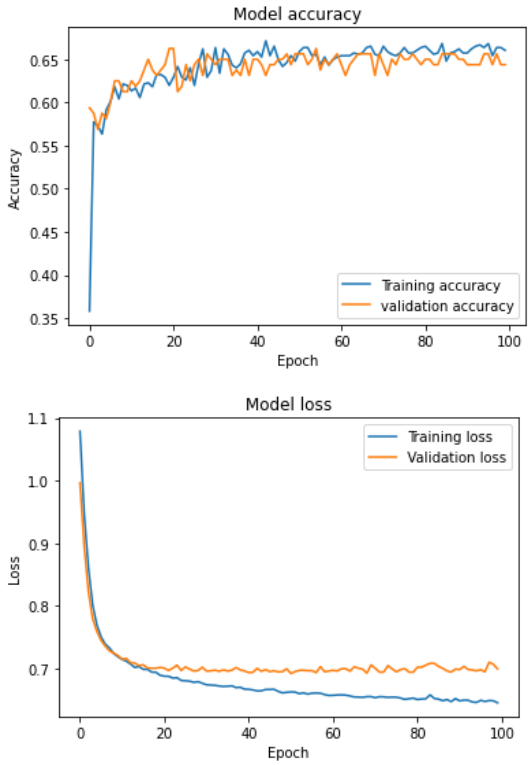

Figure 4.1: Accuracy and Loss on training set and validation set

The figures 4.1 shows the accuracy and the loss over the period of 100 epochs. For the model not to overfit or underfit, the validation loss and training loss has to be of almost same value. For the current network architecture, both the training loss and validation loss remains roughly equal and the values converges over the time. Thus the model is perfect fitting. To check how well the model has predicted the classes on a new data, confusion matrix has been used.

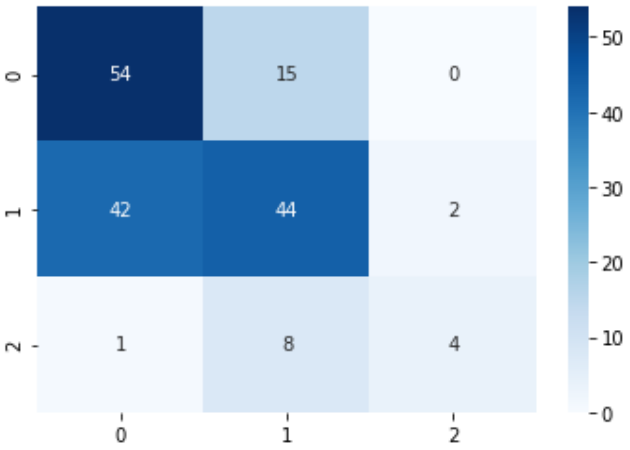

Figure 4.2: Confusion matrix of the model.

The confusion matrix in the figure 4.2 shows the prediction of the model for different classes. The model has correctly predicted 47 values of good roads as good roads, 53 fair roads as fair roads and 3 poor roads as poor roads. The classification report for the model has been taken whose values are mentioned in the table 4.2. The model has an accuracy of 64.3% on the validation set. It could be seen that the model has worked fairly well in predicting the classes. The model has worked well on predicting the classes that belongs to good and fair and poorly on predicting the class 3 as could be seen from the F1 score in the table.

Classes	Precision	Recall	F1-score
Good	0.63	0.71	0.67
Fair	0.66	0.65	0.65
Rough	0.67	0.31	0.42

Table 4.2: Classification report for the network

The other evaluation metric that has been used is Cohen Kappa score. For the current network, score of 0.37 is obtained. Kappa score in the range of 0.21 to 0.40 states that the model fairly predicts the classes [15].

4.1.2 Multi-layer perceptron with 3 hidden layers

An architecture is built with 3 hidden layers whose parameters are given in the table 4.3.

