A Machine Learning Approach Based on Automotive Engine Data Clustering for Driver Usage Profiling Classification

Cephas Alves da Silveira Barreto[1], **João C. Xavier-Júnior**[1], **Anne M.P. Canuto**[2] **and Ivanovitch M.D. da Silva**[1]

[1]Digital Metropolis Institute - Federal University of Rio Grande do Norte (UFRN)
Av. Senador Salgado Filho, 3000 – 59.078-970 – Natal – RN – Brazil

[2]Informatics and Applied Mathematics Department - Federal University of Rio Grande do Norte (UFRN) – 59.078-970 – Natal – RN – Brazil

cephasax@gmail.com, {jcxavier,ivan}@imd.ufrn.br, anne@dimap.ufrn.br

Abstract. *The potential for processing car sensing data has increased in recent years due to the development of new technologies. Having this type of data is important, for instance, to analyze the way drivers behave when sitting behind steering wheel. Many studies have addressed the drive behavior by developing smartphone-based telematics systems. However, very little has been done to analyze car usage patterns based on car engine sensor data, and, therefore, it has not been been explored its full potential by considering all sensors within a car engine. Aiming to bridge this gap, this paper proposes the use of Machine Learning techniques (supervised and unsupervised) on automotive engine sensor data to discover drivers' usage patterns, and to perform classification through a distributed online sensing platform. We believe that such platform can be useful used in different domains, such as fleet management, insurance market, fuel consumption optimization,* CO_2 *emission reduction, among others.*

1. Introduction

The number of vehicles grows considerably each year across the globe. In Brazil, for instance, it has already reached 45.4 million vehicles in 2015, which proportionally represents one car for every 4.4 inhabitants. This increase brings serious problems to everyday life, e.g., deaths caused by accidents, traffic jam, and above all high pollution rates, which affects everyone by causing diseases, such as asthma, pulmonary Cancer and Cardiovascular problems [World Health Organization 2015].

Among the aforementioned problems, the emission of pollutants and deaths caused by car accidents need to be evaluated considering two different aspects, which are the drivers usage of vehicles and their behaviors. The first aspect can be investigated by analyzing automotive engine sensing data that can be transmitted to smartphones or tablets via On-Board Diagnostic (OBD-II). On the other hand, the second aspect is related to the way drivers behave when sitting behind steering wheel, e.g., handling of steering wheel and pedals, high speed and change of lane or existing road [Kumagai and Akamatsu 2006], [Amsalu et al. 2015], [Ly et al. 2013]. In this sense, both aspects have become popular in domains, such as: fleet management and car insurance market.

Therefore, this work intends to address the driver usage patterns problem by considering only the multiple measurement sensors that are located in the car engine. We strongly believe that a system (smartphone-based or online platform) can provide valuable usage information by capturing, fusing, analyzing, and discovering patterns within this data. In this way, business segments can benefit from the use of Machine Learning (ML) techniques for the discovery of drive usage patterns within an organization. By applying effective techniques for drive usage pattern, companies can better evaluate their drivers, and also their vehicles.

Recently, supervised Machine Learning techniques (e.g., Naive Bayes, MLP, SVM) have been applied to automotive engine sensing data [Zhang et al. 2016], [Ly et al. 2013], [Meseguer et al. 2013], [Al-Sultan et al. 2013]. However, as the number of sensors in a car engine is very high, there is no public available databases that are composed of fused data from all of them that can be used for training and testing a classification system. On the other hand, others studies have applied unsupervised ML techniques (e.g. k-Means) over data composed of a specific subset of engine sensors in order to create clusters within the data, and consequently use them as classes [Higgs and Abbas 2015], [Ly et al. 2013].

The main of this paper is to apply Machine Learning techniques (supervised and unsupervised) on automotive engine sensor data to discover drivers usage patterns. After having the sensor data, different unsupervised ML techniques will be applied in order to generate partitions which will be analyzed under certain criteria and used as class labels. Then, the second step is to apply supervised techniques to classify the drivers usage patterns. To the best of the author knowledge, this is the first work to propose the use of both unsupervised and supervised ML techniques for clustering engine sensor data, and performing classification of drive usage pattern in an online platform.

