Driver Scoring System

Kaavish Report
presented to the academic faculty
by

Wahaj Nafees Baig	wb02463
Talha Javed	tj02904
Naufil Siddiqui	ns02399
Arsalan Khawaja	ak02413
Danyal Admani	da02231

In partial fulfillment of the requirements for
Bachelor of Science
Computer Science

Dhanani School of Science and Engineering

Habib University

Spring 2020

Kaavish

This Kaavish project was supervised by:

Dr. Musabbir Majeed
Internal Supervisor
Faculty of Computer Science
Habib University

Approved by the Faculty of Computer Science on _____.

Dedication

We would like to dedicate this study to our parents who are responsible for bringing us to the point where we are able and self-sufficient. However, this study is strongly dedicated to the women in our life to whom we owe much of who we are today. Our mothers, without whom we wouldn't be the people we are today. Our sisters for their support and our future wives with whom we will share this with.

Acknowledgements

The team would like to extend the first gratitude to our supervisor, Dr. Musabbir Abdul Majeed. For not only his constant guidance, but also his compassion and kindness in dealing with whatever issues we might be facing. For keeping us motivated; for the countless advises about life in general; and for being as fair as it was possible to be. Most importantly, for not giving up on us despite our many shortcomings.

The team would also like to thank the CS faculty for equipping us with the knowledge and tools to go about attempting tasks which would seem impossible had we not taken the opportunity to study under these great professors and had they not dispensed some of their knowledge to us.

The team would also like to take this opportunity and extend our gratitude towards the Habib foundation, the Habib University for making accessible to us, the technology, the services, and the infrastructure that most universities in the city do not have.

Abstract

The World Health Organization's (WHO) global status report on road safety in 2013, 1.24 million traffic related fatalities occur around the world each year. This makes it the leading cause of death for people ages 15-29 years [1]. The WHO's "Decade of Action is Road Safety (2011-2020)" program has resulted in various initiatives around driver assistance and safety awareness which have resulted in a significant improvement in the road safety conditions. However, with the continually increasing need for mobility and exponentially growing number of registered vehicles the problem is far from solved.

An interesting direction to take in solving this problem is using the trend of self-quantification to implement driver tracking and driver scoring systems. The impact of this technology in this case will be much more significant and can help make mobility safer. In this project, we will use methodologies and techniques described by research papers in the field to develop a standalone web/hybrid app which takes as input the sensor data and tracks the motion and scores the driver after a certain number of driven kilometers. The maneuver identification and scoring will be done at the end of each trip and a score will only be determined and updated on completion of a certain travelled distance.

There is little doubt that the future of the automotive industry is electric, self-driving, ride hailing services, but it is hard to imagine a road without some percentage of human drivers, at least for a while. Driving style and driver analysis helps address that human factor.

Contents

1. Introduction

1.1 Background

The problem we hope to try to solve or contribute to solving is the problem of mobility. The current system of mobility is severely broken, with around 1.24 million automobile related fatalities every year, 98% of the fuel put in to an internal combustion engine (ICE) propelled car wasted, vehicles which carry 1.1 person on average being designed to seat 5 people, and vehicles with top speeds of 180-220 km/h being driven at average speeds of 20-30 km/h in urban centers. It is evident that problems exist in the design of the system and the design of the automobile. So, rightly so, alot of of effort is being put into the redesigning of both the system of mobility and the blueprint of the automobile.

Current solutions that are being extensively explored are, electric battery propelled vehicles, autonomous self driving vehicles, and the redesign of mobility as a service. However, all of these solutions still are still a long way from having a sizeable impact on the resolution of these problems. As for these solutions to actually make mobility fuel efficient, safer, and less time and space consuming they need to have a very high penetration into the market. Autonomous vehicles will only significantly reduce accidents when a majority of the vehicles on the road are autonomous and working in harmony with each other. Similarly, mobility as a service models like Uber, Lyft, Careem, and Waymo will only reduce congestion in cities and urban centers when a large majority of the populations uses these mobility service providers rather than personal vehicles.

On the contrast, these solutions will actually making the problems of mobility worse in the short term. This can be verified by the fact that ride hailing services like Uber have actually increased congestion in cities, which was actually something the started out to reduce. Similarly autonomous vehicles when put on the road with

human drivers, the behaviors of which is much too complicated to be completely accounted for by a computer program, cause greater accidents. These autonomous vehicles have been designed with the law in mind and common driving behaviours in mind, however human drivers often disregard both of these and thus these two types of drivers do not always work in harmony with each other.

Similarly, solutions aimed to redesign automobiles altogether whether autonomous or as electric, lightweight, smaller pods will require some time to mature as a technology and gain traction and infiltrate the market. Autonomous vehicles themselves have been in the development phase for almost a decade now and still are facing problems being completely road ready.

1.2 Problem Statement

Waymore (a play on Waymo by Google) aims to tackle the problems associated with mobility for the near future, and in a manner that can be used in countries like Pakistan. What sets Pakistan apart is the lacking road infrastructure, extremely high population density in cities like Karachi, and the economic conditions. All three of these factors contribute to self driving cars not being a viable option for LMICs in general, to tackle the risks of driving. As such, Waymore aims to develop an economical and practical implementation of driver scoring using just smartphone sensors.

1.3 Proposed Solution

Our goal, was to design something that is aimed to contribute to solving the problems of mobility with low intervention or redesign of systems, and have immediate impact. The solution that we came up with was driving scoring, which we aimed to use to help solve the problem of fuel inefficiency and of exposure to risk. The solution is essentially based on the fact that both of these problems of mobility have significant contributions from the human use of the technology, rather than the limitations of the technology itself. The correlations between driving style and both fuel efficiency and exposure to risk has been widely studied and verified. Around 90% of road accidents are caused due to human error, and based on driving style, fuel efficiency can vary about 30%.

This correlation, gave us validation for our goal of trying to fist understand the driving behaviour and style of a driver, and then capture it in multiple scores and

convey it back to the driver in a way that enables and motivates change.

Our solution is designed to have minimal intervention in the current way of driving, so as to ensure that our solution could be easily adopted and incorporated into the user flow. It is implemented using a 3+1 module system, where the first three modules; Data collection, motion tracking, and maneuver classification are the main enabling modules which will feed into the fourth module i.e. driver scoring. The data for a single trip is collected from the IMU and GPS units of the user's smartphone. This is then processed to calculate a high accuracy lane level mapping of the trajectory of the vehicle, using the motion tracking module, and various predefined maneuvers that occur during the trip are classified using the maneuver classification module. The output from both of these modules is then used to calculate multiple driving scores for various metrics, and a complete overall driving score. A detailed description of each module of the system is presented later in Chapter 1.

1.4 Intended User

In the real world, driver scoring has gained traction in 3 primary markets, Individual Users, Fleet operators and insurance providers. Our initial assumption was that in the 21st CE movement of self quantization, a large market would exist, or be created for individuals who wanted to score themselves and track their usage patterns and driving behaviors that they had built up while living in their localities.
Unfortunately, that assumption remains largely untested from our perspective, due to our difficulty in accelerating the speed at which we could deploy our product which was aggravated by a lack of data. However, there is enough evidence to suggest that, on an individual level, very few people in our initial target market (Karachi) would be receptive of a service in which their own driving would be quantified, analyzed and presented to them because of sociological factors which can largely be summed up with one word: indifference. On the other hand, Fleet managers have provided us with positive reinforcement that such a system would be beneficial for them as it would allow them to significantly reduce costs by scoring each of their drivers and helping them improve. Similarly, insurance providers are also quick keen to use such a system as it allows them to score drivers and provide personalized risk-based insurance policies.

Furthermore, a very large, and natural target market for this implementation was the government. Karachi is on track to become one of the largest megacities in the

world, and if nothing else, COVID-19 has highlighted the remarkable importance of a well maintained public service infrastructure, which in megacities like Karachi, rely almost entirely on a well maintained road infrastructure. As an example, consider the necessity of roads in a city of 50 million people, and ambulances which need to cover large distances, with a reasonable amount of safety in very short periods of time. In addition to that, we have seen various countries investing heavily in collecting Naturalistic Driving Data so that they can make informed policies, and our system could help serve as an entry point for such a project.

Keeping these stakeholders, i.e. the government, fleet operators, and insurance providers in mind, our product should:

1. be lightweight in terms of computational cost, to reduce latency and battery consumption
2. work without any additional hardware, utilizing just the user's smartphone
3. present the collected data on a dashboard in a way that it produces real world results for the end user
4. store backup for those data points (if required).

1.5 Key Challenges

The lack of data was the biggest challenge that we foresaw for our project and it was well found. We were initially working with a Dataset of 4 trips with very limited set of marked maneuvers to process and use for our purpose. The validation and verification of our results was also a problem as we only had the list of marked maneuvers that we could use to validate our results, which felt insufficient. Additionally, due to the difference in the road infrastructure of Karachi and the location of the original Dataset, we found that our models performed poorly on the Dataset that we collected.

We addressed the aforementioned issues by creating a Dataset of our own, consisting of IMU and GPS data for 4 trips with the average length of around 10 minutes. For each trip, we also collected an associated video recording that was then used to mark each maneuver by our team, and it also allows us to validate our results.

2. Literature Review

In this chapter, we present the recent state of the art in the field of Driver Scoring. As our project consists of 3 main components i.e. Data Collection, Maneuver Detection and Classification and Driver Scoring, this chapter will be organized according.

The initial motivation behind WayMore was to contribute to fixing mobility and to envision the mobility of the future. Before we get into how to we aim to fix mobility through driver scoring, we must first understand how mobility is broken. There are four main problems with mobility that we have identified; Energy inefficiency, exposure to risk, time inefficiency, and space inefficiency.

The first problem of fuel inefficiency is one that doesn't require much explanation. Around 95% of the vehicles sold in the United States are based on internal combustion engines (ICE), this percentage is almost 100% when we talk specifically about Pakistan and other third world countries. Additionally, these vehicles are quite heavy, weighing about 1300 kg, where as a human weighs 70 kg on average. Since these ICE engines give about 30% fuel efficiency, and the passenger makes up about 5% of the total weight of the vehicle, only 1.5% of the fuel that is put in to a vehicle is actually used in moving the passenger. This figure might seem alarming on itself, but put it in the context of the fact that 91% of the vehicles sold globally in 2019 are propelled by internal combustion engines and it is clear that mobility is severely broken.

The incompetency of the current system doesn't stop there, as according to the World Health Organization's (WHO) global status report on road safety in 2013, 1.24 million traffic related fatalities occur around the world each year. This makes it the leading cause of death for people ages 15-29 years **??**. The WHO's "Decade of Action is Road Safety (2011-2020)" program has resulted in various initiatives around driver assistance and safety awareness which have resulted in a significant improvement in the road safety conditions. However, with the continually increasing need for

mobility and exponentially growing number of registered vehicles the problem is far from solved. This exposure to risk is one of the biggest problems of mobility, that needs innovative solutions that leverage new technology to make mobility safer.

The fact that mobility is deeply broken and needs a complete redesigning has not gone unnoticed and many big players in the automobile and tech industry are working on solutions. These solutions lie in three broad categories:

1. Electric Vehicles
2. Autonomous Vehicles
3. Mobility as a Service

The first solution aims to replace the internal combustion engine with an electric battery powered system which is much more efficient and thus tackling the fuel inefficiency problem. Various new companies, the likes of Tesla have based their whole business on these electric vehicles and a majority of the old automobile giants are working to develop and put out these electric vehicles on the road.

The second solution is aimed at reducing the exposure to risk by automating the process, and having smart computers and algorithms designed to abide by the law and make safe decisions in various driving scenarios. This eliminates human error that can potentially cause accidents but introduces the possibility of a software malfunction or bugs in the code which could result in undesirable results.

Despite the significant interest and traction, it is going to be a while before electric vehicles make up sufficient enough percentage of the market to have a sizeable impact on the problem of inefficiency.

1. High Penetration required
2. It will get worse before it gets better

Waymo was initiated by google in 2009 as project that incorporated all the major solution into one package. It aimed to make autonomous, electric vehicles that would be a part of a ride hailing, mobility as a service fleet. However, even after a decade of development, and major shifts in policy and funding, the project still hasn't yielded any practical results.

The field of driving style and driver analysis is undergoing major research and a plethora of various types of driver monitoring have emerged. They can be classified into three main categories:

1. **Vehicle Motion Tracking:** The speed and direction of movement and location is tracked to track the motion and various driving maneuvers are identifies such as acceleration, deceleration, turns, and lane changing(maneuvers are usually identified using a range of machine learning techniques). Maneuver Data can be used in conjunction with artificial intelligence to classify driving patterns and driving style. This is done using an inertial measurement unit (IMU) and a Global Positioning System (GPS) sensor. These sensors exist in almost all smartphones and sometimes in vehicles as well, and both can be exploited.

2. **Driver Behavior Tracking:** This is where physiological and psychological state of the driver is tracked. For example distraction, drowsiness, aggression, arousal etc. Cameras and computer vision plays a major role in sensing and classifying this data. Cameras can be installed or smartphones can also be docked. This is however commonly limited to research and not an everyday deployed solution as of yet.