Parameters	Values
Hidden layers	3
Activation function-hidden layers	ReLU
Activation function-output layer	softmax
Nodes at first hidden layer	32
Nodes at second hidden layer	16
Nodes at third hidden layer	12
Number of epochs	100
Batch size	16

Table 4.3: The parameters for the network

The accuracy and the loss over 100 epochs is plotted as done with the previous network.

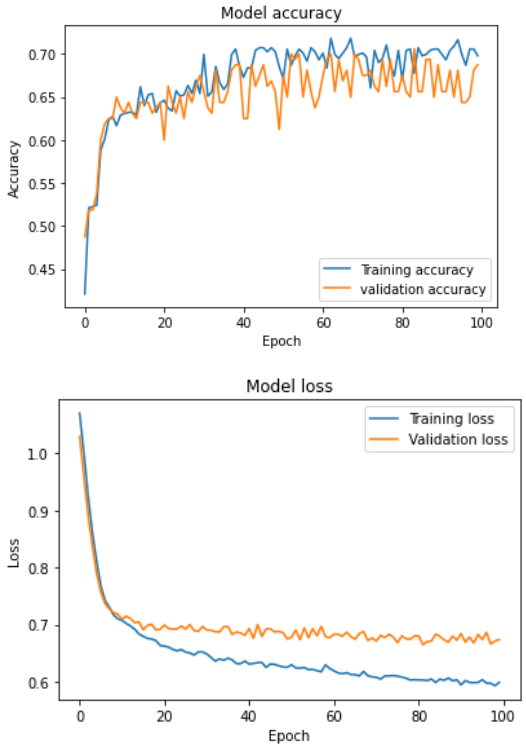

Figure 4.3: Accuracy and Loss on training set and validation set

The plots 4.3 shows the accuracy and the loss of the current network. As seen from the loss, the model doesn't overfit or underfit. As for the accuracy, the model has an accuracy of 68.75%. The confusion matrix for the network has been plotted as well to see how well the model has predicted the new classes.

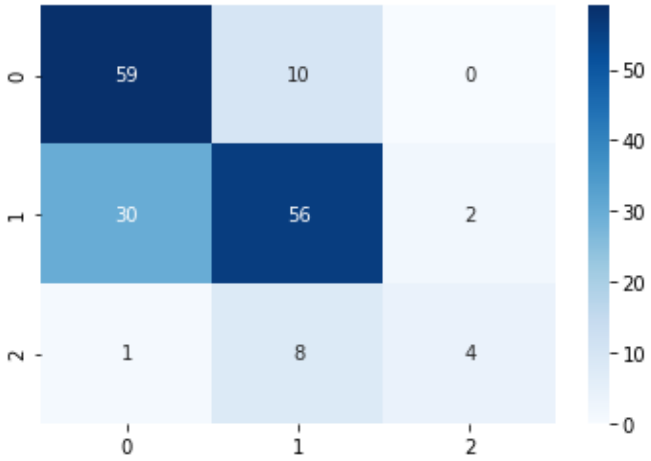

Figure 4.4: Confusion matrix of the model.

The model has predicted 58 good roads, 49 fair roads and 3 rough roads correctly. The precision, recall and f1 score obtained from the classification report has been given in the table 4.4

Classes	Precision	Recall	F1-score
Good	0.64	0.88	0.74
Fair	0.76	0.64	0.69
Rough	0.67	0.31	0.42

Table 4.4: Classification report for the network

The model works well in predicting the classes that belongs to good and fair but not for the classes that belongs to rough. The Cohen Kappa score obtained for the model is 0.43 which falls under the range of 0.41 to 0.60 which is considered as moderated. The model's performance has improved by a small notable amount when compared to the two hidden layer model.

Metrics	Two hidden network	Three hidden network
Training Loss	0.63	0.62
Validation loss	0.69	0.67
Training Accuracy	67%	70%
Validation Accuracy	64%	68%
Cohen Kappa score	0.37	0.43

Table 4.5: Metrics for two and three hidden layer network

The table 4.5 shows the accuracy, loss and Cohen Kappa score obtained for both the networks.