2. Background

2.1. Automotive Control Specifications

In order to deal with various types of automotive components and to standardize the communication between them, two important standards have been defined. The first one, called Controller Area Network (CAN) serial bus system, was created by Bosch in 1986, and adopted as an ISO standard in 1993. This specification allows communication between micro-controllers and devices without a host computer. It is currently used as a worldwide standard in communication between vehicle components.

The second one, called On-Board Diagnostic (OBD-II), is the standard protocol used across most vehicles to retrieve vehicle diagnostic information [Barreto and Mooers 2017]. In this standard, data is generated by a engine control unit (ECU or also called engine control module) within a vehicle. OBD-II provides vehicle self-diagnosis and reports capabilities for repair technicians to access subsystem information for the purpose of performance monitoring and repairing [Barreto and Mooers 2017].

In general, ODB-II is used to wirelessly transmit relevant vehicle engine information to smartphones or tablets (e.g., coolant temperature, load, RPM, intake manifold pressure, speed, throttle position, air flow, short term fuel trim bank, timing advance, fault codes, etc.). In this investigation, fifteen (15) different engine sensors values are transmitted via OBD-II to a smartphone to be stored. Then, these values are sent to a distributed online platform which is able to store, fuse and analyze records for all registered users.

2.2. Machine Learning Techniques

In general, Machine Learning techniques are data dependent. On one hand, unsupervised techniques are used to discover natural groupings in a set of examples without knowledge of class labels. Clustering is one of the most known techniques in this group. On the other hand, supervised techniques are used to identify to which set of labels a new example (instance) belongs, based on a training set of data that contains examples whose labels are known. Classification is one of the most known techniques in this group [Mitchell].

As automotive engine data has no class label, but only attributes that describe the features of each type of sensor, we apply different clustering algorithms in order to create groups of objects, or clusters, in such a way that objects in one cluster are very similar, and objects in different clusters are quite distinct. After creating clusters or groups, we use them as class labels in a classification task [Alsmadi and Alhami 2015]. In this sense, we use three clustering algorithms which are described as follows: **k-Means**, **Expectation-Maximization (EM)**, and **Hierarchical clustering**.

After having created the class labels, we applied six different classification algorithms which are described as follows: **Decision Tree (J48)**, **k-NN (k-Nearest Neighbours)**, **Multilayer Perceptron (MLP)**, **Naive Bayes (NB)**, **Random Forest (RF)** and **Support Vector Machine (SVM)**.

3. Related Work

3.1. Smartphone sensing data

In general, the vast majority of the existing studies use smartphone sensing data to predict the behavior of car drivers, such as in [Eren et al. 2012, Johnson and Trivedi 2011, Castignani et al. 2015]. In Eren et al. [Eren et al. 2012], for instance, the authors proposed a driver behavior classification method that predicts two different driver classes, unsafe and safe drivers. They also used a Dynamic Time Warping (DTW) model to compute the similarity of each event to template data, and a Bayesian classification model (Naive Bayesian) to decide whether the driver was unsafe or safe. In Johnson et al. [Johnson and Trivedi 2011], the authors also used DTW to measure data from smartphone sensors, GPS and camera. Their work evaluated the performance of different sensor fusion sets to detect lateral and longitudinal movements. However, no classification algorithm was used in this work.

In You et al. [You et al. 2012], the authors proposed the CarSafe platform, a smartphone application which fuses information obtained from front and rear cameras, sensors and GPS to detect dangerous driving events. According to them, drowsiness can be detected by using front cameras and image processing algorithms.

Finally, in Castignani et al. [Castignani et al. 2015], the authors proposed Sense-Fleet, a driver profile platform that is able to detect risky driving events independently from mobile device and vehicle. They used a fuzzy system to compute a score for each driver by using data from Accelerometer, Magnetometer, Gravity Sensor, GPS, and also route topology and weather information.

3.2. Automotive engine sensing data

Aiming to provide more useful information for drivers, some studies have combined data from smartphone sensors and engine sensors [Araujo et al. 2012, Meseguer et al.

