Mestrado em Engenharia Informática
Estágio
Relatório Intermédio

Aumente o QI do seu smartphone: mobilidade ao serviço da segurança na estrada

João Carlos de Lima Campos
campos@student.dei.uc.pt

Orientador:
António Jorge da Costa Granjal
jgranjal@dei.uc.pt

Co-Orientador:
Bruno Miguel Brás Cabral
bcabral@dei.uc.pt

Co-Orientador:
Nuno António Marques Lourenço
naml@dei.uc.pt

Data: 3 de julho de 2017

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

Traffic incidents kill about 1.3 million people per year and the number of deaths has been rising for the past decade. Our contributions in the field of detecting and profiling a driver behavior represent an attempt to positively impact this problem.Current state of the art is untested in real usage. To tackle this problem, we developed a simulator that is capable of replicating real usage. This dissertation is the result of an internship at Sentilant. They are looking to improve their current driving behavior detection algorithm by incorporating machine learning into the existing system in order to improve the feedback given to the users. Our findings suggest that taking into consideration not only past samples but also look ahead samples results in an increased performance on supervised algorithms. A set of well engineered samples was all it took to go from mediocre results to having the model behaving as expected when oriented vertically. A novel balancing technique is here presented and demonstrated to rival oversampling but without increasing the size of the dataset, achieving lower training times. Our best model is capable of achieving 74% precision for acceleration, 49% precision for brakes and 18% precision for turns. In real validation trips, the algorithm behaved as expected when the phone was placed vertically, despite the sensibility it demonstrated to turns. This was only possible because we devised a new post-processing mechanism that only allows a model to classify a sample as aggressive behavior after the same event has appeared consecutively a pre-defined amount of times.

Keywords: driving behavior, driving profiling, driving event detection

Contents

Abstract ii

List of Figures 1

List of Tables 3

1 Introduction 7
- 1.1 Context . . . 8
- 1.2 Goals . . . 8
- 1.3 Contributions . . . 9
- 1.4 Remaining structure . . . 10

2 State of the art 11
- 2.1 Background . . . 11
- 2.2 Distance based classification . . . 16
- 2.3 Machine Learning algorithms . . . 21
- 2.4 Summary of aforementioned work . . . 27

3 Research goals and methodology 31
- 3.1 Research goals . . . 31
- 3.2 Methodology . . . 32
- 3.3 Applying post-processing on top of a simulator . . . 42

4 Experimental setup and results 45
- 4.1 Experiment 1 . . . 46
- 4.2 Experiment 2 . . . 51
- 4.3 Experiment 3 . . . 54
- 4.4 Experiment 4 . . . 56
- 4.5 Experiment 5 . . . 62

5 Work Plan and risk analysis **65**
5.1 Work plan 65
5.2 Risk analysis 66

6 Conclusions **69**

Appendices **71**

Appendix A Comparison of the original feature set with the modified feature set "A" **71**

Appendix B Comparison of the original feature set with the modified feature set "no gyroscope" **73**

Appendix C Comparison of the original feature set with the modified feature set "no gyroscope-A" **75**

Appendix D Comparison of the original feature set with the modified feature set "A2" **77**

Appendix E Comparison of the original feature set with the modified feature set "no gyroscope-A2" **79**

Appendix F Description of all the features engineered **81**

Appendix G List of seeds used **91**

Bibliography **92**

List of Figures

2.1 Typical Hidden Markov model representation. 16
2.2 Six types of abnormal driving behaviors: (a) Weaving, (b) Swerving, (c) Sideslipping, (d) Fast U-turn, (e) Turning with a wide radius, (f) Sudden braking. 27

3.1 Support for each class of the dataset provided by Sentilant. 38
3.2 Support for each class of the dataset acquired in April. 38
3.3 Illustration of the process used to balance our datasets. 39
3.4 Illustration of the process used to predict an arbitrary instance. On the left we have the current approach described on the literature. On the right we have our proposed approach. The color blue denotes samples that are available to the algorithm. Red stands for the sample currently being classified. 41
3.5 Illustration of a misclassified event being corrected in Case 1. Case 2 displays the threshold function. The height of the bars is proportional to the number of consecutive labels necessary to achieve the threshold set for that event. The color blue denotes normal behaviour and red aggressive behaviour. 43

4.1 Average F1 score for SVM (OVO) trained with 22 features. The colors represent the class balancing technique used. Blue for random undersampling, red for random oversampling and green for our technique, binary undersampling. 48
4.2 Average F1 score for random forests trained with 22 features. The colors represent the class balancing technique used. Blue for random undersampling, red for random oversampling and green for our technique, binary undersampling. 48
4.3 Normalize confusion matrix of the best performing SVM shown in the above graph. 48
4.4 Normalized confusion matrix of the best performing random forest in the above graph. 48

4.5 Average F1 score of SVMs trained with 25 features. The labels on the x axis identify the pre-processing mechanism and the color distinguishes the different balancing strategy. 49
4.6 Average F1 score of random forests trained with 25 features. The labels on the x axis identify the pre-processing mechanism and the color distinguishes the different balancing strategy. 49
4.7 Average F1 score of all the classifiers tested. To balance the dataset we used binary undersampling and the number of features was 25. The color denote different pre-processing mechanisms. 50
4.8 Average F1 score for SVMs, XGBoost and random forests trained with feature set "A". The colors stand for the scaling method used. 52
4.9 Average F1 score for SVMs, XGBoost and random forests trained with feature set "no gyroscope". The colors stand for the scaling method used. 53
4.10 Average F1 score for SVMs, XGBoost and random forests trained with feature set "no gyroscope-A". The colors stand for the scaling method used. 53
4.11 Average F1 score for XGBoost on two different feature sets for the iOS platform on a vertical position. 58
4.12 Average F1 score for XGBoost on two different feature sets for the Android platform on a vertical position. 58
4.13 Confusion matrix for the best performing xgboost model on iOS using the "A2" feature set. 58
4.14 Confusion matrix for the best performing xgboost model on iOS using the "1-gyroscope-A3" feature set. 58
4.15 Identified events on iOS during a trip to iPark, located in Antanhol, Coimbra. Green markers symbolize turns, red markers brakes and dark purple accelerations. 60
4.16 Identified events on an Android during a trip to iPark, located in Antanhol, Coimbra. Green markers symbolize turns, red markers brakes and dark purple accelerations. 61
4.17 Identified events on an Android during a trip Cernache. Green markers symbolize turns, red markers brakes and dark purple accelerations. 62

5.1 Gantt chart of the tasks done in the first half of this dissertation. . 65
5.2 Gantt chart of the tasks scheduled for the second half of the work. . 66

List of Tables

2.1 Summary of all mobile-only approaches considered. 28

3.1 Summary of the datasets used for this dissertation 36

4.1 Detail of the dataset used in Experiment 1 46
4.2 Detail of Experiment 1 46
4.3 A comparison of the recall of the current system vs our best model so far. 50
4.4 Details of the dataset used in Experiment 2 51
4.5 Details of Experiment 2 51
4.6 Details of the dataset used in Experiment 3 54
4.7 Detail of Experiment 3 55
4.8 Results in tabular form for Experiment 3. Colors have different meanings, green represents the horizontal position, yellow vertical and grey the hand. The gradient from red to green represents the average of the average F1 score. Values closer to green are better. The last column contains the feature set used, blue for “A2” set and orange for “no gyroscope-A2”set. 56
4.9 Details of the dataset used in Experiment 3 56
4.10 Detail of Experiment 4 57
4.11 Performance report on iOS trip used to validate the model, without post-processing 59
4.12 Performance report on iOS trip used to validate the model, with post-processing 59
4.13 Performance report on iOS trip used to validate the model, with post-processing 63

A.1 Comparison of the original feature set and the feature set ”A“. . . . 71

B.1 Comparison of the original feature set and the “no gyroscope” feature set. 73

C.1 Comparison of the original feature set and the “no gyroscope-A” feature set. 75

D.1 Comparison of the original feature set and the feature set ”A2“. . . . 77

E.1 Comparison of the original feature set and the “no gyroscope-A” feature set. 79

F.1 Description of each feature engineered. 81

G.1 List of seeds used. 91

Acronyms

AI Artificial Intelligence.

CAN Controller Area Network.

DTW Dynamic Time Warping.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

GSNS Global Satellite Navigation System.

HMM Hidden Markov Model.

K-NN K-Nearest Neighbor.

ML Machine Learning.

OVA One-vs-All.

OVO One-vs-One.

PAA Piecewise Aggregate Approximation.

RBF Radial Basis Function.

SAX Symbolic Aggregate ApproXimation.

SDK Software Development Kit.

SVM Support Vector Machine.

Chapter 1

Introduction

According to the World Health Organization [29], road injuries are the seventh cause of death and has increased over 30% in the last decade, killing about 1.3 million people and harming between 20 to 50 million per year. Road injuries do not only cause suffering to the victims and people around them, but they also have an impact on the countries' economy. It is estimated that 3 to 5% of the gross national product of each country is used to take care of such injuries [28]. These numbers show that the research field of detecting a driver's behavior is of great importance.
From a psychological standpoint, there was no guarantee that the user would listen and follow guidelines provided by a digital system until an article in the Journal of Organizational Behavior Management was published [12]. The authors of this article conducted an experiment to determine if being monitored had any effect on the driving behavior of truck drivers, whose driving skills and performance impact their job. The study reached significant results and managed to find a positive association between being monitored and less aggressive behaviors.
Car insurance companies are starting to look at machine learning techniques to classify a driver's behavior and use the score given by the algorithm, along with other factors, to determine the insurance rate [33] not only for professional drivers but also for the day to day driver.
The problem with current solutions that monitor the driver's behavior is that they are expensive and require a lot of sensors. Recently, there has been an emerging interest to transition from such expensive systems to smartphones because of their availability and low cost. This type of solutions are only now being considered due to the exponential increase of computational power that smartphones have seen in the last years [5] and improvements in the sensors carried by them.

1.1 Context

This dissertation is the result of an internship at Sentilant, a company established in Instituto Pedro Nunes, in Coimbra. The work presented here is part of an ongoing effort by the company to continuously study innovative and disruptive ways to use mobile devices. Currently the following products:

- Drivian - a personal driving coach platform that has an application for smartphones which offers realtime feedback regarding safety, economy and on the road alerts as well as driving insights that show the user driving style progress over past trips.
- Drivian Tasks - an operation and fleet management platform that provides insights on transportation and logistics, security services, retail and commerce, services and operations.

The work produced here may be used in the Drivian Software Development Kit, which will influence the Drivian and Drivian Tasks platforms by providing enhanced event detection and promote road safety.

1.2 Goals

Sentilant is looking to improve their current driving behavior detection algorithm by incorporating Artificial Intelligence (AI), more specifically, Machine Learning (ML) into the existing system in order to improve the feedback that the existing system is providing to the user. We studied three behaviours that we think constitute the basics of driving: accelerating, braking and turning. In more detail, the goals of this dissertation are the following:

1. The algorithm shall achieve the best results possible. For the first semester we focused on recall because that was the only metric we could extract from the current system. Recall is the ability to correctly detect a driving pattern. For the second semester, our focus was on precision, which is the metric that matters the most for the company. A high precision value is desirable because feedback should only be given when we are certain that an aggressive maneuver actually happened. Precision gives us information about false positives, which is what we want to minimize;
2. The algorithm developed must be platform agnostic (no distinction between Android and iOS);

3. The software produced shall be able to achieve results that do not differ on the performance metric (check Chapter 3 to see the chosen metric and why we chose it) on multiple hardware configurations, namely on the Android platform where not all the sensors are mandatory (we were concerned about the lack of a gyroscope because it is an important sensor for the current system);

1.3 Contributions

The contributions laid out in this dissertation can be summarized as follows:

1. The available dataset had four highly unbalanced classes and with simple random oversampling or undersampling we ran into multiple problems. To leverage all the available data without running into those problems, we devised a new class balancing algorithm. Our approach was to first encode our labels as binary (normal and aggressive behaviour), do random undersampling, and then decode them back to the original multiclass problem. We show that this method, for our problem, gave results similar to random oversampling while being more time efficient;

2. We used previous research by Sentilant to create a new set of features, different from the current literature, and combined them with state of the art machine learning techniques to achieve promising results;

3. To prevent the occasional misclassification we introduced post-processing as one more layer of logic. In post-processing we constrain our model to classify a set number of consecutive samples as one unique abnormal event before outputting it;

4. To test our contributions we implemented a framework. The framework developed allowed us to test many things quickly and introduce new changes with minimal intervention. Common steps such as choosing a classifier, feature scaling, feature engineering and feature selection, among other settings, became easy to change and test;

5. We intent to publish a paper. For now we have in mind two conferences and/or two journals. The conferences are the International conference on Communication Systems and Networks, COMSNET and the European conference on Machine Learning & Principles and Practice of Knowledge Discovery in Databases, ECML PKDD. The journals we are debating are the IEEE Transactions on Intelligent Transportation Systems and the International Journal of Computer Applications.

1.4 Remaining structure

The remaining dissertation is structured as follows. Chapter 2 presents the current state of the art in the field of vehicle telematics from a user behavior perspective and is restricted to work that uses mobile devices only. Chapter 3 details the research proposal, the study of the existing system and the research methodology followed. Chapter 4 presents the current work done and statistical validation for it. Chapter 5 specifies the work plan stipulated for the thesis, a risk analysis and a mitigation plan. Chapter 6 contains a summary of our findings and contributions.

Chapter 2

State of the art

In this chapter the relevant work for this dissertation is summarized and critically analyzed with the purpose of gathering knowledge and methods that use machine learning in the field of driving security based on driver behavior feedback, particularly approaches that focus on smartphones. Section 2.1 contains brief descriptions of the terms that will be used throughout this thesis and whose understanding is important for the comprehension of the work here presented. In Section 2.2 and Section 2.3 we introduce related work on the two major different approaches to this problem, Distance based classification and Machine learning algorithms, respectively. In Section 2.4 we present a summary of all referred work and make a critical analysis of the field as a whole. Note that the current system from Sentilant is not described due to privacy and confidentiality concerns.

2.1 Background

Here are described the essential sensors, techniques and terms present in this thesis, so that the reader can better understand the following sections.

Data acquisition

The following subsections briefly summarize all the sensors commonly used in the industry that can potentially offer insightful sensory information about the car or the surrounding environment.

Accelerometer: sensor that measures the acceleration in 3 axis. It can be useful to measure changes in velocity (acceleration is the first time derivative of the

velocity) and changes in position by integrating the signal. However, integrating the signal adds an ever increasing amount of noise to the final result, making this device only useful for short term positioning correction.

Magnetometer: measures the force of the Earth magnetic field. It can be used as a compass in the north, east, south and west axis.

Gyroscope: measures changes in the 3-axis orientation vector (yaw, pitch and roll) of the mobile phone. In conjunction with the magnetometer it can use the Earth axis instead of the phone axis.

Global Navigation Satellite System: it gives the location of an object on the Earth surface, including longitude, latitude, altitude and velocity. Some of our devices support the American system (Global Positioning System (GPS)) and the Russian (Global Satellite Navigation System (GSNS)).

Controller Area Network-bus: the Controller Area Network (CAN) is a standard protocol used in vehicles that allows messages to be transmitted between multiple micro-controllers without the need of a computer. It is possible to intercept these messages and retrieve the state of several car sensors such as: speed, steering wheel angle, air conditioning on/off, engine temperature and other sensors that the car is equipped with.

Driving profiling approaches

The research community has taken two different approaches to our problem. Initial attempts focused on carefully collecting a small sample of isolated events that were then compared against batches of time series containing the sensory information of a person's driving using a similarity measurement. When the similarity measurement would reach a pre-defined threshold, usually found empirically, the algorithm would classify the time series being analyze as containing an aggressive event. We denominate this class of classification as distance based classification. Recently, there has been an increasing number of use for algorithms that adapt their decision boundaries according to the data provided. This algorithms require more data than the previous approach but tend to yield more accurate models. Our work focus on the latter. For a more in depth analysis of each algorithm please refer to [11].

Distance based classification

We call algorithms that use a similarity metric to measure how correlated two time series are distance based classifiers. A brief explanation of the techniques considered is presented in this subsection.

Dynamic Time Warping

Dynamic Time Warping (DTW) is an algorithm to measure the similarity of two time series. It warps them non-linearly and tries to find an optimal match between them in order to determine their similarity. As expected, due to non-linear transformations involved, it does not guarantee the triangle inequality [25], meaning that the sum of the differences sample wise is not guaranteed to be the final result. A low value means a high correlation, because the distance between the two signals being compared is small. This is what algorithms that compare reference patterns to driving episodes are looking for.

K-Nearest Neighbor

K-Nearest Neighbor (K-NN) is a method that receives two parameters, the metric used to compute the distance between two samples and the number of neighbors to consider. This classifier simply outputs the majority result of the K-Nearest Neighbor that surround the input variables.

General machine learning terms

Here we give a brief definition of important terms that will be used in the next chapters that the reader should get familiar with before proceeding. These are the common terms involved in a machine learning project.

Supervised learning

Supervised learning is a class of algorithms that use labelled data to infer a function that maps input variables (features) to output variables.

Unsupervised learning

In unsupervised learning, the task of the algorithm is to infer the structure of the unlabelled data and classify each sample. There are many types of unsupervised learning but for this thesis, we are only interested in clustering methods.

Data collection

This is the first step in any machine learning project. It involves the compilation and collection of enough data to have a statistically valid model. The amount of

data collected is typically a result of many factors, such as processing power (if we do not possess computational power then it might not be worth to have huge amounts of data), financial constraints or data availability. In supervised learning, for an algorithm to be able to translate the results obtained during the learning phase to the inference phase, it needs to be trained on data that is similar to the real world scenario, hence, different amounts of data are necessary, depending on the complexity of the task.

Pre-processing
The second step is to pre-process all the compiled data from the previous step into a standard format that is easy to handle for the researcher. In this step, the data can be normalized and new features can be added.

Training, validation and test sets
The data compiled before is split in training, validation and test sets. A common approach is to split it in 70/15/15, meaning that 70% of our data goes into training, 15% into validating our model and fitting hyper parameters and the rest into testing. Note that the values given here are not applicable to all scenarios, check [26] for more information.

Feature selection
Feature selection is the process of using techniques to eliminate or compose the input data so that there is no redundant information being given to the classifier. This step helps produce simpler models, shorter training times and reduce overfitting, thus, helping with generalization.

Overfitting
When an algorithm suffers from overfitting, it means that the algorithm is creating decision boundaries that fit the training set too much and is not capable of inferring on new situations given to it by the validation set.

Machine learning algorithms

The next section exposes the techniques and algorithms that are capable of adjusting their internal parameters using data. This type of approaches differ from distance based classification because this methods are data-driven.

Fuzzy logic systems
Fuzzy logic addresses uncertainty in a different way from probability theory. It

deals with subjective probabilities, using the concept of fuzzy set membership theory to answer how much a variable belong to a given set (i.e., how much is the property of being round in a given object? [not round, a little, a lot]). It is capable of creating simple hard science with IF-THEN rules. More complex rules can also be achieved by adding disjunctive or conjunctive ("and" or "or") elements to the rules.

Decision Trees
The algorithm creates a decision tree based on the information gain that each feature provide according to the entropy. To build the tree the algorithm computes the normalized information gain ratio that would be obtained by splitting it, for each attribute. The feature with the highest information gain defines a new decision node. With the use of recursion, the algorithm can be applied as many times as possible until all the attributes are placed in nodes. An open source implementation of the C4.5 decision tree algorithm can be found in Weka [37] under the name J48.

Random Forest
Random forests are an ensemble technique that use multiple decision trees built randomly to classify a given task. The degree of randomization is introduced in feature selection and in the number of samples used for training. Ensembles of decision trees tend to provide a higher degree of generalization than a single decision tree [18].

Bagging classifier
A bagging classifier is an ensemble of classifiers where each estimator is trained on random subsets of the original dataset and their prediction comes from the outcome of an internal vote.

Gradient boosting classifier
Boosting is a technique used in machine learning to create an ensemble of weak classifiers. The construction of a boosting classifier is done iteratively. At each iteration the weight of the samples is updated according to misclassifications and correct predictions. This way, new weak classifiers can be trained on samples that are getting the worst results, therefore improving the accuracy of the overall estimator over time. The type of weak classifiers is assumed to be a decision tree.

Hidden Markov model
Hidden Markov models have hidden states, events or actions that can not be observed directly, and observations. Each state represents a measurement or an

action. To go from one state to another, the process is not deterministic but rather probabilistic. A typical representation of HMM is shown below in Fig. 2.1, where the X's represent a random variable at time t and the Y's represent the observed measurements/conditions at time t.

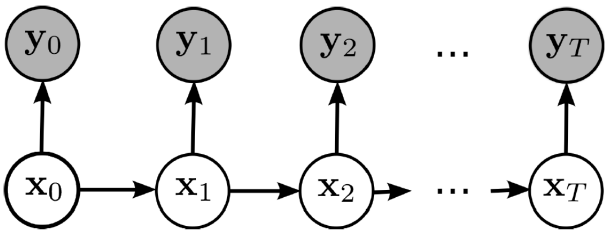

Figure 2.1: Typical Hidden Markov model representation.
Source: http://iacs-courses.seas.harvard.edu/courses/am207/blog/hmm.png

Bayes Classifier
Using Baye's Theorem and the conditional probability it is possible to compute the posterior probability of an event happening given the prior probability, the likelihood and some evidence. For the equation below, $x = (x_1, ..., x_n)$, represents the n number of features. C_k represents class number k.

$$p(C_k|X) = \frac{p(C_k) * p(X|C_k)}{p(X)}$$

Support Vector Machine
Support Vector Machine (SVM) try to find the hyperplane vector that maximizes the distance between the set of points that is closer to the hyperplane. This only gives a linear SVM of hard margin. Because not all data sets are linearly separable we can introduce a loss function with a hyper-parameter that can be controlled. Moreover, to obtain a non-linear classifier, a kernel trick can be applied, mapping the inputs to a higher dimension, where they are linearly separable and mapping them back to the original dimension, returning the final class label.

2.2 Distance based classification

For this section, we provide a summary of all the articles found to be insightful for our work in order to provide some context of what is happening right in the field. In the end of each summary there is a brief critical analysis where we discuss some topics about the article, whether it flaws in the study or important insights.

According to a study published by *Banerjee et al.*, **"How's My Driving? A Spatio-Semantic Analysis of Driving Behavior with Smartphone Sensors"**, [1], driving is a skill heavily influenced by the ambient context. The authors distinguish between two types of contexts: a static and a dynamic one. They are described as follows:

1. Static Context: includes attributes that remain mostly unchanged throughout a substantial amount of time and over multiple trips.
2. Dynamic Context: includes parameters that change frequently, across the same path and several trips.

The author comes up with a definition for good and bad driving behavior so that later he can determine the influence of the ambient context on the driving behavior. To demonstrate the need to consider the surrounding context, an experiment was conducted using Dynamic Time Warping to compare good and bad templates against the driving episode. The results show that under different conditions, different speeding profiles exist.
This led the author to consider three groups of attributes: road network (straight, turns, roundabout, bends), road neighborhood (school, traffic signal, market place, no label) and road surface condition (smooth, bump, pothole). To get this rich environment, a couple of third party information providers were used, mainly open source projects.
All the data available was fused to bring rich road segments to the cross-trip majority voting algorithm. This voting system clusters the GPS data points into two clusters: smooth and bumpy road. For every point that is near each other and from different trips, they cast a vote on each other. This voting mechanism is able to find static points on road segments, reducing false positives and reducing user induced noise. In the end, density based clustering is employed on the resultant road segment found before discovering different driving profiles.
This article tried to prove that in chaotic road settings, like the ones in India, the ambient context assumes a highly relevant importance for supervised learning methodologies.
The precision achieved is close to random, [50-60%], but the recall values are much higher [80-90%], which means that this algorithm is an aggressive method where most of the abnormal behaviors are identified at the cost of false positives.

The car and phone model are mentioned, however, it lacks phone variety, driver variety, code and data publicly released. This study introduces a new feature that most other studies do not consider: ambient context. This feature was important

to determine the driving profile of a driver. It also introduces two branches (analysis of dynamic and static context) for further research, with only static context being studied here and the other left for a future work. The intermediate step of enriching the map with augment information is valuable but due to time constraints it will not be applied in this thesis.

In this article, **"A Comparison of Driving Behaviour Prediction Algorithm Using Multi-Sensory Data on a Smartphone"** [31], the authors point that the large majority of accidents on the highway come from vehicle condition, human error and road condition. In Thailand, 75% of highway accidents come from improper lane change. The proposal of the author is to provide warning signs that are low cost and can serve a large amount of the population, and for those reasons, smartphones seem to be the logical choice.
The first step is to collect sensory data from the user smartphone and reference pattern. To get the sensory data of the user two sampling frequencies are used, 5 Hz for the accelerometer, gyroscope and magnetic sensor and 1Hz for the GPS.
The second step is to reduce noise from raw data, caused by the vibration of the vehicle or the roughness of the highway. A simple moving average is employed to solve this problem with the added benefit of providing the trend direction of the data. The third step is to compute the standard deviation in the raw data and if the result is greater than a pre-defined threshold, it triggers the pattern matching algorithm. While the algorithm is not triggered, new data is collected. Next, Dynamic Time Warping is used on each feature to determine which reference pattern looks similar to the detected driving episode. The reference patterns are data collected from a previous experiment. To differentiate between multiple matches a table with threshold values and simple logic is pre-defined with values that best suited the data set. If in the end, there is still more than one match, the algorithm chooses the result according to the accuracy of each sensor.
The driving events being monitored are brake, sudden brake, acceleration, sudden acceleration, left turn, sudden left turn, right turn, sudden right turn, left lane change, sudden left lane change, lane change right and sudden change right.
The data for the study was collected in a car with four people. Three people were assessors and their job was to mark events during the ride. If two or more people agreed on an event, then the event was recorded. The fourth person was the driver. The total distance of the route was 71.3 Km, divided into urban, traffic jam and highway roads around the University of Thammasat.
The results vary from 81% accuracy, using only the accelerometer to 71% using all the sensors.

This paper leaves crucial information out. There is no mention of the phone model

or its position inside the car nor on how/when or even if it is necessary to reset the raw data collection window if no event is detected. It was assumed that the phone must be in a fixed arbitrary position. It is also mentioned that the algorithm does not need to process all the sensory data in order to be effective but no experimental validation is shown or the sentence proven. The reference driving patterns are said to be collected from an experiment but there is no reference pointing to it. The final results are unexpected. With an increase in the number of features, we predict the accuracy of the algorithm to also rise, specially if the features added are expected to contain relevant information, but this does not happen and the authors do not address this subject. Although the study has some flaws, it can help us understand how to achieve good results using only the most common sensors available, which is a big goal of this thesis.

The authors of this article, **"Driving Style Recognition Using a Smartphone as a Sensor Platform"** [14], demonstrate the same motivation as we do related to whether or not the driving behavior is safer when people are being monitored and feedback is provided about potentially aggressive moves. This concern is answered by the Journal of Organizational Behavior Management in [12].
To collect the data necessary, the latest generation smartphones were considered because of the multiple sensors embedded in them such as: multiple cameras, microphones, 3-axis accelerometer, 3-axis gyroscope, proximity, ambient light, touch, magnetometer and GPS available to them. During the data collection trips, the phone was mounted on the car to prevent any rotation that might influence the data.
The detection was divided into two categories, lateral (T) and longitudinal (L) movements. By convention the gyroscope has $G = \{g_x, g_y, g_z\}$ in rad/s, accelerometer $A = \{a_x, a_y, a_z\}$ in m/s^2, the device Euler angle $E = \{e_x, e_y, e_z\}$ in radians, $T = \{g_x, a_y, e_x\}$ and $L = \{g_y, a_z\}$. The types of events detected are right turns, left turns, U-turns, aggressive right turns, aggressive left turns, aggressive U-turns, aggressive acceleration, aggressive braking, aggressive swerve right, aggressive swerve left, device removal and excessive speed. Non aggressive lane changes(swerve) are not being detected because the force exerted on the device was not enough to distinguish it from noise.
An iPhone 4 was used because it contained all the sensors that the study targeted. GPS is used only to determined speed and event location. The application has two modes: active and passive.
In active mode, the system monitors the driving episodes but only records sensor and video when potentially-aggressive behaviors are detected. Before storing the event, the user needs to confirm that an event happened and classify it. A speech synthesizer is used to make alerts audible via software and warn users of their be-

haviour. A driver is considered aggressive if he/she exceeds an arbitrary number of aggressive events over a predetermined time window.
In passive mode, the system records all data for further analysis, segments it into 5 minutes windows and synchronizes the video with the sensors to allow for later testing.
One worry the authors had about using smartphones to collect information was that the shaking motion of the car could affect the quality of the data. To dismiss this matter, the authors compared the correlation between the same trips recorded used an iPhone and the CAN bus of the vehicle. They found no statistical difference between both time series, indicating that smartphones are, statistically, as good as the CAN bus to record data.

Data from accelerometer and gyroscope was sampled at 25 Hz. To detect the beginning of a event, a simple moving average of the rotational energy in the x-axis for a window of size k for the current sample i was used as follows:

$$SMA = \frac{g_x(i)^2 + g_x(i-1)^2 + ... + g_x(i-k-1)^2}{k}$$

If the simple moving average is greater than a threshold t_u then the window considered has an event observed inside. Events detected are then compared to reference patterns using DTW. From the moment an event is detected, the algorithms keeps concatenating new frames until the moving average is less than a threshold t_l and just to prevent severe failure, if the length of a window exceeds 15 seconds, the event is discarded.G_x is used because prior experimentation showed that rotation is easier to distinguish than accelerometer on all the recorded events.
This system was used in three different vehicles, with three different drivers, resulting in 200 events in urban, rural and highway roads.
In the end, the accuracy of A, G and T feature sets were computed, achieving 77, 79 and 91%, respectively. This shows that a combination of the x-axis rotation rate, the y-axis accelerometer and pitch are the signals that best suit this algorithm.

The baseline defined in the study established that a smartphone is an approach that should be considered as it can replace on board expensive sensors.This is a study that employed sensor fusion and feature selection to reach very good results in the end. Although the article only mentions one phone model being used, the approach employed seems general enough to be used in multiple models, but further analysis is required.
General clues for feature selection are retrieved from this study for analysis in the 2nd half of this thesis.
When the authors describe what type of events are being monitored, they mention

"device removal" but they never explain what they mean by it, how to detect it or provide any other references for the reader.

2.3 Machine Learning algorithms

This section follows the same structure as the previous one. For each article that was insightful for our work we present a summary and a critical analysis, always in this order.

In the work developed in **"Driver Behavior Profiling Using Smartphones: A Low-Cost Platform for Driver Monitoring"** [2], we see that profiling the driving behavior of the general public in a cheap way has received an increase relevance for a variety of application domains such as fleet management and car insurance. To do so, detected events are combined with environmental factors to score a certain amount of points obtained through a scoring function.
The sensors used are GPS, accelerometer, magnetometer and gravity sensor. The internal linear accelerometer is used to compute the jerk, which is the rate of change of the accelerometer with respect to time. Kalman filters were used in an attempt to distinguish between longitudinal and lateral movement. However it was not possible to distinguish longitudinal nor lateral movements of the car from raw accelerometer data, so the only accelerometer feature considered was the magnitude. This limitation makes the accelerometer axis indistinguishable but allows the rotation and manipulation of the device without any constraint.
The orientation vector includes yaw, pitch and roll and serves to describe the rotation of the vehicle around the Earth axis. The yaw rate is the only measure considered as it gives the steering of the vehicle on the earth surface. However this measurements come with the high cost of having electromagnetic interference and device vibration. To overcome this problem, the raw data of the motion sensors are fused with GPS data to improve accuracy and reduce noise.
In order to deal with different sampling sizes the window slices considered have a fixed duration of 1 s. The final features are the speed variation (accelerometer), the bearing (angle between the magnetic North and the vehicle) variation, the average yaw rate and the jerk standard deviation. The jerk standard deviation is considered to mitigate the effects of a phone vibrating.
To detect the events, a fuzzy system was built. The fuzzification phase contains the feature described above and the rules are obtained manually by analyzing the different input variables in a controlled scenario.
To make this process independent of the device and vehicle, a calibration phase is necessary to adjust the fuzzy membership functions of the jerk and yaw rate be-

cause speed variation and bearing rate can be fixed regardless of the smartphone or the vehicle (combustion or electric engine). The values set for the adaptive features are obtained by getting the last percentile of the cumulative distribution function of the samples.
The environmental variables considered are weather information, speed limit and time of day in relation to sunrise and sunset.
The system scores each trip with 100 points at the start and removes an arbitrary number of points when it detects an aggressive behavior. The number of points deducted is based on the number of accidents that happen due to environmental factors [24] and the type of aggressive behavior detected. The deductions values are 2, 4, 6 and 8, according to the environment (low, medium, high and extreme probability of accident). After an arbitrary 0.5 Km without any event, the score increases by one point.
The experimental phase started by determining what is the effect of calibrating the fuzzy sets using different time periods. It was verified that after 17 minutes of calibration, the number of false positives was below 10% but the number of false negatives was still relatively high, at 20%. However as time went by, all the metrics mentioned before decreased to 1% and 10%, respectively, after 30 minutes. All of this caused the raise of the true positive ratio from around 75% to 90%.
A full factorial experiment was conducted, this means that all the possible combinations of scenarios were recorded while varying all the variables (weather, speed limit, time of day and aggressive behavior type). Twenty five minutes was the calibration time frame chosen, which is equivalent to n=1500 samples. One lap in the chosen path was done to calibrate and two to record events, the first one being calm and the second one being aggressive.
With a sample size of 10, it is observable that the final score always managed to distinguish by at least 17 points, in a scale of 0-100. By overlaying the location and quantity of aggressive behaviors on a map, it can be observed that there are some areas that are more prone to aggressive behaviors, indicated by red spots on the heat map.
In the end, the participants were asked to rate subjectively their score from 1 to 5, with 1 being high risk and 5 being the safest behavior. When the results are clustered across 5 centers using K-means it is easy to see that as the score increases so does the safety rank. There was a 90% match between the predicted subjective score (1-5 range) and the clustered score if the match distance considered was ± 1.
For the experiment two phones were tested and it was noticeable that, although the performance and sampling rate of both phones is very different, the number of events detected is very similar, after calibration.
It is recognized by the authors that an obvious problem with this approach is the use of a non representative calibration phase, which distorts the fuzzy mem-

bership functions. The solutions suggested involve using a dynamic calibration process that stops after a given condition or to continually calibrate.

New domains for the use of driving behavior profiling are suggested and this study focus on those domains by quantifying the risk that each driver takes in its everyday life by analyzing aggressive maneuvers while taking into consideration the environmental factors. Obvious limitations of the approach are considered and possible solutions, are suggested. The experiment analyses all the variables considered and reaches compelling evidence despite the small sample size, although it is much bigger than most studies in the field, which prevents the author to have a statistically strong conclusion.
For the final application that will integrate the company product, a scoring function may be necessary. Although the domain application is different, the scoring function serves the same purpose.
The reflection of this article closely matches the intention behind this thesis and as such it has a greater relevance than other studies. In here the authors also describe what techniques were tried before reaching a successful implementation. This is important because it allow us not to repeat the same path in our line of research and possibly achieve better results in a shorter amount of time.

In this article, **"Estimating Driving Behavior by a Smartphone"** [8], a system using Bayes classification is proposed to determine a driver behavior. The algorithm records data continuously from the accelerometer, gyroscope and magnetometer. The data gathered is sliced into smaller windows of size m, where the energy of the signals is computed for each sample.
If the energy E of one window is higher than an empirically determined fixed threshold, then the window is discarded. Signals that are discarded in this step proceed to the next.
The signals from the various sensors are forwarded to a DTW module in order to calculate the best matching template, chosen according to the test data manually. For this study, the accelerometer provides position and speed, while the gyroscope measures lane departure and turning events. Data from accelerometer comes in range [-1, 1] while the data from the gyroscope ranges between [-180°, 180°]. A high pass filter is then applied to it in order to highlight sudden variations:

$$R_{x,y,z} = accel.(x, y, z) * filtsbt + R_{x,y,z} * (g - filtsbt) * (accel.(x, y, z))$$

where accel.(x,z,y) is the accelerometer data, g is the gravitational accelerometer, which is constantly g=1 and filtsbt, which is the frequency rate of the gravity.
The classifier adopts Bayesian inference with two classes, safe and unsafe driving for each template output by the DTW algorithm. The final output is based on the

maximum posteriori estimate across all the events considered, based on the fact that our previous Bayes classification gave us probabilities based on the steering wheel angle, accelerometer, slowdown and lane change. The classification is done on the signal that results from a moving average filter.
To test the experiment, 15 drivers were recruited, 5 of which were experienced drivers, 5 novice and 5 others randomly chosen. Each driver drove two times in order to experience different weather and road conditions. All the experiments have lane change, instant accelerometer and braking, left and right turns and suspicious behaviors.
For the task of binary classification (safe or unsafe trip), the proposed algorithm achieved 93.3% accuracy on the test set and took 3.6 seconds to classify the entire trip. This result is compared against others present in earlier work. The algorithms considered were: Random Forest [23], J48 [23] and HMM [27]. Note that HMM use features directly extracted from the car.
Random forests got 93% correctly classified instances in 24.4 seconds, J48 obtained 90.6% accuracy in 78.8 seconds while HMM got 85.7% of the instances in an unknown time. The time reported in the this article refers to the time each algorithm took to classify the entire trip.

As the route taken is only shown in a map, people unfamiliar with the city where the tests were conducted don't know the road conditions, traffic or other environmental constraints. There is also no mention about what car was driven. The author says that the results are applicable to more brands of phones but the use of different sensors was not a subject of this study, which leaves the claim unproven.

In this work, **"Leveraging Sensor Information from Portable Devices towards Automatic Driving Maneuver Recognition"** [32], the authors state that despite new laws prohibiting the use of mobile devices while driving, new applications and uses for such devices have been rampant, making them a big cause car accidents. Due to the difficulty and challenges in accessing the CAN bus of a car, the use of smartphones for this domain has been in increasing demand. To help passengers and drivers become more secure, the goal of this study is to detect dangerous maneuvers for drivers. This test attempts to verify if the results reported by a cheap smartphone are the same as other expensive equipment.
This experiment compares the accuracy of data retrieved from an instrumented vehicle to that of a portable device. The instrumented car has multiple cameras, a microphone array, a second microphone, GPS, optical distance sensor, gas/ brake pedal pressure sensor, CAN bus OBD II and a data acquisition unit to synchronously record data. The portable device has front and back camera, microphones, GPS, 3-axis accelerometer, 3-axis gyroscope, digital compass (mag-

netometer), ambient light and proximity sensor.
The route selected for test drives is driven in both directions and is big enough to contain all the events tested: right turn, left turn, right lane change, left lane change, right road curve, left road curve, straight and stop.
The device is mounted on the windshield, placing it in a position that aligns the vehicle axis with the phone/tablet axis. To label events, video recordings are being taken and synchronized with sensor information.
From the CAN bus signals it was possible to extract and decrypt vehicle speed, steering wheel angle, engine RPM and gas/brake pedal pressure.
For the portable device, a series of features can be extracted using raw sensor data and fusion information, which gives 8 sets of features: 3-axis accelerometer, 3-axis gyroscope, GPS, 3-axis magnetometer, 3-axis orientation, 3-axis gravity, 3-axis linear accelerometer (similar to accelerometer data but without the gravity component) and 3-axis rotation vector, which measures the orientation of the device relative to a fixed orientation. All these sensors and derived information are captured at 50 Hz and down sampled to 1 Hz to reduce noise.
Combining the 5 signals from the CAN bus with the sensory information of the mobile phone we have a total of 28 signals, each having 16 features: difference between the maximum and the mean, difference between the mean and the minimum, the median, the mean, the minimum, the max, the difference between the maximum and the minimum, standard deviation, variance, root mean square, amplitude of the difference between the first and last sample, variance of error in a 10^h order linear prediction analysis, entropy, direct current value and energy. This gives a total of 700 features, of which 125 are from the CAN bus and 575 from the portable device.
To test if the loss in accuracy from portable devices (when compared to the CAN bus) had any impact on the output two algorithms were used. The first one is the k-nearest neighbour and the second one are support vector machines, in this case, using a Gaussian radial basis kernel in a one-versus-all strategy.
For the experiment, all maneuvers were classified using either SVM or k-NN and with the use of linear discriminant analysis (LDA) or sequential feature selection (SFS) or none. The best results for the CAN bus and the portable device are 74% and 89% accuracy, respectively. The best configuration for information extracted from the CAN bus is a combination of LDA plus k-NN while the best methodology for smartphones is only to use a SVM.

This study reports results that indicate that approaches using smartphones can be at least as good as directly retrieving information from the car. Two supervised algorithms (K-NN and SVM) are compared against each other while also employing feature selection and feature reduction, something that few articles mention

or use. This and other articles using SVM indicate that RBF SVMs should be used initially in the preliminary work. The results suggest that using an external device (smartphones) to collect data is better than using the internal CAN bus of the car. Despite this interesting information, there is no analysis of this result.

In this paper **"D3: Abnormal Driving Behaviors Detection and Identification Using Smartphone Sensors"** [6], the sensors used were a 3-axis accelerometer and 3-axis orientation sensor. The vehicle axis and the phone axis were aligned manually. Twenty drivers were recorded for four months in their daily driving activity with five different phones, placed arbitrarily in the vehicle or in a fixed position with the phone axis aligned with the vehicle axis. This allowed to experiment on the axis correction procedure, which revealed a decrease in accuracy by 2%.
The author uses a Radial Basis Function (RBF) SVM with 16 features on 6 months of driving data only to classify the behavior of drivers as abnormal or normal. Then the approach was refined to classify 6 maneuvers that are considered dangerous: weaving, swerving, sideslipping, fast U-turn, turning with wide radius and sudden braking. See Fig.2.2 for an illustration of the maneuvers. Although each feature is not separable from all the others, every pair of events has separable features.
The average accuracy was 95.36% for the fine-grained system and 99.41% for the 2 class classification task.

Although an axis correction method is mentioned, there is no explanation on how to apply it nor any reference to an external article. The sample size and time recorded provide a huge data set with multiple phone models, cars and roads and thus introduces more variety on the input features than most studies and so, it has proven to be generalizable. This study emphasizes the use of SVMs as a valid verification algorithm for the preliminary work.

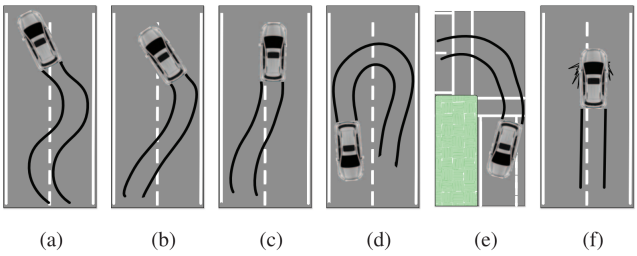

Figure 2.2: Six types of abnormal driving behaviors: (a) Weaving, (b) Swerving, (c) Sideslipping, (d) Fast U-turn, (e) Turning with a wide radius, (f) Sudden braking.
Source: D3: Abnormal Driving Behaviors Detection and Identification Using Smartphone Sensors [6]

2.4 Summary of aforementioned work

In this section we will take a look at the results mentioned previously in Section 2.2 and Section 2.3. The summary ends with Table 2.1. Afterwards, there is a critical analysis that applies to all the work revised here. We conclude the chapter by expressing known limitations brought to our work by the flaws presented earlier. The accuracy of the studied methods ranges from 71% to 95.36% with one article not available for direct comparison (it uses other performance metrics) [1]. The papers that achieve high accuracy (90%+) consistently all use machine learning. From this analysis it appears that approaches based on Machine Learning yield the best results and supports the use of Artificial Intelligence in our work.
SVMs have performance indicators close to or above 90% and because of the low variance (6%) between multiple studies, it was the algorithm of choice for the preliminary work.
As for the devices used, five out of eleven studies do not reveal what kind of phone was used (Android or iOS). Four out of eleven use Android and from these, one specifies the API level while the others do not. The remaining two are iOS devices without any API level described. The device usage can be seen to approximately mirror the market share [36].
Four out of eleven rows mention the use of only one feature set that was achieved either through logical reasoning about real world scenarios or simply an agglomerate of the most used sensors (GPS + magnetometer + gyroscope). The remaining seven rows in the table test more than one feature set in a search for the most descriptive features.
As for the number of unique devices and cars reviewed, it is visible in the table that seven out of eleven use a single device for their studies, two out of eleven are not available while the rest collected data uses multiple devices. Four out of eleven

drove in one car and tests were conducted in a defined path; seven drove multiple cars. It can be seen that having a variety of devices and drivers/cars might lead to an accuracy increase for predicting the user driving behaviour, as all studies with multiple phones/cars have 89%+ accuracy. Notice that the last row in the table is the only experiment that uses a multitude of drivers and cars and is the article that has achieved the greatest accuracy, beating the closest algorithm by 2.36%, with 95.36% accuracy.

Table 2.1: Summary of all mobile-only approaches considered.

N events	**Algorithm**	**OS version**	**Sensor**	**N unique devices/ cars**	**Ref.**	**Metric**
2	Cross Trip Voting + DBSCAN	Android API- NA	GPS + accel.	1/1	[1]	55% prec. 85% recall
12	DTW	NA	accel.	NA/3	[14]	77% acc.
12	DTW	NA	$gravity_x$ + $accel._y$ + $Euler_x$	NA/3	[14]	91% acc.
12	DTW	NA	accel.	1/1	[31]	81% acc.
12	DTW	NA	accel. + GPS + gyroscope	1/1	[31]	71% acc.
12	DTW	NA	accel. + magnetometer + GPS + gyroscope	1/1	[31]	71% acc.
4	Fuzzy Inference System	Android Api- NA	accel. + magnetometer + GPS	1+/1+	[2]	90% acc.
2	Random Forest	iOS Api- NA	light + gyroscope + accel. + GPS	1/15	[8]	93% acc.

2	J48	iOS Api- NA	light + gyroscope + accel. + GPS	1/15	[8]	90.6% acc.
8	RBF SVM	Android Api- 11-15	accel. + gyroscope + GPS + magne-tometer	1/100+	[32]	89% acc.
6	RBF SVM	Android Api NA-	accel. + gyroscope	5/20	[6]	95.36% acc.

*NA- Not available; N- number; prec. - precision; accel. - accelerometer; acc.- accuracy

Artificial intelligence is going through a phase of open access articles and code, as it is wildly recognized that half the work nowadays is spent collecting and pre-processing useful data [10]. As such, it would be extremely useful to know the amount of data each study has collected. However, what is observed is that the majority of the papers do not present a description regarding the raw data collected, much less the ratios used for the train/test/validation sets. This is a worrying trend as this data is fundamental to reproduce all studies. If the data is not public, at least the magnitude of data used should be specified.
The lack of a standard benchmark dataset for this field of research makes the results presented above slightly less relevant and not directly comparable between each other or to this thesis. This will cause a limitation on the evaluation of our work because we will not have a direct measure against other state of the art algorithms, only against the baseline system, defined in Section1.2. Despite the limitations, we plan to contribute to this field by improving the detection and prediction over the current state of the art.

Chapter 3

Research goals and methodology

The following chapter defines the research goals. We are aware that more research goals exist and those are identified later in this dissertation for future work. Next, we describe our methodology to achieve the research goals. This includes a description of how we collected new data, how it was pre-processed into a tabular format and all the machine learning steps that came afterwards.

3.1 Research goals

The major goal is to get the best model possible that has a similar performance on a multitude of situations, including, but not limited to, multiple drivers, cars, devices and road surfaces. One of the conclusions to take from analyzing the state of the art in Chapter 2 is that a Machine Learning model should be used. A Machine Learning approach can have several steps such as: pre-processing, feature engineering, feature selection, feature scaling, data balance and the choice of the classifier. We did not find any research on a few topics such as feature scaling or how to balance the data.
From the related work present in Chapter 2 it is visible that multiple factors affect the performance of the algorithm. As important factors we have identified the number of unique devices and the number of cars with unique characteristics. For this thesis, the following was investigated:

1. First, we tried to produce a model that could take data from iOS and Android devices during the training process and produce similar results during the test/validation phases.

    - A sub-goal was to know the differences between models that were trained

with data collected by multiple devices and models trained with data collected by a single device. For this comparison, it was important to distinguish between different operating systems (iOS and Android) to determine if the operating system and underlying hardware specifications had any impact on the performance of the models.

- Another sub-goal is to try to remove the features provided by the gyroscope with minimal impact on the accuracy of the system as the majority of the Android phones still do not have a gyroscope. To test this sub-goal a smartphone without gyroscope should be used instead of just removing the features provided by it. This is because the intermediate calculations done by Android to provide some of the features used are different based on the sensors available.

From this goal we are going to determine if there is a device or devices whose importance is crucial for capturing relevant information during future data gathering sessions.

2. Secondly, we studied the impact of having smartphones placed in different positions. We recorded data while having them placed on the floor, on our hands, vertically and horizontally. We attempted to study if one model could recognize driving events correctly without aligning the smartphone axis with the car axis.
   Achieving this goal would mean that our model could be delivered to smartphones without a gyroscope, which is something desirable for the Android market.

Due to lack of data, it was not possible to study nor produce models that take into consideration multiple drivers and multiple cars characteristics.

3.2 Methodology

Before diving into a detailed explanation of what we did let us take an overview of what happened. For the first half of our work, the data used for in this dissertation was provided by Sentilant. It contained data recorded by two smartphones in a fixed position. As the labelling process was producing incorrect labels, we tried to fix it by using unsupervised learning. Although this method seemed to work in that particular dataset, it later proved to not generalize to other datasets. A major milestone for the first half of our work was to determine what techniques should be applied to pre-process the dataset, how to balance it, what classifiers

looked promising and start feature engineering.

The second half of this thesis was mostly focused on feature engineering, collecting new data on multiple devices with smartphones in different positions, validating the approach in real scenarios using a simulator and developing a simple post-processing mechanism to improve the algorithm classification performance under continuous usage.

We will now be detailing our methodology, starting by describing the way used to acquire new data followed by how we managed to balance the dataset, how we produced new features and what classifier we ended up testing by describing the different approaches that could have been taken at each step including their pros, cons and our final choice. Each one of the following subsections is a crucial step necessary to understand the solution developed.

3.2.1 Events monitored and data collection

For the number of events monitored, the literature provided several choices. The granularity of events detected should depend on the final application that the model will integrate. D. Banerjee and Eren *et al.* in their work, "How's My Driving? A Spatio-Semantic Analysis of Driving Behavior with Smartphone Sensors" [1] and "Estimating Driving Behavior by a Smartphone" [8] respectively, considered a binary approach to the problem. This approach has demonstrated results that achieve high accuracy (above 90%) even when using a single phone to acquire data. One study [6] reports 99.41% accuracy when using multiple devices and drivers for binary classification vs 96.36% for classifying six types of events.

The remaining approaches that use Machine Learning support a number of different maneuvers ranging from four to eight unique events. There is limited research but studies suggest that for the same setup, having fewer classes to predict leads to an increase in accuracy. Hence there is a trade-off to chose between the number of events and the accuracy of the model. Sentilant specified the number of events to be detected in the first half of the work as four: normal behaviour, sudden acceleration, sudden braking and sudden turns (no distinction between left and right).

The majority of papers use a smartphone in a fixed position to acquire their data. No one ever mentions how they control and account the human factor during their recordings. It would have been helpful to know how they avoided the problem of the temporal disparity between the labels and the actual event. This approach

has clear downsides as it requires precise human intervention, something that can not be guaranteed to always work.
The other method involved the use of cameras, the CAN bus, placeholders for smartphones and specially designed cars to hold all this equipment. In this approach only the driver is required to be in the car as the labelling process is done afterwards by driving experts.
Both approaches have upsides and downside: on the one hand we have a method to gather data that is inexpensive, and requires almost no setup, but might get inferior labelling results. On the other hand we have a more expensive and harder to setup method, that provides more accurate labels. We discussed with Sentilant both approaches, and decided to go with the former, i.e., using a more inexpensive but easy to setup method for data acquisition.

The data used in the first half of our work was collected by Sentilant [34], with the phone in a horizontal position, screen always facing up and events labelled on the spot with the help of an event recording application. Once the application receives a command to start recording, it starts to collect information from the sensors available at the fastest frequency possible to that phone and writes that data to a text file. In the end we get a large text file where each row is the information of one sensor.
On the car used to acquire raw data from the smartphones were at least two people: one driver and the assistants. The task of the driver was to make dangerous moves on purpose and tell the assistant which move he was about to make before executing it. The task of the assistant was to hold one or several devices in a position agreed before the beginning of the trip. One of the assistants was in charge of indicating to the application that an event was happening. In the application there are several buttons, one for each aggressive event studied, with two modes, on or off. After the driver tells that he is about to make an aggressive maneuver, the assistant in charge of labelling the events would select the appropriate button, corresponding to the event that is about to take place, on the recording application. The beginning and the end of an event are defined by the labelling assistant. There would occasionally be mistakes such as pressing the wrong button, pressing a button when no event was happening or other human errors. Additionally, due to the high frequency of readings by the sensors, we can see in the data a disparity between the pressing of the button and the actual event happening. In the first half of our work we tried to correct this temporal disparity with unsupervised learning but latter, learned that the results were much worse than manual labelling.

The data contained all kinds of roads: urban, rural, highways and variants such as traffic congestion and speed bumps. For a given timestamp, if no event is labelled,

we assume that the driver is expressing a driving pattern that is considered "normal". The threshold to what is considered a sudden event is controlled implicitly in the labelling process.

The dataset that we were working with is composed by 22 features:

- The acceleration exerted on the phone in the x, y and z axis;
- The force of gravity in the x, y and z axis of the smartphone;
- The rotation of the device around the x, y and z axis of the smartphone;
- The speed, course, altitude, longitude and heading of the vehicle, as reported by the GSNS;
- The yaw, pitch and roll of the device;
- The timestamps in which each measurement was recorded;
- The acceleration being exerted on the phone without the gravity component in the x, y and z axis;

For the second half of our work we considered the addition of more events to match the state of the art, but new trips revealed problems with our approach and we decided to stick with the previous three maneuvers. During this time new data was collected in April and June using the same method as before. We did not change the way any feature was calculated with the exception of the timestamp, which was determined by the GPS, with a resolution of one second, during the first half of our work but for the second half it became determined by the accelerometer, which has a much higher resolution. Starting with the second dataset, we developed features that were using timestamps for their calculations. This rendered the first dataset acquired in the first unfit for use. The data for the first few trips of the second half of this dissertation were acquired in April and real world validation was done in June.

Table 3.1: Summary of the datasets used for this dissertation

Dataset Name	Number of phones	Phone positions	Total amount of data	Total amount of data converted to time units
Sentilant	1 Android 1 iOS	Horizontal	Normal: 192736 Accel: 6824 Brake: 166 Turn: 386	Normal: 3h 18min Accel: 7min 6s Brake: 10s Turn: 24s
April	3 Android 2 iOS	Hand, Horizontal, Vertical	Normal: 396874 Accel: 426 Brake: 223 Turn: 127	Normal: 6h 53min Accel: 26s Brake: 14s Turn: 8s
June	1 Android 1 iOS	Vertical (iOS) Horizontal (Android)	Normal: 44705 Accel: 4775 Brake: 2197 Turn: 1306	Normal: 46min Accel: 4min 40s Brake: 2min 15s Turn: 1min 18s

3.2.2 Pre-processing

Although the majority of the papers reviewed used a method similar to ours when collecting their data, no one mentions how they deal with the errors caused by the human assistant when labelling. For the first half of the work, we tried a novel way to approach this problem.

In this step we were looking to automatically correct the temporal disparity introduced on the labels during our data collection trips. For the duration of each event, we are guaranteed to have only two labels, normal and aggressive behaviour. The normal behaviour is also guaranteed to be in the beginning and at the end of each consecutive event labelled.
This two assumptions give us the number of clusters to find within each event, which is two, one center for normal behaviour and another for aggressive. For this task we employed k-means, with $k = 2$ and run the algorithm independently on each aggressive maneuver. K-means was the clustering algorithm of choice because the number of centroids was known. The centroid of each cluster was in the end defined as the average of each centroid for all the aggressive driving maneuvers labelled. This technique was correctly modifying some segments in the beginning and at the end of some labelled maneuvers for the dataset used in the first half of our work. This hinted that the process may be working well but without ground truth labels to compared it was hard to be sure.

Although this technique seemed to work for the first dataset, it was failing to cor-

rectly work for the remaining. The technique proved to not generalized to all our datasets. With the trips from April we realized that a delay caused by the GSNS signal was making our labelling process more erratic than the human component. The delay of this signal was enough to make an aggressive behaviour displaced by dozens of samples, causing the model to not learn what it was suppose to. An unsupervised approach could not work without major modifications. Our labels were mere clues about when an event took place.

Since the moment this problem was understood, the temporal disparity introduced on the labels by the way we collect data was corrected manually by one person. By only having one person correct the labels of the datasets, it introduced a clear bias to what constitute an aggressive behaviour as noticed by Sentilant when testing the algorithm in real scenarios.
In the future, multiple people should be in charge of the labelling process to avoid such bias. We would recommend an odd number of people and the use of a majority voting mechanism to decide which label should a sample get.

3.2.3 Balancing the dataset

After we have our data on a tabular format and correctly labelled to the best of our ability, we tackled the problem of balancing the dataset. As no one in the literature reviewed mentions how they balanced their dataset, there is no starting point for this problem, on this specific domain.
A first look at the data reveals that the dataset is imbalanced, as seen in Fig 3.1. This issue needs to be addressed or it will cause the classifier to favor one type of events over the others, reducing the generalization of the algorithm and performing worse on data that it has never seen (test and validation sets). The outcome of evaluation metrics that do not take this imbalance into consideration produce results that are skewed towards the class that has greater representation.

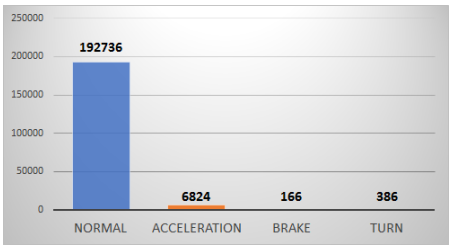

Figure 3.1: Support for each class of the dataset provided by Sentilant.

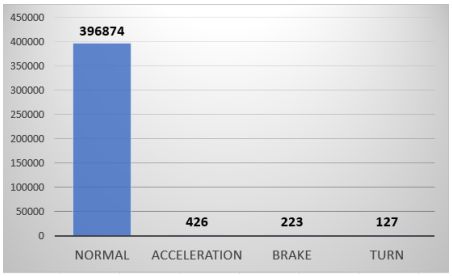

Figure 3.2: Support for each class of the dataset acquired in April.

Generally we can have three approaches to this problem. We can either use oversampling, downsampling or a mixture of both using a variety of techniques.
If one would apply oversampling techniques on our datasets the computational cost of doing that process would be immense because the process of generating or sampling points from the least represented classes would need to be repeated until they reach the same support as the most represent class. Because our dataset has a wide disparity between each class support this technique is costly. The consequent cost of training a model with such a huge amount of points would make the training time of the model unbearable. Nevertheless, the first experiment had random oversampling as one of parameters subjected to study.

Undersampling also has the ability to balance our dataset but applying downsampling when our least represented class only has 127 points (Figure 3.2) would exclude too many instances from our training data. The obvious advantage of reduced training time comes at the cost of inferior performance from the model. Despite the observed pitfalls we also included random downsampling on our first experiment.

To overcome some of the problems identified we devised a new technique that attempted to undersample the most represented class while keeping all of the instances for the dangerous maneuvers (least represented classes) intact. We also wanted this technique to retain the model performance of oversampling and the training time as low as possible but avoid the pitfalls of both techniques.
To achieve this we start by mapping our labels to a binary state, as shown in Step 1 of Figure 3.3 while retaining their original state in a temporary variable. The class corresponding to normal behaviour was left intact while the least supported classes were temporarily agglomerated into a single class representing aggressive maneuvers. At this point our labels represent normal and aggressive behaviour, as

exhibited in the box containing Step 2. To reach Step 3, we apply random down-sampling and the result we get is 50% of the training data with normal behaviour and the other 50% containing all the aggressive maneuvers recorded during the data acquisition trips. The final step is to map to their original values the labels that were classified as aggressive into their original event (acceleration, brakes and turns). We denominated this technique binary undersampling.

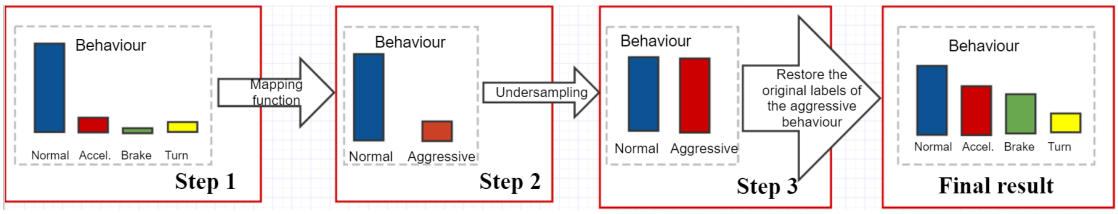

Figure 3.3: Illustration of the process used to balance our datasets.

For random undersampling and binary undersampling, we need to devise a new way of doing validation because of the low support in the class least represented (brake with 166 samples for the first dataset, or approximately 10 seconds, given that, on average, 16 samples were collected per second). Splitting the dataset in the traditional 70% for training, 15% for testing and 15% for validation would leave an accuracy resolution of $\frac{100}{166/16}$, which is approximately 4%. This means that every sample in the brake class is worth approximately 4% during test and validation. As Andrew NG. argues in the seventh chapter of his book [26], small test and validation sets might not be able to account for improvements in the model. To overcome this we decided to use a different seed per run on the data balancing step of the process in order to sample the dataset in different random ways. This is known as Monte Carlo cross-validation and allow us to split the dataset in only two ways, training and test, with a distribution of 70% of samples for training and 30% for testing.

3.2.4 Choosing a classifier

In the first experiment we tried to assess which algorithm or set of algorithms looked promising for our task on our dataset. Note that as mentioned in the end of Section 2.4, the lack of a standard benchmark test makes direct comparison with state of the art difficult. However, we see that SVM and decision trees (or ensembles) have received some attention from the research community. On the one hand. because SVM work based on the minimization of a distance (the distance between the hyperplane separating each pair of classes), it is sensitive to data

transformations [13]. On the other hand, decision trees and ensembles of decision trees do not require feature scaling. Because the performance of SVMs are affected by data transformations, there is a need to run a small experiment to test a some approaches. Three approaches were taken in consideration. The first method is to simply skip this step, the second is to rescale to a specific range, in our case it was [0-1], using a MinMaxScaler and the third option is to standardize the features by removing the mean and scaling to unit variance ([-1,1]).

Based on the current state of the art we have decided to experiment with SVM, K-NN, and a few ensemble methods including gradient boosting, a bagging of decision trees and random forests for the first experiment.

For SVMs to work with multiclass problems we need to reduce the problem to a series of binary problems. We considered a one-vs-one and a one-vs-all strategy in our experiments. A one-vs-all (also known as one-vs-rest or one-against-all) approach trains one classifier per class. During the training process the samples from that class are marked as positive and rest as negative. In a one-vs-one strategy we train N(N-1)/2 classifiers, with N being the number of classes. For the training process we provide a pair of classes and the classifier learns to distinguish between only these two classes.

3.2.5 Feature engineering

The literature provided three different approaches to feature engineering. The first is the easiest, we simply give raw data as input to the classifier. Given a sufficiently high amount of data, some models have been able to achieve state of the art performance on a multitude of tasks in different domains by using raw data. Unfortunately for this field, the data tends to be confidential and not shared among peers, which makes it hard to collect big amounts of data. For our first experiment we also fed the classifier raw data.

The second approach is to engineer new features based on domain knowledge. This tends to be a cumbersome method for finding useful features to feed the classifier. It tends to be a hit or miss approach in which we have no guaranteed of succeed. From Chapter 2, papers using this technique tend to achieve higher results than by simply inputting raw data into the classifier.

The third and last way to create new features that appears in the literature is to agglomerate consecutive instances and produce new features from these small time series. The features derived from this approach can come from signal processing,

averages, medians and other mathematical formulations that can capture relevant information from time series. This is the technique used by authors of the best paper reviewed in Chapter 2 to get their results.

Sentilant decided to go with the second approach for this internship. Adding to the 23 raw data features present in the dataset, 94 more features were added. This features can be consulted in Appendix F. They all are either the difference between the current value observed and the value recorded the past designated time frame (e.g. the difference between the current registered speed and the speed recorded two seconds ago), cumulative sum of a given feature for the past designated time or the value of a given feature observed in a designated time period (e.g. the value of the speed registered on the smartphone 2 seconds ago).
If we add to the algorithm a buffer that holds the information recorded in the last couple seconds and delay the initial inference steps just a few iterations we can do inference for the sample that sits in the middle of the buffer and use what we denominate look ahead samples to help classify, as illustrated on Figure 3.4. Although this technique prevents the method from being used in real time applications, given that current smartphones sample data at high frequencies the time it takes to buffer four samples, which the maximum amount of look ahead points currently being used, is 80 ms, considering the accelerometer is sampling at 50 Hz.

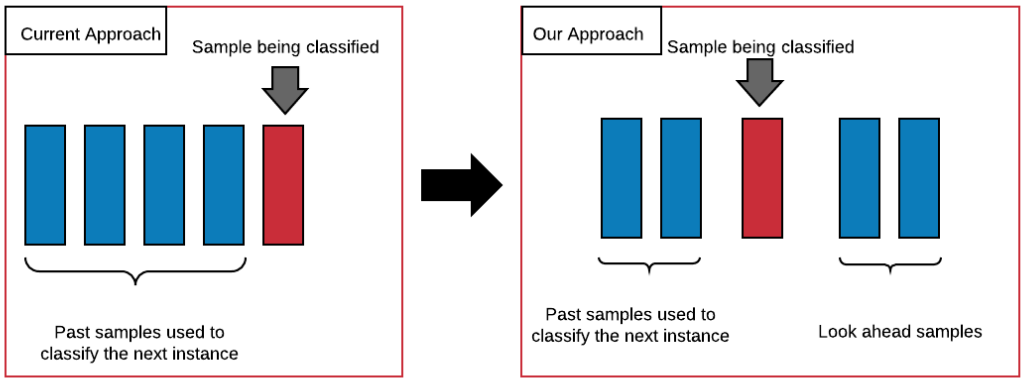

Figure 3.4: Illustration of the process used to predict an arbitrary instance. On the left we have the current approach described on the literature. On the right we have our proposed approach. The color blue denotes samples that are available to the algorithm. Red stands for the sample currently being classified.

3.3 Applying post-processing on top of a simulator

To test our model as if it were in a production environment, a simulator was developed. This simulator is capable of grouping consecutive labels with the same value and produce some statistics for it. With this it was possible to analyze a model on a event by event case. At the end of each trip the simulator outputs some statistics about the whole trip (i.e. confusion matrix, F1 scores, precision, recall, accuracy) and the events (e.g. the precision of the classifier for each event). By the end of this thesis, when we were validating the algorithm with the simulator previously developed, it was noticed that occasionally a sample would be misclassified but the samples around it would have the right value, like shown on the left side of the first case presented in Image 3.5. To correct this mistakes and increase the precision of the algorithm (at the cost of lowering the recall), we introduced a constraint on the classifier.

As shown in Case 2 of the Image 3.5, the classifier is allowed to classify a sample as aggressive behaviour if that same label appears consecutively a specific number of times. Because we are working with three aggressive maneuvers, we had to define three thresholds, one per class. The threshold for each event was found by grid search on a discrete space in the interval [0-35] to find the best precision, the performance metric that matters to Sentilant. A threshold of zero means that as soon as an event is detected, the classifier can present that result immediately. A threshold higher than zero means that the classifier needs to get x more times the same label consecutively, with x being the number of samples.

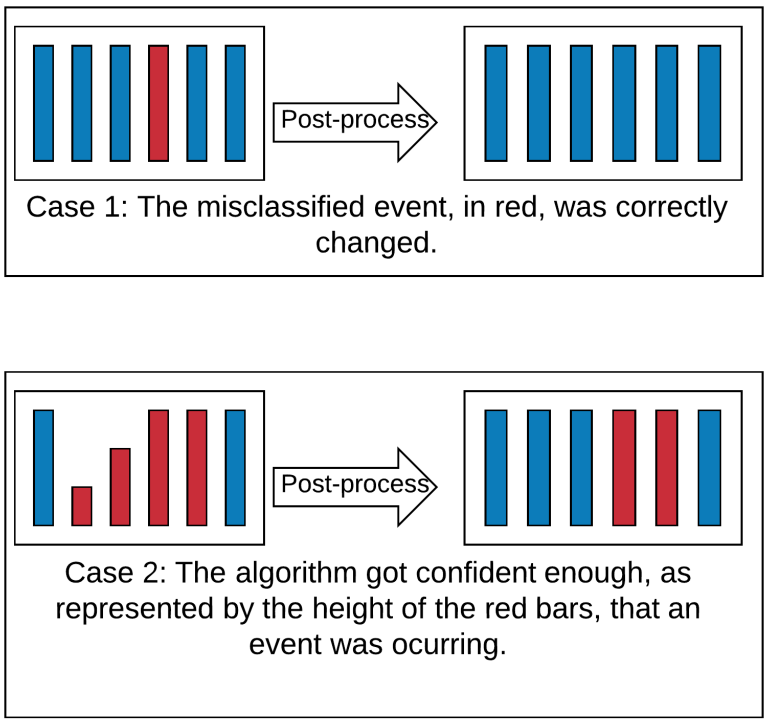

Figure 3.5: Illustration of a misclassified event being corrected in Case 1. Case 2 displays the threshold function. The height of the bars is proportional to the number of consecutive labels necessary to achieve the threshold set for that event. The color blue denotes normal behaviour and red aggressive behaviour.

Chapter 4

Experimental setup and results

In this chapter we take a look at the experiments and report their results. For each experiment there is a description of the purpose of conducting the study followed by the experimental setup. We finish the description by discussing and reporting the results. All the result in this chapter were measured in the testing set using the Monte Carlos methodology described in Chapter 3.

The purpose of the first experiment was to test our class balancing method as well as the pre-processing technique with several existing classifiers.
The second experiment was done to cover some tests not done in the first one. This study introduces a new classifier, three different ways to scale the data and tests the impact of removing the features that are captured by the gyroscope. After the second experiment, we acquired new data.
For our third experiment, on the new dataset, the impact of the phone position was the study subject.
Then there was another study conducted to test how different devices impacted the results.
The fourth experiment was conducted to test a new set features. The number of features and modifications done when compared to the last iteration of the previous feature set is quite drastic. The new feature set has an increased concentration of features that provide information about the samples around it, as described in Section 3.2.5. The last experiment was done to study the impact of feature selection on a real world scenario, provided by the developed simulator.

4.1 Experiment 1

For our first experiment we wanted to start discarding techniques that did not look promising. The goal of this study was to see how our new techniques would compare to the approach taken by the majority of the authors in Chapter 2.
Because the dataset provided had labels that do not correspond to the their true value, we tried to approximate it by using our label correction mechanism, as described in Subsection 3.2.2.

Sentilant revealed three features from it's own research: **delta speed**, **delta course** and **delta timestamp**, all referring to the difference between the current state and the previous state of the car/device. This experiment tests the original 22 features and adds the three features previously explained, making a total of 25. Therefore, there are two sets of features, which we will denominate by original feature set, composed of 22 features and modified feature set "A", with 25. The "A" stands for addition, meaning that to the original set of features a few more were added. A comparison between each feature set can be found in Appendix A.
For SVMs we had to reduce our multiclass problem to a series of binaries classification problems, as described in the Methodology, Chapter 3. This test was run with 30 different seeds using the Monte Carlos approach to split the dataset in 70% training and 30% testing sets. The performance baseline for this experiment is the current Sentilant system. To compare any model with that system the metric adopted was recall as that was the only metric possible to extract from the system. The remaining setup can be found in the following tables:

Table 4.1: Detail of the dataset used in Experiment 1

Dataset	**Description**
Dataset	Provided by Sentilant
Smartphone position	Horizontal

Table 4.2: Detail of Experiment 1

Process	**Parameter**
Label correction mechanism	None, K-means
Feature set used	original, "A"
Classifiers	SVM(OVO), SVM(OVA), bagging decision trees, gradient boosting, K-NN, random forests

Class balacing	Random oversampling, random undersampling, binary undersampling
Feature scaling	Only on SVMs, MinMaxScaler range [0-1]
Post-processing	None

4.1.1 Results of Experiment 1

To compare the results between each model we used the average F1 score of all the identifiable maneuvers i.e, we compact the F1 score obtained by the normal, acceleration, brake and turn classes into one number, using the following mathematical formulation: $\frac{F1_{normal}+F1_{acceleration}+F1_{brake}+F1_{turn}}{4}$. When deemed necessary we will look at other metrics.

To analyze the results that concern our data balancing technique we will use SVM (OVO) and random forests. The results of the remaining classifiers are either the same or approximately equal to those of random forests, so we analyze those at the end considering the best scenario we found for SVMs and random forests.

We will start by analyzing how our balancing strategy compares against random oversampling and random undersampling when using only the original 22 features. From Figure 4.1 and Figure 4.2, it can be seen that random undersampling always produces the worst results, no matter the pre-processing mechanism. In all the tests, binary undersampling always performed better, on average, than random undersampling. When comparing to random oversampling, binary undersampling was able to stay just a couple of percentage points away in three out of the four scenarios illustrated. In the other, the performance was comparable to random undersampling.

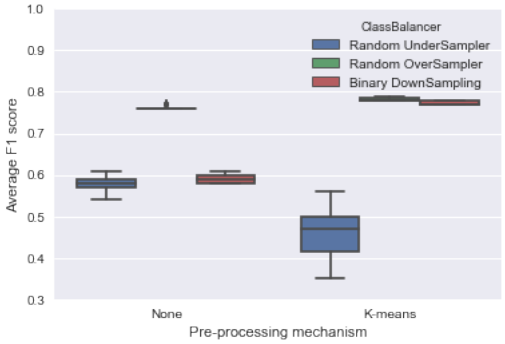

Figure 4.1: Average F1 score for SVM (OVO) trained with 22 features. The colors represent the class balancing technique used. Blue for random undersampling, red for random oversampling and green for our technique, binary undersampling.

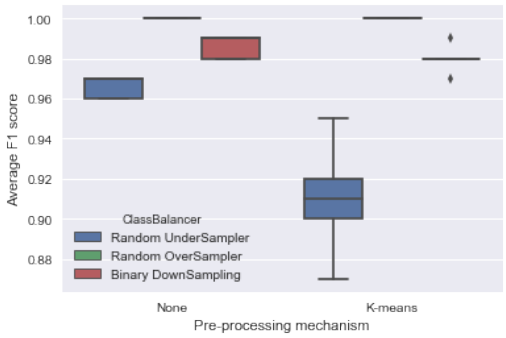

Figure 4.2: Average F1 score for random forests trained with 22 features. The colors represent the class balancing technique used. Blue for random undersampling, red for random oversampling and green for our technique, binary undersampling.

From the pictures above and the confusion matrices in Figure 4.3 and Figure 4.4, showing the best SVM and the best random forest, it is visible that random forests outperform SVMs in all the scenarios present here, in fact, random forests were able to accurately predict the entire test set.

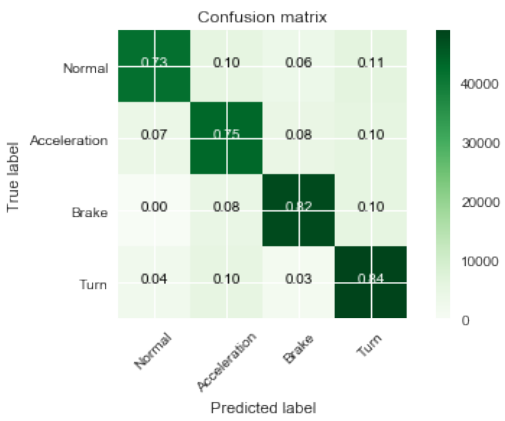

Figure 4.3: Normalize confusion matrix of the best performing SVM shown in the above graph.

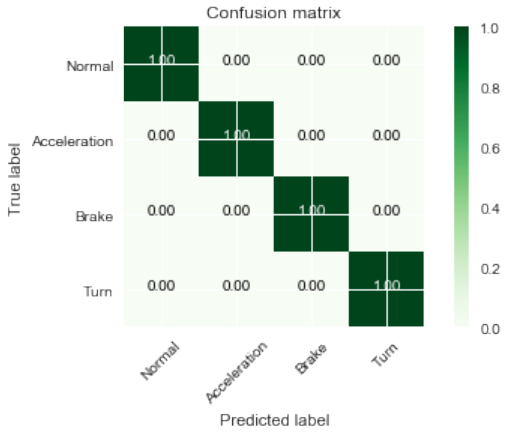

Figure 4.4: Normalized confusion matrix of the best performing random forest in the above graph.

Now we will analyze how our balancing strategy compares against random oversampling and random undersampling when using the original 22 features plus the 3 engineered, giving us a total of 25 features. Figures 4.5 and 4.6 show a vastly different landscape. Here SVMs always achieve a perfect score independently of the method used to balance the dataset. Random forests also achieve good results except when combining random undersampling and our pre-processing mechanism, where the F1 score drops to 96%.

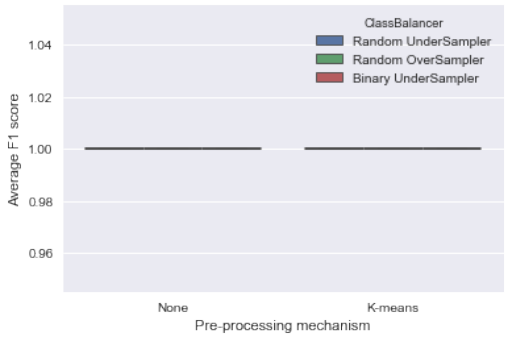

Figure 4.5: Average F1 score of SVMs trained with 25 features. The labels on the x axis identify the pre-processing mechanism and the color distinguishes the different balancing strategy.

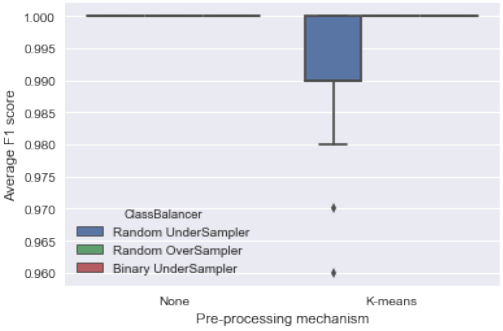

Figure 4.6: Average F1 score of random forests trained with 25 features. The labels on the x axis identify the pre-processing mechanism and the color distinguishes the different balancing strategy.

From this analysis we decided that it would be best to discard some techniques. To balance our dataset we decided to keep only our method, binary undersampling. With the right features it was able to match random oversampling and always be better than random undersampling. This decision also greatly reduces the training time of the models.
The use of 25 features also proved to be valuable as it allow us to use binary undersampling and still achieve the perfect score on the test set. In all the explored scenarios, the addition of the three engineered features improved the average F1 score, unless it was already at 100%.

Upon deciding the best scenario based on two different classifiers, Figure 4.7 shows a comparison of all the classifiers used in the experiment when running under the previously described scenario. Since all classifiers performed equally well on this scenario we decided to keep random forests and SVM (OVO). The reason for this choice is because those are the classifiers that appear more often in the literature.

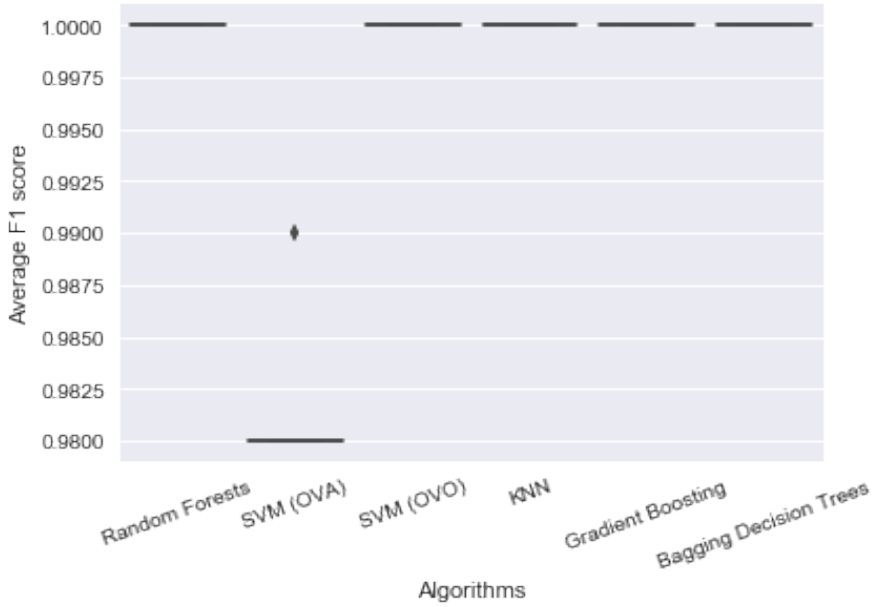

Figure 4.7: Average F1 score of all the classifiers tested. To balance the dataset we used binary undersampling and the number of features was 25. The color denote different pre-processing mechanisms.

On the current dataset, as shown previously, the models here produced are capable of achieving a perfect score on the test set. As seen in Table 4.3, ML models are already outperforming the current system from Sentilant. The great results of this experiment can probably be attributed to the lack of diversity portrayed on the dataset in conjunction with the fact that the same trips were used for training and testing can be the reason that explain this results.

Table 4.3: A comparison of the recall of the current system vs our best model so far.

Event	**Sentilant (%)**	**Our model (%)**
Normal	NA	100
Acceleration	92.3	100
Brake	27.8	100
Turn	60.9	100
Average	60.3	100

*NA- Not available.

4.2 Experiment 2

For our second experiment we introduced a new classifier, extreme gradient boost (also known as XGBoost), a different implementation of the gradient boosting trees used in Experiment 1.
The impact of feature scaling, not covered in the previous experiment, is tested here with three different methods. As the purpose of this experiment was to verify the first sub-goal of the research goal number one, the features calculated by using the gyroscope were also a study subject. The baseline for this experiment is the best result obtained in Experiment 1.

This gives us three feature sets to test. The previous set of features "A", and two new sets. The new sets are the original and modified sets without the features that are obtained by the gyroscope, which we denominate by modified feature set "no gyroscope" and "no gyroscope- A", respectively. The detailed features used can be consulted in Appendices B and C.

This experiment was run with the same 30 seeds as before using the Monte Carlo technique to split the dataset in two ways: 70% for training and the remaining 30% for testing.

Table 4.4: Details of the dataset used in Experiment 2

Dataset	**Description**
Dataset	Provided by Sentilant
Smartphone position	Horizontal

Table 4.5: Details of Experiment 2

Process	**Parameter**
Label correction mechanism	K-means
Feature set used	"A", "No gyroscope", "No gyroscope-A"
Classifiers	SVM(OVO), XGBoost, random forest
Class balacing	Binary undersampling
Feature scaling	None, MinMaxScaler , StandardScaler
Post-processing	None

4.2.1 Results of Experiment 2

This experiment had three variables: the method used to rescale the feautres, which features to use and three classifiers. By using some color when plotting, just three graphs are enough to analyze the results.

As suggested in a paper published by Chih-Wei Hsu *et al.* [13], SVMs were sensible to feature scaling. Excluding the outliers, the best performing SVM, as shown in all scenarios (Figure 4.8, 4.9 and 4.10), used the suggested method: a MinMaxScaler to rescale the data to be between the interval [0-1].

As expected, the ensembles methods produce results completely identical independently of the scaling technique used.

It should be noted that both tree ensembles outperformed the SVM when the features dependant on the gyroscope were removed. Between the ensembles, XGBoost always achieved 100% on the score F1 for all the maneuvers while random forests get 100% based on the value of the seed.

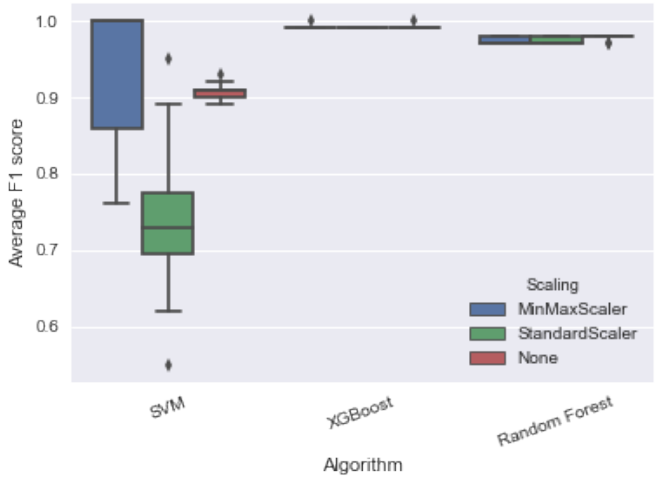

Figure 4.8: Average F1 score for SVMs, XGBoost and random forests trained with feature set "A". The colors stand for the scaling method used.

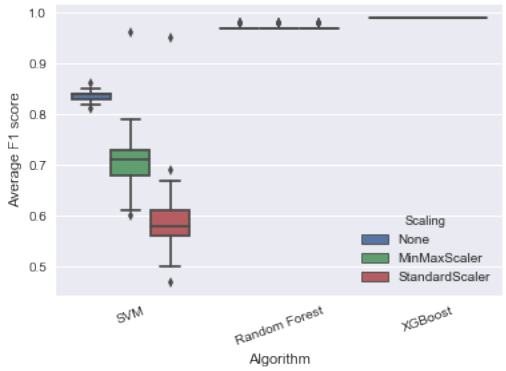

Figure 4.9: Average F1 score for SVMs, XGBoost and random forests trained with feature set "no gyroscope". The colors stand for the scaling method used.

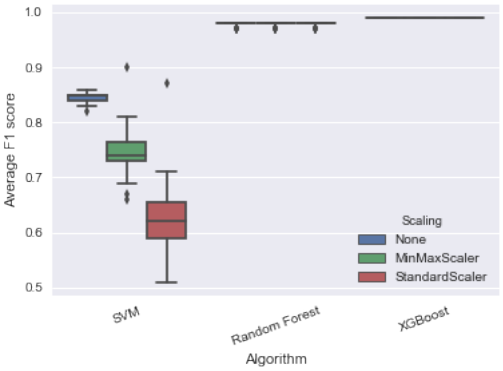

Figure 4.10: Average F1 score for SVMs, XGBoost and random forests trained with feature set "no gyroscope-A". The colors stand for the scaling method used.

From this experiment we can take a couple of conclusions. If the smartphone is in a fixed horizontal position, then without the gyroscope, there is a model that can correctly classify all the samples on the testing dataset. This result was expected because, according to related work, the utility provided by the gyroscope is to change the axis of the information being measure, so that for example, instead of working with the axis the device we can work with axis of the vehicle.
XGBoost also demonstrated to be quite good at fitting the training data we provided, always outperforming SVMs and being marginally better on the average F1 score of the random forests. Extreme gradient boosting was also fitting the training data faster than SVMs. Since there was no downgrade on the baseline scenario from Experiment 1 and a substantial improvement on the scenarios defined in Experiment 2, XGBoost will be the classifier used for future experiments. The second sub-goal of the first goal defined in the Chapter 3 was partially accomplished. It is still necessary to test the model on a different dataset. Note that, as explained in the conclusions of the previous experiment, this results are very optimistic because the dataset lacks diversity in many domains, such as the number of drivers, cars or devices.

4.3 Experiment 3

The purpose of Experiment 3 was to test how the models would perform to positions they had never seen before, which is the second goal written in Chapter 3. For this experiment we had to collect more data. The dataset acquired in April was used here. The labels were corrected manually by one person.

A person had to correct the labels manually because the k-means technique was not working properly. It simply was not changing enough values on the new dataset. To confirm this issue the person manually labelling implemented a script to check the number of different labels between the manually corrected April's dataset and the unsupervised approach. The unsupervised approach probably worked in the Sentilant' dataset because the events were loosely marked while in the April's dataset we tried to have tighter labels. For the April's dataset it is visible that the delay caused by the GSNS is enough to cause an event to not be labelled correctly for maneuvers that do not last for long, such as braking. This will be the last experiment containing the dataset from Sentilant as it contains to much uncertainty on the labels.

Because of the low amount of samples for the classes that represent aggressive behaviour in the April's dataset we decided to keep the data from previous experiments.

Due to the interest in studying the position of the smartphone, the April dataset was split in three ways, according to the position of smartphone: held in the hand, horizontal and vertical.

The training procedure for this experiment always contained data from the dataset provided by Sentilant and from the April dataset. For each dataset the method used to split the data in training and testing was same as previously explained, using the Monte Carlos technique to split the dataset in two ways with 70% of the samples, from each dataset, used for training and the remaining samples for testing purposes, as indicated in Table 4.9.

Table 4.6: Details of the dataset used in Experiment 3

Dataset	**Description**
Dataset 1	Provided by Sentilant (70% for training, 30% for test)
Dataset 2	Acquired in April (70% for training, 30% for test)
Smartphone position	Horizontal, Vertical, held in hand

The number of features increased by three. To the previous three features sets, we added a **delta course**, the **norm of the acceleration**, the **cosine of course** in radians and the **delta cosine course**. The delta features are simply the difference between the current state of the variable and the last sample. The radians of cosine of the course were introduced to prevent a drastic change in values when the course changed from 360^{o} to 0^{o}. This second iteration of the feature set is denominated "A2", to which we can add the remaining restrictions, such as the removal of the features that depend on the gyroscope. For a complete list of the features used in each set check Appendices D and E for the feature sets of "A2" and "no gyroscope-A2", respectively.

Table 4.7: Detail of Experiment 3

Process	**Parameter**
Label correction mechanism	K-means for Sentilant' dataset. Manual for April's dataset.
Feature set used	"A2","no gyroscope-A2"
Classifiers	XGBoost
Class balacing	Binary undersampling
Feature scaling	None
Post-processing	None

4.3.1 Results of Experiment 3

Surprisingly, adding data from the April's dataset always reduces the average F1 score for the Sentilant dataset. One would expect that training a model with a higher degree of diversity would increase results. This might not have happened because the labels in of the Sentilant' dataset do not have the correct value.

No matter the feature set used, classifying a maneuver with the smartphone placed in a different position than those trained does not yield good results. However, classifying events with the smartphone in one of the positions used to train gives results close to 70%. To further proceed we should mix all the available positions and validate with new data, from different trips.

Table 4.8: Results in tabular form for Experiment 3. Colors have different meanings, green represents the horizontal position, yellow vertical and grey the hand. The gradient from red to green represents the average of the average F1 score. Values closer to green are better. The last column contains the feature set used, blue for "A2" set and orange for "no gyroscope-A2"set.

Datasets used for training (70% from each)							
Sentilant' dataset	April's Dataset	Results for dataset 1 (30% of Sentilant' dataset)	Results for test 2 (30% of April's dataset)	New phone position	Results	Avg	Feature set
horizontal	horizontal	0.704030007	0.782590412	hand	0.454994	0.6472	no gyroscope-A2
horizontal	horizontal	0.704030007	0.782590412	vertical	0.341166	0.6093	no gyroscope-A2
horizontal	horizontal	0.722609909	0.878259041	hand	0.458116	0.6863	A2
horizontal	horizontal	0.722609909	0.878259041	vertical	0.364439	0.6551	A2
horizontal	vertical	0.600924634	0.876212586	hand	0.465387	0.6475	no gyroscope-A2
horizontal	vertical	0.600924634	0.876212586	horizontal	0.539424	0.6722	no gyroscope-A2
horizontal	vertical	0.691294487	0.929392256	hand	0.448796	0.6898	A2
horizontal	vertical	0.691294487	0.929392256	horizontal	0.46762	0.6961	A2
horizontal	hand	0.599790649	0.862017769	horizontal	0.367641	0.6098	no gyroscope-A2
horizontal	hand	0.599790649	0.862017769	vertical	0.386718	0.6162	no gyroscope-A2
horizontal	hand	0.665125611	0.898828742	horizontal	0.454584	0.6728	A2
horizontal	hand	0.665125611	0.898828742	vertical	0.454584	0.6728	A2

4.4 Experiment 4

For our fourth experiment we collected more data. We named it June's dataset. The purpose of this experiment is to validate the hypothesis raised in Experiment 3 about mixing all the available positions in a single model and validate it with new trips. The dataset used to train the model was the April's dataset. For validation we used the dataset acquired in June.

Table 4.9: Details of the dataset used in Experiment 3

Dataset	**Description**
Dataset used for training	April's dataset
Dataset used to validate	June's dataset

The number of features dramatically increases for this experiment, going from 26 to 94. We named this set of features "1-gyroscope-A3". The one in the name is the number of features that depend on the gyroscope and A3 simply means the number of iterations in which we have increased the number of features. Here is where we apply the procedures described of the subsection "Feature Engineering", located in Chapter 3. A detailed list can be consulted in Appendix F. The new set of features does not only add new features, it also removes absolute values, such

as absolute longitude or latitude and converts it to relative values.

Table 4.10: Detail of Experiment 4

Process	**Parameter**
Label correction mechanism	Manual
Feature set used	"A2","1-gyroscope-A3"
Classifiers	XGBoost
Class balacing	Binary undersampling
Feature scaling	None
Post-processing	None

4.4.1 Results of Experiment 4

We see a massive increase in the average F1 score, jumping from 0.4% to 0.8%, in Figure 4.11 for the iOS platform when the phone is vertical. As for the android platform, the phone was positioned horizontally as performance does not even reach 0.3% F1 score. With so little trips to validate our model it is not possible to reach any significant conclusions. The gap observed between platforms can be either because of the hardware, the software or the position of the phone. Further research is necessary.

Looking at Figure 4.13, it can be seen that the model using the old feature set is classifying most samples as normal behaviour and can not distinguish between normal and aggressive behaviour. The massive amount of features is therefore beneficial to our model for the iOS platform.

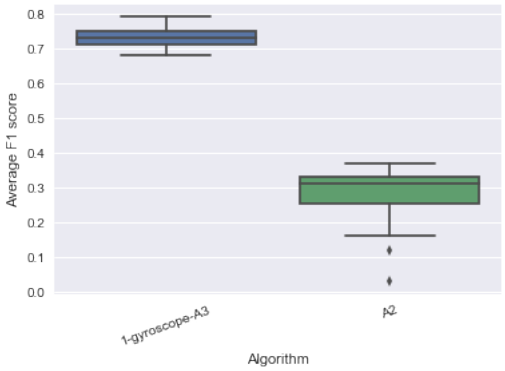

Figure 4.11: Average F1 score for XGBoost on two different feature sets for the iOS platform on a vertical position.

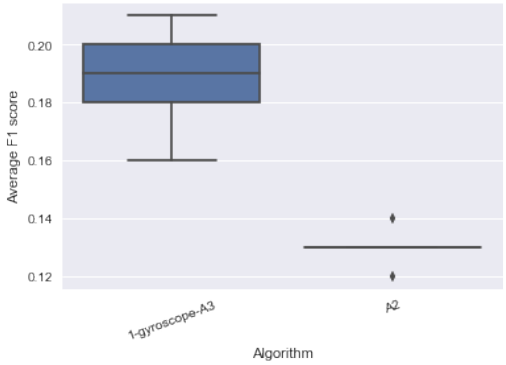

Figure 4.12: Average F1 score for XGBoost on two different feature sets for the Android platform on a vertical position.

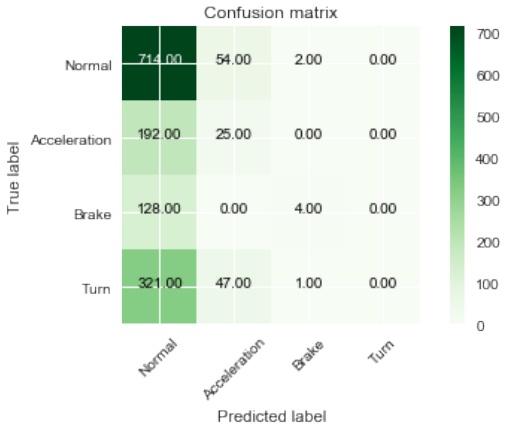

Figure 4.13: Confusion matrix for the best performing xgboost model on iOS using the "A2" feature set.

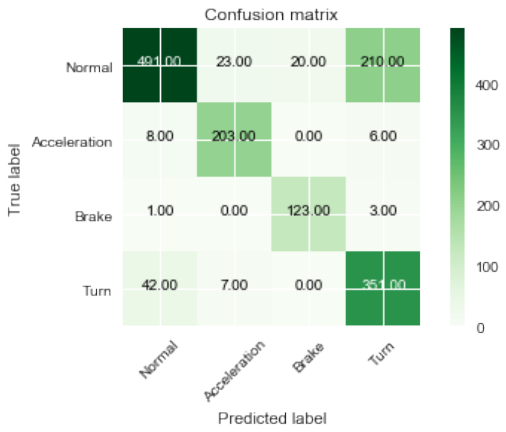

Figure 4.14: Confusion matrix for the best performing xgboost model on iOS using the "1-gyroscope-A3" feature set.

The model that uses the feature set "1-gyroscope-A3" classifies most classes well except for the turning event. Manually labelling turns is not a easy task as there is no apparent feature that can tell exactly when a turn is occurring.

4.4.2 Running our model on real scenarios

To test real scenarios we used the simulator developed. The only trip worth looking in detail are the ones that used a smartphone oriented vertically, as Figure 4.12 reveals that the Android platform, at least when placed horizontally requires further research.
Without using our post-processing technique we obtain the results in Table 4.11. By applying the post-processing technique the model increases the desired performance metric that is desirable to Sentilant, precision. The results in Table 4.12 show an increase in the precision of accelerations but at the cost of lowering recall. This is fine because for Sentilant applications, not giving false positives to the user is of much importance.

Table 4.11: Performance report on iOS trip used to validate the model, without post-processing

	Precision	**Recall**	**F1-Score**	**Support**
Normal	0.94	0.71	0.81	14233
Accel	0.52	0.79	0.62	763
Brake	0.45	0.84	0.59	419
Turn	0.19	0.59	0.28	1297
avg / total	0.85	0.71	0.75	16712

Table 4.12: Performance report on iOS trip used to validate the model, with post-processing

	Precision	**Recall**	**F1-Score**	**Support**
Normal	0.88	0.85	0.86	14233
Accel	0.74	0.27	0.40	763
Brake	0.49	0.31	0.38	419
Turn	0.18	0.35	0.24	1297
avg / total	0.81	0.77	0.78	16712

During the trip to iPark, located in Antanhol, Coimbra, our model detected the events shown in Figure 4.15, when using a iOS device. All the accelerations and brakes that were aggressive got well identified, i.e., when someone swerved to our lane and forced us to suddenly stop, on top of bridge. We discussed with Sentilant and turns may be too sensitive but this might be because it was only a single person correcting the labels, which might have introduced a bias. Nevertheless,

this is something that can be fine tuned in future work.

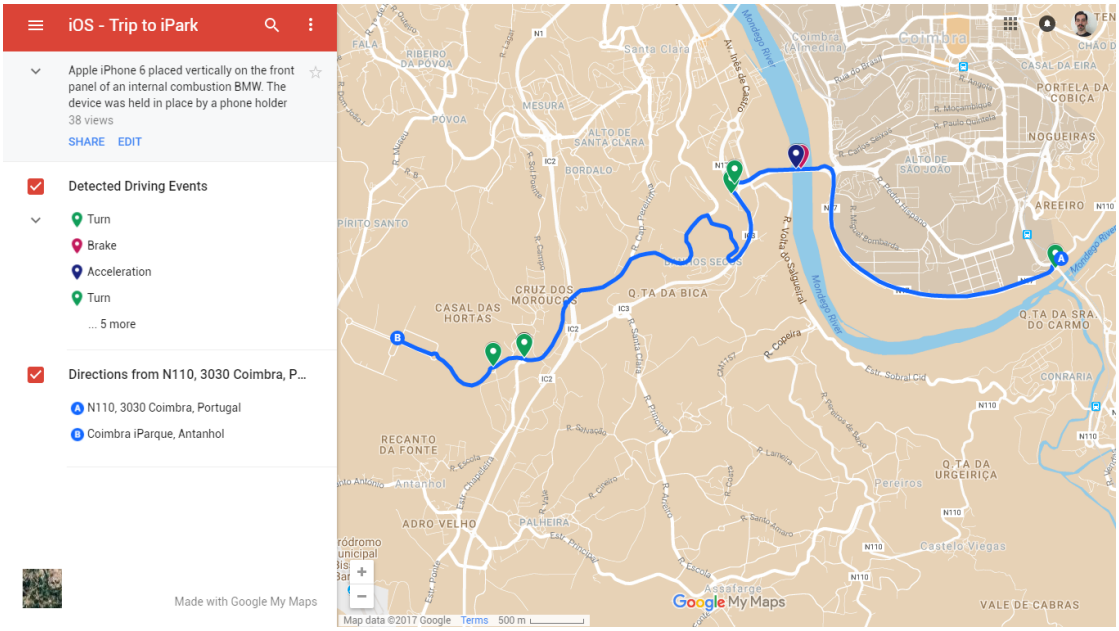

Figure 4.15: Identified events on iOS during a trip to iPark, located in Antanhol, Coimbra. Green markers symbolize turns, red markers brakes and dark purple accelerations.

The android device was oriented horizontally during the trip to iPark. All the identified aggressive maneuvers did not happen. This orientation needs more research.

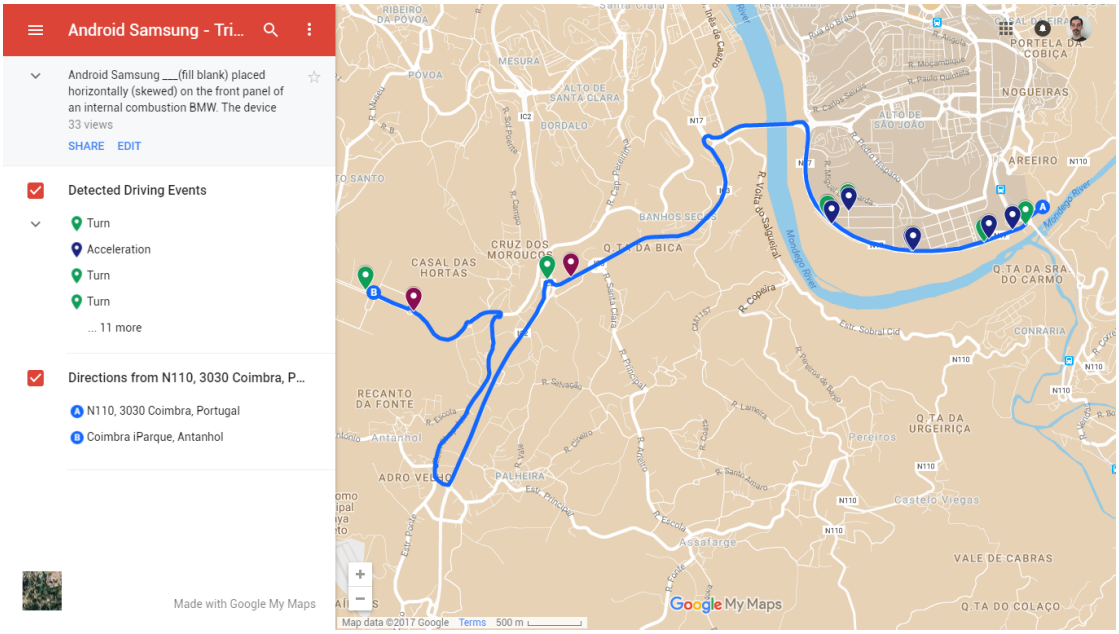

Figure 4.16: Identified events on an Android during a trip to iPark, located in Antanhol, Coimbra. Green markers symbolize turns, red markers brakes and dark purple accelerations.

The phone was oriented vertically, like the iOS device and it worked perfectly on the trip tested. We should have the same trip with an Android and a iOS device with the same orientation to compare both on equal terms. We do not have this kind of data but it appears that phones placed vertically are behaving as expected.

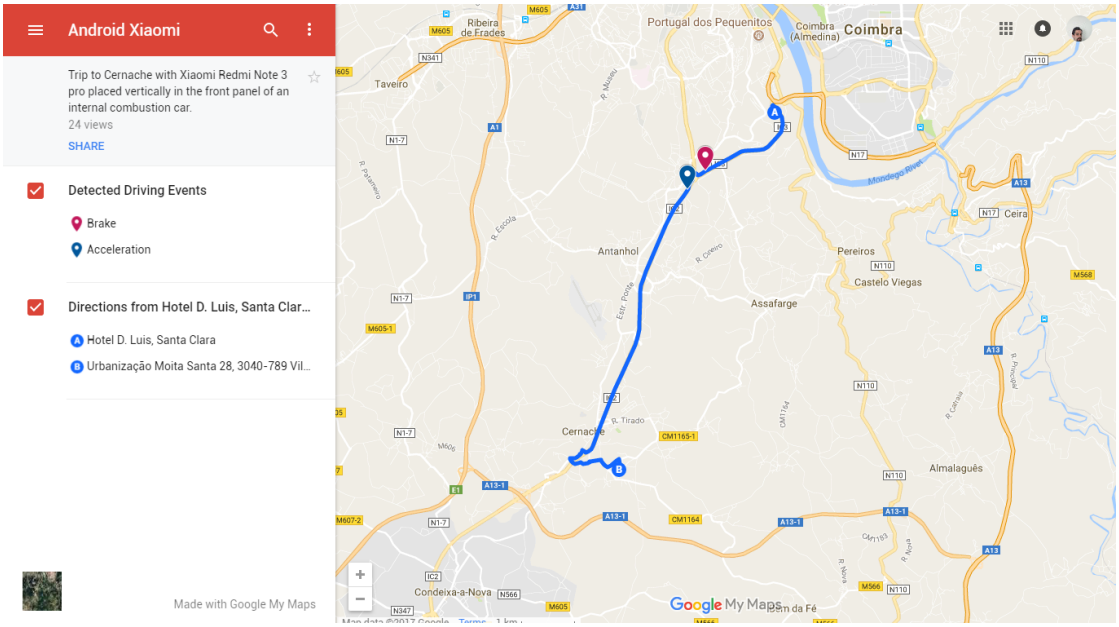

Figure 4.17: Identified events on an Android during a trip Cernache. Green markers symbolize turns, red markers brakes and dark purple accelerations.

To conclude, data from real trips, on different roads never before seen by the model seem to suggest that when placed vertically, our classifier is identifying well the aggressive maneuvers, apart from turns, which seem to be too sensitive. The events identified were always in the correct place and at the right time. For other orientations we need more data to make strong arguments, but the data available seem to suggest that when the phone is oriented horizontally, the system does not perform as expected.

4.5 Experiment 5

Our last experiment dealt with feature selection. Recursive feature elimination was performed on the training set of the dataset acquired in April. It was determined that 10 features was the minimum amount of features that one could retain before the classifier started to lose performance on the average F1 score. The following features were deemed the most important during this process:

- speed;
- cumSum_accelNorm – Cumulative sum of norm of acceleration;

- cumSum_delta_speed_1 – Cumulative sum of the differences in the current speed and the speed observed 1 second ago;
- speed_look_6 – Value of the speed observed 6 samples ago;
- accelNorm_look_6 – Value of the norm of the acceleration as observed 6 samples ago;
- courseCosRads_look_6 – Cosine of course, in radians, as observed 6 samples ago;
- cumSum_delta_speed_1_look_24 – Cumulative sum of the differences in the current speed and the previous speed value, as observed 24 samples ago;
- delta_courseCosRads_1_look_-4 – Difference between the current cosine of course, in radians, and the previous cosine value, as observed 4 sample ahead;
- cumSum_absDelta_courseCosRads_1_look_14 – Cumulative sum of the absolute difference between the current cosine of the course, in radians, and the observed 14 samples ago;
- cumSum_absDelta_courseCosRads_1_look_18 – Cumulative sum of the absolute difference between the current cosine of the course, in radians, and the observed 18 samples ago;

When ran through the simulator, without post-processing enabled we obtained the results in Table 4.13. When comparing to the results in Table 4.11, it is visible that removing features reduces both precision and recall for all aggressive maneuvers while retaining features that depend on the GSNS. Feature selection should be excluded in this situation because it brings no benefit.

Table 4.13: Performance report on iOS trip used to validate the model, with post-processing

	Precision	**Recall**	**F1-Score**	**Support**
Normal	1.00	1.00	1.00	396874
Accel	0.35	0.46	0.40	426
Brake	0.12	0.27	0.16	127
Turn	0.15	0.48	0.23	223
avg / total	1.00	1.00	1.00	397650

Chapter 5

Work Plan and risk analysis

This thesis is based on the innovation of a product, achieved through research and development. Here we detail the work plan created during our first meeting and analyze the risks involved in that plan.

5.1 Work plan

During the first half of the work here produced, the focus was to on study the state of the art, analyze the provided dataset, run preliminary experiments and devised a plan to solve or lessen the effect of the identified issues on the final model, as shown in Fig. 5.1. We followed a basic machine learning approach where we implemented techniques we thought would work the best for our task and then proceeded to validate our ideas.
In machine learning it is hard to define boundaries as to when a phase is finished because there might be a need to revisit any step of the process (modify pre-processing, pos-processing, features or classifiers) and that is why we see overlaps in our Gantt chart.

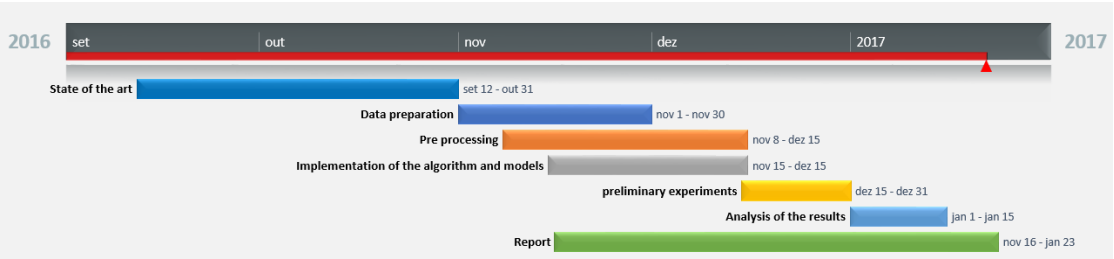

Figure 5.1: Gantt chart of the tasks done in the first half of this dissertation.

In the second half of the work here developed, we finished the research on the first dataset, which was lacking some tests on feature scaling. New data was acquired to validate the model so far developed. When trying to validate the model we ran into some issues with a risk identified below where our pre-processing mechanism was not doing it's task correctly. During the next couple of months we tried to modify it but to no success. We ended up doing manual label correction. While this validation was still ongoing we built a simulator to test how the model would work in real scenarios. We kept on developing new features for the simulator while doing feature engineering to improve the model. This was not enough and we had to implement a post-processing technique to achieve the desirable results. We once again acquired new data and validated our model. The last couple of weeks were dedicate to the writing of the report.

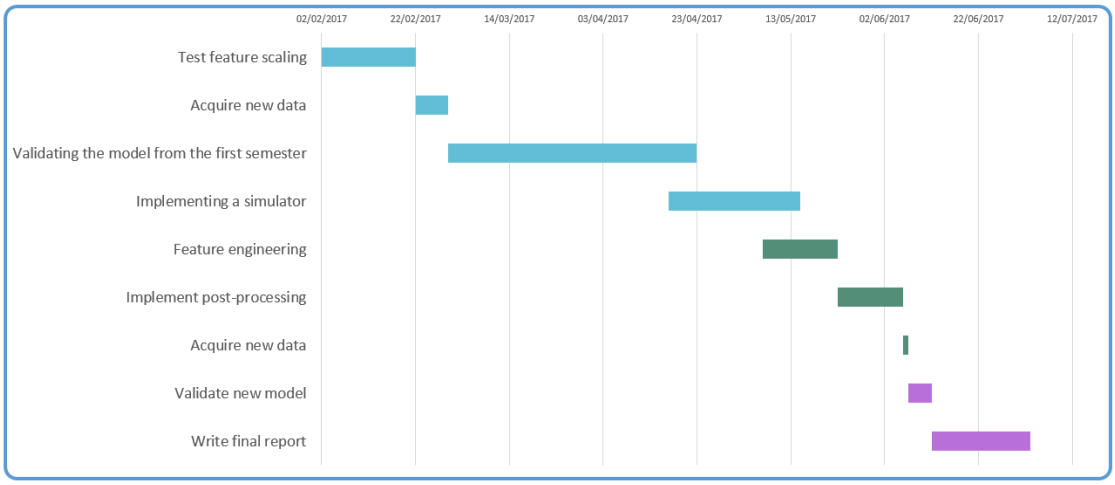

Figure 5.2: Gantt chart of the tasks scheduled for the second half of the work.

5.2 Risk analysis

Over the last few chapters we have been identifying some of the risks that can pose an obstacle to the success of this thesis. In this section we make a comprehensive list of the risks that we encountered or could have met.

1. Lack of reproducibility- as mentioned at the end of Chapter 2, the reproducibility of state of the art techniques is hard to achieve in machine learning because there is no access to the data used in every study. For the subject of this dissertation, there is no standard benchmark to compare our results

against the most advanced algorithms. This problem is not easy to solve as it depends on the community of researchers that are studying this field. Our comparisons will be made against our own models and the current production environment;

2. Long training times- our experiments can take up to two weeks to be fully complete. This can severely delay our work plan. As a countermeasure, we have used our custom data balancing class, presented in Chapter 3, to iterate over our models faster at the cost of potentially lower accuracy;

3. Lack of generalization- this risk arises from the first step of our process, the data collection. The dataset available was recorded with one car and with few Android devices. Although the Android specification dictates a range for all the properties of a sensor, some of the specifications are optional. The risk of recording the events we monitor with just one car is that our model does not have data of other types of automobiles. To mitigate this risk, more data should be collected. We need variety and not only quantity.
   We are also at risk when using our pre-processing mechanism because it is a novel approach, not proven to work on a multitude of datasets. The mitigation strategy for this problem is to simply correct the value of the labels by hand, even if takes a lot of time;

4. Lack of data- the lack of was a constant struggle throughout the thesis. It severely hindered progress. A contingency plan for this problem is to collect more diverse data and to devise new strategies to overcome this problem.

Chapter 6

Conclusions

Profiling the behavior patterns of a driver is of great importance because it can help save human lives. According to [12], the success of this work will help drivers reduce their over speeding and aggressive incidents. As a consequence of reducing the number of accidents on the road, this product has the potential side effect of reducing the public spending on health care, as pointed out in [28].

Commercially, the final model here developed is ready to be tested on the field. The data seem to suggest that more research may be necessary to make the algorithm work on all possible orientations of the device. By making our model almost independent of hardware (no gyroscope necessary) and software (iOS and Android) we are increasing the number of people that can have a satisfactory experience with the resultant applications that use a version of the Drivian Core Software Development Kit (SDK) containing our models. By incorporating AI into a core product of the company and improving a driver's profiling, we are providing a solution that has direct application in key areas such as: fleet management, business-to-consumer (B2C) and insurance, among others. All of this is going to be provided while still respecting the business requirements.

Scientifically, our most important contributions are the proposed techniques to balance the dataset and our feature engineered. The novel balancing technique here proposed is a new way to look at the dataset and the features here engineered represented a breakthrough to generalize the models here investigate, as seen during real usage. Binary undersampling was specifically created to tackle this task and was demonstrated to rival with over sampling but without artificially increasing the size of the dataset. The lack of a public dataset means that the results claimed here are only comparable to the current production environment. The lack of data makes our scientific conclusions weak as they should be verified in more scenarios.

Data suggests that the research goal one might have been accomplished for a restrictive scenario, the phone must be vertically oriented. As for the others positions the phone could take there was not enough data to validate our model in that realm. Goal number one was partially successful as it verified in the last experiment that when placed vertically, phones from different platforms behave as expected. The sub-goal number one can not be validated either because we lack two phones in the same position in our validation set. Sub-goal number two was not attained as we could not remove one last feature that depends on the GSNS.

Appendix A

Comparison of the original feature set with the modified feature set “A”

Table A.1: Comparison of the original feature set and the feature set ”A“.

Feature	**Feature set “A”**
Raw acceleration on the x-axis of the phone	Not modified
Raw acceleration on the y-axis of the phone	Not modified
Raw acceleration on the z-axis of the phone	Not modified
Gravity on the x-axis of the phone	Not modified
Gravity on the y-axis of the phone	Not modified
Gravity on the x-axis of the phone	Not modified
Rotation of the device around the x-axis	Not modified
Rotation of the device around the y-axis	Not modified
Rotation of the device around the z-axis	Not modified
Speed of the device	Not modified
Course of the device	Not modified
Altitude	Not modified
Latitude	Not modified
Longitude	Not modified
Heading	Not modified
Pitch	Not modified
Yaw	Not modified
Roll	Not modified

Timestamp of the measurement	Not modified
Acceleration on the x-axis of the vehicle	Not modified
Acceleration on the y-axis of the vehicle	Not modified
Acceleration on the z-axis of the vehicle	Not modified
Delta speed	Added
Delta course	Added
Delta timestamp	Added

Appendix B

Comparison of the original feature set with the modified feature set "no gyroscope"

Table B.1: Comparison of the original feature set and the "no gyroscope" feature set.

Feature	**"No gyroscope" feature set**
Raw acceleration on the x-axis of the phone	Not modified
Raw acceleration on the y-axis of the phone	Not modified
Raw acceleration on the z-axis of the phone	Not modified
Gravity on the x-axis of the phone	Removed
Gravity on the y-axis of the phone	Removed
Gravity on the x-axis of the phone	Removed
Rotation of the device around the x-axis	Removed
Rotation of the device around the y-axis	Removed
Rotation of the device around the z-axis	Removed
Speed of the device	Not modified
Course of the device	Not modified
Altitude	Not modified
Latitude	Not modified
Longitude	Not modified
Heading	Removed
Pitch	Removed
Yaw	Removed
Roll	Removed

Timestamp of the measurement	Not modified
Acceleration on the x-axis of the vehicle	Removed
Acceleration on the y-axis of the vehicle	Removed
Acceleration on the z-axis of the vehicle	Removed

Appendix C

Comparison of the original feature set with the modified feature set "no gyroscope-A"

Table C.1: Comparison of the original feature set and the "no gyroscope-A" feature set.

Feature	**"No gyroscope-A" feature set**
Raw acceleration on the x-axis of the phone	Not modified
Raw acceleration on the y-axis of the phone	Not modified
Raw acceleration on the z-axis of the phone	Not modified
Gravity on the x-axis of the phone	Removed
Gravity on the y-axis of the phone	Removed
Gravity on the x-axis of the phone	Removed
Rotation of the device around the x-axis	Removed
Rotation of the device around the y-axis	Removed
Rotation of the device around the z-axis	Removed
Speed of the device	Not modified
Course of the device	Not modified
Altitude	Not modified
Latitude	Not modified
Longitude	Not modified
Heading	Removed
Pitch	Removed
Yaw	Removed
Roll	Removed

Timestamp of the measurement	Not modified
Acceleration on the x-axis of the vehicle	Removed
Acceleration on the y-axis of the vehicle	Removed
Acceleration on the z-axis of the vehicle	Removed
Delta speed	Added
Delta course	Added
Delta timestamp	Added

Appendix D

Comparison of the original feature set with the modified feature set "A2"

Table D.1: Comparison of the original feature set and the feature set "A2".

Feature	**Feature set "A2"**
Raw acceleration on the x-axis of the phone	Not modified
Raw acceleration on the y-axis of the phone	Not modified
Raw acceleration on the z-axis of the phone	Not modified
Gravity on the x-axis of the phone	Not modified
Gravity on the y-axis of the phone	Not modified
Gravity on the x-axis of the phone	Not modified
Rotation of the device around the x-axis	Not modified
Rotation of the device around the y-axis	Not modified
Rotation of the device around the z-axis	Not modified
Speed of the device	Not modified
Course of the device	Not modified
Altitude	Not modified
Latitude	Not modified
Longitude	Not modified
Heading	Not modified
Pitch	Not modified
Yaw	Not modified
Roll	Not modified

Timestamp of the measurement	Not modified
Acceleration on the x-axis of the vehicle	Not modified
Acceleration on the y-axis of the vehicle	Not modified
Acceleration on the z-axis of the vehicle	Not modified
Delta speed	Added
Delta course	Added
Norm of the acceleration	Added
Cosine of the course, in radians	Added
Delta course cosine, in radians	Added

Appendix E

Comparison of the original feature set with the modified feature set "no gyroscope-A2"

Table E.1: Comparison of the original feature set and the "no gyroscope-A" feature set.

Feature	**"No gyroscope-A" feature set**
Raw acceleration on the x-axis of the phone	Not modified
Raw acceleration on the y-axis of the phone	Not modified
Raw acceleration on the z-axis of the phone	Not modified
Gravity on the x-axis of the phone	Removed
Gravity on the y-axis of the phone	Removed
Gravity on the x-axis of the phone	Removed
Rotation of the device around the x-axis	Removed
Rotation of the device around the y-axis	Removed
Rotation of the device around the z-axis	Removed
Speed of the device	Not modified
Course of the device	Not modified
Altitude	Not modified
Latitude	Not modified
Longitude	Not modified
Heading	Removed
Pitch	Removed
Yaw	Removed
Roll	Removed

Timestamp of the measurement	Not modified
Acceleration on the x-axis of the vehicle	Removed
Acceleration on the y-axis of the vehicle	Removed
Acceleration on the z-axis of the vehicle	Removed
Delta speed	Added
Delta course	Added
Delta timestamp	Added
Norm of the acceleration	Added
Cosine of the course, in radians	Added
Delta course cosine, in radians	Added

Appendix F

Description of all the features engineered

When using all the features from the table below, we denominated that set of features by "1-gyroscope-A3". This is the last iteration of our feature engineering work. The one on the name is the number of features that depend on the gyroscope and A3 simply means that this is the third iteration of feature engineering where we add more feature.

Table F.1: Description of each feature engineered.

Feature name	**Description**
location_timestamp	Unix timestamp of the row, as given by the accelerometer.
speed	Speed, as given by the Global Satellite Navigation System.
cumSum_accelNorm	Cummulative sum of the norm of the acceleration for the last second.
delta_speed_1	Difference between the current speed and the speed registered one second ago.
cumSum_delta_speed_1	Cummulative sum of the differences between the current speed and the speed registered one second ago.
delta_courseCosRads_1	Difference between the current course, as given by the GSNS and the course registered one second ago, in radians.

cumSum_delta_courseCosRads_1	Cummulative sum of the differences between the current course, as given by the GSNS and the course registered one second ago, in radians.
delta_accelNorm_2	Difference between the current norm of the acceleration and the norm of the acceleration registered two second ago.
cumSum_delta_accelNorm_2	Cummulative sum of the differences between the current norm of the acceleration and the norm of the acceleration registered two seconds ago.
absDelta_courseCosRads_1	Absolute difference between the current course and the course registered one second ago, in radians.
cumSum_absDelta_courseCosRads_1	Cummulative sum of the absolute differences between the current course and the course registered one second ago, in radians.
speed_look_-4	Speed registered 4 sample ahead (uses our look ahead sample strategy).
speed_look_-2	Speed registered 2 sample ahead (uses our look ahead sample strategy).
speed_look_6	Speed registered 6 sample ago.
speed_look_10	Speed registered 10 sample ago.
speed_look_14	Speed registered 14 sample ago.
speed_look_18	Speed registered 18 sample ago.
speed_look_24	Speed registered 24 sample ago.
accelNorm_look_-4	Norm of the acceleration registered 4 samples ahead (uses our look ahead sample strategy).
accelNorm_look_-2	Norm of the acceleration registered 2 samples ahead (uses our look ahead sample strategy).
accelNorm_look_6	Norm of the acceleration registered 6 samples ago.
accelNorm_look_10	Norm of the acceleration registered 10 samples ago.
accelNorm_look_14	Norm of the acceleration registered 14 samples ago.

accelNorm_look_18	Norm of the acceleration registered 18 samples ago.
accelNorm_look_24	Norm of the acceleration registered 24 samples ago.
cumSum_accelNorm_look_-4	Cumulative sum of the norm of the acceleration as registered 4 samples ahead (uses our look ahead sample strategy).
cumSum_accelNorm_look_-2	Cumulative sum of the norm of the acceleration as registered 2 samples ahead (uses our look ahead sample strategy).
cumSum_accelNorm_look_6	Cumulative sum of the norm of the acceleration as registered 6 samples ago.
cumSum_accelNorm_look_10	Cumulative sum of the norm of the acceleration as registered 10 samples ago.
cumSum_accelNorm_look_14	Cumulative sum of the norm of the acceleration as registered 14 samples ago.
cumSum_accelNorm_look_18	Cumulative sum of the norm of the acceleration as registered 18 samples ago.
cumSum_accelNorm_look_24	Cumulative sum of the norm of the acceleration as registered 24 samples ago.
courseCosRads_look_-4	Course, in radians, as registered 4 samples ahead (uses our look ahead sample strategy).
courseCosRads_look_-2	Course, in radians, as registered 2 samples ahead (uses our look ahead sample strategy).
courseCosRads_look_6	Course, in radians, as registered 6 samples ago.
courseCosRads_look_10	Course, in radians, as registered 10 samples ago.
courseCosRads_look_14	Course, in radians, as registered 14 samples ago.
courseCosRads_look_18	Course, in radians, as registered 18 samples ago.
courseCosRads_look_24	Course, in radians, as registered 24 samples ago.

delta_speed_1_look_-4	Difference between the current speed and the speed registered 4 samples ahead (uses our look ahead sample strategy).
delta_speed_1_look_-2	Difference between the current speed and the speed registered 2 samples ahead (uses our look ahead sample strategy).
delta_speed_1_look_6	Difference between the current speed and the speed registered 6 samples ago.
delta_speed_1_look_10	Difference between the current speed and the speed registered 10 samples ago.
delta_speed_1_look_14	Difference between the current speed and the speed registered 14 samples ago.
delta_speed_1_look_18	Difference between the current speed and the speed registered 18 samples ago.
delta_speed_1_look_24	Difference between the current speed and the speed registered 24 samples ago.
cumSum_delta_speed_1_look_-4	Cumulative sum of the differences between the current speed and the speed registered 4 samples ahead (uses our look ahead sample strategy).
cumSum_delta_speed_1_look_-2	Cumulative sum of the differences between the current speed and the speed registered 2 samples ahead (uses our look ahead sample strategy).
cumSum_delta_speed_1_look_6	Cumulative sum of the differences between the current speed and the speed registered 6 samples ago.
cumSum_delta_speed_1_look_10	Cumulative sum of the differences between the current speed and the speed registered 10 samples ago.
cumSum_delta_speed_1_look_14	Cumulative sum of the differences between the current speed and the speed registered 14 samples ago.
cumSum_delta_speed_1_look_18	Cumulative sum of the differences between the current speed and the speed registered 18 samples ago.

cumSum_delta_speed_1_look_24	Cumulative sum of the differences between the current speed and the speed registered 24 samples ago.
delta_courseCosRads_1_look_-4	Difference between the course, in radians, observed 1 second ago and the course registered 4 samples ahead (uses our look ahead sample strategy).
delta_courseCosRads_1_look_-2	Difference between the course, in radians, observed 1 second ago and the course registered 2 samples ahead (uses our look ahead sample strategy).
delta_courseCosRads_1_look_6	Difference between the course, in radians, observed 1 second ago and the course registered 6 samples ago.
delta_courseCosRads_1_look_10	Difference between the course, in radians, observed 1 second ago and the course registered 10 samples ago.
delta_courseCosRads_1_look_14	Difference between the course, in radians, observed 1 second ago and the course registered 14 samples ago.
delta_courseCosRads_1_look_18	Difference between the course, in radians, observed 1 second ago and the course registered 18 samples ago.
delta_courseCosRads_1_look_24	Difference between the course, in radians, observed 1 second ago and the course registered 24 samples ago.
cumSum_delta_courseCosRads_1_look_-4	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 4 samples ahead (uses our look ahead sample strategy).
cumSum_delta_courseCosRads_1_look_-2	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 2 samples ahead (uses our look ahead sample strategy).

cumSum_delta_courseCosRads_1_look_6	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 6 samples ago.
cumSum_delta_courseCosRads_1_look_10	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 10 samples ago.
cumSum_delta_courseCosRads_1_look_14	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 14 samples ago.
cumSum_delta_courseCosRads_1_look_18	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 18 samples ago.
cumSum_delta_courseCosRads_1_look_24	Cumulative sum of the difference between the course, in radians, observed one second ago and the value observed 24 samples ago.
delta_accelNorm_2_look_-4	Difference between the norm of the acceleration observed two seconds ago and the value observed 4 samples ahead (uses our look ahead sample strategy).
delta_accelNorm_2_look_-2	Difference between the norm of the acceleration observed two seconds ago and the value observed 2 samples ahead (uses our look ahead sample strategy).
delta_accelNorm_2_look_6	Difference between the norm of the acceleration observed two seconds ago and the value observed 2 samples ago.
delta_accelNorm_2_look_10	Difference between the norm of the acceleration observed two seconds ago and the value observed 10 samples ahead.
delta_accelNorm_2_look_14	Difference between the norm of the acceleration observed two seconds ago and the value observed 14 samples ahead

delta_accelNorm_2_look_18	Difference between the norm of the acceleration observed two seconds ago and the value observed 18 samples ahead
delta_accelNorm_2_look_24	Difference between the norm of the acceleration observed two seconds ago and the value observed 24 samples ahead
cumSum_delta_accelNorm_2_look_-4	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 4 samples ahead (uses our look ahead sample strategy).
cumSum_delta_accelNorm_2_look_-2	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 2 samples ahead (uses our look ahead sample strategy).
cumSum_delta_accelNorm_2_look_6	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 6 samples ago.
cumSum_delta_accelNorm_2_look_10	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 10 samples ago.
cumSum_delta_accelNorm_2_look_14	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 14 samples ago.
cumSum_delta_accelNorm_2_look_18	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 18 samples ago.
cumSum_delta_accelNorm_2_look_24	Cumulative sum of differences between the norm of acceleration observed two seconds ago and 24 samples ago.
absDelta_courseCosRads_1_look_-4	Absolute difference between the course observed one second ago and the value observed 4 samples ahead (uses our look ahead sample strategy).
absDelta_courseCosRads_1_look_-2	Absolute difference between the course observed one second ago and the value observed 4 samples ahead (uses our look ahead sample strategy).

absDelta_courseCosRads_1_look_6	Absolute difference between the course observed one second ago and the value observed 6 samples ago.
absDelta_courseCosRads_1_look_10	Absolute difference between the course observed one second ago and the value observed 10 samples ago.
absDelta_courseCosRads_1_look_14	Absolute difference between the course observed one second ago and the value observed 14 samples ago.
absDelta_courseCosRads_1_look_18	Absolute difference between the course observed one second ago and the value observed 18 samples ago.
absDelta_courseCosRads_1_look_24	Absolute difference between the course observed one second ago and the value observed 24 samples ago.
cumSum_absDelta_courseCosRads_1_look_4	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 4 samples ahead (uses our look ahead sample strategy).
cumSum_absDelta_courseCosRads_1_look_2	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 2 samples ahead (uses our look ahead sample strategy).
cumSum_absDelta_courseCosRads_1_look_6	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 6 samples ago.
cumSum_absDelta_courseCosRads_1_look_10	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 6 samples ago.
cumSum_absDelta_courseCosRads_1_look_14	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 6 samples ago.

cumSum_absDelta_courseCosRads_1_look_18	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 6 samples ago.
cumSum_absDelta_courseCosRads_1_look_24	Cumulative sum of the absolute differences between the course observed one second ago and the value observed 6 samples ago.

Appendix G

List of seeds used

The seeds in this list were randomly generated using atmospheric noise from the website random.org.

Table G.1: List of seeds used.

Number	**Seed**
1	96336637
2	39505458
3	55459387
4	12771562
5	11459385
6	90037278
7	92467704
8	77648643
9	5269032
10	840601
11	81112458
12	64232047
13	41929190
14	87618330
15	4386231
16	2132609
17	4777222
18	78775539
19	47624036
20	23045465
21	77250062

22	51536209
23	81081104
24	14651079
25	37770447
26	30854037
27	71537772
28	71591351
29	82543978
30	69428507
31	3562973

Bibliography

[1] D. Banerjee and N. Banerjee. "How's My Driving? A Spatio-Semantic Analysis of Driving Behavior with Smartphone Sensors". In: *10th International Conference, MOBIQUITOUS* (2013).

[2] G. Castignani et al. "Driver Behavior Profiling Using Smartphones: A Low-Cost Platform for Driver Monitoring". In: *IEEE Intelligent transportation systems magazine* (2015), pp. 91–102.

[3] P. Chaovalit, C. Saiprasert, and T. Pholprasit. "A method for driving event detection using SAX with resource usage exploration on smartphone platform". In: *EURASIP Journal onWireless Communications and Networking* (2014).

[4] P. Chaovalit, C. Saiprasert, and T. Pholprasit. "Method for Driving Event Detection Using SAX on Smartphone Sensors". In: *International Conference on ITS Telecommunications* (2013).

[5] *Chart of smartphone processing power over the years.* URL: i.dailymail.co.uk/i/pix/2015/05/26/23/29171A1F00000578-3098315-image-a-11_1432677790789.jpg (visited on 12/15/2016).

[6] Z. Chen et al. "Abnormal Driving Behaviors Detection and Identification Using Smartphone". In: *12th Annual IEEE International Conference on Sensing, Communication, and Networking* (2015).

[7] A. Chowdhury, T. Chakravarty, and P. Balamuralidhar. "Estimating True Speed of Moving Vehicle using Smartphone-based GPS Measurement". In: *IEEE International Conference on Systems, Man, and Cybernetics* (2014).

[8] H. Eren et al. "Estimating Driving Behavior by a Smartphone". In: *Intelligent Vehicles Symposium* (2012).

[9] A. Ghose et al. "An Enhanced Automated System for Evaluating Harsh Driving Using Smartphone Sensors". In: *International Conference on Distributed Computing and Networking* (2016). DOI: dx.doi.org/10.1145/2833312.2849555.

[10] A. Halevy, P. Norvig, and F. Pereira. "The Unreasonable Effectiveness of Data". In: *IEEE Intelligent Systems, March/April* (2009).

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Second Edition. 2008. URL: statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf (visited on 12/15/2016).

[12] J. Hickman and E. Geller. "Self-Management to Increase Safe Driving Among Short-Haul Truck Drivers". In: *Journal of Organizational Behavior Management* 23.4 (2004-2005), pp. 1–20. DOI: 10.1300/J075v23n04_01.

[13] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. *A Practical Guide to Support Vector Classification*. URL: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

[14] D. Johnson and M. Trivedi. "Driving Style Recognition Using a Smartphone as a Sensor Platform". In: *14th International IEEE Conference on Intelligent Transportation Systems* (2011).

[15] E. Keogh and J. Lin. *SAX*. URL: www.cs.ucr.edu/~eamonn/SAX.htm (visited on 12/15/2016).

[16] E. Keogh et al. "Dimensionality reduction for fast similarity search in large time series databases". In: *Journal of Knowledge and Information Systems* (2000), pp. 263–286.

[17] H. Chul Kim et al. "Support vector machine ensemble with bagging". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2388 (2002), pp. 397–408. DOI: 10.1007/3-540-45665-1_31.

[18] C. Lakhmi, B. Himansu, and M. Durga (eds.) M. Jyotsna. *Comparative Analysis of Decision Tree Algorithms: ID3, C4.5 and Random Forest*. 1st ed. Smart Innovation, Systems and Technologies 31. Springer India, 2015. ISBN: 978-81-322-2204-0,978-81-322-2205-7.

[19] G. Lemaître, F. Nogueira, and C. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning". In: *CoRR* abs/1609.06570 (2016). URL: arxiv.org/abs/1609.06570.

[20] J. Lin et al. "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms". In: *Proceedings of the 8th ACM SlGMOD Workshop on Research 1ssues in Data Mining and Knowledge Discovery* (2003).

[21] M. Van Ly, S. Martin, and M. Trivedi. "Driver Classification and Driving Style Recognition using Inertial Sensors". In: *IEEE Intelligent Vehicles Symposium, June* (2013).

[22] S. Martiny M. Van Lyy and M. Trivediy. "Driver Classification and Driving Style Recognition using Inertial Sensors". In: *IEEE Intelligent Vehicles Symposium* (2013).

[23] J. Magaña and M. Organero. "Artesima: Using Android device as an Eco-Driving assistant". In: *Journal of Selected Areas in Mechatronics, June Edition* (2011), pp. 1–8.

[24] D. Massie. "Analysis of accident rates by age, gender, and time of day based on the 1990 Nationwide Personal Transportation Survey". In: (1993).

[25] Müller and Meinard. *Dynamic Time Warping*. URL: www.springer.com/cda/content/document/cda_downloaddocument/9783540740476-c1.pdf?SGWID=0-0-45-452103-p173751818 (visited on 12/15/2016).

[26] Andrew Ng. *Machine Learning Yearning*. Draft V0.5. 2016. URL: www.mlyearning.org/ (visited on 12/15/2016).

[27] N. Oliver and A. Pentland. "Graphical Models of Driver Behavior Recognition and Prediction in a Smart Car". In: *IEEE of Intelligent Vehicles Symposium, Cambridge* (2000), pp. 7–12.

[28] World Health Organization. *Road traffic injuries*. URL: www.who.int/mediacentre/factsheets/fs358/en/ (visited on 12/15/2016).

[29] World Health Organization. *The top 10 causes of death*. URL: www.who.int/mediacentre/factsheets/fs310/en/ (visited on 12/15/2016).

[30] P. Patel et al. *Mining Motifs in Massive Time Series Databases*. URL: cs.gmu.edu/~jessica/publications/motif_icdm02.pdf (visited on 12/15/2016).

[31] T. Pholprasit, W. Choochaiwattana, and C. Saiprasert. "A Comparison of Driving Behaviour Prediction Algorithm Using Multi-Sensory Data on a Smartphone". In: *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing* (2015).

[32] A. Sathyanarayana, S. Sadjadi, and J. Hansen. "Leveraging Sensor Information from Portable Devices towards Automatic Driving Maneuver Recognition". In: *15th International IEEE Conference on Intelligent Transportation Systems* (2012).

[33] Kaz Sato. *Using machine learning for insurance pricing optimization*. URL: https://cloud.google.com/blog/big-data/2017/03/using-machine-learning-for-insurance-pricing-optimization (visited on 06/17/2017).

[34] *Sentilant*. URL: www.sentilant.com (visited on 12/15/2016).

[35] *SKlearn Random Forest Classifier*. URL: scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (visited on 12/15/2016).

[36] *Smartphone market share*. URL: www.idc.com/prodserv/smartphone-os-market-share.jsp (visited on 12/15/2016).

[37] *Weka*. URL: www.cs.waikato.ac.nz/ml/weka/ (visited on 12/15/2016).