4.2 Support Vector Machines

The SVM is the another model whose performance is evaluated on the same dataset. For the SVM, the performance of the model with 4 kernels with default parameter is evaluated. Among them, RBF kernel produced better accuracy. The result section explores the results obtained with RBF kernel and the hyper parameters used. The metrics that has been used of MLP has been used for SVM as well. In addition to that, AUC-ROC curve has been used. The table 4.6 shows the hyper parameters used to get the results which are later explained in this section.

Parameters	Values
Kernel	RBF
C	1000
Gamma	5
Tolerance	0.001
Decision function shape	One vs One

Table 4.6: Parameters for SVM

With this parameter, the model achieved an accuracy of 72.9%. The confusion matrix is plotted to see how much the model is able to predict the classes which is given in the figure 4.5.

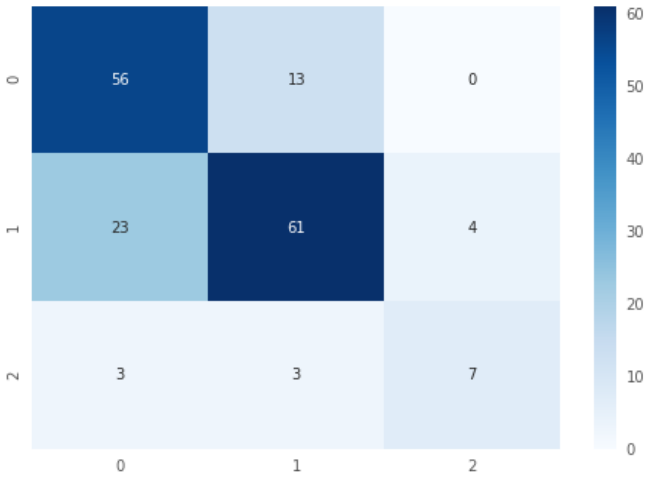

Figure 4.5: Confusion matrix of the model.

The SVM with RBF kernel is able to predict 56 classes of good roads, 61 classes of fair roads and 7 of rough roads correctly. The classification reports gives insight into the precision, recall and f1 score.

Classes	Precision	Recall	F1-score
Good	0.70	0.80	0.74
Fair	0.75	0.71	0.73
Rough	0.64	0.58	0.61

Table 4.7: Classification report for SVM

From the f1 score, the model has performed better than the MLP model in predicting three classes. A scatter plot has been made for both the features and decision surface is drawn using RBF kernel with the parameters mentioned above. The figure 4.6 shows the decision boundaries drawn by the SVM with RBF kernel to separate the three classes. It was able to draw decision boundaries for most of the classes. The red decision boundary is to separate the rough classes, blue is to separate the good road and the rest for the fair road. There are few points at which two classes lies close to each other. The reason for this will be explained in the discussion section.