3. **Environment Tracking:** This is where the external environment is tracked for example, road condition, bumps, and the weather and traffic conditions. These require external sensors to be mounted, such as LIDARs and external cameras. This has a big application in terms of crowdsensing and collecting data.

We intend to work in the **Vehicle Motion Tracking** domain where we will be using telemetry data from smartphones to score drivers. The first step for this is Data collection. In our literature review we found possible candidates for our research. UAH-DriveSet, presented by Romera et a. [8], is a data set consisting of raw and processed data of smartphone inertial sensors, GPS and Camera that was collected using their app DriveSafe. It is publicly available with the intent to encourage people to research into the field of driver scoring. The inertial smartphone sensor data in this dataset is processed by employing an Adaptive Fuzzy classifier [2] to event driving maneuvers. Ferreira et al. [3] present a "Driver Behaviour Dataset", which is a dataset of 4 trips, approximately 13 minutes each, which contain smartphone's sensor data along with start and end time of various maneuvers such harsh breaking, acceleration. There are several other datasets that are being used in the field currently such as Honda Research Institute Dataset [7] and UDrive Dataset [1] but these either need to requested or are not applicable for our use case. Hence, for our project, we chose to work with the data set presented in [3] as it most resembled the structure of data that will be collected by us.

Once we have decided on the dataset that we will be working with, it is important

to decide on the maneuvers that we will be detecting and using to score the drivers. Lopez et al. [4] provide a list of risky maneuvers, called Figure of Metrics (FoMs), out of which we selected Harsh Breaking, Harsh Acceleration, lane changes and Turns as the maneuvers that we will be detecting. The authors also do a comparison of various techniques that are generally used to classify maneuvers, listing the pros and cons of each techniques. These include, SVM Regressor, RF Regressor, GP and MLP. In another study, Woo and Kulic' [9] present an approach to classify maneuvers based on Support Vector Machines (SVM) and sliding window technique. Achieving a performance of 0.8874 average accuracy. They also employ Principle Component Analysis (PCA) to reduce the dimensions of the raw data. Meiring [5] does an in-depth review of various driver analysis systems, and concludes that Fuzzy Logic Systems, HMM and SVMs show most promising results in classifying maneuvers.

In the paper accompanied with the dataset that we have decided to work with, the authors evaluated 4 ML techniques to classify maneuvers [3]. And found Random Forests to perform the best, achieving 0.999 mean AUC value. Hence, in our project, we have decided to employ RF models to classify maneuvers. Additionally, it can be seen that Support Vector Machine is a technique that is recurrent across literature. Which is why we have employed them as well.

3. Software Requirement Specification (SRS)

This chapter provides detailed specifications of the system under development.

3.1 Functional Requirements

This section describes each function/feature provided by our system. These functions are logically grouped into modules based on their purpose/users/mode of operations etc (as per our system). A functional hierarchy may look like:

- The system can collect and store data from the black box unit
- The system can process and fuse raw data to detect and store driving events
- The system should start the ride with the press of a button
- The system should end the ride with the press of a button
- The system should notify when the ride has started
- The system should keep track of the location of the vehicle
- The system should present the maneuvers performed by the vehicle at the end of the trip
- The system should give the driver their firsts score after completing 40 km of travelling

3.2 Non-functional Requirements

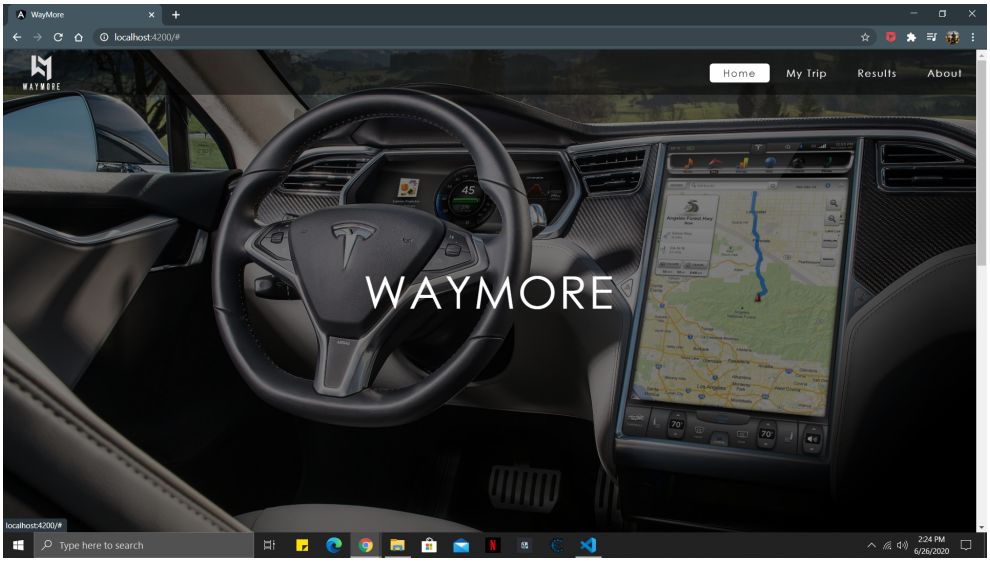
This sections mentions the specific non-functional requirements of our system. These generally address performance, scalability, safety, availability, deployment etc.

- The system can be operated easily by people without the knowledge about the technology behind it
- The system should allow for an easy addition of new components or exchange of existing algorithms
- The system should work on its own once the driver presses the start button

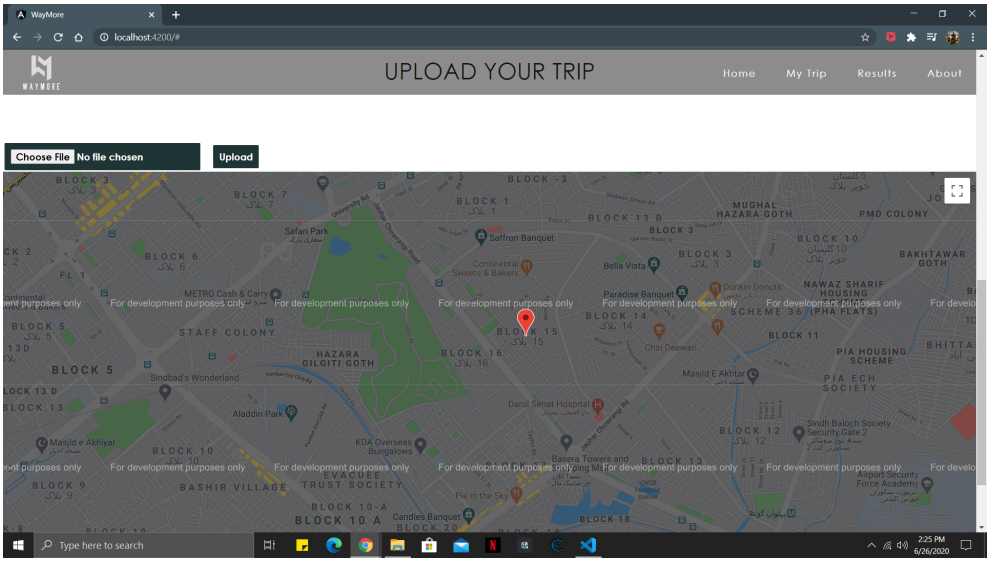
3.3 External Interfaces

3.3.1 User Interfaces

This section includes our mock-up screens and briefly explains them.



This is homepage/landing page of the WayMore Web Application. (Dashboard)



This is the embedded Google Maps API on which trips are plotted and bad maneuvers indicated with a marker.

3.3.2 Hardware Interface

This section describes our project's specific hardware/network interfaces.

- Arduino UNO
- MPU6050 (IMU Unit)
- Adafruit ultimate GPS breakout

3.4 Use Cases

This section presents detailed use cases of our system.

Use cases

- A Personal User:
  - Sign up/Sign in
  - Access dashboard → Total distance driven, average frequency of poor maneuvers, average frequency of good maneuvers
  - View average score of day, week, month, year, custom range

  - View all trips → trip profile
  - Access trip profile for route, maneuver breakdown → each maneuver has time, location, logged against a graph, scoring process i.e. intensity and duration (information already displayed on graph)
  - Badges?
- A Fleet Employee:
  - All the above
  - Segment trips via destinations, miles covered in specific states/cities/-counties, positive reinforcement of all good maneuvers.
- Fleet Manager:
  - All of the above
  - Access central dashboard where they can view
    * Each driver's profile
    * Each driver's trips' profiles
    * Cumulative fleet performance
    * Cumulative destinations visited/miles logged in area
    * each maneuver breakdown (for focused efforts) for entire fleet
  - Asset tracking (segment routes/active trips of a particular shipment (for eg: Twinkies en route to xyz location)), and how many trucks are active, which ones are in the depot, which ones are inactive for maintenance, which ones have been serviced between trips etc., mileage per vehicle (nice in entirety)

3.5 Datasets

This section describes the specific dataset(s) used to build our system. An appropriate snapshot of the dataset(s) is also included. Futher details, when needed, are presented in the appendix

For our project, we are working with Trip Datasets from two sources. Both of these Datasets come with an accompanying file that contains the details of maneuvers in

each trip.

3.5.1 D1: Driver Behaviour Dataset

The "Driver Behaviour Dataset" is publicly available at this link. This Dataset is optimal for our use case as it contains sensor readings for accelerometer, magnetometer and gyroscope; which are the sensors that we are primarily working with to classify different maneuvers. The sensor data is stored in separate CSV files. This dataset also comes with a paper that describes how the authors applied different Machine learning techniques (Support Vector Machine, Random Forest, Neural Networks) to this dataset to classify manuevers such as harsh acceleration, harsh braking etc. which can serve as a good guideline for us.

	timestamp	uptimeNanos	x	y	z
0	14/05/2016 10:54:33	11537640270059	-0.161602	0.120174	-0.209893
1	14/05/2016 10:54:33	11537650128140	-0.122628	0.315638	-0.380996
2	14/05/2016 10:54:33	11537659894659	-0.178777	0.330181	-0.360696
3	14/05/2016 10:54:33	11537679549779	0.016043	0.038759	-0.278204
4	14/05/2016 10:54:33	11537699204899	0.141716	-0.162492	-0.049796
5	14/05/2016 10:54:33	11537719165222	0.094701	-0.208684	0.060976
6	14/05/2016 10:54:33	11537738820341	-0.093779	0.126911	-0.190100
7	14/05/2016 10:54:33	11537758475461	-0.204955	0.198113	-0.337521
8	14/05/2016 10:54:33	11537778069541	-0.070380	0.081215	-0.273695
9	14/05/2016 10:54:33	11537797724661	0.047234	-0.219240	-0.018233
10	14/05/2016 10:54:33	11537817349260	0.085305	-0.322752	-0.106739

Accelerometer Data for a single trip

	evento	inicio	fim
0	evento_nao_agressivo	2.0	6.5
1	curva_direita_agressiva	19.5	23.5
2	evento_nao_agressivo	30.0	33.5
3	curva_direita_agressiva	95.0	98.0
4	curva_esquerda_agressiva	247.0	251.5
5	curva_esquerda_agressiva	348.7	352.3
6	evento_nao_agressivo	485.0	489.0
7	curva_esquerda_agressiva	496.0	499.5
8	curva_direita_agressiva	587.0	590.0
9	curva_esquerda_agressiva	750.0	753.8
10	curva_direita_agressiva	840.7	844.0

Event Data for a single trip

3.5.2 D2: Computer Science Center Dataset

The "Computer Science Center Dataset" is publicly available at this link. This Dataset contains about a 1000 kilometer of telemetry and video data along with JSON files that outlines when a maneuver occurs. This dataset is also accompanied by a Dashboard that can be used to visualize video and telemetart data simultaneously, which aids in analysis and validation of results. Each trip's telemetry data is stored in a single CSV file, but the format is very difficult to understand at first glance.

