A Project Report On

Automatic Detection of Helmet on Public Roads

Submitted in partial fulfillment of the requirement for the 8th semester

Bachelor of Engineering

in

Computer Science & Engineering

VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM

Submitted by

AMANDEEP RATHEE	**[1DS13CS012]**
KRISHNANGINI KALITA	**[1DS13CS048]**
MAHIMA SINGH DEO	**[1DS13CS050]**

Under the guidance of

Dr. Ramesh Babu D R

Vice Principal and HoD, CSE DSCE

2016-2017

Department of Computer Science & Engineering

DAYANANDA SAGAR COLLEGE OF ENGINEERING

BANGALORE - 560078

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

DAYANANDA SAGAR COLLEGE OF ENGINEERING

Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore - 560078

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project work entitled **"Automatic Detection of Helmet on Public Roads"** is a bonafide work carried out by **Amandeep Rathee [1DS13CS012], Krishnangini Kalita [1DS13CS048]** and **Mahima Singh Deo [1DS13CS050]** in a partial fulfilment for the 8^{th} semester of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the year 2016-2017. The project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for Bachelor of Engineering Degree.

Dr. Ramesh Babu D R	**Dr. Ramesh Babu D R**	**Dr. C P S Prakash**
(Internal Guide)	Vice Principal & Head of	Principal,
Vice Principal & HoD,	Department, CSE, DSCE	DSCE
Department of CSE, DSCE		
Signature:.....	Signature:.....	Signature:.....

Name of the Examiners: Signature with date:

1..... .....

2..... .....

Abstract

Road safety is often neglected by riders worldwide leading to accidents and deaths. To address this issue most countries have laws which mandate the use of helmets for two wheeler riders. Despite the law there is a large proportion of population that does not use helmet as a safety gear. Traffic police discourages this behavior by issuing a traffic violation ticket. As of now this process is manual and tedious. This project aims to solve this problem by automating the process of detecting the riders who are riding without helmets and also extracts the license plate so that it could be used to issue traffic violation tickets.

The system implements machine learning and image processing techniques to detect two-wheeler riders who are not wearing helmets. To solve this problem the system takes a video of traffic at public roads as the input and detects moving objects in the scene. A machine learning classifier (SVM) is applied on the moving object to detect if the moving object is a two-wheeler. If it is a two-wheeler then the same classifier is applied again to the head region of the rider to detect if the rider is wearing a helmet. The license plate area is provided as the output in case the rider is not wearing helmet.

Acknowledgement

A successful project is a fruitful culmination of efforts by many people, some directly involved and some others who have quietly encouraged and extended support from the background.

A project is not complete if one fails to acknowledge all these individuals who have been instrumental in the successful completion of the project.

We would like to begin by expressing our sincere gratitude to our guide, **Dr. Ramesh Babu D R**, Vice Principal and Head of Department, Department of Computer Science & Engineering and for his excellent and timely guidance, constant encouragement and strive for perfection that he brought to our project.

We express our sincere gratitude to him for his relentless support and helped us to make critical decisions during the course of our project.

We would also like to extend our thanks to **Dr. C.P.S. Prakash**, Principal of our college, DSCE for his advice and guidance, that helped us to complete this project with success.

Finally, its a pleasure and happiness to the friendly co-operation showed by all the staff members of Computer Science Department, DSCE.

Amandeep Rathee [1DS13CS012]

Krishnangini Kalita [1DS13CS048]

Mahima Singh Deo [1DS13CS050]

List of Figures

Contents

1 Introduction

The proposed system aims to provide complete safety for bike riders. Recently helmets have been made compulsory but still people drive without helmets.

Machine Learning is a ”Field of study that gives computers the ability to learn without being explicitly programmed”. It deals with the problem of extracting features from data so as to solve many different predictive tasks.

Image processing is processing of images using mathematical operations by using any form of signal processing for which the input is an image, a series of images, or a video, such as a photograph or video frame, the output of image processing may be either an image or a set of characteristics or parameters related to the image.

We describe a system in machine learning and image processing domain for detecting if bike riders are wearing helmets while riding on roads. The system is in for public interest and good.

1.1 Motivation

Statistics state that 1,37,000 people were killed in road accidents in our country in 2013 alone. This number is more than the number of people killed in our wars put together. Two wheelers account for 25% of the total road crashes and Bengaluru ranks fourth in the list of cities with highest number of accidents.There was a 20% increase in the number of road accidents in the state during 2016 compared to 2015.Hence there is need for a system or mechanism that can help reduce deaths due to road accidents. Human life is very precious and there is a need to protect it from unaware conditions and siuation.

1.2 Objective of Work

The amount of deaths has been rising as we can see from the charts below.Hence, for public safety there needs to be a mechanism for automatic helmet detection which can extract the number plates for those not wearing helmets on roads.mages depicting various objects moving on the roads. The images must be captured and background images must be subtracted during preprocessing. There comes a bounding box defining the object of interest in a single frame, our goal here is to test the presence of the helmet on the bounded object. If no helmet is detected, next step comes here to extract the number plate from that very frame.

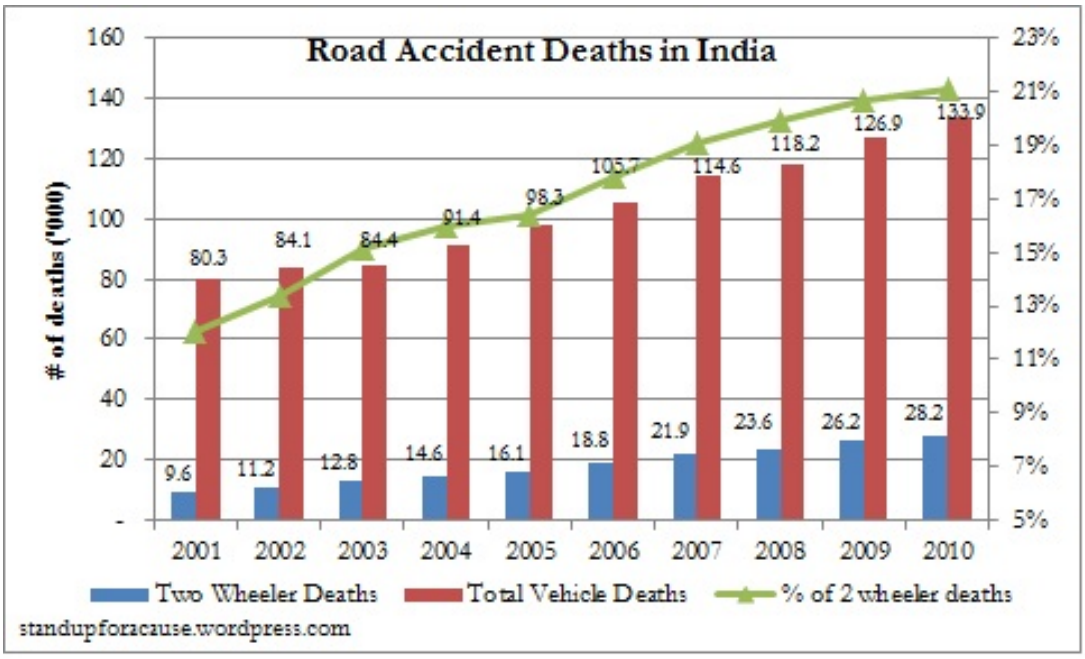

Figure 1: Road Accident Deaths in India

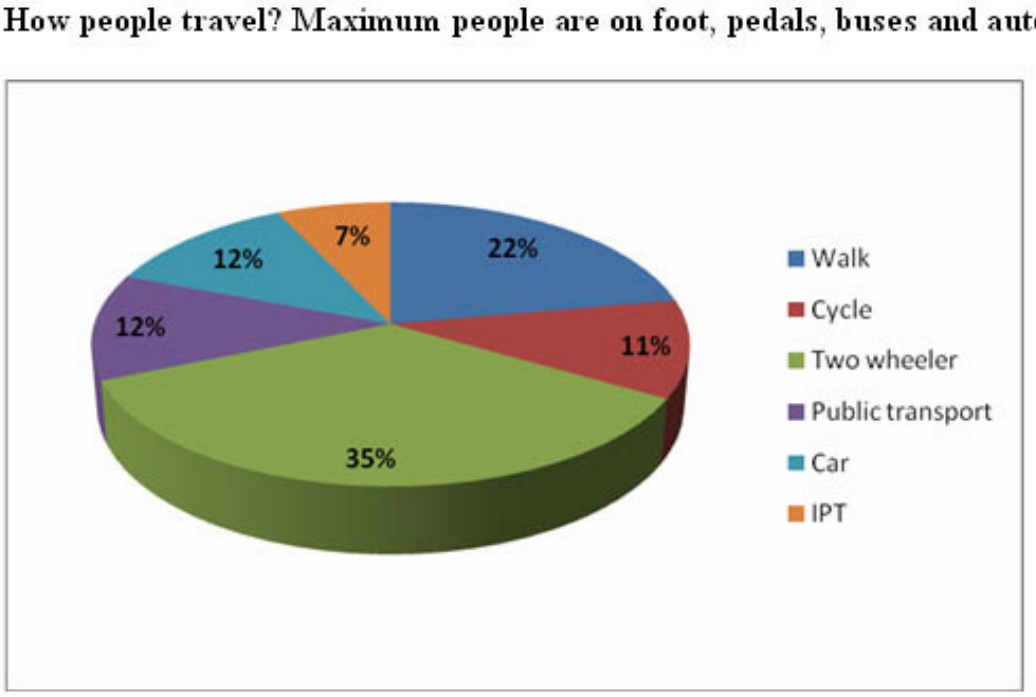

Figure 2: Percentage of deaths by various modes of vehicles

2 Problem Statement and Proposed Solution

2.1 Problem Solution

The amount of deaths has been rising as we can see from the charts below. Hence, for public safety there needs to be a mechanism for automatic helmet detection which can extract the number plates for those not wearing helmets on roads so as to have road accidents minimized. The citizens should be made aware of the need to wear helmets while riding on two-wheelers.

2.2 Methodology Adopted

1. Moving Object Detection: There is an input video,from which we then detect certain moving objects around the vehicle in real time. The ForegroundDetector System object compares a color or grayscale video frame to a background image to determine whether individual pixels are part of the background or the foreground. It then computes a foreground object. By using background subtraction,we can detect foreground objects in an image that has been taken from a stationary camera.

2. Vehicle classification: For vehicle classification,we use feature extraction algorithm,HOG(histogram of oriented gradients) for detection and compaison of features. We divide the datasets into train and test, and henceforth train the classifier into classifying our input image into motorcycle and non-motorcycle.

3. Helmet Detection: After classification into motorcycle,we train the classifier into dividing the input cropped head part that has a helmet and that which doesnot have a helmet.

4. License plate extraction: The next step consist of, if not detected wearing a helmet in the previous step, to crop the lincense plate of the motorcycle in that very frame as the output.

2.3 Organisation of the Report

In the next chapter, Chapter 2, a Survey of the related work has been presented. Chapter 3, states the System Requirement Specification. Chapter 4 consists of System Design. Chapter 5 deals with System Implementation. Chapter 6 deals with Verification and Validation. Chapter 7 has the Results of the project while Chapter 8 states the Conclusion and Future Works for the same.Chapter 9 is the Reference section which lists down all the study material that has been referenced for the project. Chapter 10 is the Appendix which contains the source code of the project.

3 Literature Survey

3.1 Existing State of Art

1. There exists an intelligent system, Smart Helmet, which automatically checks whether the person is wearing the helmet and has non- alcoholic breath while driving. Here we have a transmitter at the helmet and the receiver at the bike. There is a switch used to make sure the wearing of helmet on the head. The ON condition of the switch ensures the placing of the helmet in proper manner. An alcohol sensor is placed near to the mouth of the driver in the helmet to detect the presence of alcohol. The data to be transferred is coded with RF encoder and transmitted through radio frequency transmitter. The receiver at the bike receives the data and decodes it through RF decoder. The engine should not ON if any of the two conditions is violated.MCU controls the function of relay and thus the ignition, it controls the engine through a relay and a relay interfacing circuit.

2. There is a proposed circle/circular arc detection method based upon the modified Hough transform,

and can apply it to the detection of safety helmet for the surveillance system by the camera. Since the safety helmet location will be in the set of the obtained possible circles/circular arcs (if any exists). We use geometric features to verify safety helmet exists in the set.

Here, first the camera scans the image of the driver and sends it to the controller.

L=2*(3.14)(r0)*(Rac)

The ratio of the arc length to the complete circle is denoted Rac and r0 is the radius.

3.2 Survey

- Title: Automatic Detection of motorcyclists without helmet

  Author: Romuere Silva, Kelson Aires, Thiago Santos, Kalyf Abdala, Rodrigo Veras, André Soares

  Conference Name: 2013 XXXIX Latin America Computing Conference (CLEI)

  Year:2013

  Abstract: This paper aims to explain and illustrate an automatic method for motorcycles detection and classification on public roads and a system for automatic detection of motorcyclists without helmet. For this, a hybrid descriptor for features extraction is proposed based on Local Binary Pattern, Histograms of Oriented Gradients and the Hough Transform descriptors. Traffic images captured by cameras were used. The best result obtained from classification have an accuracy rate of 0.9767, and the best result obtained from helmet detection have an accuracy rate of 0.9423.

- Title:The safety helmet detection for atm's surveillance system via the modified hough transform
  Author:C.-Y. Wen, S.-H. Chiu, J.-J. Liaw, and C.-P. Lu
  Conference Name:IEEE 37th Annual International Carnahan Conference on SecurityTechnology
  Year:2003
  Abstract: Wen et al. suggested a circle arc detection methodbased upon the Hough transform. They applied it to detect helmet on the surveillance system of the Automatic Teller Machine.
  The weakness of this work is that they only use geometric features to verify if any safety helmet exists in the set.Geometric features are not enough to find helmet. The people head can be mistaken with a helmet too.