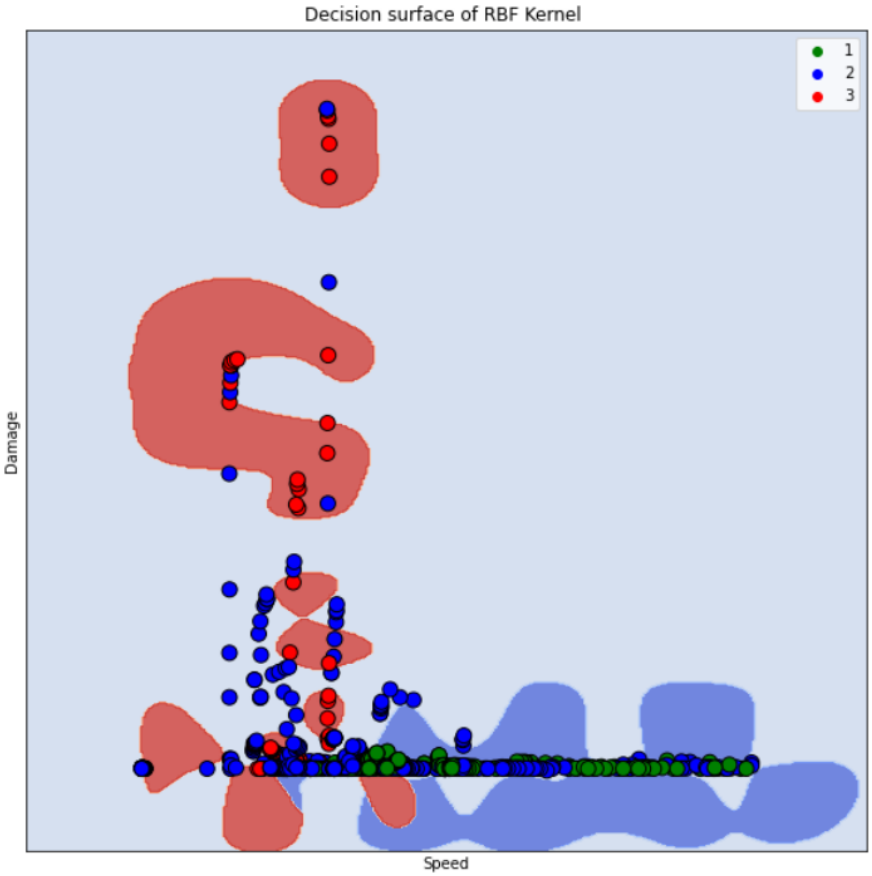

Figure 4.6: Decision boundaries drawn by the SVM.

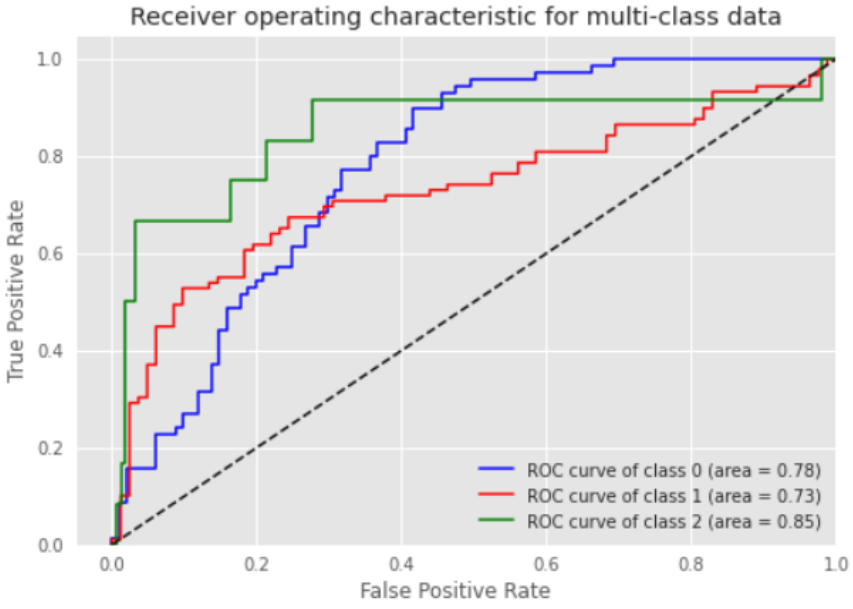

Figure 4.7: ROC-AUC curve.

The plot 4.7 shows the plot and the score obtained for three classes. When the value is 1, the classifier is able to perfectly the positive and negative classes and when the value is 0.5, the classifier could be said as random classifying the points. The SVM with RBF kernel and other hyperparameter settings, it is able to achieve a good score for all the three classes. From comparing the evaluation metrics of MLP and SVM, it is evident that SVM with RBF kernel has performed well on the dataset.