3.5.3 D3: The Processed Dataset

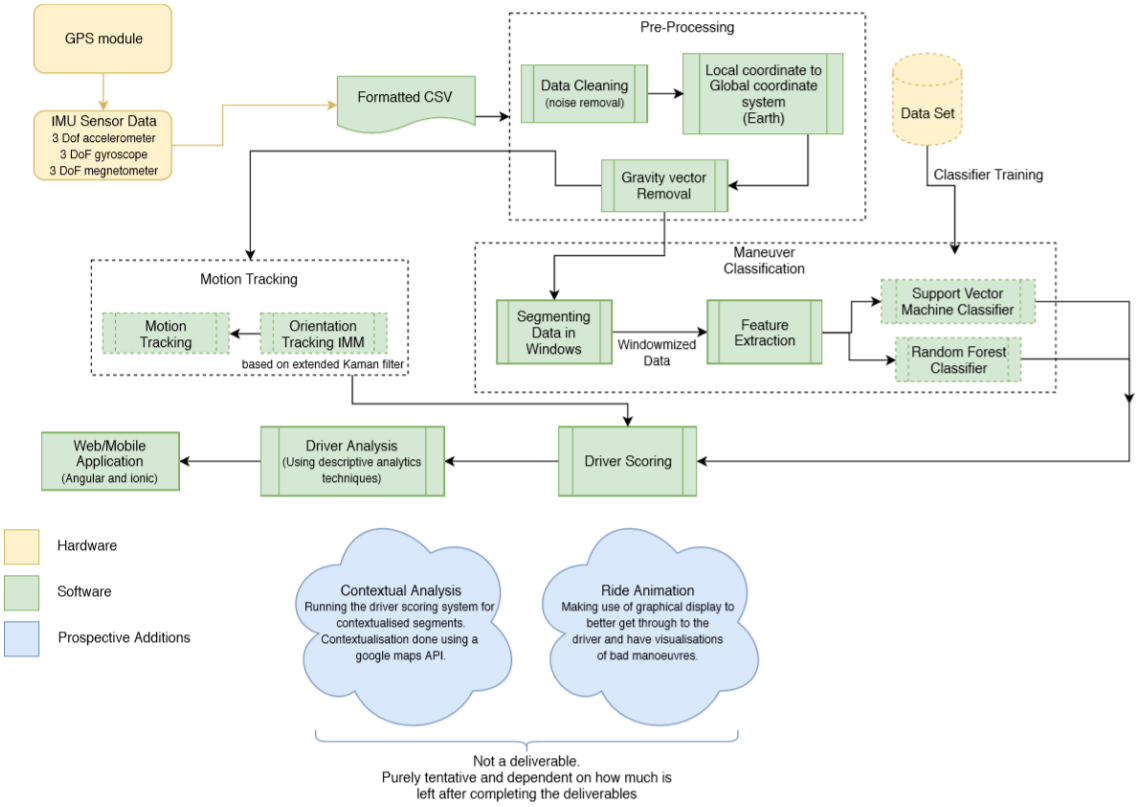
The final form of the Dataset that we will be working with is shown below. We have implemented different pre-processing pipelines to bring the two Datasets into this processed form so that they can be seamlessly passed to the next modules. Additionally, we have also collected and created a dataset containing 100 kilometer of video and telemetry data. Which is also processed to this format. Each trip has 4 CSV files associated with it, accel, gyro, magnet and geo.

timestamp	x	y	z	Event
2020-06-26 16:29:26.750000	0.004017176436588965	0.005755207188340337	-0.013795656222527812	Aggressive left lane change
2020-06-26 16:29:26.875000	0.0008900467867249697	0.010826789115954472	-0.024147199000881844	Aggressive left lane change
2020-06-26 16:29:36	0.005877458911718317	-0.006448538007626573	0.05129575567583191	Aggressive left 90-turn
2020-06-26 16:29:36	0.00884260473573601	-0.006490066912444052	0.0699950330874281	Aggressive left 90-turn
2020-06-26 16:29:36.125000	0.013642301057108765	-0.0025030793271334805	0.07654155649103482	Aggressive left 90-turn
2020-06-26 16:29:36.250000	0.012489844425176505	0.0007430171993826313	0.08997953050114885	Aggressive left 90-turn
2020-06-26 16:29:36.375000	0.014242176558051968	0.005766270194870449	0.10865591592634624	Aggressive left 90-turn
2020-06-26 16:29:36.500000	0.01573528114148341	0.01102723518379935	0.13110022939492563	Aggressive left 90-turn
2020-06-26 16:29:36.625000	0.02223442473965275	0.015136617315737002	0.1615172492791682	Aggressive left 90-turn
2020-06-26 16:29:36.750000	0.019977907342891045	0.026776115460730627	0.17756072693003194	Aggressive left 90-turn

Gyroscope Data of Processed Dataset

3.6 System Diagram

This diagram gives a high-level view of the different components of our system and the interactions between them. Each component and the particular tools/technologies/libraries used to build it are described.



- The System Data Diagram is divided into 3 main heavy components. Pre-processing, Motion Tracking and Maneuver Classification. Pre-processing has to do with processing of the data our system acquires from our hardware modules and applying specific transformations; in order to convert the data in required csv format, filters; in order to clean the raw data of noise and other factors that are not relevant to our purpose, removal of the gravity factor from our 3DoF accelerometer for example.

- This processed data is fed separately feeded in both our modules, Motion Tracking and Maneuver classification. The Maneuver Classification module first segments our time-series data in windows of varying size, this windowmized data is then fed in for feature extraction. The windowmized data is run through several algorithms to fetch all relevant features like mean, max, and other relevant features. This featured and classified data is then used to feed in and train our classifiers, Support Vector Machine and Random Forrest, and then tested on the same. Motion Tracking module first achieves orientation tracking using

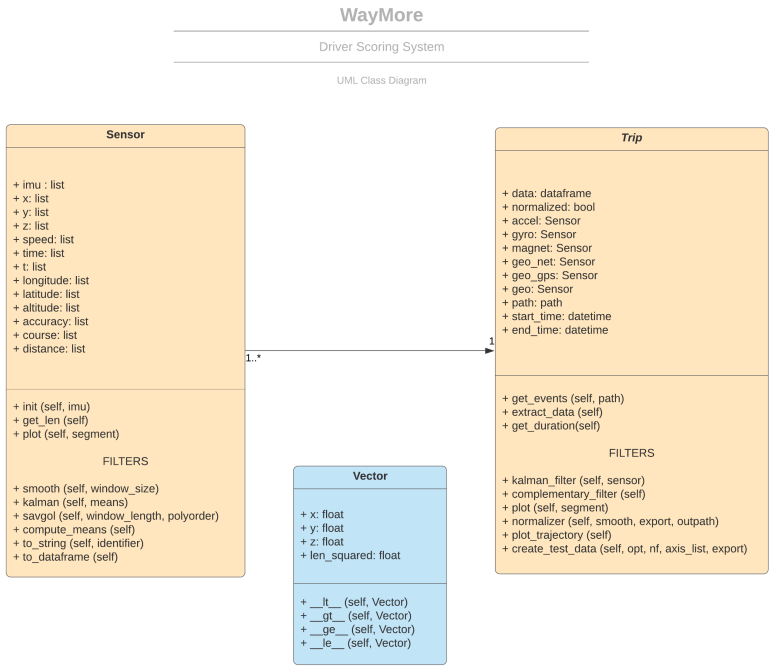
an IMM, which runs based on extended Kalman filter. Having once achieved that, the same data is further fed into the Motion tracking module, which tracks the displacement with respect to the origin point, in a local coordinate system which can then be further used to transform to our desired coordinate frames.

- The results from both these modules, which are classified maneuvers with their sliding time windows, and a complete trajectory in 3 axis, are used based on already existing algorithms, to compute maneuvers and assign each one a score, which will in turn compute the score for the entire trip and over generations of a few trips, driver scoring will be achieved. This segmented, classified, and meaningful data about trips and the drivers will be further fed into the Driver Analytics module. And using this driver profile, the application will be developed.

4. Software Design Specification (SDS)

This chapter provides important artifacts related to design of our project.

4.1 Software Design



UML.png UML.png

4.1.1 Dashboard

A Web Application was created using the Angular framework as a dashboard, aiming to serve as the first landing page for use of our application. The Web Application allows the user to upload a specific trip in the form of a .csv file and as a result allows them to see a trajectory of the trip they made plotted on to the Google Maps API, as well as markers pinpointing the locations of all bad maneuvers the driver made onto the Google Maps API. The user also receives a summary of the uploaded trip in form quantified data indicating the duration of the trip, and other parameters like the quantity of each maneuver that took place during the trip.

4.2 Technical Details

4.2.1 Data Preprocessing

Data pre-processing is the first step performed when we get the data in the raw form from our IMU and GPS. This data pre-processing part, takes raw data from the IMU+GPS and converts them into a list of dataframes corresponding to each sensor. In order to correctly execute the pre-processing step, we provide the function with

- Source path i.e. the path containing the raw data
- Destination path i.e. path to the location where the data set is stored
- Number of frames that make up the time window
- List of axis that will be used for classification

  This function returns the list of dataframes corresponding to each sensor. These dataframes are then used by machine learning algorithms for classification of various maneuvers.

4.2.2 Maneuver Classification

This is the step that will take the data frames from the previous step and identify various maneuvers performed by the vehicle. For the classification part, we have used to two algorithms, SVM and Random Forest .Support vector machines also referred as SVM and Random Forest are widely used machine learning algorithms for classification of labelled data. Thorough our pipeline, SVM receives the clean labelled data from the sensors i.e accelerometer, gyroscope. This data is normalized so that each column of the received data have an approx of 0 mean and standard

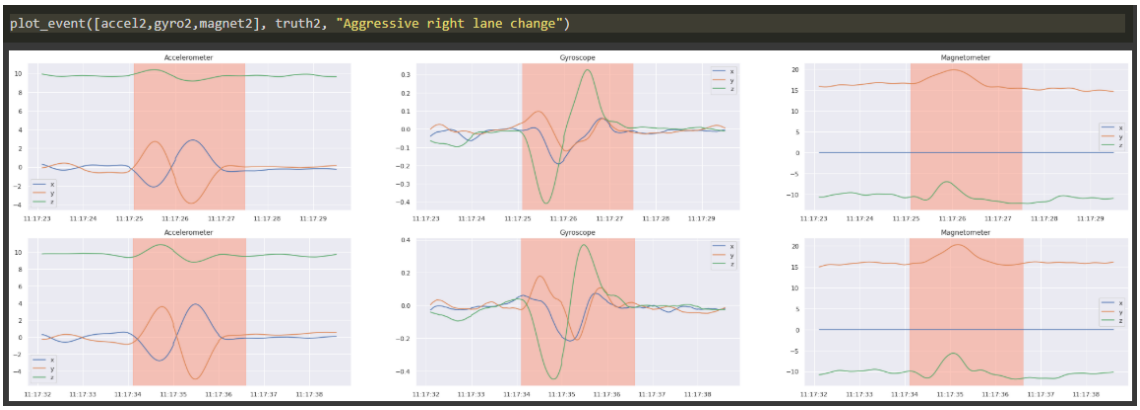
deviation of 1. The normalized data is then split into test and training set. The training set is used to train both algorithms while test set is used to check the accuracy of the predictions on unseen or untrained data set. The test makes sure that our algorithms does not overfit the data. The final result of this step is the prediction of the maneuver performed by the vehicle in each row.

5. Experiments and Results

After pre-processing our Datasets in our required format, we start our analysis by visualizing the data stream for different maneuvers to identify if there was a visual pattern that could be observed.

5.1 Maneuver's Visual Patterns

5.1.1 Aggressive Right Lane Change



We can observe a sine wave in accel y, and a flipped sine wave in accel x, with amplitude around 5. Additionally,we can observe a flipped sine wave in gyroscope z, with an amplitude > 0.4. Plotting all samples also show that the gyroscope sine wave being flipped or not is a good way to distinguish between left and right lane changes.

5.1.2 Aggressive Left Lane Change

We can observe a low peak in gyro z and high peak in gyro x, with amplitude around 0.8 and higher. This is a good indicator for the maneuver as this pattern isn't observed in other maneuvers. The Accelerometer pattern exhibits a similar pattern to the lane change maneuvers, with x value reaching an amplitude around 5 during the maneuver.

5.1.3 Aggressive Right Turn

We can observe a low peak in gyro z and high peak in gyro x, with amplitude around 0.8 and higher. This is a good indicator for the maneuver as this pattern isn't observed in other maneuvers. The Accelerometer pattern exhibits a similar pattern to the lane change maneuvers, with x value reaching an amplitude during the maneuver.

5.1.4 Aggressive Left Turn

A similar pattern is observed with "Aggressive right turn" with the main difference being that the x-z roles are reversed in gyroscope data.

5.1.5 Aggressive Breaking

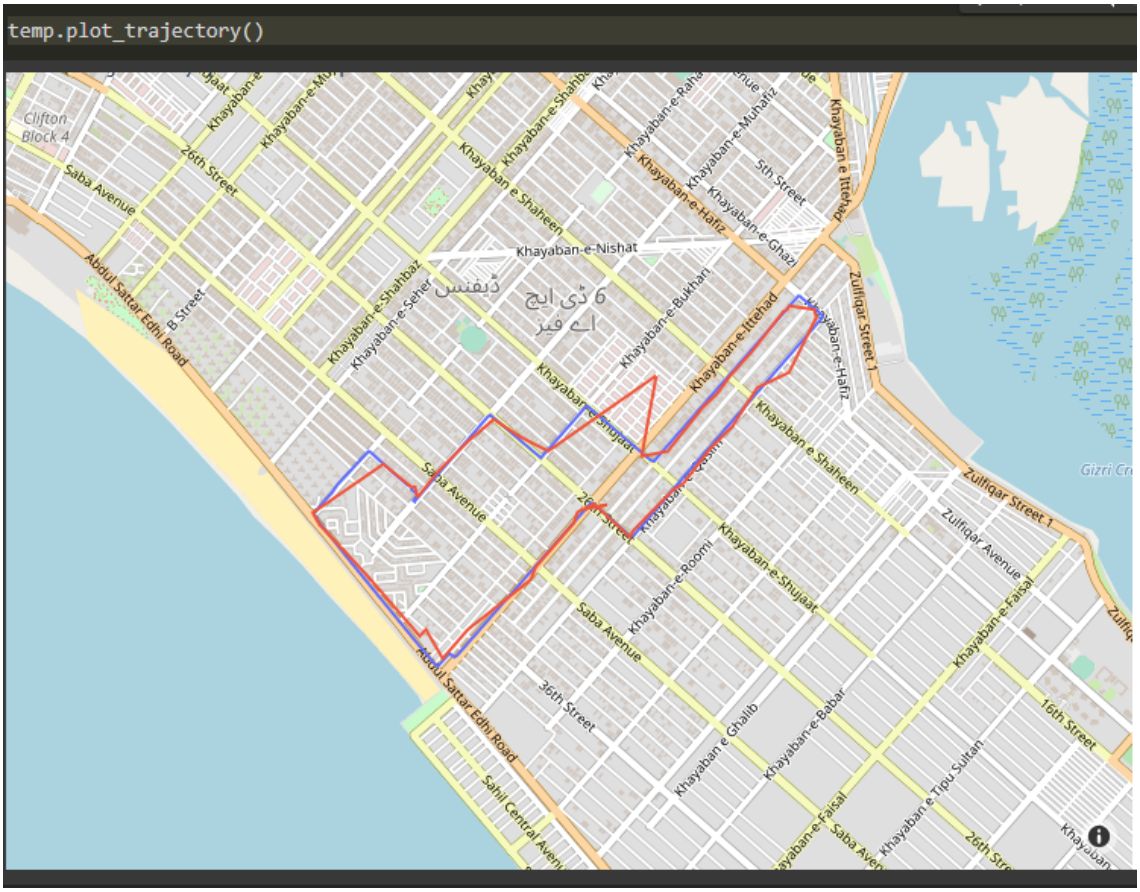
The gyroscope data doesn't contribute to analysis Accelerometer shows a unique α shape which can serve as a marker for this maneuver, ranging from +-5

5.1.6 Aggressive Acceleration

Although the small rise in accel x is exhibited in both instances of the maneuver and makes sense as well, we cannot rule this out as a distinguishing feature.

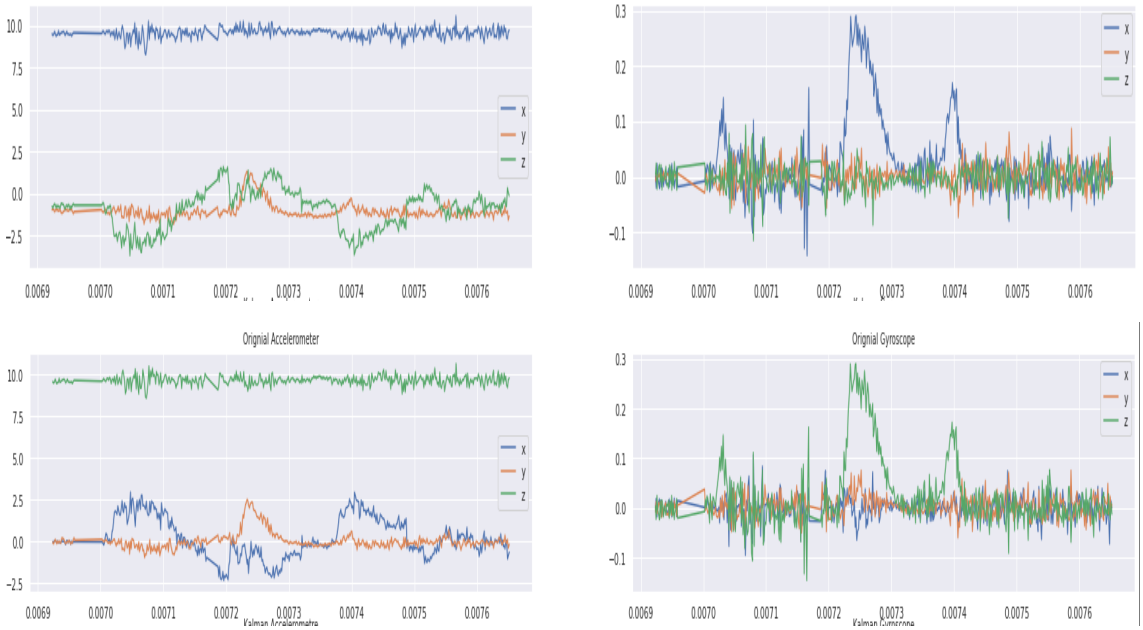
5.2 Plotting Trajectory

Using our data, we were able to plot an approximate trajectory of the trip, mapped to the real world which can enable contextual analysis in the future.



5.3 Re-orienting Sensors

Since our primary data source are smartphones, it was important to standardize our orientation, without any added work on the user's part. We were able to develop a re-orientation algorithm that would always align Z-axis with gravity. The results can be seen below.



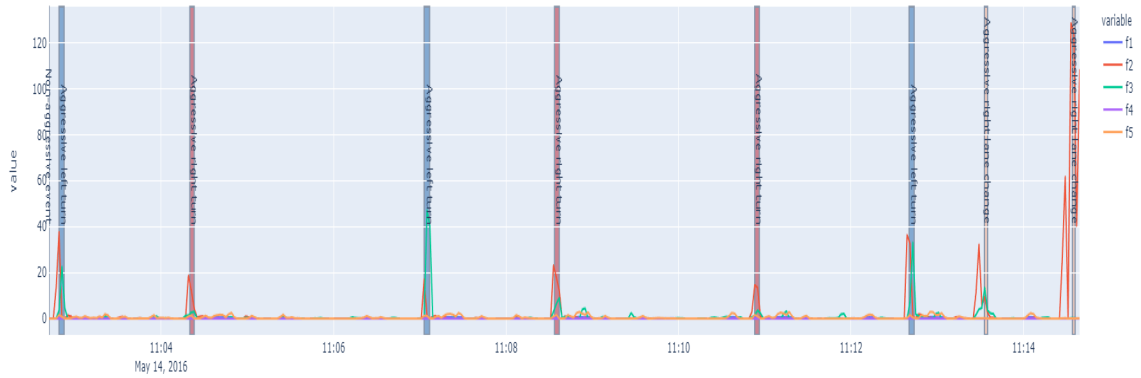
We can see that after re-orientation, we are getting a value of around 9.81 on the z-axis.

5.4 Removing Noise

We developed a smoothing algorithm based on the moving average technique, and compared it with a library implementation of Kalman filter. The results are show below.

5.5 Identifying windows for Maneuver

The next step is to identify time windows in the Trip data when a maneuver is most likely to occur. For this purpose, we downsample our dataset to 2Hz i.e. 2 readings per second, and segment our dataset into sliding windows of 3 seconds. Thus, we have 6 readings for each window. We then calculate the 5 features outlined in the Math section.



We plotted these features and added markers to highlight regions where we know a maneuver has occurred. It can be seen that at least one of the features shows a spike during these regions. More importantly, it can be seen that during windows when no maneuver has occurred, these features have an almost constant value. Thus, these features are sufficient to distinguish between windows with a maneuver and without a maneuver.

5.6 Calculating attribute vectors for Maneuver Classification

After identifying the segments of time where a maneuver might have occurred, we filter our original data to only contain those segments of data. We then create windows containing varying number of frames. And then we calculate the attribute vector for each window as defined in [3]. Here, 1 frame contains data of 1 second.

M0_z	M1_z	M2_z	MD0_z	MD1_z	MD2_z	SD0_z	SD1_z	SD2_z ▼	T1_z	T2_z	mane
-0.014390230779367824	-0.10137904383409013	-0.27051069619436957	-0.013900883859653337	-0.031243563823642243	-0.16055148199289876	0.009553632988229651	0.12653915420965234	0.2684485032040775	13.089981653309357	42.304672541302 52	Aggre right tu
0.0012632911482656462	-0.07211826287721208	-0.24131396233404867	0.002366127862585703 4	-0.02445124307047993	-0.08847611668634457	0.016414967771370824	0.10626690661255556	0.2609754257975946	-115.17520494182546	-458.88500211815034	Aggre right tu
-0.047054977098555446	0.08327973242651605	0.19006059254007865	-0.04978544685566891	0.08523524989807907	0.2151551485499734 5	0.07719725973152684	0.1489452990657964	0.1966892993275533 6	-4.539677949564771	-8.577675256792224	Non-aggres event
-0.13942129774725578	-0.09323813742290565	0.0090460557019254	-0.15112543410430004	-0.11564914870700296	-0.04978544685566891	0.028641970210371473	0.07423158615747938	0.1615697767005497 3	0.337502073634812 55	-1.532150721612344 2	Non-aggres event
-0.013162064315846057	-0.013776147547606945	-0.07197338399467544	-0.012179907174161196	-0.01287780925687725	-0.021424449352525975	0.012825959698128 5	0.011269568807936604	0.1115102420546606 9	1.0933110820650795	14.311422005591691	Aggre right tu
-0.003348271100667013 7	-0.010424899762006825	-0.0491949228503039	-0.005263963544518203	-0.003894936862769732	-0.009896394430734166	0.013435067503314736	0.015103571867972076	0.0928489099369968 2	-0.377296554037689 6	43.45520793513542	Aggre right tu
-0.05613089575586675	-0.09777609675156121	-0.08086905686722598	-0.05766418524677982	-0.07995186823444152	-0.07346849111403 61	0.012918926735397 18	0.04733114169624116	0.0634379238220289 1	2.4838601962393163	8.838307966849741 6	Non-aggres event
0.0037912302814032 45	0.009642913661556302	0.0049597713768112115	0.003486624554841 41	0.007086642997842034 5	0.004200476252115915	0.002023115802963647	0.006929357300618865	0.0100306773451796 71	4.086957502347853	-1.162291094343125 5	Non-aggres event
0.0017918076748042638	0.0027915189781037536	0.007025878332638954	0.002169237583092318 4	0.00261312220909882	0.003909014418631 7	0.001411650843061951	0.002005509483477912	0.0068079909389709 36	2.115868982321106	8.647466611282361	Non-aggres event
0.0004959583730054071	0.0014138516284462638	0.0023995939967454997	-0.000137261989744137 87	0.001474102834969640 9	0.002467742666453258 2	0.005537886693793742	0.003223986168450999	0.0030181972792600125	3.612818680620861 3	7.644250985075943	Non-aggres event

The sample above shows the attribute vector for just gyroscope's z-axis, with a

window of 3 frames. We have tried different combinations of sensors and axis in our analysis, the results of which are provided in the subsequent sections. In general, an attribute vectors contains $x = 3 * 3 * nf - 1$ attributes associated with a particular sensor's axis.

5.7 Maneuver Classification

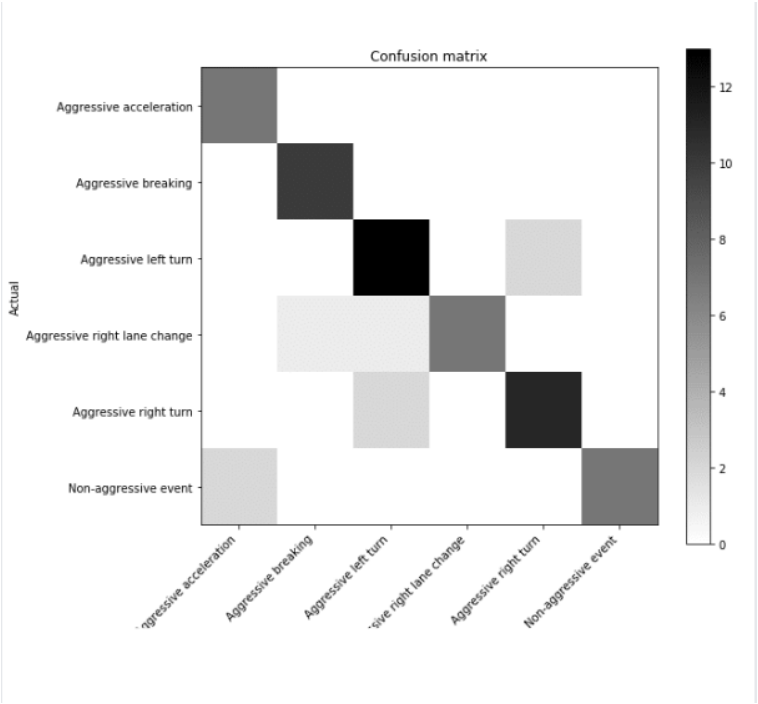
This is the step that will take the data frames from the previous step and identify various maneuvers performed by the vehicle. For the classification part, we have used two algorithms in the start. Random Forest and SVM. After some initial testing and results, we realized that SVM was surpassing the Random Forests both in the time complexity and the accuracy. So, we decided to stick with SVM for rests of our experimentation.

5.7.1 Jr Jain Dataset

Acceleration Based Classification

	precision	recall	f1-score	support
Aggressive acceleration	0.82	0.90	0.86	10
Aggressive breaking	1.00	0.64	0.78	11
Aggressive left turn	0.78	0.88	0.82	16
Aggressive right lane change	0.75	1.00	0.86	3
Aggressive right turn	0.85	0.85	0.85	13
Non-aggressive event	1.00	1.00	1.00	10
micro avg	0.86	0.86	0.86	63
macro avg	0.87	0.88	0.86	63
weighted avg	0.87	0.86	0.86	63

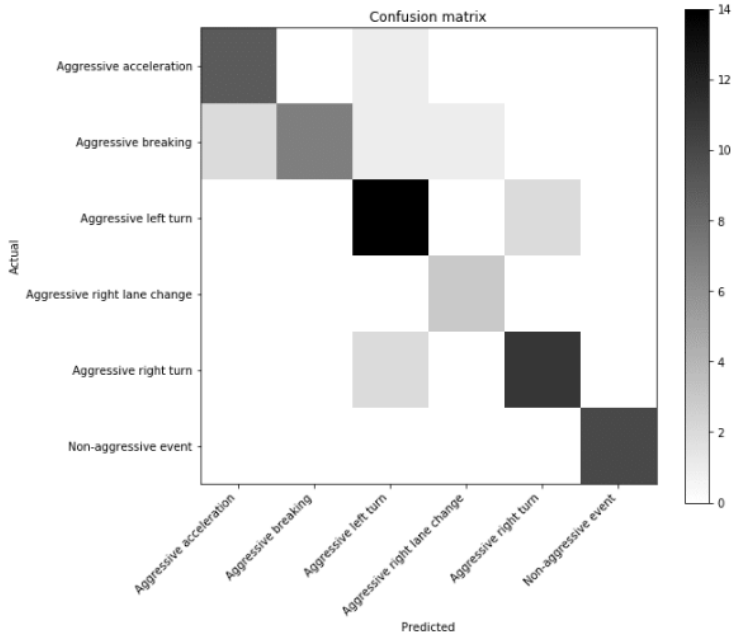
0.8571428571428571



Gyro Based Classification

	precision	recall	f1-score	support
Aggressive acceleration	0.78	1.00	0.88	7
Aggressive breaking	0.91	1.00	0.95	10
Aggressive left turn	0.81	0.87	0.84	15
Aggressive right lane change	1.00	0.78	0.88	9
Aggressive right turn	0.85	0.85	0.85	13
Non-aggressive event	1.00	0.78	0.88	9
micro avg	0.87	0.87	0.87	63
macro avg	0.89	0.88	0.88	63
weighted avg	0.88	0.87	0.87	63

0.873015873015873



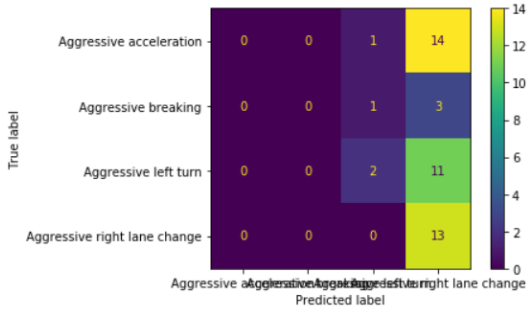
5.7.2 Driver Behaviour Dataset

Acceleration 5 frames

	precision	recall	f1-score	support
Aggressive left U turn	0.00	0.00	0.00	15
Aggressive left lane change	0.00	0.00	0.00	4
Aggressive left turn	0.50	0.15	0.24	13
Aggressive right turn	0.32	1.00	0.48	13
accuracy			0.33	45
macro avg	0.20	0.29	0.18	45
weighted avg	0.24	0.33	0.21	45

```
0.3333333333333333
[[ 0  0  1 14]
 [ 0  0  1  3]
 [ 0  0  2 11]
 [ 0  0  0 13]]
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

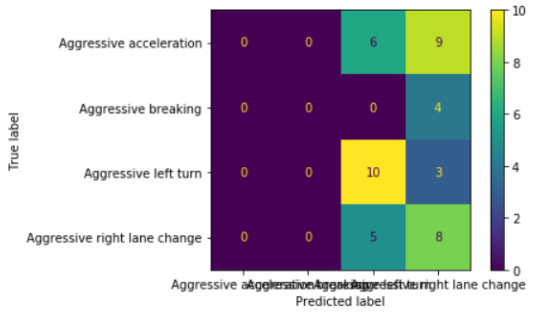


Gyro 5 frames

	precision	recall	f1-score	support
Aggressive left U turn	0.00	0.00	0.00	15
Aggressive left lane change	0.00	0.00	0.00	4
Aggressive left turn	0.48	0.77	0.59	13
Aggressive right turn	0.33	0.62	0.43	13
accuracy			0.40	45
macro avg	0.20	0.35	0.26	45
weighted avg	0.23	0.40	0.29	45

```
0.4
[[ 0  0  6  9]
 [ 0  0  0  4]
 [ 0  0 10  3]
 [ 0  0  5  8]]
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

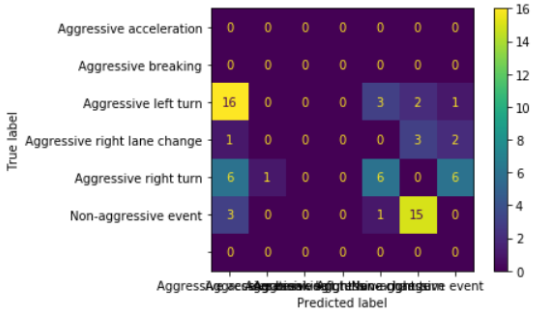


Acceleration 8 frames

	precision	recall	f1-score	support
Aggressive acceleration	0.00	0.00	0.00	0
Aggressive breaking	0.00	0.00	0.00	0
Aggressive left U turn	0.00	0.00	0.00	22
Aggressive left lane change	0.00	0.00	0.00	6
Aggressive left turn	0.60	0.32	0.41	19
Aggressive right turn	0.75	0.79	0.77	19
Non-aggressive event	0.00	0.00	0.00	0
accuracy			0.32	66
macro avg	0.19	0.16	0.17	66
weighted avg	0.39	0.32	0.34	66

```
0.3181818181818182
[[ 0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0]
 [16  0  0  0  3  2  1]
 [ 1  0  0  0  0  3  2]
 [ 6  1  0  0  6  0  6]
 [ 3  0  0  0  1 15  0]
 [ 0  0  0  0  0  0  0]]
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```



5.7.3 Computer Science Center DataSet

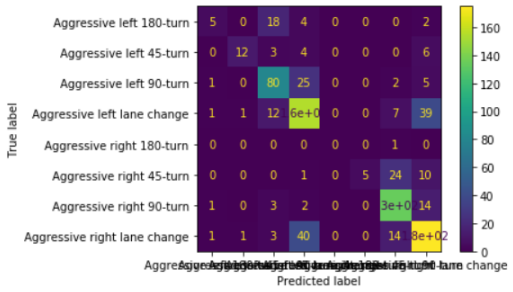
Acceleration 3 frames

```
{'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.56	0.17	0.26	29
Aggressive left 45-turn	0.86	0.48	0.62	25
Aggressive left 90-turn	0.67	0.71	0.69	113
Aggressive left lane change	0.67	0.72	0.70	215
Aggressive right 180-turn	0.00	0.00	0.00	1
Aggressive right 45-turn	1.00	0.12	0.22	40
Aggressive right 90-turn	0.74	0.87	0.80	154
Aggressive right lane change	0.70	0.75	0.72	234
accuracy			0.70	811
macro avg	0.65	0.48	0.50	811
weighted avg	0.71	0.70	0.68	811

```
0.6979038224414303
[[  5   0  18   4   0   0   0   2]
 [  0  12   3   4   0   0   0   6]
 [  1   0  80  25   0   0   2   5]
 [  1   1  12 155   0   0   7  39]
 [  0   0   0   0   0   0   1   0]
 [  0   0   0   1   0   5  24  10]
 [  1   0   3   2   0   0 134  14]
 [  1   1   3  40   0   0  14 175]]
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```



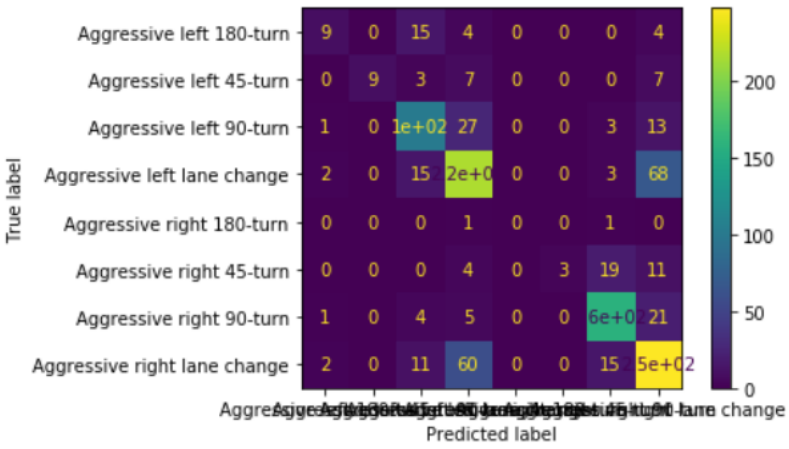
Acceleration 4 frames

```
{'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                           precision    recall  f1-score   support

    Aggressive left 180-turn       0.60      0.28      0.38        32
     Aggressive left 45-turn       1.00      0.35      0.51        26
     Aggressive left 90-turn       0.68      0.70      0.69       146
 Aggressive left lane change       0.67      0.71      0.69       305
   Aggressive right 180-turn       0.00      0.00      0.00         2
    Aggressive right 45-turn       1.00      0.08      0.15        37
    Aggressive right 90-turn       0.79      0.84      0.81       189
Aggressive right lane change       0.67      0.74      0.70       336

                    accuracy                           0.70      1073
                   macro avg       0.68      0.46      0.49      1073
                weighted avg       0.71      0.70      0.68      1073


0.6952469711090401
[[  9   0  15   4   0   0   0   4]
 [  0   9   3   7   0   0   0   7]
 [  1   0 102  27   0   0   3  13]
 [  2   0  15 217   0   0   3  68]
 [  0   0   0   1   0   0   1   0]
 [  0   0   0   4   0   3  19  11]
 [  1   0   4   5   0   0 158  21]
 [  2   0  11  60   0   0  15 248]]
```



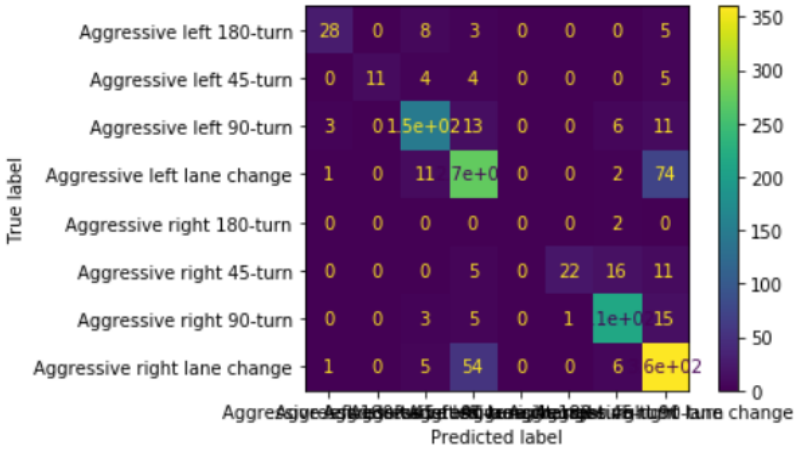
Acceleration 5 frames

```
{'C': 10, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=10, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.85	0.64	0.73	44
Aggressive left 45-turn	1.00	0.46	0.63	24
Aggressive left 90-turn	0.83	0.82	0.82	180
Aggressive left lane change	0.76	0.76	0.76	361
Aggressive right 180-turn	0.00	0.00	0.00	2
Aggressive right 45-turn	0.96	0.41	0.57	54
Aggressive right 90-turn	0.87	0.90	0.88	238
Aggressive right lane change	0.75	0.85	0.79	426
accuracy			0.79	1329
macro avg	0.75	0.60	0.65	1329
weighted avg	0.80	0.79	0.79	1329

```
0.7938299473288186
[[ 28   0   8   3   0   0   0   5]
 [  0  11   4   4   0   0   0   5]
 [  3   0 147  13   0   0   6  11]
 [  1   0  11 273   0   0   2  74]
 [  0   0   0   0   0   0   2   0]
 [  0   0   0   5   0  22  16  11]
 [  0   0   3   5   0   1 214  15]
 [  1   0   5  54   0   0   6 360]]
```



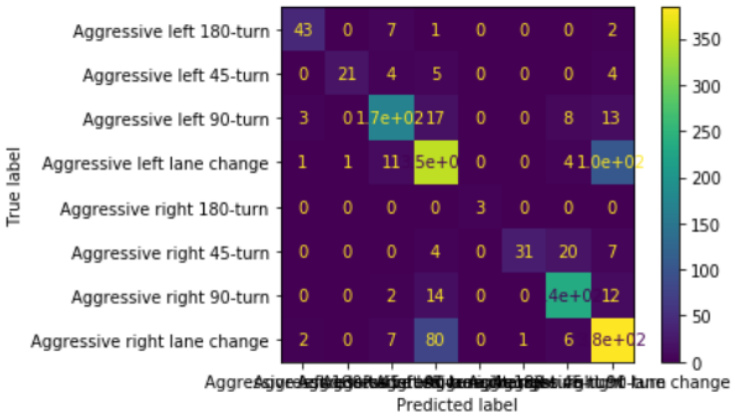
Acceleration 6 frames

```
{'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       0.88      0.81      0.84        53
     Aggressive left 45-turn       0.95      0.62      0.75        34
     Aggressive left 90-turn       0.85      0.81      0.83       211
 Aggressive left lane change       0.74      0.74      0.74       469
   Aggressive right 180-turn       1.00      1.00      1.00         3
    Aggressive right 45-turn       0.97      0.50      0.66        62
    Aggressive right 90-turn       0.86      0.89      0.88       264
Aggressive right lane change       0.73      0.80      0.76       480

                    accuracy                           0.78      1576
                   macro avg       0.87      0.77      0.81      1576
                weighted avg       0.79      0.78      0.78      1576


0.7836294416243654
[[ 43   0   7   1   0   0   0   2]
 [  0  21   4   5   0   0   0   4]
 [  3   0 170  17   0   0   8  13]
 [  1   1  11 347   0   0   4 105]
 [  0   0   0   0   3   0   0   0]
 [  0   0   0   4   0  31  20   7]
 [  0   0   2  14   0   0 236  12]
 [  2   0   7  80   0   1   6 384]]
```



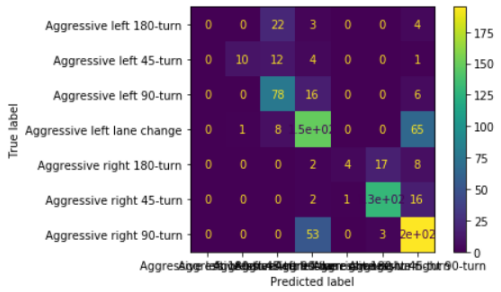
Gyro 3 frames

```
{'C': 1000, 'gamma': 1, 'kernel': 'linear'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.00	0.00	0.00	29
Aggressive left 45-turn	0.91	0.37	0.53	27
Aggressive left 90-turn	0.65	0.78	0.71	100
Aggressive left lane change	0.65	0.67	0.66	223
Aggressive right 45-turn	0.80	0.13	0.22	31
Aggressive right 90-turn	0.87	0.87	0.87	149
Aggressive right lane change	0.66	0.78	0.71	251
accuracy			0.70	810
macro avg	0.65	0.51	0.53	810
weighted avg	0.68	0.70	0.68	810

```
0.6987654320987654
[[  0   0  22   3   0   0   4]
 [  0  10  12   4   0   0   1]
 [  0   0  78  16   0   0   6]
 [  0   1   8 149   0   0  65]
 [  0   0   0   2   4  17   8]
 [  0   0   0   2   1 130  16]
 [  0   0   0  53   0   3 195]]
```

```
C:\Users\talha.javed\Anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

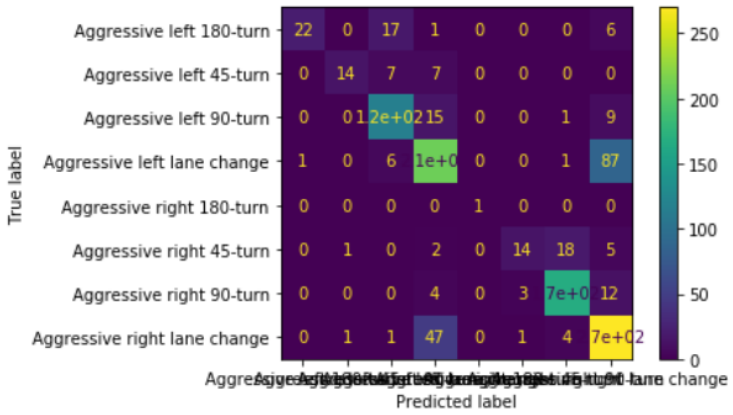


Gyro 4 frames

```
{'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.96	0.48	0.64	46
Aggressive left 45-turn	0.88	0.50	0.64	28
Aggressive left 90-turn	0.79	0.82	0.81	142
Aggressive left lane change	0.73	0.69	0.71	305
Aggressive right 180-turn	1.00	1.00	1.00	1
Aggressive right 45-turn	0.78	0.35	0.48	40
Aggressive right 90-turn	0.87	0.90	0.89	186
Aggressive right lane change	0.69	0.83	0.76	324
accuracy			0.76	1072
macro avg	0.84	0.70	0.74	1072
weighted avg	0.77	0.76	0.75	1072

```
0.7602611940298507
[[ 22   0  17   1   0   0   0   6]
 [  0  14   7   7   0   0   0   0]
 [  0   0 117  15   0   0   1   9]
 [  1   0   6 210   0   0   1  87]
 [  0   0   0   0   1   0   0   0]
 [  0   1   0   2   0  14  18   5]
 [  0   0   0   4   0   3 167  12]
 [  0   1   1  47   0   1   4 270]]
```



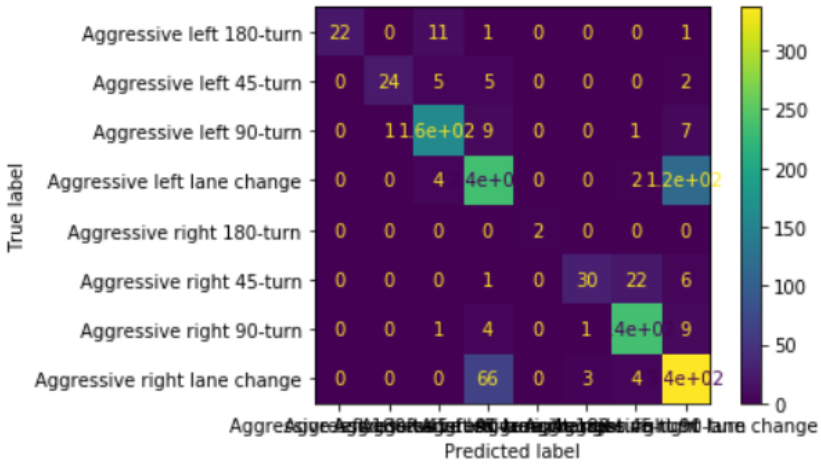
Gyro 5 frames

```
{'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                              precision    recall  f1-score   support

    Aggressive left 180-turn       1.00      0.63      0.77        35
     Aggressive left 45-turn       0.96      0.67      0.79        36
     Aggressive left 90-turn       0.88      0.90      0.89       177
 Aggressive left lane change       0.73      0.66      0.69       358
   Aggressive right 180-turn       1.00      1.00      1.00         2
    Aggressive right 45-turn       0.88      0.51      0.65        59
    Aggressive right 90-turn       0.89      0.94      0.91       251
Aggressive right lane change       0.70      0.82      0.76       410

                    accuracy                           0.79      1328
                   macro avg       0.88      0.77      0.81      1328
                weighted avg       0.79      0.79      0.78      1328

 0.7868975903614458
 [[ 22   0  11   1   0   0   0   1]
  [  0  24   5   5   0   0   0   2]
  [  0   1 159   9   0   0   1   7]
  [  0   0   4 235   0   0   2 117]
  [  0   0   0   0   2   0   0   0]
  [  0   0   0   1   0  30  22   6]
  [  0   0   1   4   0   1 236   9]
  [  0   0   0  66   0   3   4 337]]
```



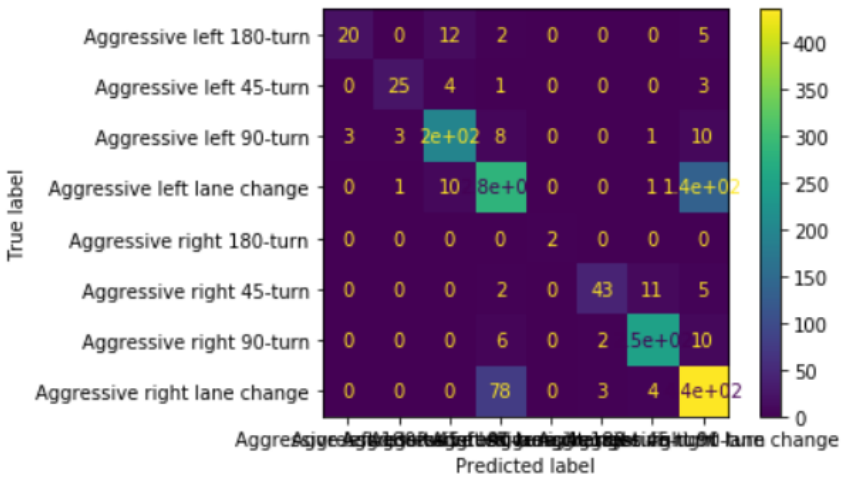
Gyro 6 frames

```
{'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       0.87      0.51      0.65        39
     Aggressive left 45-turn       0.86      0.76      0.81        33
     Aggressive left 90-turn       0.88      0.89      0.89       222
 Aggressive left lane change       0.74      0.66      0.70       430
   Aggressive right 180-turn       1.00      1.00      1.00         2
    Aggressive right 45-turn       0.90      0.70      0.79        61
    Aggressive right 90-turn       0.94      0.93      0.93       267
Aggressive right lane change       0.72      0.84      0.78       521

                    accuracy                           0.80      1575
                   macro avg       0.86      0.79      0.82      1575
                weighted avg       0.80      0.80      0.80      1575


0.7968253968253968
[[ 20   0  12   2   0   0   0   5]
 [  0  25   4   1   0   0   0   3]
 [  3   3 197   8   0   0   1  10]
 [  0   1  10 283   0   0   1 135]
 [  0   0   0   0   2   0   0   0]
 [  0   0   0   2   0  43  11   5]
 [  0   0   0   6   0   2 249  10]
 [  0   0   0  78   0   3   4 436]]
```



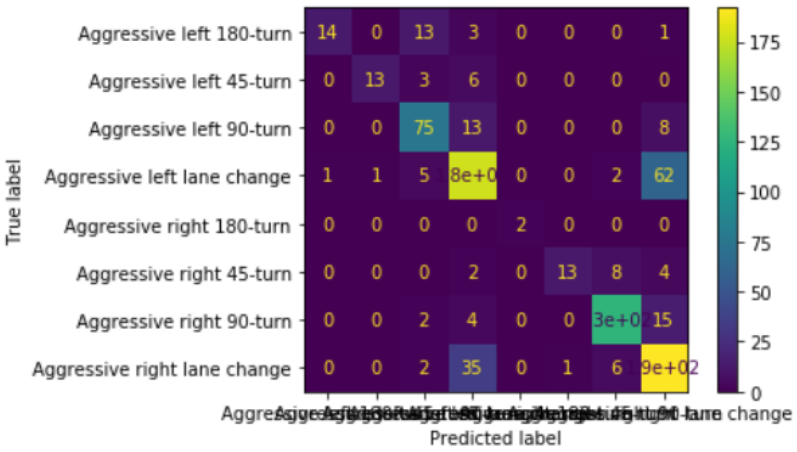
Gyro+Acc 3 frames

```
{'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       0.93      0.45      0.61        31
     Aggressive left 45-turn       0.93      0.59      0.72        22
     Aggressive left 90-turn       0.75      0.78      0.77        96
 Aggressive left lane change       0.74      0.71      0.73       249
   Aggressive right 180-turn       1.00      1.00      1.00         2
    Aggressive right 45-turn       0.93      0.48      0.63        27
    Aggressive right 90-turn       0.89      0.86      0.87       147
Aggressive right lane change       0.68      0.81      0.74       236

                    accuracy                           0.76       810
                   macro avg       0.86      0.71      0.76       810
                weighted avg       0.77      0.76      0.75       810


0.7567901234567901
[[ 14   0  13   3   0   0   0   1]
 [  0  13   3   6   0   0   0   0]
 [  0   0  75  13   0   0   0   8]
 [  1   1   5 178   0   0   2  62]
 [  0   0   0   0   2   0   0   0]
 [  0   0   0   2   0  13   8   4]
 [  0   0   2   4   0   0 126  15]
 [  0   0   2  35   0   1   6 192]]
```

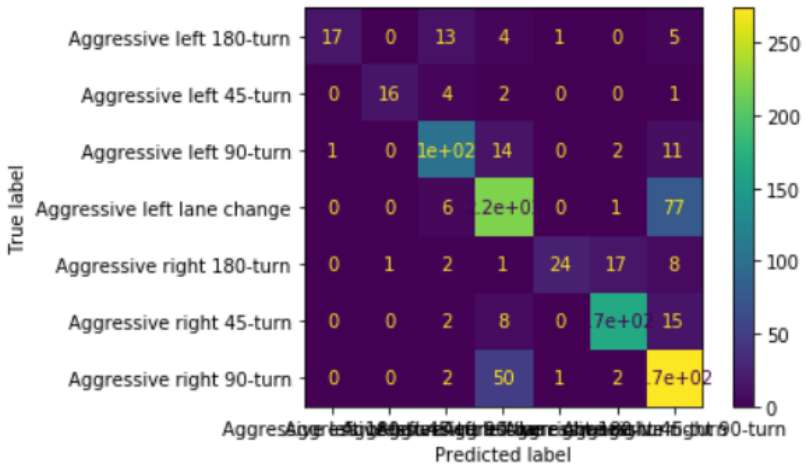


Gyro+Acc 4 frames