2013, Zhang et al. 2016]. In Araujo et al. [Araujo et al. 2012], for instance, a smartphone application has been proposed to help drivers to reduce fuel consumption in vehicles by using GPS data and engine data (e.g., speed, acceleration, altitude, throttle, fuel consumption, rpm).

In Meseguer et al. [Meseguer et al. 2013], a sensing platform (DrivingStyles) has been proposed to perform classification of driving styles by analyzing the driver behavior using the following data: speed, acceleration, and revolutions per minute of the engine (rpm), among others features collected via OBD-II. According to the authors, DrivingStyles is able to characterize the type of road (city, suburban or highway) on which the vehicle is moving, as well as the degree of aggressiveness of each driver by using a neural network (MLP). Their goal was to assist drivers at correcting some bad habits when driving, while offering helpful tips to improve fuel economy for a car.

Another sensing platform has been proposed in Zhang et at. [Zhang et al. 2016]. This platform uses a Support Vector Machine (SVM) classifier to classify drivers based on engine sensor data collected from OBD-II and also smartphone sensor data. According to the authors, the platform is able to classify drivers behaviors by using each sensor data (car and phone) independently, and combining both sensors' data.

3.3. Discussion

Most of the studies (first group - smartphone sensing data) above focus on smartphone sensing data to predict the behavior of drivers [Eren et al. 2012, Johnson and Trivedi 2011, Castignani et al. 2015]. In this sense, data collected from smartphones (e.g., acceleration, gyroscope, magnetometer, GPS and camera) is used for defining whether the drivers are risky or safe. However, only one study has used a classification algorithm (Naive Bayes) [Eren et al. 2012], whereas the others have used different techniques (e.g., DTW or fuzzy system).

The second group of studies focus on automotive engine sensing data or the combination of both types of data (engine and smartphone) [Araujo et al. 2012, Meseguer et al. 2013, Zhang et al. 2016]. In this group, two studies have used classification algorithms (MLP and SVM) to classify driver behaviors [Meseguer et al. 2013, Zhang et al. 2016], whereas another one has focused on reduction of fuel consumption [Araujo et al. 2012]. Differently from the first two studies, in the third study, the authors have used a fuzzy system. On top of that, the engine sensor data used in the studies above is limited to few sensor features (e.g., speed, throttle and rpm), which does not exploit the full potential of engine sensor data.

Finally, it is important to emphasize that none of the studies has used clustering techniques to create groups in the sensor data, and use those groups as labels for classification purposes. Moreover, a data pre-processing phase is required when using this type of data (no labels). In this sense, a manual labeling process is used in most of the cases [Meseguer et al. 2013, Zhang et al. 2016]. To the best of our knowledge, there is no sensing platform which uses data from all possible engine sensors, and also uses both supervised and unsupervised ML techniques to predict driver usage patterns.

4. The Proposed Automotive Engine Sensing Platform

The main aim of this paper is to propose an automotive engine sensing distributed platform capable of collecting, clustering and performing classification of drivers' usage. By performing such complex task this platform will be useful in fleet management, insurance market, fuel consumption optimization and CO_2 emission reduction.

As depicted in Figure 1, the general architecture of the Automotive Engine Sensing Platform is divided in four modules. The first one, called Monitoring, is responsible for extracting and sending the engine sensors data to the next module. The second module, called Application, is responsible for collecting, performing data pre-processing, and storing data to be classified. The third module (Service) is responsible for collecting data from lower level module, providing classification, and storing analytic results obtained by this module. The last module, called Client Application Prototype, is responsible for providing access to the Service module, and therefore it is able to receive requests from different applications regarding the analytic results of the driving evaluation.

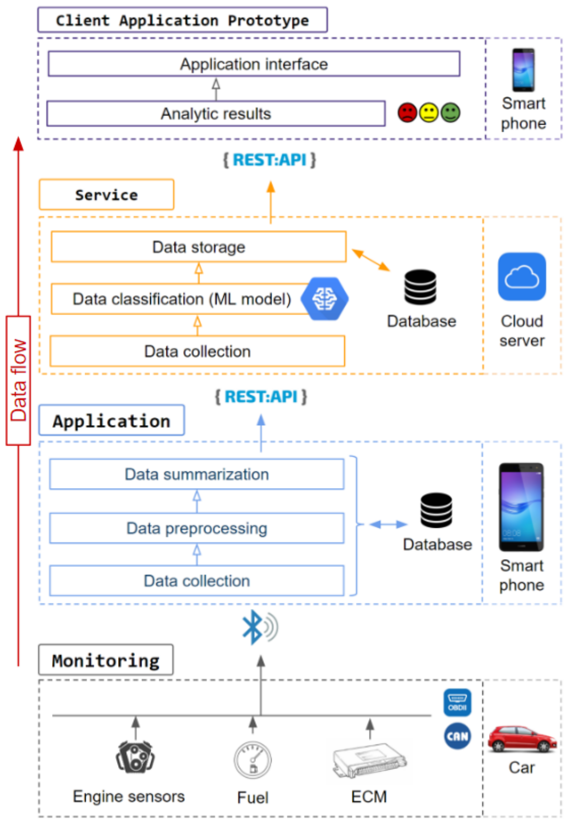

Figure 1. Automotive Engine Sensing Platform architecture.

4.1. Monitoring

In general, the majority of the OBD-II (On-Board Diagnostics) readers function as regular gateway by collecting data from the engine sensors and sending it via Bluetooth to a particular destination. In order to receive data from an OBD-II reader, we implemented structural modifications to an open source application, called Smart Fleet Android [Viana 2018a], [Viana 2018b], to perform data acquisition of a subset of fifteen (15) sensors over a given time window (10 seconds). After extracting data from each sensor, which will be called as feature, this module sends it (a vector of features) to the upper level module. The features are described as follows:

$$V_d = \left\{ \begin{array}{c} ep[i], bp[i], ect[i], fl[i], fp[i], ft[i], el[i], \\ e_{RPM}[i], e_{MAP}[i], e_{MAF}[i], s[i], tp[i], \\ e_{LTFT1}[i], e_{LTFT2}[i], er[i] \end{array} \right\}_{i=1}^{n} \tag{1}$$

where V_d (vector data) is composed by ep (engine power), bp (barometric pressure), ect (engine coolant temperature), fl (fuel level), fp (fuel pressure), ft (fuel type), el (engine load), e_{RPM} (engine RPM - Revolutions Per Minute), e_{MAP} (MAP - Manifold Absolute Pressure), e_{MAF} (MAF - mass air flow), s (speed) and tp (throttle position), e_{LTFT1} (engine LTFT - Long Term Fuel Trim 1), e_{LTFT2} (engine LTFT - Long Term Fuel Trim 2), er (equivalent ratio is related to air fuel), whilst n represents the number of sampling instants. In fact, V_d represents the vehicle operation in a window of 10 seconds.

4.2. Application

As mentioned earlier (Section 4.1), we implemented important changes in the Smart Fleet Android application, allowing it to collect, send, and more important, process engine sensing data. By using our version, the driver's smartphone collects engine sensors data transmitted via Bluetooth, processes it and sends the resulting data to the upper level. Moreover, this module is designed to collect data from the monitoring module, which is extracted in a vector format ($V_{d_1}, V_{d_2}, ..., V_{d_n}$), and store it on demand. However, before making the data available for classification, this module needs to perform two procedures (Data Pre-processing and Data Summarization) which will be detailed below. After that, the organized data is ready to be sent to the upper level (Web service) model when WiFi connection is available.

1. Data Pre-processing: As row data cannot be used directly for performing an analysis process, Application (module) needs to provide some data pre-processing techniques. Basically, two main pre-processing techniques are applied, missing value definition and normalization. After the end of a driver's route, this module seeks for missing, noisy (erroneous and outliers) and inconsistent data. Then, it replaces the values of each attribute accordingly to its characteristics (categorical or numeric) by using Mean, Median or Mode. Finally, the values of all numeric attributes are normalized [0:1].
2. Data Summarization: After applying the data pre-processing procedure, data summarization is applied on the resulting data. The aim is to map $V_{d_1}, V_{d_2}, ..., V_{d_n} \to F_d$, where $F_d \in \mathbb{R}^{n_f}$ represents the set of n_f characteristics:

$$F_d = \left\{ \begin{array}{c} \overline{ep}, \overline{bp}, \overline{ect}, \overline{fl}, \overline{fp}, \overline{ft}, \overline{el}, \overline{e_{RPM}}, \overline{e_{MAP}}, \\ \overline{e_{MAF}}, \overline{s}, \overline{tp}, \overline{e_{LTFT1}}, \overline{e_{LTFT2}}, \overline{er} \end{array} \right\} \tag{2}$$

   where $\overline{ep}$ is the average of this particular feature (engine power), and so on.
   As each vector V_{d_1} represents the values of a 10 second window, the best way to summarize all the V_d's values was to use the average operator and to use the average values of all 15 features. This is represented by Eq.(2).

4.3. Web Service

This module is responsible for performing classification based on drivers' usage data received from the Application module. It collects the summarized data from the Application module and applies the already trained ML model. In other words, based on the summarized data, the trained ML model will classify the drivers usage profiling based on a driver profiling (F_d). In this way, F_d can be mapped as:

$$F_d \rightarrow \{L, M, H\} \tag{3}$$

where L, M e H represents "Low", "Mid" and "High", respectively (see Figure 2).

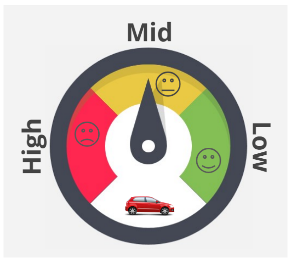

Figure 2. Graphical representation of the classes.

After classifying the driver's usage pattern, it stores that information for later use by any type of application.

5. Methodology

This section will describe some important aspects of the experimental methodology, such as the creation of the database, the selection process of the clustering algorithm, and more importantly the selection process of the most suitable classification algorithm, since the main purpose of the proposed engine sensing platform is classification.

5.1. Experimental Testbed

A 2015 Chevrolet S10 (206 hp with 2.5 L flex-fuel) collected all the data used in this study. This model is popular, and has characteristics common to many other models of this type, which reinforces the possibility of applying this experiment to other popularly used vehicles.

The chosen route corresponds to an urban stretch, depicted in Figure 3, located in the city of Natal (Brazil). The total distance is approximately 18.8 kilometer, taking around 34 minutes, according to the fastest route projection done by Google Maps[1]. We collected data from 19 participant drivers who have volunteered to take part in the experiment.

5.2. Methods and Materials

Figure 4 presents the overall experimental methodology used in this work. After collecting data from all nineteen (19) drivers, we placed it into a table with fifteen (15)

[1]https://goo.gl/A7KViD

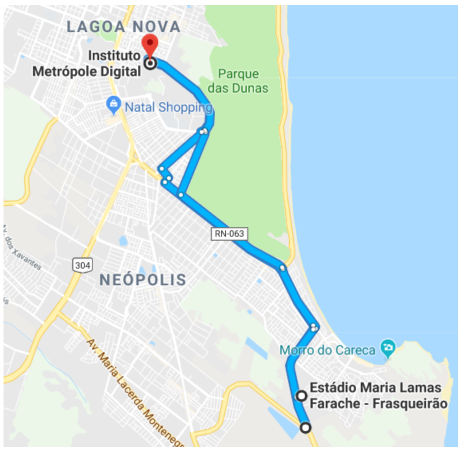
LAGOA NOVA
Instituto Metrópole Digital
Parque das Dunas
Natal Shopping
RN-063
304
NEÓPOLIS
Av. Maria Lacerda Montenegro
Morro do Careca
Estádio Maria Lamas Farache - Frasqueirão

Figure 3. Predefined route for data collection.

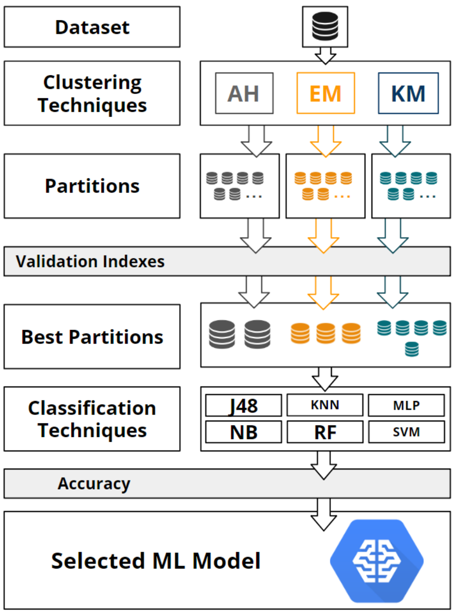

Figure 4. Experimental methodology.

columns (features) and 8.245 rows (instances), representing 434 instances per drive on average. Data pre-processing techniques have been applied to this table, resulting in a dataset ready to be used for a ML model. Three different clustering (unsupervised) techniques have been used over the pre-processed dataset, resulting in several partitions (each model can provide more than one partition). Then, two validation indexes have been applied in order to choose the best partitions, the ones that best represent the original data. Then, the clusters defined by the winning partitions were transformed into class labels. After this phase, six classification (supervised) techniques have been applied over the best partitions in order to analyze the best overall model, based on mean accuracy rate over all datasets. Finally, the selected model has been trained and tested using the best partition. The next subsections will describe, in more details, the important aspects of the experimental methodology.

5.2.1. Clustering

As the aforementioned dataset has no class label (class attribute), we applied three different clustering algorithms (e.g., k-Means, EM and Agglomerative Hierarchical) in order to obtain different partitions representing the drivers' profiling. Initially, we used only the EM algorithm with no predefined k (number of cluster - $k = -1$). As a result, we obtained a partition with 11 clusters (groups). Based on this result,from this first experiment, we varied values of k from 2 to 11.

As the clustering algorithms have different characteristics (deterministic and probabilistic), we performed 10 executions for each value of k for the probabilistic ones (k-Means and EM). Therefore, a total of 30 partitions were generated (3 algorithms and 10 executions).

5.2.2. Validation Measures

Once all partitions were created, we applied two different clustering validation metrics (Silhouette and Davies-Bouldin) [Halkidi et al. 2002] over 30 partitions created by the clustering algorithms, and averaged the indexes values (10 executions) for each k. The results for both indexes have showed that the best partition was created by the k-Means algorithm with $k = 3$. However, we decided to perform a broader analysis choosing not one, but the top ten best partitions, described as follows: EM ($k = 3$, $k = 4$ and $k = 5$), Agglomerative Hierarchical ($k = 3$ and $k = 4$), k-Means ($k = 3$, $k = 4$, $k = 5$, $k = 6$ and $k = 7$). As it can be observed, in all three algorithms, the partitions with 3 and 4 groups were selected in the top ten list. This might be an indication that the number of classes should be selected between these two numbers. Nevertheless, for a broader analysis, we decided to evaluate the performance of the classification algorithms over the best 10 partitions.

5.2.3. Classification

These partitions (10) have been used as 10 datasets in the classification analysis, which aimed to analyze the accuracy performance of 6 classification algorithms (e.g., Decision Tree, k-NN, Multi Layer Perceptron, Naive Bayes, Random Forest and Support Vector Machine). For each classification algorithm, we used three different training/test division, 50/50 split percentage, 70/30 split percentage, and 10 fold cross validation. The mean accuracy results for all six classification algorithms over 10 datasets (partitions) will be presented and discussed in the next section (Section 6).

5.3. Implementational Issues

We used both clustering and classification algorithms that were extracted from the WEKA API, and for simplicity reasons, we used the default parameters of all six classification algorithms, which are automatically chosen by WEKA. The validation metrics (Silhouette and DB) were implemented in Java using the WEKA API. We also implemented a Java class for training and saving the ML model.

The Friedman statistical test was applied to evaluate the performance of all six classifiers over the ten partitions. It is important to emphasize that the Friedman test is applied directly on the mean accuracy rate of all classifiers. If any significant difference is detected, a post-hoc test will be applied.

Table 1. Mean accuracy rate on three test sets.

-	Classification Techniques					
Partitions	**J48**	**k-NN**	**MLP**	**NB**	**RF**	**SVM**
AH_k_3	90.01	**95.65**	90.57	84.34	91.71	88.28
AH_k_4	89.33	**95.39**	89.06	82.57	91.16	87.09
EM_k_3	96.97	93.76	97.32	**99.47**	97.68	94.84
EM_k_4	96.39	90.67	96.30	**99.17**	97.22	93.39
EM_k_5	96.97	90.21	95.98	**99.29**	96.65	93.07
KM_k_3	95.73	96.02	**99.58**	93.23	96.82	98.94
KM_k_4	94.83	93.93	**99.14**	90.61	96.09	98.07
KM_k_5	93.39	92.55	**97.01**	90.08	95.17	97.52
KM_k_6	92.84	92.17	**96.94**	91.01	94.66	97.38
KM_k_7	92.05	91.71	**97.44**	89.65	94.15	97.13
Mean Acc	93.85	93.21	**95.93**	91.94	95.13	94.57

6. Experiment Results

In this section, a comparison analysis of six different classification techniques over ten partitions with different number of clusters (groups) is performed. This analysis is based on the accuracy (classification error) of each classification technique which is the average of three distinct training/test division.

Table 1 presents the mean accuracy obtained by J48, k-NN, MLP, NB, RF and SVM. In this table, the most accurate algorithm is highlighted in bold, for each database. Note that the clustering algorithm used to create the partition/dataset has a strong influence in the performance of the classification algorithms. From Table 1, we can observe that k-NN obtained the best accuracy results in the partitions created by the Agglomerative Hierarchical algorithm (AH_k_3 and AH_k_4), whereas NB obtained in all three partitions created by EM (EM_k_3, EM_k_4 and EM_k_5), and MLP obtained in all five k-Means partitions (KM_k_3, KM_k_4, KM_k_5, KM_k_6 and KM_k_7).

Although three different classification algorithms had good performance, as shown in the last line of the table, MLP obtained the highest mean accuracy (95.93), followed by RF (95.13) and SVM (94.57). In fact, MLP provided reasonable performance for the datasets generated by the other two clustering algorithms (AH and EM), always reaching the second or third highest accuracy.

In order to conduct a more robust analysis of the results, the Friedman statistical test was used (at the conventional significance level of 5%) to determine whether or not there is a statistically significant difference between the mean accuracy rates of the classification techniques across the 10 partitions/datasets.

The Friedman statistical test was applied on the mean accuracy rates of all six classification techniques (Table 1). By analyzing the test result, we could observe that there has been a statistically significant difference among the accuracy rates ($p-values = 3.48\ x\ 10^{-07}$). For this reason, a post-hoc test has been applied. Table 2 presents the $p-values$ for all two-by-two possible comparisons. Bold and underline $p-values$ represent the significant difference between both classification algorithms (line and column). As Table 2 is a symmetric matrix, we will only present the superior part of this matrix.

As it can be observed from Table 2, the performance of MLP was statistically

Table 2. Post-Hoc Test Results

-	k-NN	MLP	NB	RF	SVM
J48	0.9742	**0.0051**	0.9062	**0.0289**	0.7181
k-NN	-	**0.0002**	0.9997	**0.0020**	0.2500
MLP		-	**0.0001**	0.9955	0.2670
NB			-	**0.0006**	0.1394
RF				-	0.5841

superior than J48, k-NN and NB. However, there has not been significant difference between the performance of MLP, RF and SVM. In addition, the performance of RF was also statistically higher than J48, k-NN and NB. Moreover, although SVM had the third highest mean accuracy rate, there has not been any significant difference between itself and other techniques (e.g., J48, k-NN and NB). Additionally, k-NN and NB delivered the worst performances of all techniques, from a statistical point of view.

Finally, based on the results shown in both tables (1 and 2), we can state that MLP obtained the best result, and supported by this analysis MLP will be selected to be trained over the best partition (KM_k_3). Then, this model will be used as classification model of our proposed automotive engine sensing platform.

7. Conclusion and Future Work

This work is motivated by the enormous potential of applying ML techniques over automotive engine sensor data to classify drivers' usage profiling. Based on this motivation, we proposed a Machine Learning approach which is able to use clustering for creating data labels in engine sensing data, and then to use this data for training and testing classification techniques in order to choose the most appropriated model.

In order to evaluate the feasibility of the proposed approach, an empirical analysis was conducted. In this analysis, first we applied three different clustering techniques for creating the data labels, and then, as a result, we obtained 10 best partitions. Secondly, we used these partitions to evaluate the accuracy performance of six classification techniques. Based on their accuracy rates, the best classification techniques has been chosen.

Finally, we can conclude that the main contribution of this work was the proposal and implementation of an ML based platform for driver usage profiling classification. As future work, we will collect data from different car models (size and engine power) in order to perform classification considering not only the data itself, but also the car model. In fact, a web service has been implemented in order to receive engine sensing data and classify drivers' usage profiling. Additionally, different drivers have already been invited to test the platform in their own cars. The initial results have been promising, and also the platform has performed within good standards regarding scalability and elasticity.

References

Al-Sultan, S., Al-Bayatti, A. H., and Zedan, H. (2013). Context-aware driver behavior detection system in intelligent transportation systems. *IEEE transactions on vehicular technology*, 62(9):4264–4275.

Alsmadi, I. and Alhami, I. (2015). Clustering and classification of email contents. *Journal of King Saud University - Computer and Information Sciences*, 27(1):46 – 57.

Amsalu, S. B., Homaifar, A., Afghah, F., Ramyar, S., and Kurt, A. (2015). Driver behavior modeling near intersections using support vector machines based on statistical feature extraction. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1270–1275.

Araujo, R., Igreja, A., de Castro, R., and Araujo, R. E. (2012). Driving coach: A smartphone application to evaluate driving efficient patterns. In *2012 IEEE Intelligent Vehicles Symposium*, pages 1005–1010.

Barreto, V. and Mooers, D. (2017). What is OBD II? History of On-Board Diagnostics.

Castignani, G., Derrmann, T., Frank, R., and Engel, T. G. (2015). Driver behavior profiling using smartphones: A low-cost platform for driver monitoring. *IEEE Intelligent Transportation Systems Magazine*, 7:91–102.

Eren, H., Makinist, S., Akin, E., and Yilmaz, A. (2012). Estimating driving behavior by a smartphone. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 234–239. IEEE.

Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2002). Clustering validity checking methods: part ii. *SIGMOD Rec.*, 31 (3):19–27.

Higgs, B. and Abbas, M. (2015). Segmentation and clustering of car-following behavior: Recognition of driving patterns. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):81–90.

Johnson, D. and Trivedi, M. (2011). Driving style recognition using a smartphone as a sensor platform. In *14th Int. IEEE Conf. Intelligent Transportation Systems*, page 1609–1615.

Kumagai, T. and Akamatsu, M. (2006). Prediction of human driving behavior using dynamic bayesian networks. *IEICE - Trans. Inf. Syst.*, E89-D(2):857–860.

Ly, M. V., Martin, S., and Trivedi, M. M. (2013). Driver classification and driving style recognition using inertial sensors. In *Intelligent Vehicles Symposium*, pages 1040–1045. IEEE.

Meseguer, J. E., Calafate, C. T., Cano, J. C., and Manzoni, P. (2013). Drivingstyles: A smartphone application to assess driver behavior. In *Computers and Communications (ISCC), 2013 IEEE Symposium on*, pages 535–540. IEEE.

Mitchell, T. M. Machine learning. 1997. 45(37):870–877.

Viana, E. (2018b). Smart fleet android.

Viana, E. (Jun, 2018a). Java OBD API: An OBD-II API written in Java.

World Health Organization (2015). Global status report on road safety 2015.

You, C.-W., Montes-de Oca, M., Bao, T. J., Lane, N. D., Lu, H., Cardone, G., Torresani, L., and Campbell, A. T. (2012). Carsafe: A driver safety app that detects dangerous driving behavior using dual-cameras on smartphones. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 671–672, New York, NY, USA. ACM.

Zhang, C., Patel, M., Buthpitiya, S., Lyons, K., Harrison, B., and Abowd, G. D. (2016). Driver classification based on driving behaviors. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pages 80–84. ACM.