5

Discussion and Conclusion

This section gives insight into the discussion of the results, future work and conclusion to the problem

5.1 Discussion

As seen with the MLP, the network with three hidden layers performed better than the network with two hidden layers. The reason could be that since the network is deep, it is able to learn the correlation between the data in the dataset. Since there are data points of different classes lies close to each other, the network is able to learn a bit better than the two hidden layer model. Increasing more than three hidden layers will lead the model to overfit. From the results of both the models, it is evident that SVM has performed much better than the MLP. One of the main reasons for this is kernel functions. SVM is able to transform the data into a higher dimension to be able to classify it. SVM comparatively has lower chances of overfitting as compared to the MLP. But it's not the case every time, SVM has an issue of taking longer training model for complex data. SVM has chances of poor performance when the data is really large and noisy. One of the reasons why SVM has performed better in this problem could be due to the fact that our dataset is not large and there are only two features that has been used to train the model. These are the potential reasons why SVM had better performance than MLP on the dataset.

From the figure 4.6, there are few points where two classes at same speed has two different pseudo-damage values. The reason for this could be that the HERE API wouldn't have access to the locations such as mines or mountains where the trucks would've been used to transport load. In these regions, the truck would've accumulated some pseudo-damage and by the time it reaches the road, the pseudo-damage value will be higher than the usual one and when it's driven on the fair road, it accumulates much more damage. This is the reason for the points to lie close to each other in-spite of different being under different roughness classes. And along the x-axis there are many points that belongs to class 1 and class 2 overlaps. The reason is that since the interval for categorizing the classes is a general one rather than selecting them based on the countries. Since the size of the data is small already, these outliers aren't removed. But when there is more data on all the three classes, the roughness range for categorizing the classes could be adjusted and this issue of overlap could be fixed.

5.2 Future Work

There are number of potential future work that can be built upon this which would improve the performance of the model. The size of the data that has been used for this problem could be increased further and including more data for the class 3 could improve the performance of the model. The accelerometer reading's resolution could be increased such that the readings are updated every one or two hour. It is known that neural network model will work better on large dataset with more data on all the three classes. Since there was more time spent on creating a proper dataset, the other classification models hasn't been explored in detail. Trucks with similar configuration can be taken and verified in future. For the future work, the performance of other classification models could be compared and playing with the different architectures of the neural network to find a better architecture than the current one. Another one would be to include the other axes as well since only Z-axis data has been used now. The roughness classes are not consistent all over the world. Each country has different roughness scales. For this thesis a common scale has been used for all the countries. A future work could be to include the roughness scale based on the country and then the labels could be created on that.

5.3 Conclusion

This thesis attempts to build a model that can predict the road roughness category for a vehicle given its accumulated accelerometer values over a period of time. Despite all the limitations and challenges mentioned earlier, the model can predict fairly well for the two categories. The predictions can be further improved by adding more data and features. The feature size could be increased by considering the vehicles with similar configurations and country based IRI. A higher resolution of accelerometer readings will also help the model in improving accuracy. Overall we conclude that with current limitations the model is achieving a reasonable level of accuracy. Such a model is useful in predicting road roughness profiles for customer's vehicles and thus assisting in recommending more appropriate vehicle configurations as per needs.

Bibliography

[1] Eriksson, Jakob Girod, Lewis Hull, Bret Newton, Ryan Madden, Samuel Balakrishnan, Hari. (2008). The Pothole Patrol: Using a mobile sensor network for road surface monitoring. MobiSys'08 - Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services. 29-39. 10.1145/1378600.1378605.

[2] Aydın, Metin Yıldırım, Mehmet Sinan Farslöf, Lars. (2018). THE USE OF SMART PHONES TO ESTIMATE ROAD ROUGHNESS: A CASE STUDY IN TURKEY. Engineering Sciences. 13. 247-257. 10.12739/NWSA.2018.13.3.1A0416.