- Title:Motorcycle detection and tracking system with occlusion segmentation
  Author:C.-C. Chiu, M.-Y. Ku, and H.-T. Chen
  Conference Name: WIAMIS '07, USA
  Year:2007
  Abstract:Here there was proposed a computer vision system aiming to detect and segment motorcycles partly occluded by another vehicle. A helmet detection system is used, and the helmet presence determines that there is a motorcycle. In order to detect the helmet presence, the edges are computed on the possible helmet region. The Canny edge detector is used.The quantity of edge points which are similar to a circle that define a helmet region. The method needs so much information(helmet radius, camera angle, camera height, etc) that must beprovided by user.

- Title:Helmet presence classification with motorcycle detectionand tracking
  Author: J. Chiverton
  Conference Name:IET, vol. 6
  Year:2012
  Abstract:Chiverton described and tested a system for the automatic classification and tracking of motorcycle riders with and without helmets. The system uses support vector machines trained on histograms. The histograms are derived from head region of motorcycle riders using both static photographs and individual image frames from video data. A high accuracy rate was obtained but the number of test images were insufficient.

- Title:A computer vision system forthe detection and classification of vehicles at urban road intersections

Author: S. Messelodi, C. Modena, and M. Zanin

Conference Name: Pattern Analysis and Applications, vol. 8

Year: 2005

Abstract:A segmentation and classification vehicle system was pro-posed in this paper. It also computes the approximated speed ofthe vehicle. Three-dimensional models are created based onvehicle size. The size depends on the vehicle class: car, bus,pedestrian, etc. The generated models are compared to thecomputed models to classify the vehicles. The main drawback of this paper is that a single model is used for both bicycles and motorcycles.

- Title:Bicycle detection usingpedaling movement by spatiotemporal gabor filtering

  Author: K. Takahashi, Y. Kuriya, and T. Morie

  Conference Name: TENCON 2010

  Year: Nov. 2010

  Abstract:Takahashi et al. introduced a computer vision system for bicycle, pedestrian and motor-cycle detection. The system detects moving objects (horizontal motion) and pedaling move-ment (vertical motion) using the Gabor filtering. The HOG descriptor is computed and the SVM clas-sifier classifies theobjects into two classes: two-wheel vehicle and pedestrian. At the end, the vertical motion is computed and the two-wheel vehicle are classified into motorbike and bicycle.

- Title:Moving ob-jects detection at an intersection by sequential background extraction

  Author:S. Sonoda, J. K. Tan, H. Kim, S. Ishikawa, and T. Mori

  Conference Name:ICCAS 2011

  Year: Oct'2011

  Abstract: Sonoda et al. proposed a system to detect moving objects. The system aims to detect moving objects at an intersection (like vehicles and pedestrians), and to warn the driver. They used the Mixture of Gaussians to detect the moving objects and the Lucas-Kanade Traker algo-rithm for pedestrian tracking.

- Title: Machine vision techniques for motorcycle safety helmet detection in Image and Vision

  Author: R. Waranusast, N. Bundon, V. Timtong, C. Tangnoi, and P. Pattanathaburt

  Conference Name: Image and Vision Computing New Zealand (IVCNZ), 28th International

  Year: 2013

  Abstract: The most recent work on helmet detection was proposed by Waranusast et al. In this

work the classification is made by geometric and color information. These attributes are used by the K-Nearest Neighbors (KNN) to classify images of people based on helmet use. Moreover, the system counts the number of passengers on the motorcycle. According to the authors, the passenger counter had a total of 83.82was 89

- Title: Text Extraction for Sri Lankan Number Plates
  Author: J. M. N. D. B. Jayasekara , W. G. C. W. Kumara
  Conference name: 8th International conference on Ubi-media computing(UMEDIA)
  Year :2015
  Abstract: Automatic text extraction from number plates is used in most of the countries in present. In this paper a semi-automatic text extraction from Sri Lankan vehicle number plates is presented.Number plate first corrected as can be seen from a camera situated in front of the camera. Then the area of the numbers is extracted. Texts and numbers are then separated in to separate images. Finally separated texts and numbers are matched with the original template created using template matching. A data set consisted of 93 number plates is used to demonstrate the usability of the proposed method.

- Title: Safeguarding of Motorcyclist through helmet recognition
  Author:G. Sasikala, Kiran Padol, Aniket A. Katekar and Surender Dhanasekaran
  Conference Name:2015 International Conference on Smart Technologies and Management for computing, Communication, Controls, Energy and Materials (ICSTM).
  Year: 2015
  Abstract: Motorcycles being an obvious choice a convenient transportation mode, it has a major contribution to road accident casualties and injuries.Despite the Government traffic regulation, people still avoid using helmet. The proposed system is an effort to create awareness in a society by endorsing use of helmet and lead people to safety. This paper proposes effective enforcement of use of helmet by implementing RF communication based helmet detection system.

- Title: Development of Helmet Detection System and Smart seat belt
  Author: S. Anil Babu ,Sumathi Ayyalusamy, Rejin Ranjit Singh , Sreejin Dharmarijan , Jason James and Mohamed Anas
  Conference Name: IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
  Year: 2015
  Abstract: The driver of the vehicle involves in a high speed accident without wearing a helmet and

seat belt.It is highly dangerous and can cause death. Wearing a seat belt and helmet can reduce shock from the impact and may save a life. The aim of this research work is to development of smart helmet detection system and seat belt detection system for dune buggy to avoid or reduce the accident fatigue on drivers during accident. Driver will be unable to start vehicle without wearing seat belt and helmet.

- Title: Extraction and segmentation of helmets in motorcycles
  Author: G.S.Gopika, R.Monisha and S.Karthik
  Conference Name: International Journal of Trend in Research and Development
  Year: Mar- Apr 2016
  Abstract: The work is to detect whether the motorcycle riders are having the helmets. Even though motorcycles are convenient means of transportation, it is not so safer. Here, we have give a traffic video as input. In this video, we apply background extraction algorithm. This is used to extract the foreground objects in the video which is then extracted as frames. In the next stage, the SIFT (Scale Invariant Feature Transform) algorithm is used to detect a motion object motorcycle. Using the Region of Interest (ROI), it chooses the location where the helmet can be found. This area is extracted and the helmet is detected.

4 Architecture and Design

4.1 System Requirement Specification

4.1.1 Minimum Hardware Requirement

i.System : Intel i3 and above

ii.RAM : 4 GB

iii.Input device : Keyboard, Mouse

iv.Output device : Color monitor

v.Network hardware : Network Interface Card

4.1.2 Minimum Software Requirement

Software requirements for the implementation and testing:

i.Language: Matlab programming language

ii.Operating System: Windows 7+

iii.Software Packages : MATLAB version 8.4

4.1.3 Core Tools and Technologies

This section covers the complete development matrix. It identifies the technology elements with guidelines and specifications for specific implementations.The core tools and technologies elements used in developing the project are given below.

MATLAB includes development tools that help to implement the algorithm efficiently.These include the following:

i. MATLAB Editor and Debugger - Provides standard editing and debugging features, such as setting breakpoints and single stepping

ii. Code Analyzer - Checks the code for problems and recommends modifications to maximize performance and maintainability

iii. MATLAB Profiler - Records the time spent executing each line of code.

iv. Directory Reports - Scan all the files in a directory and report on code efficiency, file differences, file dependencies, and code coverage.

4.1.4 Language Specification

The MATLAB language supports the vector and matrix operations that are fundamental to engineering and scientific problems. It enables fast development and execution.

With the MATLAB language, to develop the program and algorithms are faster than with traditional languages because this does not need to perform low-level administrative tasks, such as declaring variables, specifying data types, and allocating memory. In many cases, MATLAB eliminates the need for 'for' loops.

As a result, one line of MATLAB code can often replace several lines of C or C++ code. At the same time, MATLAB provides all the features of a traditional programming language, including arithmetic operators, flow control, data structures, data types, Object- Oriented Programming (OOP), and debugging features

.

MATLAB provides the following types of functions for performing mathematical operations and analyzing data:

i. Matrix manipulation and linear algebra

ii. Polynomials and interpolation

iii. Fourier analysis and filtering

iv. Data analysis and statistics

v. Ordinary Differential Equations (ODEs)

vi. Partial Differential Equations (PDEs)

4.2 System Design

4.2.1 Architectural Diagram

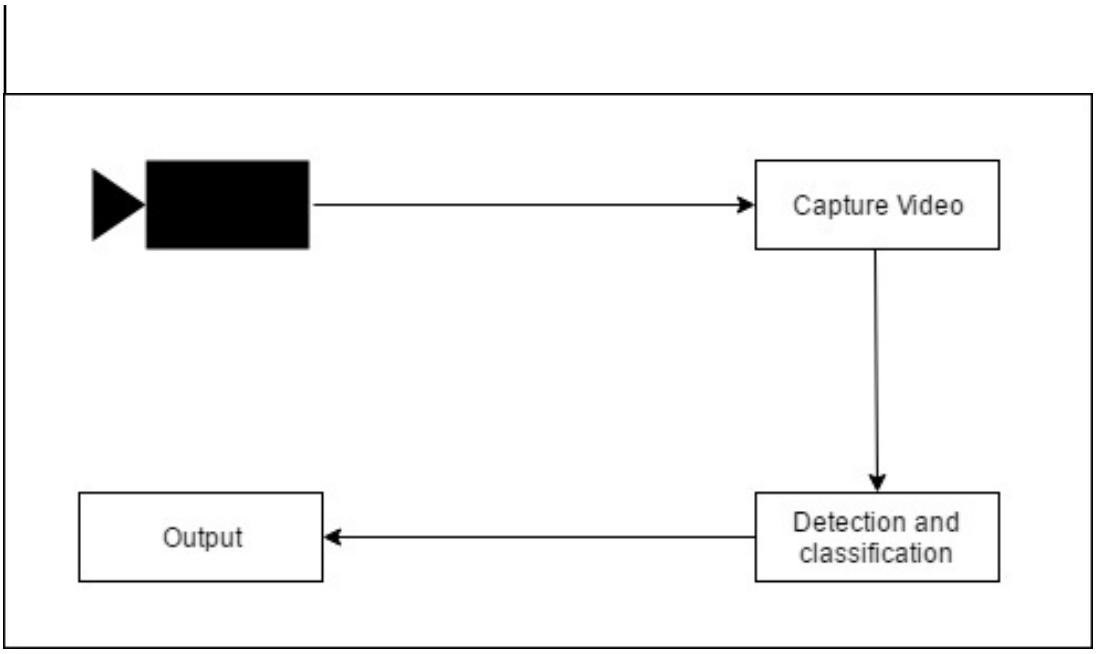

Figure 3: System Architecture

The diagram depicts the architectural structure of the system that is being built. The camera is responsible for capturing videos of traffic on public roads. The video is then fed as input to our system which performs processes like detection of moving objects and classification into two-wheeler or non two-wheeler and checks if helmet is present or not. Once this is done the output is displayed.

4.2.2 Activity Diagram

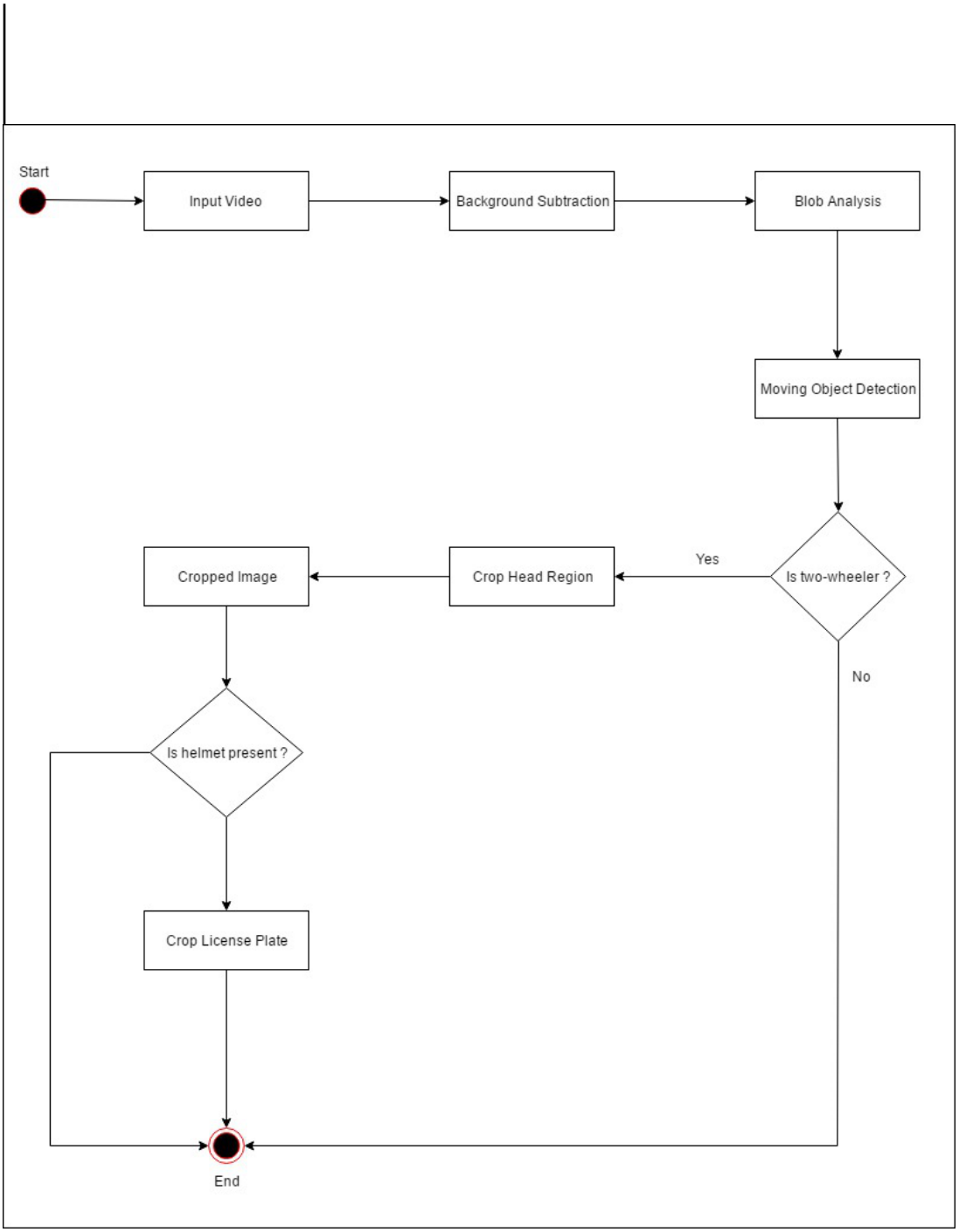

Figure 4: Activity Diagram

4.2.3 Sequence Diagram

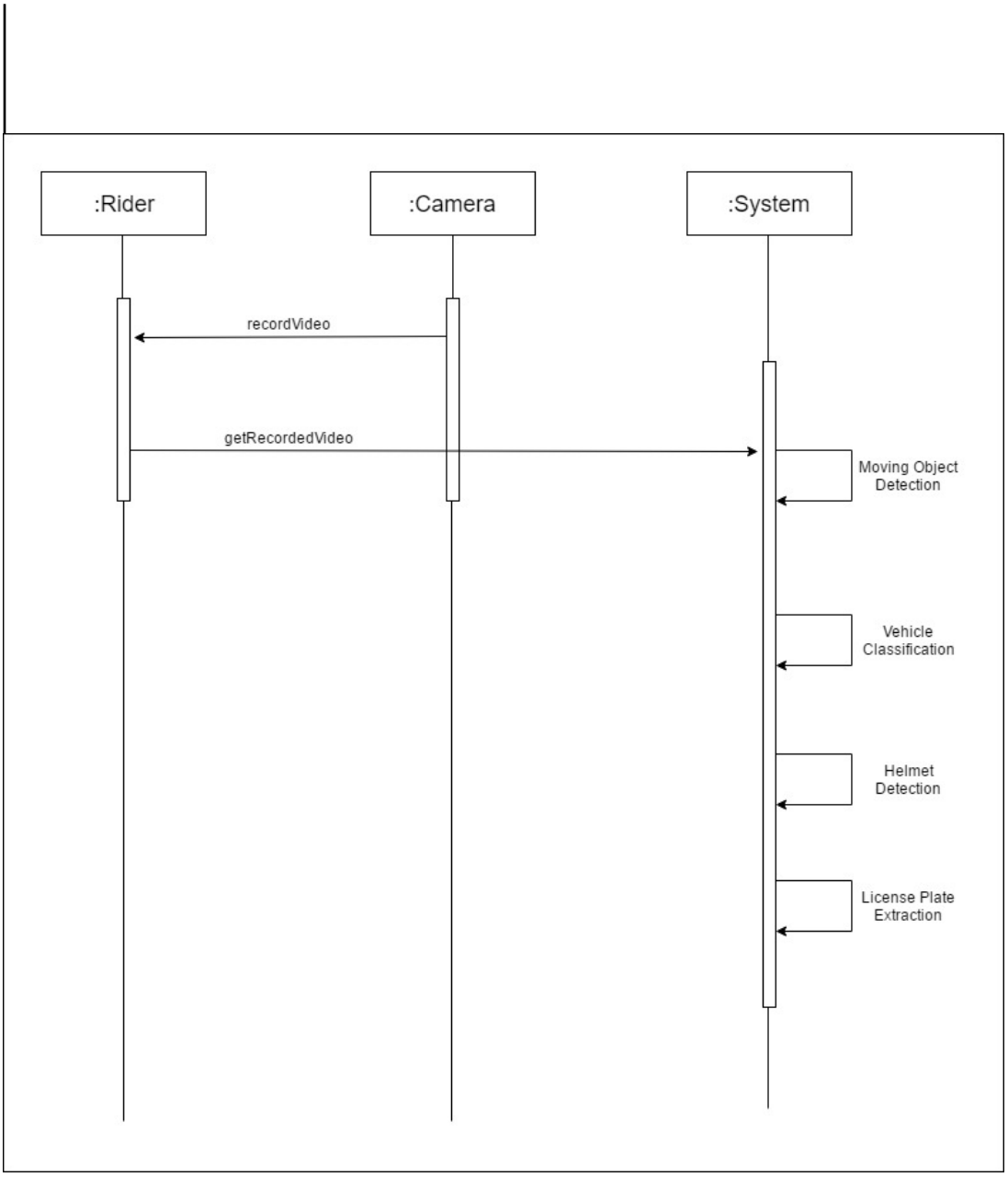

Figure 5: Sequence Diagram

4.2.4 Use Case Diagram

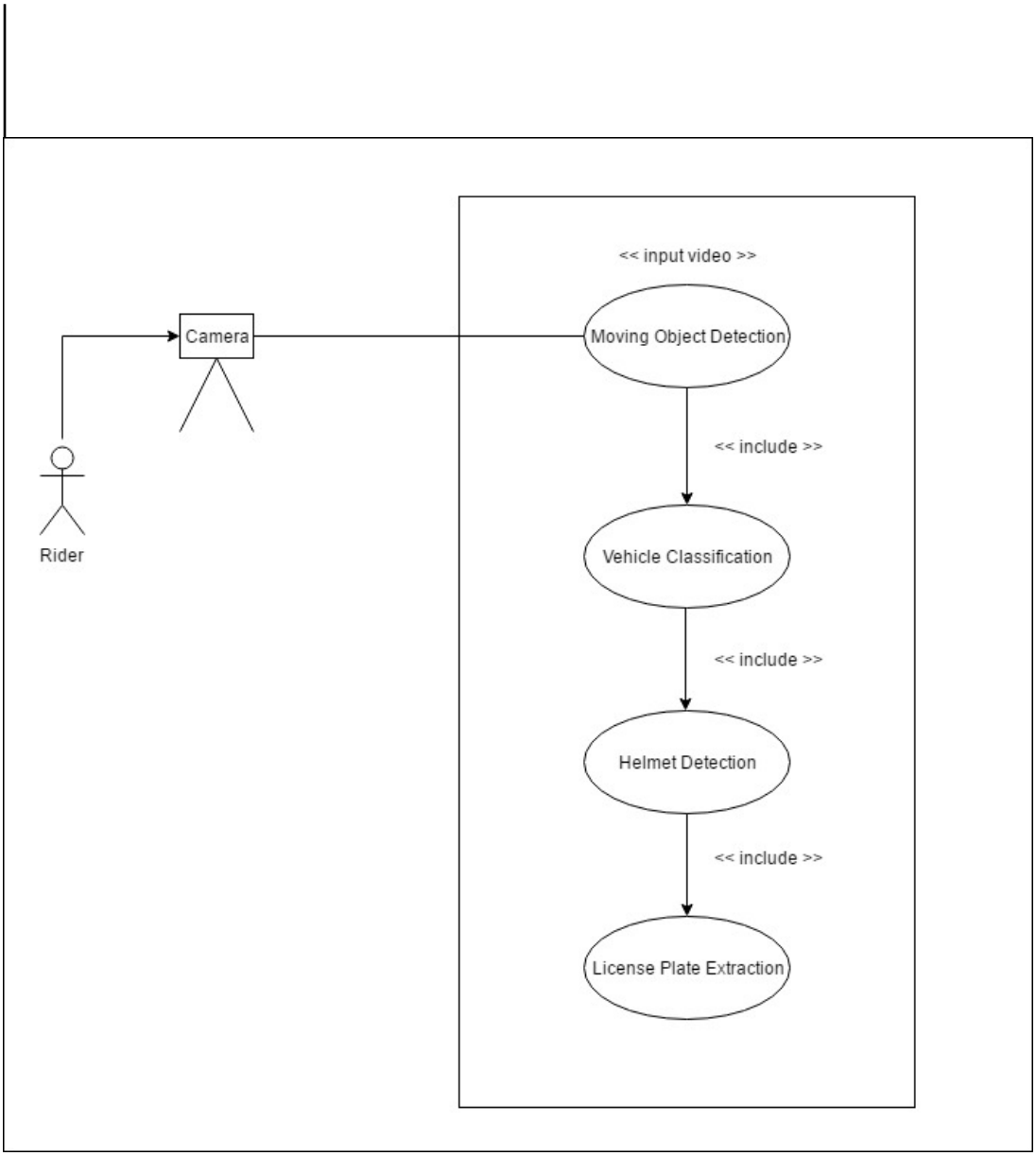

Figure 6: Use Case Diagram

5 Implementation

5.1 Design of various modules

5.2 Moving Object Detection and Backgroung Subtraction

Detecting and counting vehiles can be used to analyze traffic patterns. Detection is also a first step prior to performing more sophisticated tasks such as tracking or categorization of vehicles by their type.
Rather than immediately processing the entire video, the example starts by obtaining an initial video frame in which the moving objects are segmented from the background. This helps to gradually introduce the steps used to process the video.
The foreground detector requires a certain number of video frames in order to initialize the Gaussian mixture model.

```
    foregroundDetector = vision.ForegroundDetector('NumGaussians', 3, ...
'NumTrainingFrames', 50);

    videoReader = vision.VideoFileReader('visiontraffic.avi');
for i = 1:150 frame = step(videoReader); foreground = step(foregroundDetector, frame);
end
```

The foreground segmentation process is not perfect and often includes undesirable noise. The code below uses morphological opening to remove the noise and to fill gaps in the detected objects.

```
    se = strel('square', 3);
filteredForeground = imopen(foreground, se);
figure; imshow(filteredForeground); title('Clean Foreground');
```

Next, we find bounding boxes of each connected component corresponding to a moving car by using vision.BlobAnalysis object. The object further filters the detected foreground by rejecting blobs which contain fewer than 150 pixels.

```
    blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
'AreaOutputPort', false, 'CentroidOutputPort', false, ...
'MinimumBlobArea', 150);
bbox = step(blobAnalysis, filteredForeground);
```

The number of bounding boxes corresponds to the number of moving objects found in the video frame. We display the number of found objects in the upper left corner of the processed video frame.

5.3 Classification into motorcycle and non-motorcycle and helmet detection

Synthetic digit images are used for training. The training images each contain an object of interest surrounded by other objects of interest, which mimics how our objects(which we want to classify) are normally seen together. Using synthetic images is convenient and it enables the creation of a variety of training samples without having to manually collect them. For testing, scans of cropped image are used to validate how well the classifier performs on data that is different than the training data. Although this is not the most representative data set, there is enough data to train and test a classifier, and shows the feasibility of the approach.
syntheticDir = fullfile(toolboxdir('vision'), 'visiondata','digits','synthetic');
handwrittenDir = fullfile(toolboxdir('vision'), 'visiondata','digits','handwritten');
Prior to training and testing a classifier, a pre-processing step is applied to remove noise artifacts introduced while collecting the image samples.
The data used to train the classifier are HOG feature vectors extracted from the training images. Therefore, it is important to make sure the HOG feature vector encodes the right amount of information about the object. The extractHOGFeatures function returns a visualization output that can help form some intuition about just what the "right amount of information" means.

The fitcecoc function from the Statistics and Machine Learning Toolbox^{TM} is used to create a multiclass classifier using binary SVMs.
classifier = fitcecoc(trainingFeatures, trainingLabels);

5.4 License Plate Extraction

After the previous steps are accomplised and we find the motorbike not having a helmet,our next step is to extract the license plate of that very motorbike. We extract the region of interest from our cropped image by giving the appropraite coordinates.

5.5 Flowchart

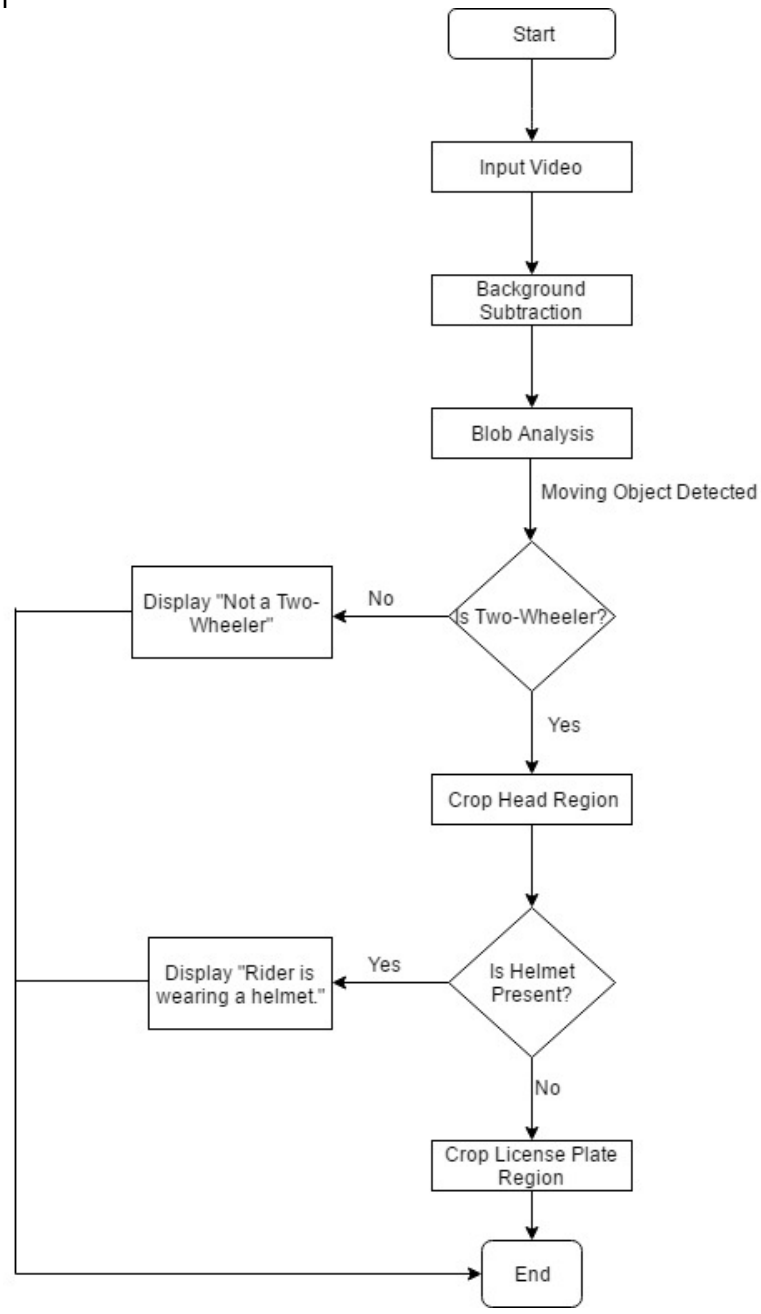

Figure 7: Flowchart

5.6 Algorithms Used

For our system we need a classification model at two stages, one for vehicle classification and other for helmet detection. The first stage is where the system will estimate whether the moving vehicle in the video is a two-wheeler or not. At the second stage of classification it estimates whether the two-wheeler rider is wearing a helmet or not. Both of these estimates will be done using some features extracted from the images using image processing techniques discussed later.

5.6.1 Logistic Regression

This machine learning algorithm was developed in the 1958 by statician David Cox. Logistic model is used to estimate the probability of a categorical variable which has only two classes. Using the probabilites and a suitable cutoff one of the two classes is chosen as the final outcome. Logistic model uses a logistic function which takes a number as input and outputs a number between zero and one. The beauty of the function is that it can take any real value and the output of the function is still between zero and one always.

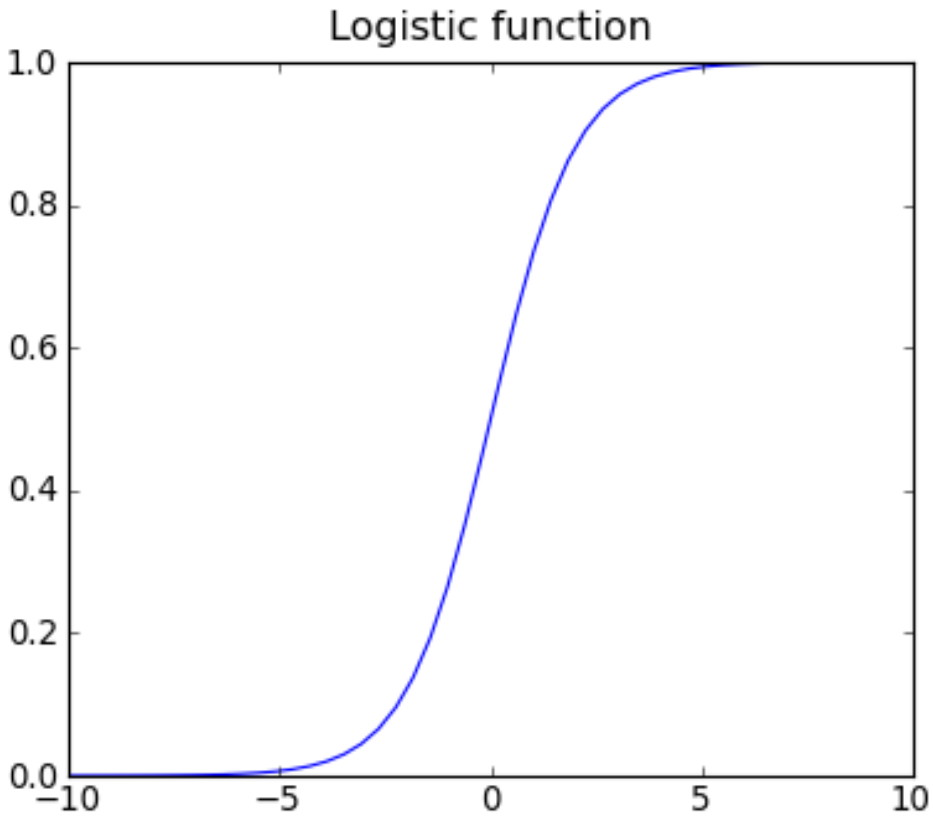

Figure 8: Logistic Regression Function

1.Representation Used for Logistic Regression Logistic regression uses an equation as the representation, very much like linear regression.

Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

y = e^(b0 + b1*x) / (1 + e^(b0 + b1*x))

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data.

The actual representation of the model that you would store in memory or in a file are the coefficients in the equation (the beta value or b's).

2.Logistic Regression Predicts Probabilities Logistic regression models the probability of the default class (e.g. the first class).

Written another way, we are modeling the probability that an input (X) belongs to the default class (Y=1), we can write this formally as:

P(X) = P(Y=1—X)

Note that the probability prediction must be transformed into a binary values (0 or 1) in order to actually make a probability prediction. More on this later when we talk about making predictions.

Logistic regression is a linear method, but the predictions are transformed using the logistic function. The impact of this is that we can no longer understand the predictions as a linear combination of the inputs as we can with linear regression, for example, continuing on from above, the model can be stated as:

p(X) = e^(b0 + b1*X) / (1 + e^(b0 + b1*X))

I don't want to dive into the math too much, but we can turn around the above equation as follows (remember we can remove the e from one side by adding a natural logarithm (ln) to the other):

ln(p(X) / 1 – p(X)) = b0 + b1 * X

This is useful because we can see that the calculation of the output on the right is linear again (just like linear regression), and the input on the left is a log of the probability of the default class.

This ratio on the left is called the odds of the default class (it's historical that we use odds, for example, odds are used in horse racing rather than probabilities). Odds are calculated as a ratio of the probability of the event divided by the probability of not the event, e.g. 0.8/(1-0.8) which has the odds of 4. So we could instead write:

$$\ln(odds) = b0 + b1 * X$$

Because the odds are log transformed, we call this left hand side the log-odds or the probit. It is possible to use other types of functions for the transform (which is out of scope, but as such it is common to refer to the transform that relates the linear regression equation to the probabilities as the link function, e.g. the probit link function.
We can move the exponent back to the right and write it as:

$$odds = e^{(b0+b1*X)}$$

All of this helps us understand that indeed the model is still a linear combination of the inputs, but that this linear combination relates to the log-odds of the default class.

3.Learning the Logistic Regression Model The coefficients (Beta values b) of the logistic regression algorithm must be estimated from your training data. This is done using maximum-likelihood estimation.

Maximum-likelihood estimation is a common learning algorithm used by a variety of machine learning algorithms, although it does make assumptions about the distribution of your data (more on this when we talk about preparing your data).

The best coefficients would result in a model that would predict a value very close to 1 (e.g. male)

for the default class and a value very close to 0 (e.g. female) for the other class. The intuition for maximum-likelihood for logistic regression is that a search procedure seeks values for the coefficients (Beta values) that minimize the error in the probabilities predicted by the model to those in the data (e.g. probability of 1 if the data is the primary class).

We are not going to go into the math of maximum likelihood. It is enough to say that a minimization algorithm is used to optimize the best values for the coefficients for your training data. This is often implemented in practice using efficient numerical optimization algorithm (like the Quasi-newton method).

4.Assumptions of Logistic Regression The assumptions made by logistic regression about the distribution and relationships in your data are much the same as the assumptions made in linear regression.

Much study has gone into defining these assumptions and precise probabilistic and statistical language is used. My advice is to use these as guidelines or rules of thumb and experiment with different data preparation schemes.

Ultimately in predictive modeling machine learning projects you are laser focused on making accurate predictions rather than interpreting the results. As such, you can break some assumptions as long as the model is robust and performs well.

Binary Output Variable: This might be obvious as we have already mentioned it, but logistic regression is intended for binary (two-class) classification problems. It will predict the probability of an instance belonging to the default class, which can be snapped into a 0 or 1 classification.
Remove Noise: Logistic regression assumes no error in the output variable (y), consider removing outliers and possibly misclassified instances from your training data. Gaussian Distribution: Logistic regression is a linear algorithm (with a non-linear transform on output). It does assume a linear relationship between the input variables with the output. Data transforms of your input variables that better expose this linear relationship can result in a more accurate model. For example, you can use log, root, Box-Cox and other univariate transforms to better expose this relationship.
Remove Correlated Inputs: Like linear regression, the model can overfit if you have multiple highly-correlated inputs. Consider calculating the pairwise correlations between all inputs and removing highly correlated inputs.
Fail to Converge: It is possible for the expected likelihood estimation process that learns the coefficients to fail to converge. This can happen if there are many highly correlated inputs in your data or the data is very sparse (e.g. lots of zeros in your input data).

5.Advantages and Disadvantages Advantages:

1. Convenient probability scores for observations.
2. Efficient implementations available across tools.
3. Multi-collinearity is not really an issue and can be countered with L2 regularization to an extent.
4. Wide spread industry comfort for logistic regression solutions.

Disadvantages:

1. Doesn't perform well when feature space is too large.
2. Doesn't handle large number of categorical features/variables well.
3. Relies on transformations for non-linear features.
4. Relies on entire data.

5.6.2 Support Vector Machines

Support Vector Machines (SVM) are also a type of classification method which can work for both, binary and multiclass classifications. They generally perform better than logistic regression but the computational cost is also very high. SVM is a non-probabilistic classifier i.e. it does not compute the probabilities of an observations belonging to a class. Instead it directly tells which class does the observation belong to based on the training set. While training, SVM creates a hyperplane that spreads through multiple dimensions in order to separate different classes in the data. The hyperplane is created in such a way that the distance between any two classes is maximum from the plane.
SVM uses kernels to compute the distance between two classes. There are many kernels available ranging from linear to non-linear kernels. Appropriate kernel can be selected by cross validation techniques.

1.Maximal-Margin Classifier The Maximal-Margin Classifier is a hypothetical classifier that best explains how SVM works in practice.

The numeric input variables (x) in your data (the columns) form an n-dimensional space. For example, if you had two input variables, this would form a two-dimensional space.

A hyperplane is a line that splits the input variable space. In SVM, a hyperplane is selected to best separate the points in the input variable space by their class, either class 0 or class 1. In two-dimensions you can visualize this as a line and let's assume that all of our input points can be completely separated by this line. For example:

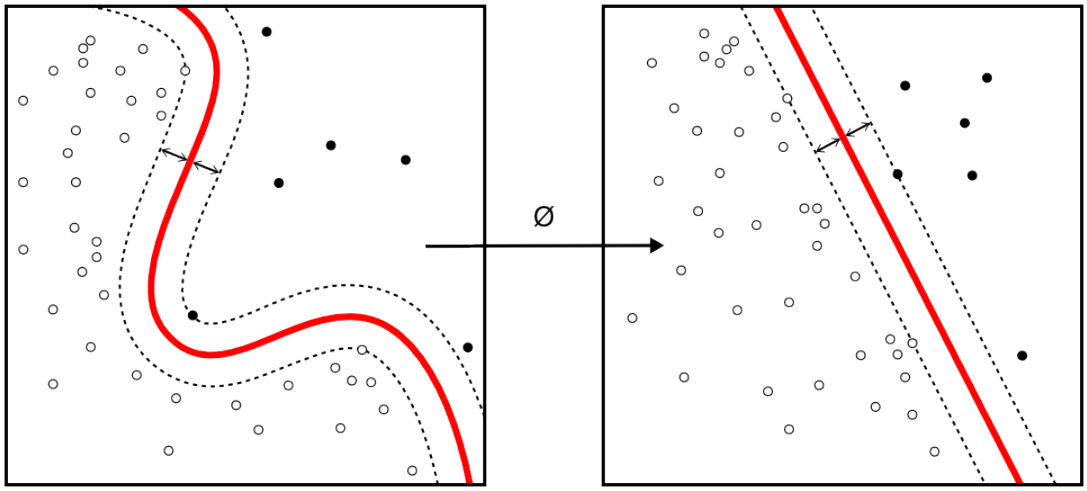

Figure 9: SVM Hyperplane Maximum Separation

B0 + (B1 * X1) + (B2 * X2) = 0

Where the coefficients (B1 and B2) that determine the slope of the line and the intercept (B0) are found by the learning algorithm, and X1 and X2 are the two input variables.

You can make classifications using this line. By plugging in input values into the line equation, you can calculate whether a new point is above or below the line.

Above the line, the equation returns a value greater than 0 and the point belongs to the first class (class 0).
Below the line, the equation returns a value less than 0 and the point belongs to the second class (class 1).
A value close to the line returns a value close to zero and the point may be difficult to classify.
If the magnitude of the value is large, the model may have more confidence in the prediction.
The distance between the line and the closest data points is referred to as the margin. The best or optimal line that can separate the two classes is the line that as the largest margin. This is called the Maximal-Margin hyperplane.

The margin is calculated as the perpendicular distance from the line to only the closest points. Only these points are relevant in defining the line and in the construction of the classifier. These points are called the support vectors. They support or define the hyperplane.

The hyperplane is learned from training data using an optimization procedure that maximizes the margin.

2.Soft Margin Classifier In practice, real data is messy and cannot be separated perfectly with a hyperplane.

The constraint of maximizing the margin of the line that separates the classes must be relaxed. This is often called the soft margin classifier. This change allows some points in the training data to violate the separating line.

An additional set of coefficients are introduced that give the margin wiggle room in each dimension. These coefficients are sometimes called slack variables. This increases the complexity of the model as there are more parameters for the model to fit to the data to provide this complexity.

A tuning parameter is introduced called simply C that defines the magnitude of the wiggle allowed across all dimensions. The C parameters defines the amount of violation of the margin allowed. A C=0 is no violation and we are back to the inflexible Maximal-Margin Classifier described above. The larger the value of C the more violations of the hyperplane are permitted.

During the learning of the hyperplane from data, all training instances that lie within the distance of the margin will affect the placement of the hyperplane and are referred to as support vectors. And as C affects the number of instances that are allowed to fall within the margin, C influences the number of support vectors used by the model.

The smaller the value of C, the more sensitive the algorithm is to the training data (higher variance and lower bias).
The larger the value of C, the less sensitive the algorithm is to the training data (lower variance and higher bias).

3.Support Vector Machines (Kernels) The SVM algorithm is implemented in practice using a kernel.

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra, which is out of the scope of this introduction to SVM.

A powerful insight is that the linear SVM can be rephrased using the inner product of any two given observations, rather than the observations themselves. The inner product between two vectors is the sum of the multiplication of each pair of input values.

For example, the inner product of the vectors [2, 3] and [5, 6] is 2*5 + 3*6 or 28.

The equation for making a prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

f(x) = B0 + sum(ai * (x,xi))

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm.

- **Linear Kernel SVM** The dot-product is called the kernel and can be re-written as:

  K(x, xi) = sum(x * xi)

  The kernel defines the similarity or a distance measure between new data and the support vectors. The dot product is the similarity measure used for linear SVM or a linear kernel because the distance is a linear combination of the inputs.

  Other kernels can be used that transform the input space into higher dimensions such as a Polynomial Kernel and a Radial Kernel. This is called the Kernel Trick.

  It is desirable to use more complex kernels as it allows lines to separate the classes that are curved or even more complex. This in turn can lead to more accurate classifiers.

- **Polynomial Kernel SVM** Instead of the dot-product, we can use a polynomial kernel, for example:

  K(x,xi) = 1 + sum(x * xi)$\hat{d}$

  Where the degree of the polynomial must be specified by hand to the learning algorithm. When d=1 this is the same as the linear kernel. The polynomial kernel allows for curved lines in the input space.

- **Radial Kernel SVM** Finally, we can also have a more complex radial kernel. For example:

  K(x,xi) = exp(-gamma * sum((x – xi$\hat{2}$))

  Where gamma is a parameter that must be specified to the learning algorithm. A good default value for gamma is 0.1, where gamma is often 0 ¡ gamma ¡ 1. The radial kernel is very local and can create complex regions within the feature space, like closed polygons in two-dimensional space.

4.Advantages and Disadvantages Advantages:

1. It works really well with clear margin of separation
2. It is effective in high dimensional spaces.
3. It is effective in cases where number of dimensions is greater than the number of samples.
4. It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

Disadvantages:

1. It doesn't perform well, when we have large data set because the required training time is higher.
2. It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping.
3. SVM doesn't directly provide probability estimates.

5.6.3 Classification and Regression Trees (CART)

Decision Trees are an important type of algorithm for predictive modeling machine learning.

The classical decision tree algorithms have been around for decades and modern variations like random forest are among the most powerful techniques available.

The representation for the CART model is a binary tree.

This is your binary tree from algorithms and data structures, nothing too fancy. Each root node represents a single input variable (x) and a split point on that variable (assuming the variable is numeric).

The leaf nodes of the tree contain an output variable (y) which is used to make a prediction.

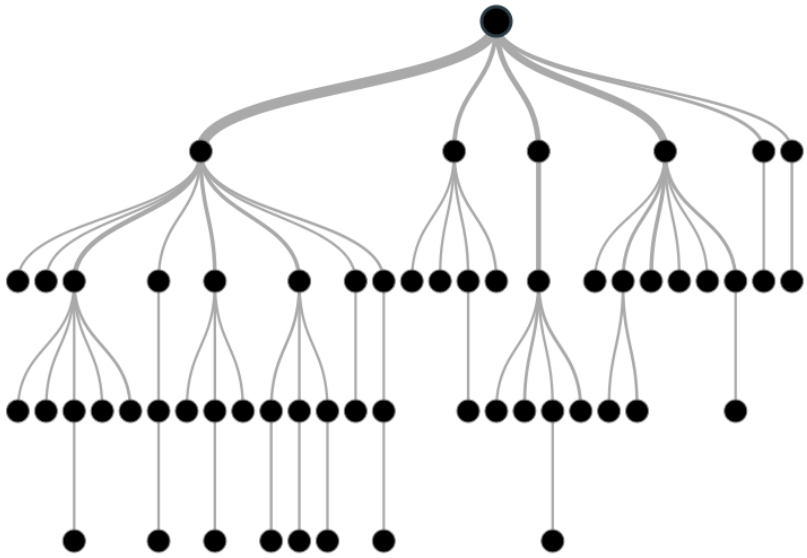

Figure 10: Decision Tree

1.Learn a CART Model From Data Creating a CART model involves selecting input variables and split points on those variables until a suitable tree is constructed.

The selection of which input variable to use and the specific split or cut-point is chosen using a greedy algorithm to minimize a cost function. Tree construction ends using a predefined stopping criterion, such as a minimum number of training instances assigned to each leaf node of the tree.

2.Greedy Splitting Creating a binary decision tree is actually a process of dividing up the input space. A greedy approach is used to divide the space called recursive binary splitting.

This is a numerical procedure where all the values are lined up and different split points are tried and tested using a cost function. The split with the best cost (lowest cost because we minimize cost) is selected.

All input variables and all possible split points are evaluated and chosen in a greedy manner (e.g. the very best split point is chosen each time).

For regression predictive modeling problems the cost function that is minimized to choose spit points is the sum squared error across all training samples that fall within the rectangle:

sum(y – prediction)$\hat{2}$

Where y is the output for the training sample and prediction is the predicted output for the rectangle.

For classification the Gini cost function is used which provides an indication of how "pure" the leaf nodes are (how mixed the training data assigned to each node is).

G = sum(pk * (1 – pk))

Where G is the Gini cost over all classes, pk are the number of training instances with class k in the rectangle of interest. A node that has all classes of the same type (perfect class purity) will have G=0, where as a G that has a 50-50 split of classes for a binary classification problem (worst purity) will have a G=0.5.

3.Stopping Criterion The recursive binary splitting procedure described above needs to know when to stop splitting as it works its way down the tree with the training data.

The most common stopping procedure is to use a minimum count on the number of training instances assigned to each leaf node. If the count is less than some minimum then the split is not accepted and the node is taken as a final leaf node.

The count of training members is tuned to the dataset, e.g. 5 or 10. It defines how specific to the training data the tree will be. Too specific (e.g. a count of 1) and the tree will overfit the training data and likely have poor performance on the test set.

4.Pruning The Tree The stopping criterion is important as it strongly influences the performance of your tree. You can use pruning after learning your tree to further lift performance.

The complexity of a decision tree is defined as the number of splits in the tree. Simpler trees are preferred. They are easy to understand (you can print them out and show them to subject matter experts), and they are less likely to overfit your data.

The fastest and simplest pruning method is to work through each leaf node in the tree and evaluate the effect of removing it using a hold-out test set. Leaf nodes are removed only if it results in a drop in the overall cost function on the entire test set. You stop removing nodes when no further improvements can be made.

More sophisticated pruning methods can be used such as cost complexity pruning (also called weakest link pruning) where a learning parameter (alpha) is used to weigh whether nodes can be removed based on the size of the sub-tree.

5.Advantages and Disadvantages Advantages:

1. Decision trees implicitly perform variable screening or feature selection.
2. Decision trees require relatively little effort from users for data preparation.
3. Nonlinear relationships between parameters do not affect tree performance.
4. The best feature of using trees for analytics. They are easy to interpret and explain to executives.

Disadvantages:

1. Decision Trees do not work well if you have smooth boundaries. i.e they work best when you have discontinuous piece wise constant model. If you truly have a linear target function decision trees are not the best.

2. Decision Tree's do not work best if you have a lot of un-correlated variables. Decision tree's work by finding the interactions between variables. if you have a situation where there are no interactions between variables linear approaches might be the best.

3. Data fragmentation : Each split in a tree leads to a reduced dataset under consideration. And, hence the model created at the split will potentially introduce bias.

4. High variance and unstable : As a result of the greedy strategy applied by decision tree's variance in finding the right starting point of the tree can greatly impact the final result. i.e small changes early on can have big impacts later. So- if for example you draw two different samples from your universe , the starting points for both the samples could be very different (and may even be different variables) this can lead to totally different results.

6 Testing

Verification and validation is an important phase in the development life cycle of the product.This is the phase where the errors are detected. Hence testing or verification and validation perform a very critical role for quality assurance and ensuring the reliability of the software. One definition of testing is "The process of questioning a product in order to evaluate it", where the "questions" are operations the tester attempts to execute with the product, and the product answers with its behavior in reaction to the probing of the tester.

6.1 Goals of Testing

1.The foremost and basic intention behind Software Testing is to make sure that the application under test is free of bugs and errors and in case it is not, it should be identified and sent for rectification. In our Android Application, we have checked every feature in our application (like faculty login, student login, attempting the quiz, etc.) and got the desired results.

2.Software testing plays a vital role to improve the quality of the deliverable by reporting bugs at different phases of development depending on the methodology followed for the project. In our Android Application, testing helped us identify several errors, the biggest among them was that earlier we were not able to run our application on an actual device but just on an emulator. When we debugged it properly, we were able to remove such errors.
3.Software testing brings with it a source of reliability that is important for any business to get hold of the market.In our initial versions of the Android Application, we did not include the code to close the application properly, system testing helped us to identify such flaws and we rectified it in the end, and helped us in making the application more reliable.
4.Software testing ensures whether the application is ready to deliver to the client or not, so it aims to find the critical bugs early in the testing phase.We tested each incremental update of our application, identified all types of errors, rectified them until we got the best possible version of the system that can be delivered to the clients.

5.The goal aims to ensure the flaws in application; methodologies used and are put forward in front of management in order to get it resolved. In our Project Development and management task were performed by the same people, hence all the flaws identified were rectified by the same.

6.2 Levels of Testing

6.2.1 Unit Testing

Individual components are tested to ensure that they operate correctly. Each component is tested independently. Each module of this application was tested against a set of inputs and the results were checked with the expected output. Unit testing focuses on verification effort on the smallest unit of the software design module. This is also known as MODULE TESTING. This testing is carried out during phases, each module is found to be working satisfactorily as regards to the expected output from the module.

6.2.2 Integration Testing

Integration testing is another aspect of testing that is generally done in order to uncover errors associated with the flow of data across interfaces. The unit-tested modules of the application are grouped together and tested in small segment, which makes it easier to isolate and correct errors. This approach is continued until we have integrated all modules to form the android application as a whole.

6.2.3 System Testing

System testing is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. We evaluated our application's compliance with the specified requirements.

6.2.4 Validation Testing

The validation testing can be defined in many ways, but a simple definition is that, validation succeeds when the software functions in a manner that can be reasonably expected by the end user. The application's results were as expected.

6.3 Problem of Overfitting and Underfitting

1 .Generalization in Machine Learning:

In machine learning we describe the learning of the target function from training data as inductive learning.

Induction refers to learning general concepts from specific examples which is exactly the problem that supervised machine learning problems aim to solve. This is different from deduction that is the other way around and seeks to learn specific concepts from general rules.

Generalization refers to how well the concepts learned by a machine learning model apply to specific examples not seen by the model when it was learning.

The goal of a good machine learning model is to generalize well from the training data to any data from the problem domain. This allows us to make predictions in the future on data the model has never seen.

There is a terminology used in machine learning when we talk about how well a machine learning model learns and generalizes to new data, namely overfitting and underfitting.

Overfitting and underfitting are the two biggest causes for poor performance of machine learning algorithms.

2. Statistical Fit

In statistics, a fit refers to how well you approximate a target function.

This is good terminology to use in machine learning, because supervised machine learning algorithms seek to approximate the unknown underlying mapping function for the output variables given the input variables.

Statistics often describe the goodness of fit which refers to measures used to estimate how well the approximation of the function matches the target function.

Some of these methods are useful in machine learning (e.g. calculating the residual errors), but some of these techniques assume we know the form of the target function we are approximating, which is not the case in machine learning.

If we knew the form of the target function, we would use it directly to make predictions, rather than trying to learn an approximation from samples of noisy training data.

3. Overfitting in Machine Learning

Overfitting refers to a model that models the training data too well.

Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.

Overfitting is more likely with nonparametric and nonlinear models that have more flexibility when learning a target function. As such, many nonparametric machine learning algorithms also include parameters or techniques to limit and constrain how much detail the model learns.

For example, decision trees are a nonparametric machine learning algorithm that is very flexible and is subject to overfitting training data. This problem can be addressed by pruning a tree after it has learned in order to remove some of the detail it has picked up.

4. Underfitting in Machine Learning

Underfitting refers to a model that can neither model the training data nor generalize to new data.

An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

Underfitting is often not discussed as it is easy to detect given a good performance metric. The remedy is to move on and try alternate machine learning algorithms. Nevertheless, it does provide a good contrast to the problem of overfitting.

5. A Good Fit in Machine Learning

Ideally, you want to select a model at the sweet spot between underfitting and overfitting.

This is the goal, but is very difficult to do in practice.

To understand this goal, we can look at the performance of a machine learning algorithm over time as it is learning a training data. We can plot both the skill on the training data and the skill on a test dataset we have held back from the training process.

Over time, as the algorithm learns, the error for the model on the training data goes down and so does the error on the test dataset. If we train for too long, the performance on the training dataset may continue to decrease because the model is overfitting and learning the irrelevant detail and noise in the training dataset. At the same time the error for the test set starts to rise again as the model's ability to generalize decreases.

The sweet spot is the point just before the error on the test dataset starts to increase where the model has good skill on both the training dataset and the unseen test dataset.

You can perform this experiment with your favorite machine learning algorithms. This is often not useful technique in practice, because by choosing the stopping point for training using the skill on the test dataset it means that the testset is no longer "unseen" or a standalone objective measure. Some knowledge (a lot of useful knowledge) about that data has leaked into the training procedure.

There are two additional techniques you can use to help find the sweet spot in practice: resampling methods and a validation dataset.

6. How To Limit Overfitting

Both overfitting and underfitting can lead to poor model performance. But by far the most common problem in applied machine learning is overfitting.

Overfitting is such a problem because the evaluation of machine learning algorithms on training data is different from the evaluation we actually care the most about, namely how well the algorithm performs on unseen data.

6.4 Solutions To Underfitting and Overfitting

The solution to over fitting and underfitting is to use resampling techniques. This resampling technique is called cross validation which is discussed below.

Cross validation is a model evaluation method that is better than residuals. The problem with residual evaluations is that they do not give an indication of how well the learner will do when it is asked to make new predictions for data it has not already seen. One way to overcome this problem is to not use the entire data set when training a learner. Some of the data is removed before training begins. Then when

training is done, the data that was removed can be used to test the performance of the learned model on "new" data. This is the basic idea for a whole class of model evaluation methods called cross validation.

Types of Cross validation techniques:

1. Holdout method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The function approximator fits a function using the training set only. Then the function approximator is asked to predict the output values for the data in the testing set (it has never seen these output values before). The errors it makes are accumulated as before to give the mean absolute test set error, which is used to evaluate the model. The advantage of this method is that it is usually preferable to the residual method and takes no longer to compute. However, its evaluation can have a high variance. The evaluation may depend heavily on which data points end up in the training set and which end up in the test set, and thus the evaluation may be significantly different depending on how the division is made.

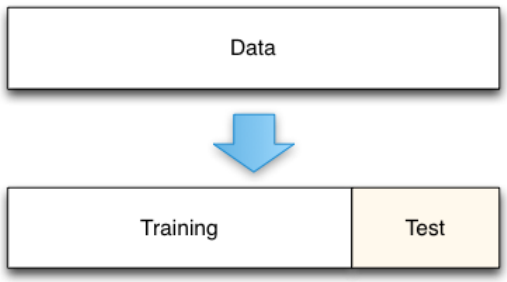

Figure 11: Holdout Method Cross Validation

2. K-fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. The variance of the resulting estimate is reduced as k is increased. The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation. A variant of this method is to randomly divide the data into a test and training set k different times. The advantage of doing this is that you can independently choose how large each test set is and how many trials you average over.

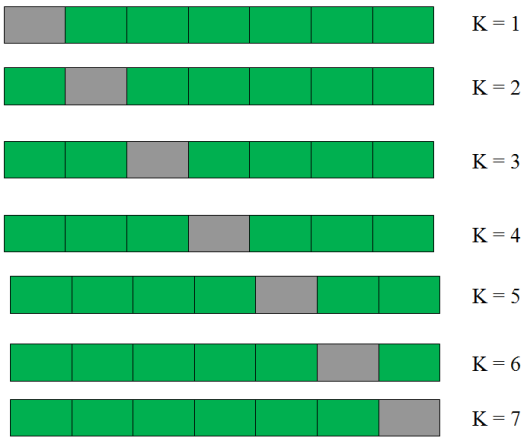

Figure 12: k-folds Cross validation

3. Leave-one-out cross validation is K-fold cross validation taken to its logical extreme, with K equal to N, the number of data points in the set. That means that N separate times, the function approximator is trained on all the data except for one point and a prediction is made for that point. As before the average error is computed and used to evaluate the model. The evaluation given by leave-one-out cross validation error (LOO-XVE) is good, but at first pass it seems very expensive to compute. Fortunately, locally weighted learners can make LOO predictions just as easily as they make regular predictions. That means computing the LOO-XVE takes no more time than computing the residual error and it is a much better way to evaluate models. We will see shortly that Vizier relies heavily on LOO-XVE to choose its metacodes.

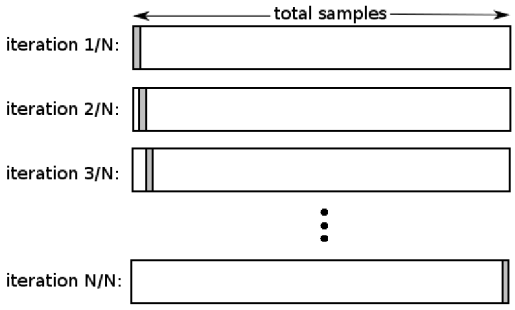

Figure 13: Leave one out cross validation

7 Experiments and Results

Example1:

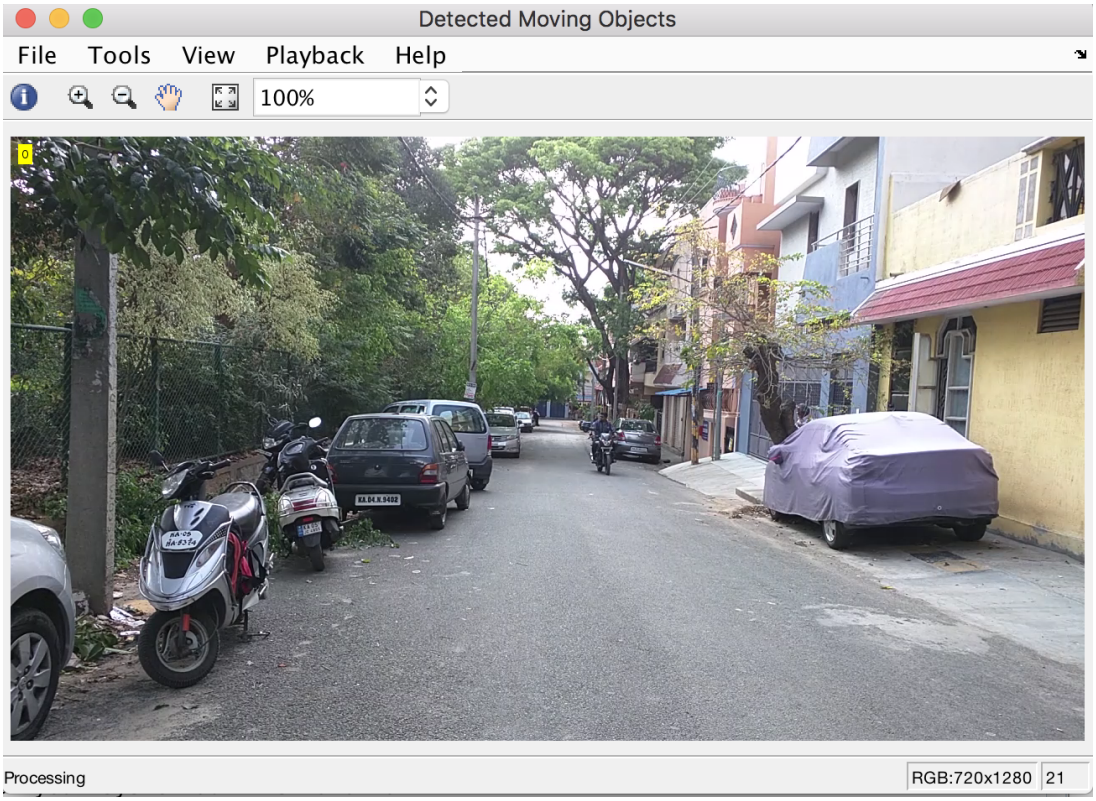

Figure 14: Image shows the input video input.

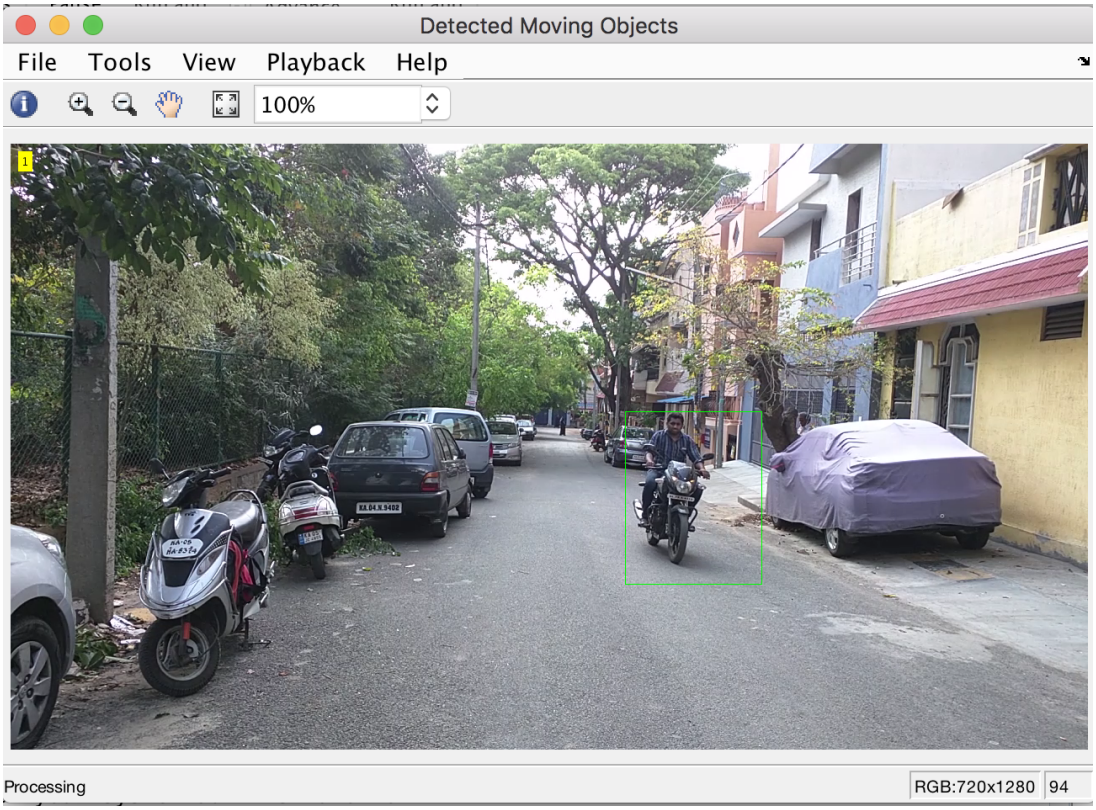

Figure 15: Image shows the bounding box created around the moving objects

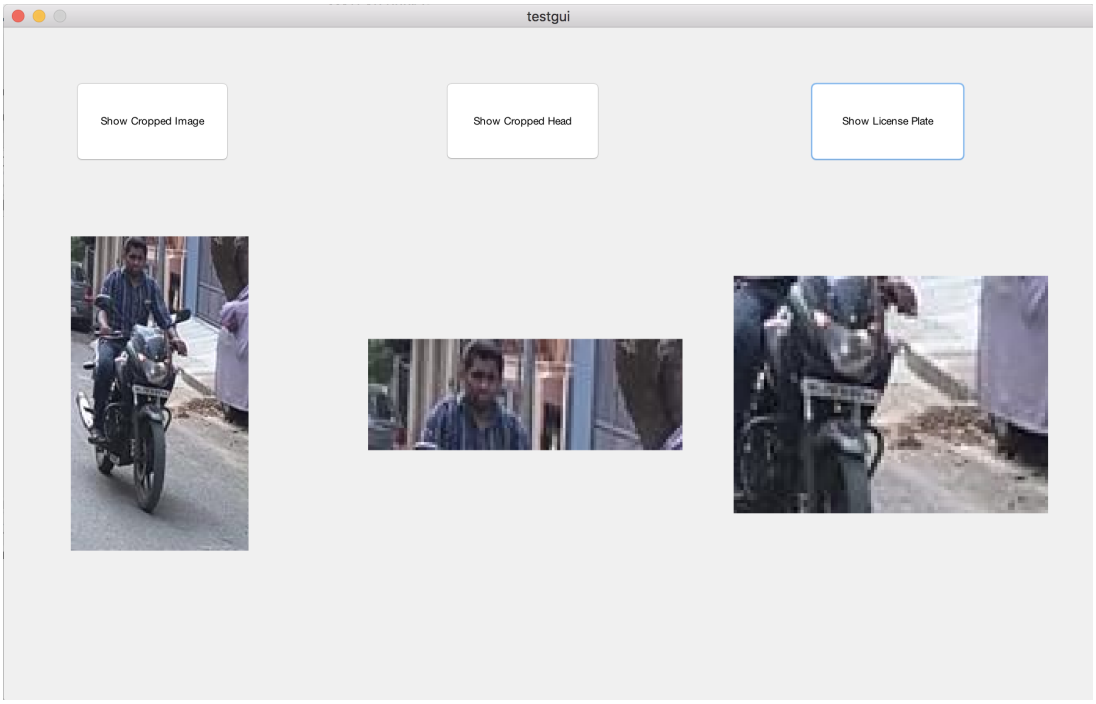

Figure 16: GUI results

This image shows the cropped image of the moving object.

The second image is the cropped head region after it is classified as a motorcycle.

The third image is the license plate region of the vehicle which is cropped if the algorithm find that a helmet is not present.

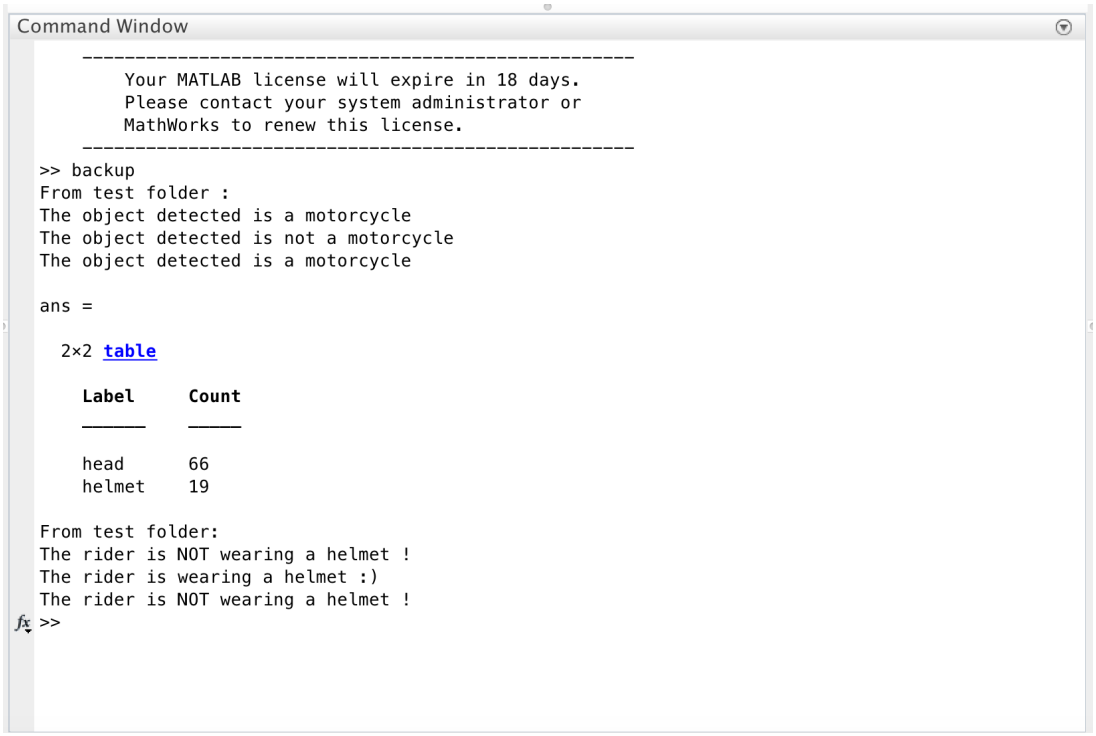

Figure 17: Command window data

The results of the algorithm on test data are displayed on the command window.

Example 2:

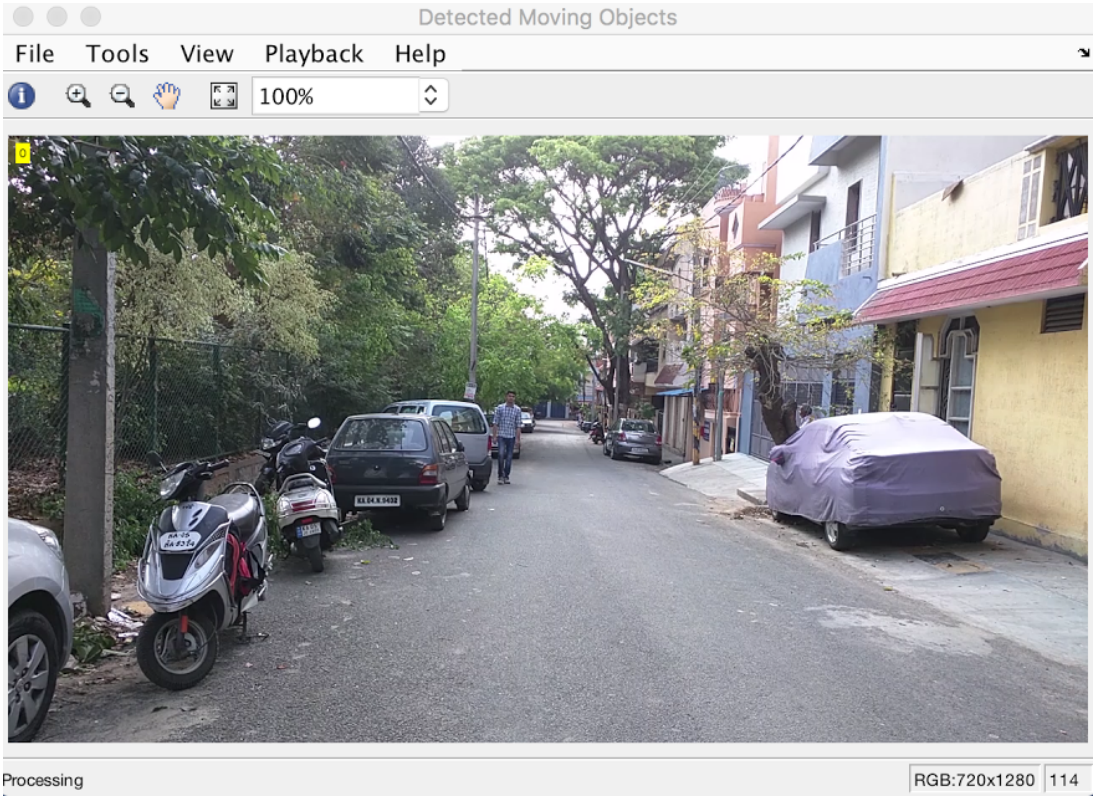

Figure 18: Background of video

This image above shows the background where the video was taken.

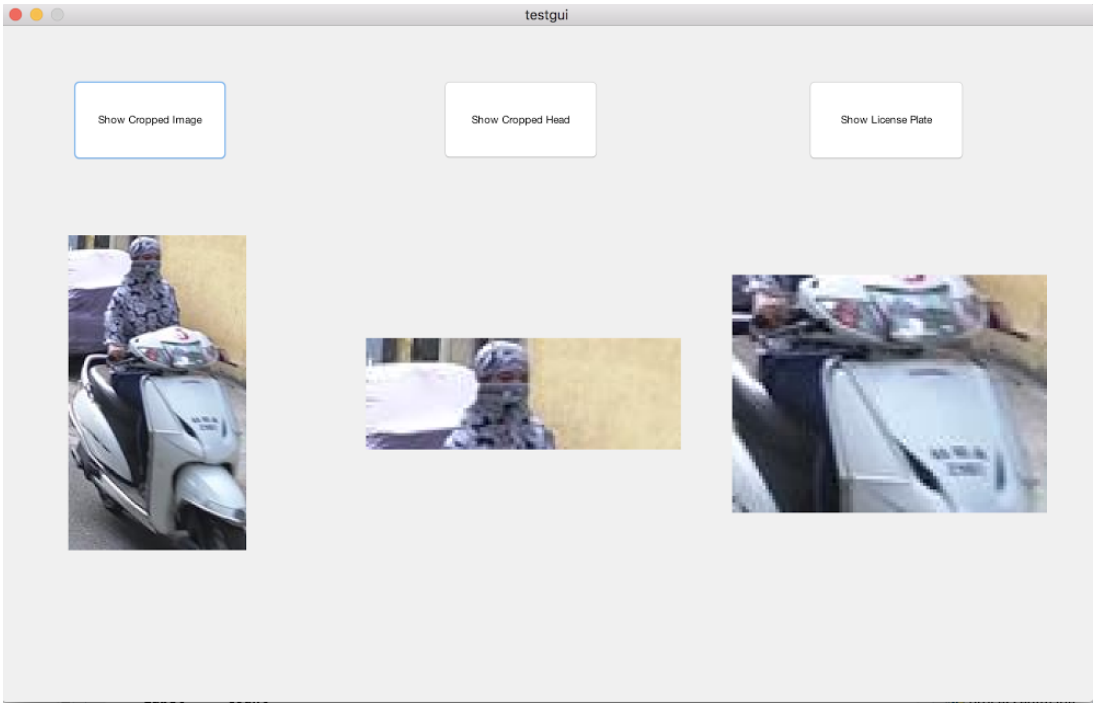

Figure 19: User Interface showing classification

The above image shows the GUI where the first shows the cropped image of the moving object being detected.

The second shows the cropped head in case the moving object is classified as motorcycle.

The third shows the cropped license plate in case the helmet is not found to be present.

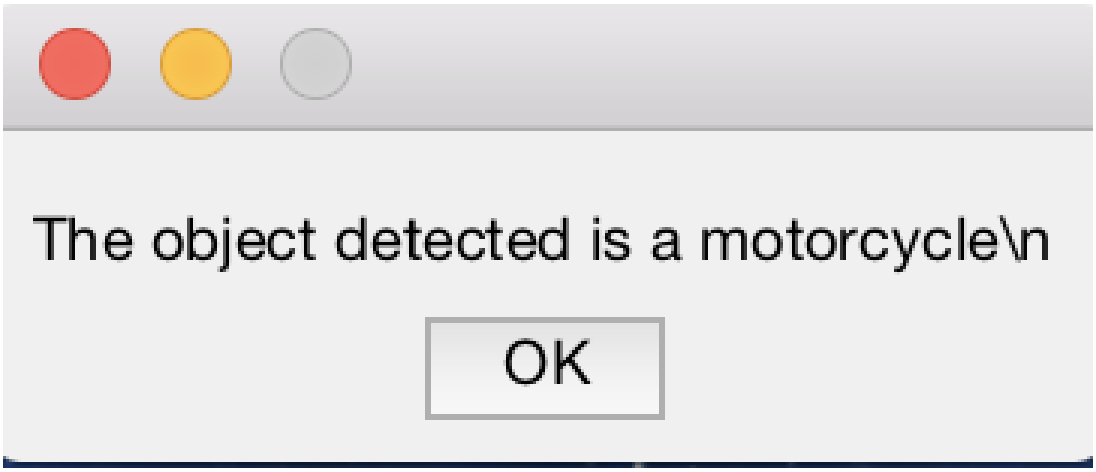

Figure 20: Dialog box that classifies image

The image above shows a dialog box showing the classification of object either into motorcycle or non-motorcycle.

Figure 21: Dialog box that detects helmet

The image above shows a dialog box showing the classification of detected two wheeler wearing a helmet or not.

Example 3:

Figure 22: Dialog Box 1

The image above shows a dialog box showing the classification of object either into motorcycle or non-motorcycle.

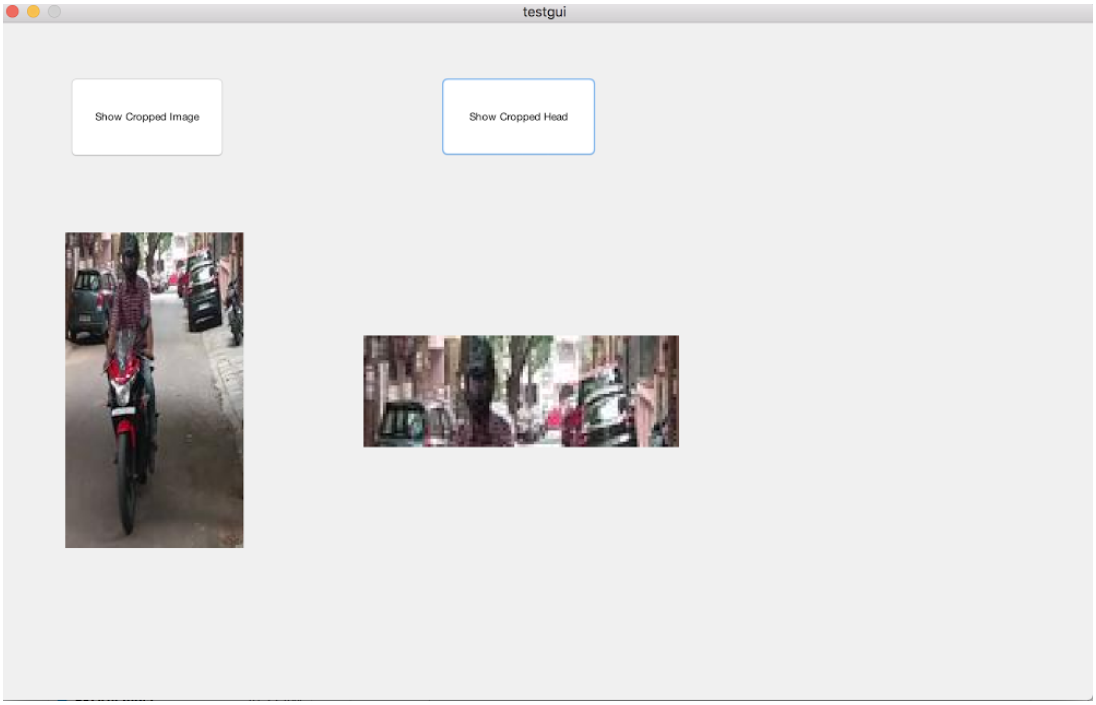

Figure 23: GUI

The above image shows the GUI where the first shows the cropped image of the moving object being detected.

The second shows the cropped head in case the moving object is classified as motorcycle.

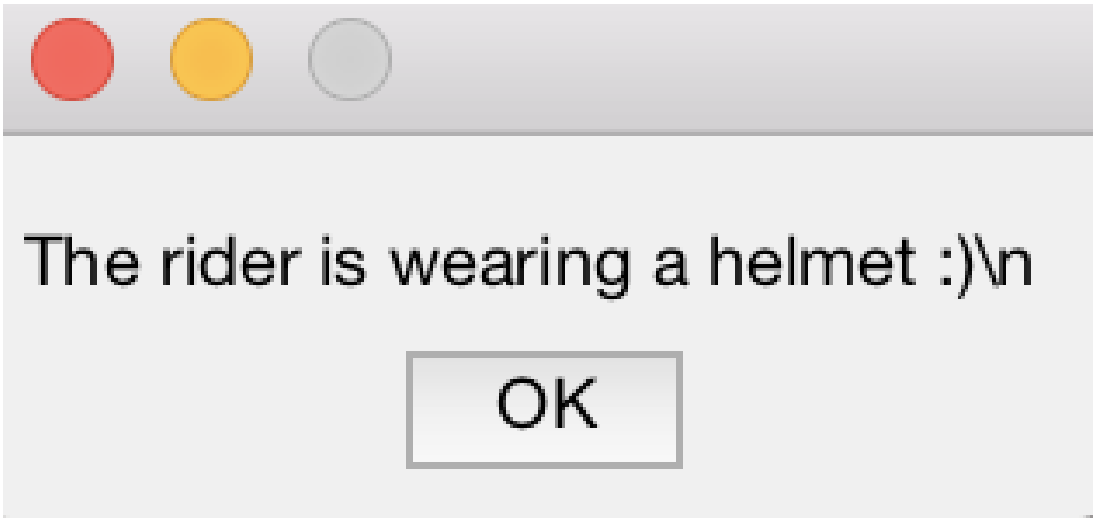

Figure 24: Dialog Box 2

The image above displays that the rider is wearing a helmet.

Statistics of results using R

Training data was collected using a smartphone camera. For two wheeler classification, a total of 255 images were captured, out of which 130 were two wheelers and 125 were four wheelers. Similarly, 230 images were collected for helmet classification task, 138 images of head without helmet and 92 pictures of head with helmet.

The accuracy of the SVM classifier was measured in statistical tool called R. The optimum value for tuning parameters of the classifier were achieved using 10-fold cross validation. The accuracy of the model on validation set for two wheeler classification and helmet detection was 97.26% and 99.13% respectively. The accuracy of the model on test data was 88% for two wheeler classification and 60% for helmet detection.

8 Conclusion

The system successfully takes in a video input of traffic conditions on public roads. It then detects moving objects and classifies them into either two-wheelers or non two-wheelers. In case of a two-wheeler, the head region is cropped and subsequently the system detects if a helmet is present or not. If helemet is not detected, the license plate region of the vehicle is cropped.

The system performs extremely well on validation set for both, two wheeler classification and helmet detection. As far as test set performance is concerned, the system achieves a very high accuracy on the two wheeler classification but the performance while detecting helmet is average. This can be improved by gathering more training data.

The system can also be employed to detect vehicles moving in both directions. We can also implement different types of classifiers and compare their accuracies. The license plate information can be extracted using OCR techniques and a ticket can be sent to the owner of the vehicle.

REFERENCES

1. International Journal of Engineering Trends and Technology (IJETT) - Vol 35 Number-5 5 May,2016

2. IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834, - ISSN: 2278-8735.Volume 10, Issue 4, Ver. II (Jul - Aug .2015), PP 55-59 www.iosrjournals.org

3. CLEI ELECTRONIC JOURNAL, VOLUME 16, NUMBER 3, PAPER 04, DECEMBER 2013

4. ISSN 2319-2518, No.2, Vol 4, April 2015

5. CLEI ELECTRONIC JOURNAL, VOLUME 16

6. NUMBER 3, PAPER 04, DECEMBER 2013

7. Face Recognition using Local Binary Patterns (LBP) By Md. Abdur Rahim, Md. Najmul Hossain, Tanzillah Wahid Pabna University of Science and Technology,Bangladesh

8. M. Fathy and M. Y. Siyal, Senior Member, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 47, NO. 4, NOVEMBER 1998 9. Algorithms for Real Time Object Detection in Images, Milos Stojmenovic 10. Automatic Motorcycle Detection on Public Roads, Romuere Silva, Kelson Aires, Rodrigo Veras,

Thiago Santos, Kalyf Lima and Andre Soaress, CLEI ELECTRONIC JOURNAL, VOLUME 16, NUMBER 3, PAPER 04, DECEMBER 2013

11. Helmet presence classification with motorcycle detection and tracking J.Chiverton School of Information Technology, Mae Fah Luang University

12. G.S. Gopika, R. Monisha and S. Karthik - “Extraction and Segmentation of Helmets in Motor Cycles”

13. Rattapoom Waranusast, Nannaphat Bundon, Vasan Timtong and Chainarong Tangnoi - “Machine Vision Techniques for Motorcycle Safety Helmet Detection”

14. Che-Yen Wen, Shih-Hsuan Chiu, Jiun-Jian Liaw, ChuawPin Lu - “The safety helmet detection for ATM’ s surveillance system via the modified Hough transform”

15. Visual Motorcycle Detection and Tracking Algorithms,MIN-YU KU, CHUNG-CHENG CHIU, HUNG-TSUNG CHEN, SHUN-HUANG HONG, Department of Electrical and Electronic Engineering, Chung-Cheng Institute of Technology

16. Pratiksha Jain, Neha Chopra, Vaishali Gupta - “Automatic License Plate Recognition using OpenCV

APPENDIX

Project Code

{main.m}

```
   clear;
   foregroundDetector = vision.ForegroundDetector('NumGaussians', 3, ...
    'NumTrainingFrames', 50);
videoReader = vision.VideoFileReader('vid2.mp4');

for i = 1:180
    frame = step(videoReader); % read the next video frame
    %figure; imshow(frame); title('Video Frame');
    foreground = step(foregroundDetector, frame);
end

%figure; imshow(frame); title('Video Frame');
%figure; imshow(foreground); title('Foreground');

se = strel('square', 3);
filteredForeground = imopen(foreground, se);
%figure; imshow(filteredForeground); title('Clean Foreground');
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', true, 'CentroidOutputPort', true, ...
    'MinimumBlobArea', 2000);
[areas, centroids, bbox] = step(blobAnalysis, filteredForeground);



result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');
numCars = size(bbox, 1);
result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...
    'FontSize', 14);
%figure; imshow(result; title('Detected Moving Objects');
videoPlayer = vision.VideoPlayer('Name', 'Detected Moving Objects');
videoPlayer.Position(3:4) = [650,400]; % window size: [width, height]
se = strel('square', 3); % morphological filter for noise removal
x = 0;
%release(videoReader); % close the video file
%videoReader = vision.VideoFileReader('vid2.mp4');
```

```
while ~isDone(videoReader)
    x = x + 1;
    frame = step(videoReader); % read the next video frame

    % Detect the foreground in the current video frame
    foreground = step(foregroundDetector, frame);

    % Use morphological op?ove noise in the foreground
    filteredForeground = imopen(foreground, se);

    % Detect the?????????
%ents with the specified
%minimum area, and
    % compute their bounding boxes
    [areas, centroids, bbox] = step(blobAnalysis, filteredForeground);
    if(mod(x,100) == 0)
        %disp(x);
        %for i=1:size(bbox, 1)
        %centroids(i,:) = [ bbox(i,1)+bbox(i,3)/2 ; bbox(i,2)+bbox(i,4)/2 ];
        %areas(i,1) = bbox(i,3)*bbox(i,4);
        %areas(:,1)
        %disp('hello');
        cropped = imcrop(frame, bbox);

        %imshow(cropped);
        cd 'test'
        cropped = imresize(cropped,[200 113]);
        imwrite(cropped,'cropped.jpg');
        testgui;
        cd ..
        %info = imfinfo('cropped.jpg');

        %cropped = imread('cropped.jpg');
        %figure; imshow(cropped); title('cropped');
        %{
        height = size(cropped,1);
        width = size(cropped,2);


        ar = width/ height;
```

```
        disp(ar);
        %}

        pause(3);
        %disp('hello222');
        %end

    end
    % Draw bounding boxes around the detected cars
    result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

    % Display the number of cars found in the video frame
    numCars = size(bbox, 1);
    result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...
        'FontSize', 14);

    step(videoPlayer, result); % display the results
end

release(videoReader); % close the video file
syntheticDir  = fullfile('train');
handwrittenDir = fullfile('test');
trainingSet = imageDatastore(syntheticDir, 'IncludeSubfolders', true, 'LabelSource',
    'foldernames');
testSet    = imageDatastore(handwrittenDir, 'IncludeSubfolders', true, 'LabelSource',
    'foldernames');
%testSet = cropped;
countEachLabel(trainingSet);
%countEachLabel(testSet)


%figure;
%{
subplot(2,2,1);
imshow(trainingSet.Files{9});

subplot(2,2,2);
imshow(trainingSet.Files{4});

%subplot(2,3,3);
```

```
%imshow(trainingSet.Files{7});

subplot(2,2,3);
imshow(testSet.Files{1});

subplot(2,2,4);
imshow(testSet.Files{2});

%subplot(2,3,6);
%imshow(testSet.Files{97});
%}
exTestImage = readimage(testSet,2);
%exTestImage = cropped;
processedImage = imbinarize(rgb2gray(exTestImage));

%figure;
%{
subplot(1,2,1)
imshow(exTestImage)

subplot(1,2,2)
imshow(processedImage)
%}
img = readimage(trainingSet, 99);
%figure;imshow(img);

% Extract HOG features and HOG visualization
[hog_8x8, vis8x8] = extractHOGFeatures(img,'CellSize',[8 8]);
[hog_16x16, vis16x16] = extractHOGFeatures(img,'CellSize',[16 16]);
[hog_32x32, vis32x32] = extractHOGFeatures(img,'CellSize',[32 32]);

% Show the original image
%figure;
%subplot(2,3,1:3); imshow(img);

% Visualize the HOG features
%{
subplot(2,3,4);
plot(vis8x8);
title({'CellSize = [8 8]'; ['Length = ' num2str(length(hog_8x8))]});
```

```
subplot(2,3,5);
plot(vis16x16);
title({'CellSize = [16 16]'; ['Length = ' num2str(length(hog_16x16))]});

subplot(2,3,6);
plot(vis32x32);
title({'CellSize = [32 32]'; ['Length = ' num2str(length(hog_32x32))]});
%}
cellSize = [16 16];
hogFeatureSize = length(hog_16x16);
% Loop over the trainingSet and extract HOG features from each image.
% similar procedure will be used to extract features from the testSet.
numImages = numel(trainingSet.Files);\\
trainingFeatures = zeros(numImages, hogFeatureSize, 'single');

for i = 1:numImages
    img = readimage(trainingSet, i);

    img = rgb2gray(img);

    % Apply pre-processing steps
    img = imbinarize(img);

    trainingFeatures(i, :) = extractHOGFeatures(img, 'CellSize', cellSize);
end

% Get labels for each image.
trainingLabels = trainingSet.Labels;
%disp(trainingLabels)
% fitcecoc uses SVM learners and a 'One-vs-One' encoding scheme.

classifier = fitcecoc(trainingFeatures, trainingLabels);
% Extract HOG features from the test set. The procedure is similar to what
% was shown earlier and is encapsulated as a helper function for brevity.

[testFeatures,testLabels] =
    helperExtractHOGFeaturesFromImageSet(testSet,hogFeatureSize,cellSize);

% Make class predictions using the test features.
```

```
predictedLabels = predict(classifier, testFeatures);

disp("From test folder :");
for i = 1:size(predictedLabels)

   if(predictedLabels(i) == "motorcycle")
        %h = msgbox('The object detected is a motorcycle\n');
        fprintf("The object detected is a motorcycle\n");

   else
  fprintf("The object detected is not a motorcycle\n");

   end
end

testgui;

if predictedLabels(3)== "motorcycle"

X2 = imcrop(cropped, [0 0 113 40]);
%figure;
%imshow(X2);title('cropped head')
cd 'test1'
croppedhead = imresize(X2,[40 113]);
imwrite(X2,'croppedhead.jpg');
testgui;
cd ..
syntheticDir  = fullfile('train1');
handwrittenDir = fullfile('test1');
trainingSet = imageDatastore(syntheticDir, 'IncludeSubfolders', true, 'LabelSource',
    'foldernames');
testSet    = imageDatastore(handwrittenDir, 'IncludeSubfolders', true, 'LabelSource',
    'foldernames');
%testSet = cropped;
countEachLabel(trainingSet)
%countEachLabel(testSet)

%{
figure;
```

```
subplot(2,2,1);
imshow(trainingSet.Files{9});

subplot(2,2,2);
imshow(trainingSet.Files{4});

%subplot(2,3,3);
%imshow(trainingSet.Files{7});

subplot(2,2,3);
imshow(testSet.Files{1});

subplot(2,2,4);
imshow(testSet.Files{2});

%subplot(2,3,6);
%imshow(testSet.Files{97});

exTestImage = readimage(testSet,2);
%exTestImage = cropped;
processedImage = imbinarize(rgb2gray(exTestImage));

figure;

subplot(1,2,1)
imshow(exTestImage)

subplot(1,2,2)
imshow(processedImage)
%}
img = readimage(trainingSet, 20)
%figure;imshow(img);

% Extract HOG features and HOG visualization
[hog_8x8, vis8x8] = extractHOGFeatures(img,'CellSize',[8 8]);
[hog_16x16, vis16x16] = extractHOGFeatures(img,'CellSize',[16 16]);
[hog_4x4, vis4x4] = extractHOGFeatures(img,'CellSize',[4 4]);
%{\\
% Show the original image
```

```
figure;
subplot(2,3,1:3); imshow(img);

% Visualize the HOG features
subplot(2,3,4);
plot(vis8x8);
title({'CellSize = [8 8]'; ['Length = ' num2str(length(hog_8x8))]});

subplot(2,3,5);
plot(vis16x16);
title({'CellSize = [16 16]'; ['Length = ' num2str(length(hog_16x16))]});

subplot(2,3,6);
plot(vis4x4);
title({'CellSize = [4 4]'; ['Length = ' num2str(length(hog_4x4))]});
%}
cellSize = [16 16];
hogFeatureSize = length(hog_16x16);
% Loop over the trainingSet and extract HOG features from each image. A
% similar procedure will be used to extract features from the testSet.

numImages = numel(trainingSet.Files);
trainingFeatures = zeros(numImages, hogFeatureSize, 'single');

for i = 1:numImages
    img = readimage(trainingSet, i);

    img = rgb2gray(img);

    % Apply pre-processing steps
    img = imbinarize(img);

    trainingFeatures(i, :) = extractHOGFeatures(img, 'CellSize', cellSize);
end

% Get labels for each image.
trainingLabels = trainingSet.Labels;
%disp(trainingLabels)
% fitcecoc uses SVM learners and a 'One-vs-One' encoding scheme.
classifier = fitcecoc(trainingFeatures, trainingLabels);
```

```
% Extract HOG features from the test set. The procedure is similar to what
% was shown earlier and is encapsulated as a helper function for brevity.

[testFeatures,testLabels] =
    helperExtractHOGFeaturesFromImageSet(testSet,hogFeatureSize,cellSize);

% Make class predictions using the test features.
predictedLabels = predict(classifier, testFeatures);

disp("From test folder:");
for i = 1:size(predictedLabels)

   if(predictedLabels(i) == "helmet")
       fprintf("The rider is wearing a helmet :)\n");
   else
      fprintf("The rider is NOT wearing a helmet !\n");
   end
end
testgui;
if (predictedLabels(3)=="head")
%X3 = imcrop(cropped, [20 65 113 70]);
testgui;
%figure;
%imshow(X3);title('license plate');
%else
 %   disp("Person is wearing helmet.");
end
%else
%    disp("Not a mototrcycle. Exiting.....")
end
\\
{testgui.m}


function varargout = testgui(varargin)
% TESTGUI MATLAB code for testgui.fig
%      TESTGUI, by itself, creates a new TESTGUI or raises the existing
%      singleton*.
%
%      H = TESTGUI returns the handle to a new TESTGUI or the handle to
```

```
%      the existing singleton*.
%
%      TESTGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TESTGUI.M with the given input arguments.
%
%      TESTGUI('Property','Value',...) creates a new TESTGUI or raises the
%      existing singleton*. Starting from the left, property value pairs are
%      applied to the GUI before testgui_OpeningFcn gets called. An
%      unrecognized property name or invalid value makes property application
%      stop. All inputs are passed to testgui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help testgui

% Last Modified by GUIDE v2.5 09-May-2017 17:46:53

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',    mfilename, ...
                   'gui_Singleton', gui_Singleton, ...
                   'gui_OpeningFcn', @testgui_OpeningFcn, ...
                   'gui_OutputFcn', @testgui_OutputFcn, ...
                   'gui_LayoutFcn', [] , ...
                   'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before testgui is made visible.
function testgui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to testgui (see VARARGIN)
 \\
% Choose default command line output for testgui
handles.output = hObject;
 \\
% Update handles structure
guidata(hObject, handles);
 \\
% UIWAIT makes testgui wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = testgui_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 % Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
hObject    handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cd 'test';
a = imread('cropped.jpg');
axes(handles.axes1);
imshow(a);
cd ..
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cd 'test1';
b = imread('croppedhead.jpg');
axes(handles.axes2)
imshow(b);
cd ..
 --- Executes on button press in pushbutton3.
{function pushbutton3_Callback(hObject, eventdata, handles)
hObject    handle to pushbutton3 (see GCBO)
 eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cd 'test';
cropped= imread('cropped.jpg');
X3 = imcrop(cropped, [20 65 113 70]);
axes(handles.axes3);
imshow(X3);
cd ..
}
```