```
{'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.94	0.42	0.59	40
Aggressive left 45-turn	0.94	0.70	0.80	23
Aggressive left 90-turn	0.78	0.78	0.78	130
Aggressive left lane change	0.74	0.72	0.73	305
Aggressive right 45-turn	0.92	0.45	0.61	53
Aggressive right 90-turn	0.88	0.87	0.88	192
Aggressive right lane change	0.70	0.83	0.76	329
accuracy			0.77	1072
macro avg	0.84	0.68	0.73	1072
weighted avg	0.78	0.77	0.76	1072

```
0.7658582089552238
[[ 17   0  13   4   1   0   5]
 [  0  16   4   2   0   0   1]
 [  1   0 102  14   0   2  11]
 [  0   0   6 221   0   1  77]
 [  0   1   2   1  24  17   8]
 [  0   0   2   8   0 167  15]
 [  0   0   2  50   1   2 274]]
```



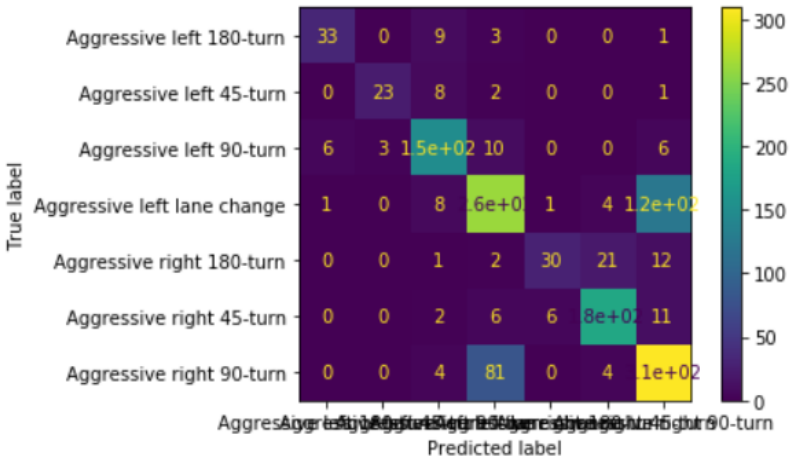
Gyro+Acc 5 frames

```
{'C': 1000, 'gamma': 1, 'kernel': 'linear'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       0.82      0.72      0.77        46
     Aggressive left 45-turn       0.88      0.68      0.77        34
     Aggressive left 90-turn       0.82      0.86      0.84       175
 Aggressive left lane change       0.72      0.66      0.69       399
    Aggressive right 45-turn       0.81      0.45      0.58        66
    Aggressive right 90-turn       0.86      0.88      0.87       210
Aggressive right lane change       0.67      0.78      0.72       398

                    accuracy                           0.75      1328
                   macro avg       0.80      0.72      0.75      1328
                weighted avg       0.75      0.75      0.75      1328


0.7477409638554217
[[ 33   0   9   3   0   0   1]
 [  0  23   8   2   0   0   1]
 [  6   3 150  10   0   0   6]
 [  1   0   8 263   1   4 122]
 [  0   0   1   2  30  21  12]
 [  0   0   2   6   6 185  11]
 [  0   0   4  81   0   4 309]]
```



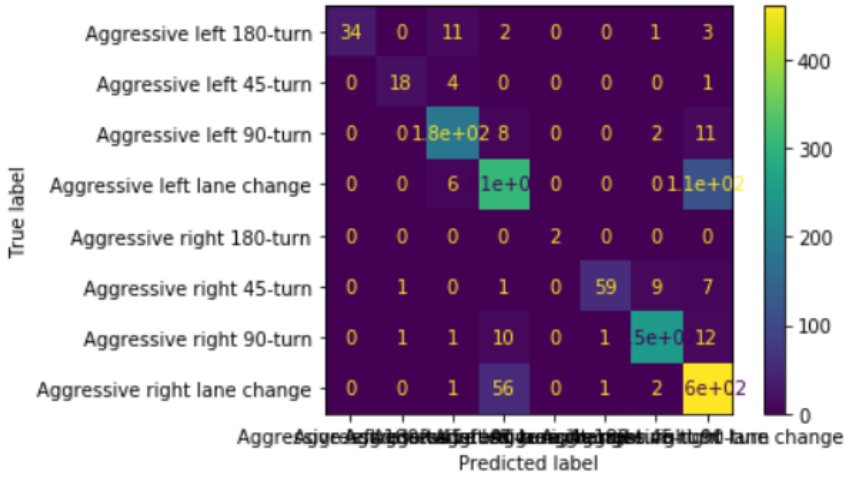
Gyro+Acc 6 frames

```
{'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       1.00      0.67      0.80        51
     Aggressive left 45-turn       0.90      0.78      0.84        23
     Aggressive left 90-turn       0.89      0.90      0.89       202
 Aggressive left lane change       0.80      0.72      0.76       426
   Aggressive right 180-turn       1.00      1.00      1.00         2
    Aggressive right 45-turn       0.97      0.77      0.86        77
    Aggressive right 90-turn       0.95      0.91      0.93       274
Aggressive right lane change       0.76      0.88      0.82       520

                    accuracy                           0.83      1575
                   macro avg       0.91      0.83      0.86      1575
                weighted avg       0.84      0.83      0.83      1575


0.8311111111111111
[[ 34   0  11   2   0   0   1   3]
 [  0  18   4   0   0   0   0   1]
 [  0   0 181   8   0   0   2  11]
 [  0   0   6 306   0   0   0 114]
 [  0   0   0   0   2   0   0   0]
 [  0   1   0   1   0  59   9   7]
 [  0   1   1  10   0   1 249  12]
 [  0   0   1  56   0   1   2 460]]
```



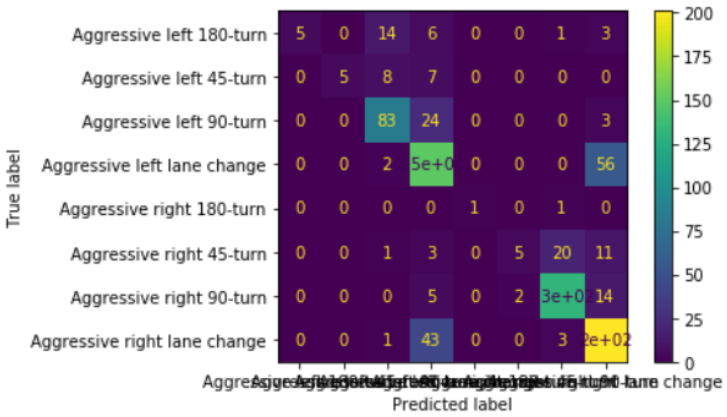
Gyro z 3 frames

```
{'C': 1000, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
                            precision    recall  f1-score   support

    Aggressive left 180-turn       1.00      0.17      0.29        29
     Aggressive left 45-turn       1.00      0.25      0.40        20
     Aggressive left 90-turn       0.76      0.75      0.76       110
 Aggressive left lane change       0.63      0.72      0.67       208
   Aggressive right 180-turn       1.00      0.50      0.67         2
    Aggressive right 45-turn       0.71      0.12      0.21        40
    Aggressive right 90-turn       0.84      0.86      0.85       153
Aggressive right lane change       0.70      0.81      0.75       248

                    accuracy                           0.72       810
                   macro avg       0.83      0.52      0.58       810
                weighted avg       0.74      0.72      0.70       810


  0.7185185185185186
  [[  5   0  14   6   0   0   1   3]
   [  0   5   8   7   0   0   0   0]
   [  0   0  83  24   0   0   0   3]
   [  0   0   2 150   0   0   0  56]
   [  0   0   0   0   1   0   1   0]
   [  0   0   1   3   0   5  20  11]
   [  0   0   0   5   0   2 132  14]
   [  0   0   1  43   0   0   3 201]]
```

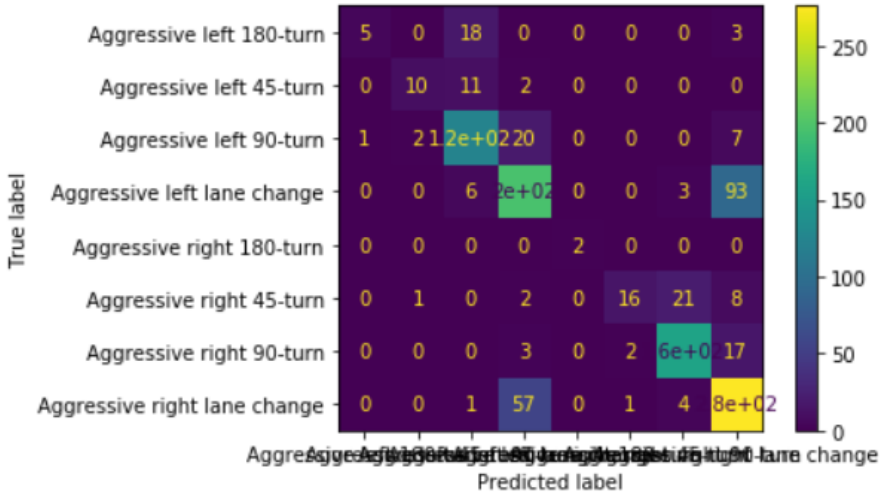


Gyro z 4 frames

```
{'C': 1000, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	0.83	0.19	0.31	26
Aggressive left 45-turn	0.77	0.43	0.56	23
Aggressive left 90-turn	0.77	0.80	0.79	153
Aggressive left lane change	0.70	0.66	0.68	299
Aggressive right 180-turn	1.00	1.00	1.00	2
Aggressive right 45-turn	0.84	0.33	0.48	48
Aggressive right 90-turn	0.85	0.88	0.86	182
Aggressive right lane change	0.68	0.81	0.74	339
accuracy			0.74	1072
macro avg	0.81	0.64	0.68	1072
weighted avg	0.74	0.74	0.73	1072

```
0.7360074626865671
[[  5   0  18   0   0   0   0   3]
 [  0  10  11   2   0   0   0   0]
 [  1   2 123  20   0   0   0   7]
 [  0   0   6 197   0   0   3  93]
 [  0   0   0   0   2   0   0   0]
 [  0   1   0   2   0  16  21   8]
 [  0   0   0   3   0   2 160  17]
 [  0   0   1  57   0   1   4 276]]
```

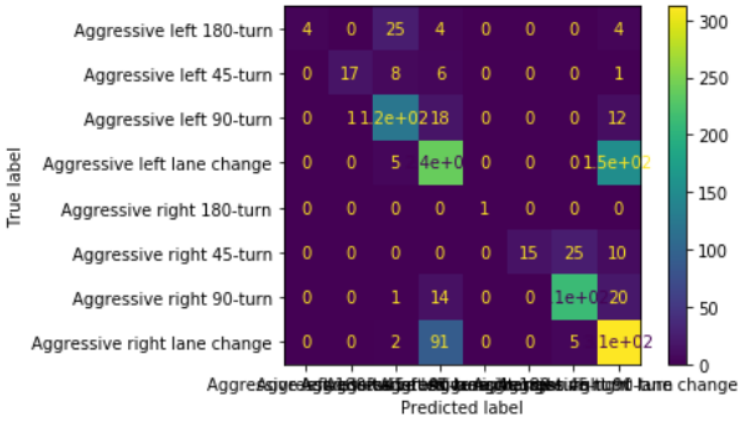


Gyro z 5 frames

```
{'C': 100, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	1.00	0.11	0.20	37
Aggressive left 45-turn	0.94	0.53	0.68	32
Aggressive left 90-turn	0.75	0.80	0.77	153
Aggressive left lane change	0.64	0.60	0.62	397
Aggressive right 180-turn	1.00	1.00	1.00	1
Aggressive right 45-turn	1.00	0.30	0.46	50
Aggressive right 90-turn	0.88	0.86	0.87	248
Aggressive right lane change	0.61	0.76	0.68	410
accuracy			0.70	1328
macro avg	0.85	0.62	0.66	1328
weighted avg	0.72	0.70	0.69	1328

```
0.6950301204819277
[[  4   0  25   4   0   0   0   4]
 [  0  17   8   6   0   0   0   1]
 [  0   1 122  18   0   0   0  12]
 [  0   0   5 239   0   0   0 153]
 [  0   0   0   0   1   0   0   0]
 [  0   0   0   0   0  15  25  10]
 [  0   0   1  14   0   0 213  20]
 [  0   0   2  91   0   0   5 312]]
```

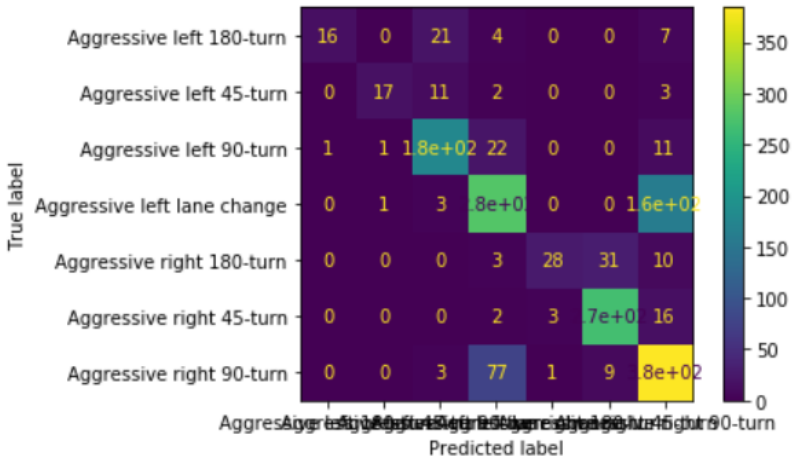


Gyro z 6 frames

```
{'C': 100, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

	precision	recall	f1-score	support
Aggressive left 180-turn	1.00	0.11	0.20	37
Aggressive left 45-turn	0.94	0.53	0.68	32
Aggressive left 90-turn	0.75	0.80	0.77	153
Aggressive left lane change	0.64	0.60	0.62	397
Aggressive right 180-turn	1.00	1.00	1.00	1
Aggressive right 45-turn	1.00	0.30	0.46	50
Aggressive right 90-turn	0.88	0.86	0.87	248
Aggressive right lane change	0.61	0.76	0.68	410
accuracy			0.70	1328
macro avg	0.85	0.62	0.66	1328
weighted avg	0.72	0.70	0.69	1328

```
0.7441269841269841
[[ 16   0  21   4   0   0   7]
 [  0  17  11   2   0   0   3]
 [  1   1 177  22   0   0  11]
 [  0   1   3 281   0   0 161]
 [  0   0   0   3  28  31  10]
 [  0   0   0   2   3 269  16]
 [  0   0   3  77   1   9 384]]
```



5.7.4 Driver Scoring

After classifying maneuvers, the final step is to assign a score to each trip. Our scoring algorithm takes as input the number of times each maneuver has occurred in a trip, along with a trip length. Each maneuver is assigned a different weights in the overall score calculation based on their correlation to risk.

Driver Scoring - Risk

Cases	Right Lane Change	Left Lane Change	Right Turn	Left Turn	Braking	Acceleration	Duration	Score
1	8	4	9	7	8	8	10 mins	1
2	0	0	0	0	0	0	10 mins	10
3	3	1	2	3	2	2	10 mins	6.45
4	2	1	2	3	2	0	10 mins	8
5	3	1	2	3	2	2	7 mins	4.31
6	3	1	2	3	2	2	15 mins	8

This is how data is stored before assiging a score. Each vector is passed to the algorithm which then ouputs the score.

Results on two trips

Test Trip 1

Trip Characteristics

- Fairly slow driving under the speed limit.
- A few Isolated acceleration and braking events
- On average 1/3 lane changes and turns were aggressive

Driving Analytics

DRIVING EVENT	VALUE	FINAL SCORE
		Risk Score
Duration	10 mins	8
No. of acceleration events	0	
No. of braking events	2	
No. of lane changes	3	Eco Score
No. of turns	5	6.5

Score for Sample trip 2

Test Trip 2

Trip Characteristics

- Speed was occasionally rated as too fast by onboard passengers.
- Number of acceleration and braking events higher than previous driver.

Driving Analytics

DRIVING EVENT	VALUE	FINAL SCORE
		Risk Score
Duration	10 mins	6.45
No. of acceleration events	2	
No. of braking events	2	
No. of lane changes	4	
No. of turns	5	4

Score for Sample trip 2

6. Conclusion and Future Work

For our project, we set out to use telemetry data from Smartphones scoring drivers based on how many aggressive maneuvers they made with respect to the duration of the trip. We successfully employed a pre-processing pipeline which would re-orient and smooth the data to convert to a form that we can work with. This is followed by a feature extraction pipepline in which we segment our data in windows and identify windows with a high probability of a maneuver occurring. This enables us to significantly reduce our search space, thereby improving performance. We then calculate a new set of features for the identified windows and pass it through one of our machine learning algorithms to classify a label to. Finally, we then pass our classified results to the driver scoring algorithm, which then assigns a risk-based score to each trip.

If we consider the driver behavior dataset, then we weren't able to have a good accuracy over them. We considered three types of data for classification, acc 5 frames, gyro 5 frames and acc 8 frames. All of them scored around 30-35% accuracy which is a poor result from our perspective. The reason for such a low accuracy can be less data set for machine learning training.

Our algorithm performed the best when we trained it using the computer science center dataset. Out of all the classifications we performed using differently pre-processed data, we got the best result by using the gyroscope and accelerometer data concatenated together for classification. The accuracy lied between 73%-83% for 3 frames to 6 frames. As we increased the number of frames, although the accuracy of the classifier kept increasing, the training time increased too. So, through all experimentation and results, we realized that our algorithm performed the best for the combined data of accelerometer and gyroscope. We can choose the number of frames depending on the resources and time constraint.

For future work, we would like to further test the driver scoring algorithm. Although

we were able the trips' trajectory to the real world, there is a lot to explore on how we can use road condition data to make our driver scoring algorithm more contextual. Additionally, factors such as time of day, weather data can also be incorporated into the pipeline, as they also affect the overall number of maneuvers one would make.

Appendix A. More Math

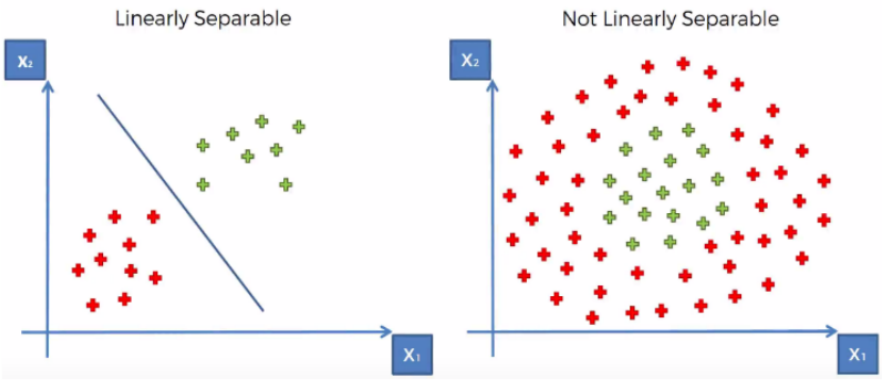
A.1 Support Vector Machines

SVM is one of the most popular machine learning algorithms that is being used for classification and regression problems. SVM tries to find a hyper plane to linearly separates the data with the maximum margin [6]

Hyper Plane:

Hyper plane divides the n-dimensional data in two components. Hyperplane is a line in 2D, a plane in 3D and so on.

Fig.3 Fig.4



In figure 3, the data points are linearly separable.

In figure 4, we cannot draw a 2d line to separate them. So in order, to linearly separate them, we increase the dimensionality of our data.

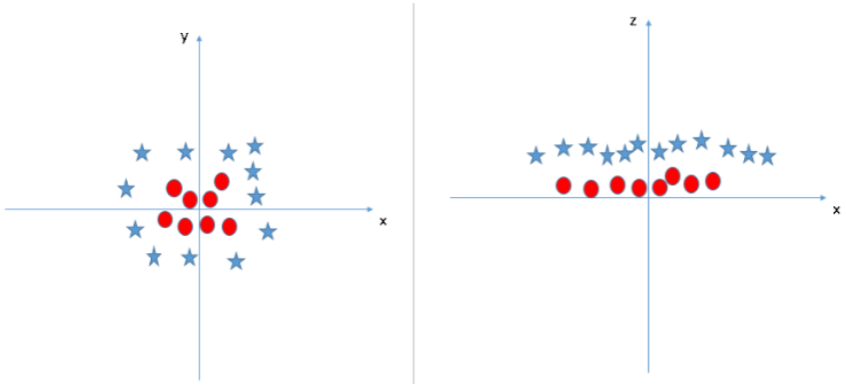


Fig.5

It is evident from fig 5 that it can be easily separated linearly.

Optimal Hyperplane

Fig 6 shows alot of hyperplanes that successfully linearly separate the data. We need to choose the optimal hyperplane so that the algorithm performs good in case of unseen data

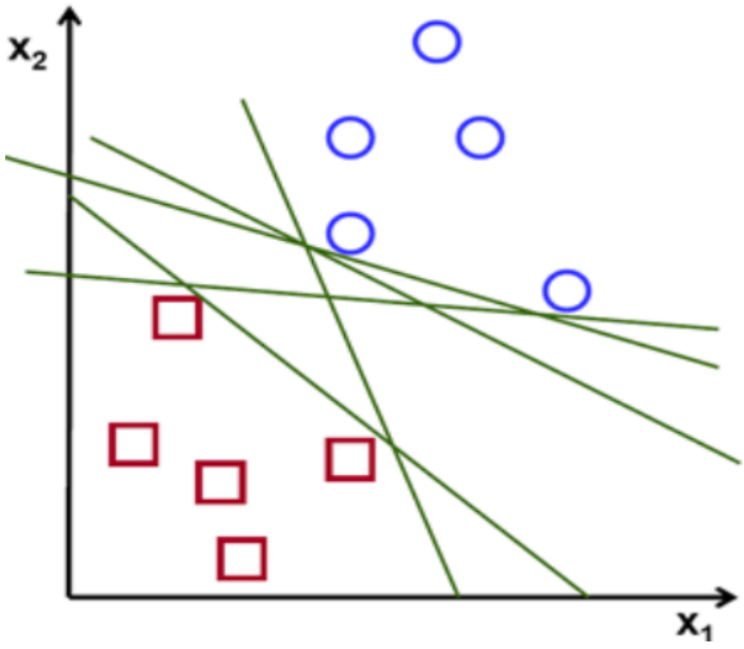


Fig.6

When choosing the optimal hyperplane, we will choose one which has highest distance from the closest data points. Margin refers to the closeness of hyperplane with the data points. Closer the data points are to hyperplane, the lower be the margin and vice versa.

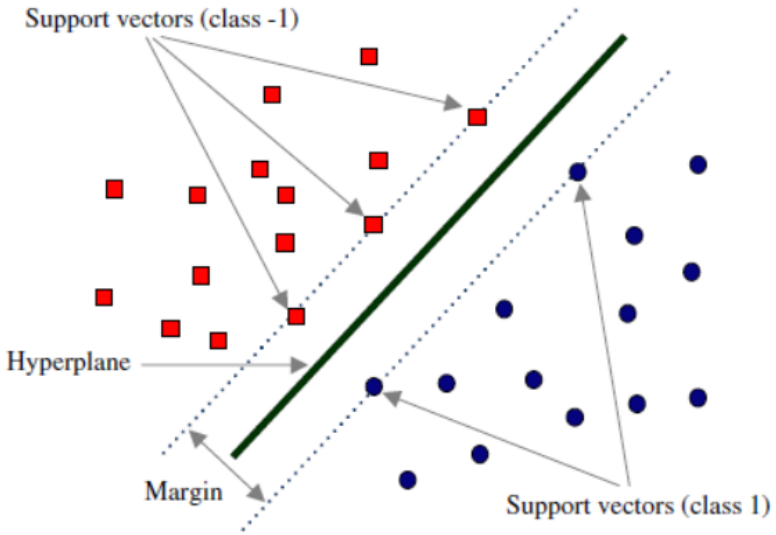


Fig.7

The goal of SVM is to such an optimal hyperplane that maximizes the margin

Appendix B. Data

Our used data sets can be found at

B.1 Driver Behaviour Dataset

B.2 Computer Science Centre Dataset

Appendix C. Code

Our code can be found at

C.1 Maneuver Classification repo

References

[1] URL: https://erticonetwork.com/wp-content/uploads/2017/12/UDRIVE-D41.1-UDrive-dataset-and-key-analysis-results-with-annotation-codebook.pdf.

[2] C. Arroyo, L. M. Bergasa, and E. Romera. "Adaptive fuzzy classifier to detect driving events from the inertial sensors of a smartphone". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2016, pp. 1896–1901. DOI: 10.1109/ITSC.2016.7795863.

[3] Jair Ferreira Júnior et al. "Driver behavior profiling: An investigation with different smartphone sensors and machine learning". In: *PLOS ONE* 12.4 (Apr. 2017), pp. 1–16. DOI: 10.1371/journal.pone.0174959. URL: https://doi.org/10.1371/journal.pone.0174959.

[4] J. R. López et al. "A Genetic Programming Approach for Driving Score Calculation in the Context of Intelligent Transportation Systems". In: *IEEE Sensors Journal* 18.17 (Sept. 2018), pp. 7183–7192. ISSN: 2379-9153. DOI: 10.1109/JSEN.2018.2856112.

[5] Gys Meiring and Herman Myburgh. "A Review of Intelligent Driving Style Analysis Systems and Related Artificial Intelligence Algorithms". In: *Sensors* 15 (Dec. 2015), pp. 30653–30682. DOI: 10.3390/s151229822.

[6] MLMath.io. *Math behind Support Vector Machine(SVM)*. Feb. 2019. URL: https://medium.com/@ankitnitjsr13/math-behind-support-vector-machine-svm-5e7376d0ee4d.

[7] Vasili Ramanishka et al. "Toward Driving Scene Understanding: A Dataset for Learning Driver Behavior and Causal Reasoning". In: *CoRR* abs/1811.02307 (2018). arXiv: 1811.02307. URL: http://arxiv.org/abs/1811.02307.

[8] E. Romera, L. M. Bergasa, and R. Arroyo. "Need data for driver behaviour analysis? Presenting the public UAH-DriveSet". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2016, pp. 387–392. DOI: 10.1109/ITSC.2016.7795584.

[9] C. Woo and D. Kulić. "Manoeuvre segmentation using smartphone sensors". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. June 2016, pp. 572–577. DOI: 10.1109/IVS.2016.7535444.