[3] Du, Yuchuan Liu, Chenglong Wu, Difei Jiang, Shengchuan. (2014). Measurement of International Roughness Index by Using Z -Axis Accelerometers and GPS. Mathematical Problems in Engineering. 2014. 1-10. 10.1155/2014/928980.

[4] Lépine, Julien Rouillard, Vincent Sek, Michael. (2016). On the Use of Machine Learning to Detect Shocks in Road Vehicle Vibration Signals. Packaging Technology and Science. 30. n/a-n/a. 10.1002/pts.2202.

[5] Allouch, Azza Koubaa, Anis Abbes, Tarek Ammar, Adel. (2017). RoadSense: Smartphone Application to Estimate Road Conditions Using Accelerometer and Gyroscope. IEEE Sensors Journal. PP. 10.1109/JSEN.2017.2702739.

[6] Gorges, Christian Öztürk, Kemal Liebich, Robert. (2018). Impact detection using a machine learning approach and experimental road roughness classification. Mechanical Systems and Signal Processing. 117. 738-756. 10.1016/j.ymssp.2018.07.043.

[7] Arun Mohan, Sumathi Poobal,Crack detection using image processing: A critical review and analysis, Alexandria Engineering Journal, Volume 57, Issue 2, 2018, Pages 787-798, ISSN 1110-0168, https://doi.org/10.1016/j.aej.2017.01.020.

[8] Nguyen Teron,Lechner Bernhard, Wong Yiik Diew, 2019/10/15, Response-based methods to measure road surface irregularity: a state-of-the-art review, European Transport Research Review,1866-8887, https://doi.org/10.1186/s12544-019-0380-6

[9] Le, Thang Garcia, Rafael Casari, Paolo Östberg, P-O. (2019). Machine Learning Methods for Reliable Resource Provisioning in Edge-Cloud Computing: A Survey. ACM Computing Surveys. 52. 1-39. 10.1145/3341145.

[10] Elghriany, Ahmed Yi, Ping Liu, Peng Yu, Quan. (2015). Investigation of the Effect of Pavement Roughness on Crash Rates for Rigid Pavement. Journal of Transportation Safety Security. 8. 00-00. 10.1080/19439962.2015.1025458.

[11] Chen, Shong-Loong Lin, Chih-Hsien Tang, Chao-Wei Chu, Liang-Pin Cheng, Chiu-Kuei. (2020). Research on the International Roughness Index Threshold of Road Rehabilitation in Metropolitan Areas: A Case Study in Taipei City. Sustainability. 12. 10536. 10.3390/su122410536.

[12] Schlecht, Joseph Kaplan, Matthew Barnard, Kobus Karafet, Tatiana Hammer, Michael Merchant, Nirav. (2008). Machine-Learning Approaches for Classifying Haplogroup from Y Chromosome STR Data. PLoS computational biology. 4. e1000093. 10.1371/journal.pcbi.1000093.

[13] Hachimi, Marouane Kaddoum, Georges Gagnon, Ghyslain Illy, Poulmanogo. (2020). Multi-stage Jamming Attacks Detection using Deep Learning Combined with Kernelized Support Vector Machine in 5G Cloud Radio Access Networks.

[14] Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names. https://gombru.github.io/2018/05/23/cross_entropy_loss/

[15] Performance Meaures: Cohen Kappa's statistic. https://thedatascientist.com/performance-measures-cohens-kappa-statistic/

[16] Evgeniou, Theodoros Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7_12.

[17] Grossi, Enzo Buscema, Massimo. (2008). Introduction to artificial neural networks. European journal of gastroenterology hepatology. 19. 1046-54. 10.1097/MEG.0b013e3282f198a0.

[18] https://www.omicsonline.org/open-access/how-the-roadway-pavement-roughness-impa
php?aid=91642

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY