The Journal of Supercomputing (2023) 79:18800–18819
https://doi.org/10.1007/s11227-023-05364-3

Driving behavior analysis and classification by vehicle OBD data using machine learning

Raman kumar[1] · **Anuj Jain**[1]

Accepted: 28 April 2023 / Published online: 19 May 2023

Abstract

The transportation industry's focus on improving performance and reducing costs has driven the integration of IoT and machine learning technologies. The correlation between driving style and behavior with fuel consumption and emissions has highlighted the need to classify different driver's driving patterns. In response, vehicles now come equipped with sensors that gather a wide range of operational data. The proposed technique collects critical vehicle performance data, including speed, motor RPM, paddle position, determined motor load, and over 50 other parameters through the OBD interface. The OBD-II diagnostics protocol, the primary diagnostic process used by technicians, can acquire this information via the car's communication port. OBD-II protocol is used to acquire real-time data linked to the vehicle's operation. This data are used to collect engine operation-related characteristics and assist with fault detection. The proposed method uses machine learning techniques, such as SVM, AdaBoost, and Random Forest, to classify driver's behavior based on ten categories that include fuel consumption, steering stability, velocity stability, and braking patterns. The solution offers an effective means to study driving behavior and recommend corrective actions for efficient and safe driving. The proposed model offers a classification of ten driver classes based on fuel consumption, steering stability, velocity stability, and braking patterns. This research work uses data extracted from the engine's internal sensors via the OBD-II protocol, eliminating the need for additional sensors. The collected data are used to build a model that classifies driver's behavior and can be used to provide feedback to improve driving habits. Key driving events, such as high-speed braking, rapid acceleration, deceleration, and turning, are used to characterize individual drivers. Visualization techniques, such as line plots and correlation matrices, are used to compare drivers' performance. Time-series values of the sensor data are considered in the model. The supervised learning methods are employed to compare all driver classes. SVM, AdaBoost, and Random Forest algorithms are implemented with 99%, 99%, and 100% accuracy, respectively. The suggested model offers a practical approach to examining driving behavior and suggesting necessary measures to enhance driving safety and efficiency.

Extended author information available on the last page of the article

Keywords Society of automotive engineers (SAE) · On-board diagnostic (OBD) · Internet of things (IoT) · Engine control unit (ECU) · Analysis of driving behavior (ADB) Body control unit (BCU) · Keyword protocol (KWP) · Data acquisition systems (DAS) · Controller area network (CAN) · Information and communication technology (ICT) · Society of automotive engineers (SAE)

1 Introduction

The standards in the automotive domain have been raised due to the rising number of public, private, transport, and non-transport vehicles, as well as advancements in communication systems and computing techniques. However, the increase in vehicles on the road poses new risks to humans and the environment both. Given this fact, research should be strengthened to develop more secure and ecological practices and strategies that address emerging ecological and safety threats. In this regard, analyzing and examining vehicular information received from in-vehicle sensors can provide valuable insights and knowledge [1]. This knowledge is useful for the consideration of driving behavior from an environmental and safety perspective. Since their first introduction in 1913, automobiles have advanced in standards and functional qualities. They now include various control units, sensors, and numerous subsystems. Along with them, they include on-chip communication protocols such as CAN and LIN. Data generated from these subsystems are useful for regulating bodies and motor activity as well as ensuring safety and security. Data retrieved by the sensors are continuously publicized for the operator on the dashboard. If there are issues with body control, the motor, or the safety of the vehicle, this framework is helpful for both operators and professionals [2]. This framework can be compared to a "black box" that has access to a vast amount of digital data for monitoring, supervision, and a variety of other purposes. Given the direct relationship between driving behavior and vehicle emissions and fuel consumption [3], collecting and analyzing data directly from the vehicle is essential for assessing and highlighting the driver's emissions and fuel consumption behavior. This would give the driver a better understanding of driving more sustainably and securely [4, 5]. To assess the condition of the ECU, ECM, or BCM, a driver or technician needs to obtain data from the car's onboard sensors. [6, 7, 9]. Vehicles are given a port that permits the specialists or technicians to attach compatible gadgets to recuperate live data and read fault codes. An OBD scanner, makes use of the protocol to gather information [6, 9, 14]. Asset owners are enthused about evaluating their driver's driving styles via firmly filtered data about asset use, which is affected by driving patterns. Likewise, insurance partnerships could put insurance premiums (extra expenses) based on somebody's previous driving records. In the future, the government might demand that drivers drive even more safely to obtain a license. Alternatively, the use of artificial intelligence [8] in the automotive industry and the associated autonomous vehicles for safer, cleaner, more intelligent, and more capable assets. Information about driving or vehicle operation is collected using a module that is installed in the vehicle [6, 9]. While driving, the vehicle's various parameters are monitored and recorded over time. These parameters include Fuel consumption, Absolute throttle

position, Engine speed, Current Gear, Steering wheel angle, Steering wheel speed, Calculated road gradient, Acceleration speed Longitudinal, Flywheel torque, Calculated load value, Intake air pressure, Engine coolant temperature, Activation of Air compressor, Vehicle speed, Indication of brake switch, Steering wheel angle, are captured and saved in PC as a CSV file, and driving information gathered is used for preparing the dataset[10]. The dataset was first analyzed in this paper with the help of visualization techniques based on MS Excel and Phyton library as mentioned in Sect. 5. A driving classification model based on a few selected parameters was framed. The classification was also done manually, and then the driving classification model was fitted into SVM, AdaBoost, and Random Forest [1, 2, 8, 16]. The proposed technique was tested for the accuracy of behavior classification.

2 Related works

In recent years, the analysis of driving behavior using ML techniques has become a prominent research topic, with rapid development observed over the past ten years. This trend has been fueled by the rapid advancement of computational technology and the proliferation of AI strategies. Moreover, the digitization of vehicles and their integration with ICT has enabled driving behavior research. Overall, this upward trend has been driving advancements in this field.

Nikolaos Peppes et. al. [1] The purpose of this study was to test and evaluate several renowned machine learning and deep learning approaches and develop a comprehensive proof-of-concept methodology for driver behavior analysis in light of the massive amount of vehicle data. To handle the large and more frequent data frames of vehicular information, a variety of machine learning and deep learning techniques were executed, implemented, and evaluated. The results showed that applying both machine learning and deep learning approaches to the provided dataset yielded satisfactory results. This is because calculations for both machine learning and deep learning can be carried out on the same hardware. A cloud-based architecture with flexible and scalable capabilities should be considered for this purpose. Future research should focus on exploring novel advanced approaches, as well as investigating the drawbacks, risks, and hazards that arise from the improvement of transportation systems. Additionally, researchers should concentrate on the computing efficiency and reliability of dynamic AI systems, as well as the implementation of robust network security measures.

Lisheng Jin et al. [2] Study was carried out on factors such as driving experience, psychology, and attitude significantly influence driving behavior, as their effects are difficult to measure directly. Survey techniques are commonly used to address this kind of issue. This paper presents the results of 59 driving behavior surveys, The study examined the relationship between driving behaviors and their causes, considering driving data and instructions, as well as the mechanism influencing the direction and degree of movement among variables, and merging actual categories of variables. The study's findings can help researchers investigate and understand the driving experience, thinking, and behavior, with real-world applications and

theoretical support for improving road traffic safety and reducing driver risk to other road users. It is also helpful for educating and preparing drivers.

Chen Chen et al. [3] The model for evaluating a driver's eco-driving behavior is presented in this work in light of data on vehicle operation and fuel consumption. First, the effect of three main vehicle functioning parameters speed, driving mode and fuel consumption is looked at. The connection between nine driving events and fuel consumption is then suggested. An eco-driving evaluation model is set up using PCA and multiple linear regression, and nine contrasting eco-driving recommendations are provided. The foundation of this model gives a quantitative and natural method for assessing the impact of driver's behavior on fuel utilization and provides several practical driving-event-based instructions. The results of this paper could be applied directly to the improvement of eco-driving assistance devices due to their basic yet precise calculations, laying the foundation for eco-driving advancements in mobility.

Rana Massoud et al. [4] To encourage more frugal driving, a driving profiling methodology has been developed, with the Serious Games approach focusing on user inspiration and conduct improvement. The established game method for driving profiling may also be useful for fleet managers, insurance companies, and driving schools. When non-eco-driving situations are found, the game directly proposes options to the player or driver. The player or driver can then modify their handling to cut fuel consumption and CO2 emissions. This response is the result of research into the information sources like TPS, RPM, vehicle speed, and vehicle jerk (i.e., the rate of change of speed over time). The purpose of this driving game is to identify a driver's level of aggression, which is linked to increased fuel usage. The game awards the driver a score after a drive that ranges from 0 (ineffective) to 100 (proficient, top score), classifying their natural driving style as one of three categories: "Fuel Saver," "Average," and "Indiscreet." This game mimics real-life driving behavior and can significantly impact productivity. The author believes that the game can balance the effects of driving style and other factors that affect fuel efficiency. In comparison with other drivers of similar types of vehicles, a higher score indicates that the driver is safer and more cautious when using the gas pedal.

Ward Ahmed Al-Hussein et al. [5] This is a paper on the field of sensor-based driving behavior that discusses the motivations that draw scientists to this field, common problems and challenges, and suggestions. Additionally, the paper offers research guidelines based on the authors' comprehension of the literature for future advancements, including guidelines for developing an effective DAS, suggestions for the most effective way to profile drivers according to their modes of behavior, and a suggestion for a deep learning-based algorithm that would classify drivers according to their laid-out profiles. Results have been made available for other professionals to use in future work. Furthermore, the paper makes comparisons between other driving demographics, such as male and female drivers, young and older drivers, working days and weekends, and driving during regular rush hour versus driving during Covid-19 traffic restrictions. The paper also presents the performance of a deep learning-based algorithm that classifies drivers according to their conduct.

Shi-Huang Chen et al. [6] This research presents an approach for driving behavior analysis using car onboard diagnostic (OBD) data and AdaBoost

algorithms. The suggested driving behavior research technique uses the OBD port to collect a variety of fundamental driving activity data, including the car's speed, RPM, throttle setting, and engine load. The technique then extensively uses AdaBoost methods to investigate how drivers behave. According to the test results, the proposed driving conduct investigation technique can achieve an average accuracy rate of 99.8% in a variety of driving circumstances. The proposed method, which uses the AdaBoost algorithm, achieves 100% accuracy in just 15 rounds. The proposed technique has the capability of being applied to a practical driver assistance framework.

Ashanira MatDeris et al. [7] A survey of utilization of SVM in machining modeling has been introduced. According to the research, SVM was frequently used to predict aspects of machining performance, such as uneven surfaces, instrument wear, apparatus breakage, and defect identification for both traditional and contemporary vehicles. SVM is another multi-objective modeling tool that may fulfil the requirements of the learning activity for locating sets of arrangements in light of the proper variable.

Charlyn Nayve Villavicencio et al. [8] SVM has been utilized in a few investigations in bioinformatics concerning sickness characterization. At present, the world is facing a pandemic called COVID-19 which is an infectious illness that can be moved through drops in the air from a contaminated individual. To prevent the infection from spreading and endangering human lives, those who carry it are concerned about being rapidly evaluated. However, COVID-19 testing kits were not effectively available, and research facility tests were not instantly feasible. Accordingly, the developers built a model to assess and predict the existence of COVID-19 in an individual in light of the adverse effects experienced by the individual using Support Vector Machine-regulated AI calculation. To create the most accurate model representation possible, R Studio was used to fine-tune the hyperparameters degree, cost, gamma, and components linear, radial, polynomial, and sigmoid. Tenfold cross-approval was used to evaluate the model, and the findings display that the polynomial component with improved hyperparameters produced an accuracy of 98.02%.

Raman kumar et al. [9] Experiments were completed using a multi-protocol OBD scanner, a Bluetooth module and a microcontroller-based board. In this study, OBD-II compatible hardware and software were used to capture important data relevant to the functioning of a vehicle. In this experiment parameters (such as vehicle velocity, engine speed, engine temperature, and throttle position sensor) that the driver can see on the dashboard are extracted. These parameters were collected and examined during this experiment. The findings showed that there is latency in the data received from the scanner, which is why only a few parameters were requested from the ECU. Additionally, there were more than 50 parameters that could be gathered from the ECU and can be used for a variety of purposes, including determining how a driver or operator behaves and making pertinent suggestions or recommendations to them, providing information and warnings about preventive and scheduled maintenance, and providing data that an insurance company can use to process claims. A few of these systems also used additional sensors and devices such as smartphones for implementation and analysis [12–14].

3 Data collection arrangement

According to a requirement set forth by the government, every light-duty vehicle and truck built after 1996 must have an onboard diagnostics system. The main purpose of the OBD system is to track the functioning of some key engine components, such as those in charge of regulating emissions and aiding in the detection of engine flaws. ISO presented the norm for the automotive sector to enable a consistent mechanism for communication and reporting of faults for car ECU. The ISO standard administrates communication between a vehicle's OBD port or connector and outside tools (such as diagnostic testers) for diagnostics about emissions and operations. The OBD II stacks are a collection of programming features or services that were employed by ISO 15031. In a few automotive applications, the OBD II protocol also supports the CAN bus [15].

3.1 DLC

The connector [9] mentioned is a 16-pin female-type connector as shown in Fig. 1.

This connector can be found where it's easy to access from the driver's or passenger's seat without the use of any additional tools, such as under the dash, in the ashtray, or underneath the driver's or passenger's seat.

3.2 Data frame structure

Two bytes must be written to the serial data bus to communicate with the ECU. The first byte of the message must indicate the mode, and the second byte must indicate the PID (parameter identifier). The message sent consists of four bytes of data, including the mode, PID, and corresponding data required. Common modes of data frames and the PID byte in the data frame with the necessary data are provided in Table 1.

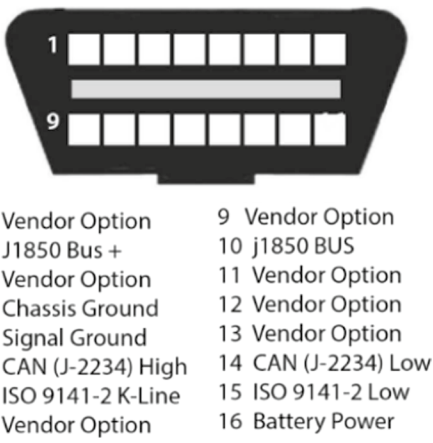

Fig. 1 Data link connector [9]

Table 1 PID and request data example

PID (hex)	Data bytes returned	Description	Units
×04	1	Engine load value	%
×05	1	Engine temperature	°C
×06	1	Short fuel trim—Bank 1	%
×07	1	Long fuel trim—Bank 1	%
×08	1	Short fuel trim—Bank 2	%
×09	1	Long fuel trim—Bank 2	%
×0A	1	Fuel pressure	kPa (gauge)
×0B	1	MAP	kPa (absolute)
×0C	2	Engine RPM	rpm
×0D	1	Vehicle speed	km/h
×0E	1	Timing advance	degree
×0F	1	Intake air temperature	°C
×10	2	MAF rate	g/s
×11	1	Throttle position	%

3.3 Response data format

The ECU response time frame consists of a mode-dependent header and four bytes of response data, as shown in Table 2.

3.4 Arrangement

The OBD-II data extraction method has two primary components: hardware and firmware. The ELM327 Bluetooth-based multi-protocol device [12] at the heart of the system is compatible with the following OBD-II protocols:

- SAE J1939 (250 kbit/s),
- SAE J1939 (500 kbit/s),
- SAE J1850 PWM,
- CAN (11-bit ID, 500 kbit/s),
- ISO 14230-4 KWP (10.4 kbit/s),
- ISO 14230-4 KWP (5 baud init),
- ISO 15765-4 CAN (11-bit ID, 250 kbit/s),
- ISO 15765-4 CAN (29-bit ID, 500 kbit/s) and
- ISO 9141-2 (10.4 kbit/s, 5 baud init)

Table 2 Response frame formats

Header	Response			
Width (configuration dependent)	Byte 1	Byte 2	Byte 3	Byte 4

The device is linked to the vehicle's OBD port to link with Bluetooth modules built on microcontroller-based boards. Its Tx and Rx pins are connected to the microcontroller board's Rx and Tx pins, respectively. Firmware consists of libraries related to ELM327 and NodeMCU [9].

4 Dataset and description

For data analysis, live vehicle data [9] were required. A dataset in the form of a CSV file was obtained [10]. The driving data were collected from the KIA Soul vehicle, which was driven by ten different drivers (A to J) on the same road trip. The trip covered a total distance of 46 km (round-trip) and had a total driving time of approximately 23 h. The dataset contains 65,535 rows of values in the time domain and almost 55 columns of values representing different parameters [5] such as:— Fuel consumption, Fuel Trim, Intake air pressure, Filtered Accelerator Pedal percentage, Absolute throttle position, Engine soaking time, Engine fuel cut off, Engine in fuel cut off, Fuel Pressure, Long Term Fuel Trim Bank, Engine speed, Engine torque after correction, Torque of friction, Flywheel torque (after torque interventions), Current spark timing, Engine coolant temperature, Engine Idle Target Speed, Accelerator Pedal value, Throttle position signal, Engine torque, Calculated LOAD value, Minimum indicated engine torque, Maximum indicated engine torque, Flywheel torque, Torque scaling factor(standardization), Standard Torque Ratio, Requested spark retard angle from TCU, TCU requested engine RPM increase, Target engine speed used in lock-up module, Glow plug control request, Air compressor ON–OFF, Torque converter speed, Current Gear, Engine coolant temperature, Wheel velocity front left-hand, rear right-hand, front right-hand and rear left-hand, Torque converter turbine speed Unfiltered, Clutch operation acknowledge Converter clutch, Gear Selection, Vehicle speed, Acceleration speed Longitudinal, Master cylinder pressure, Calculated road gradient, Acceleration speed, Steering wheel speed and Steering wheel angle Time(s), Class etc. Out of these parameters, a few of the parameters are selected for classifying driving behavior as mentioned in Table 3.

Along with above mentioned [4] paraments few derived parameters are calculated and based on which classification was done as mentioned below.

4.1 Derived parameters

4.1.1 AVG_FUEL

Average fuel consumption is directly related to driver's behavior and it negatively affects driving score. Average fuel consumption is calculated for a specific driver after taking the average of all instantaneous fuel consumption values using the formula mentioned in Eq. (1). where A=Average fuel consumption, n=Total number of dataset values in the given dataset.

Table 3 Column values (parameters)

Parameters name	Unit of measurement	Parameters description
Fuel consumption	MCC/sec	Instantaneous fuel consumption
Absolute throttle position	Percentage	The percentage to which the driver is pressing the accelerator pedal
Engine speed	RPM	Engine revolutions per minute
Current Gear	0–6	Gear engaged
Steering wheel angle	Degree	The angle of the steering wheel
Steering wheel speed	Rad./sec	Speed of rotation of steering wheel
Acceleration speed Longitudinal	M/Sec^2	Acceleration in the longitudinal direction
Calculated LOAD value	Nm	Load calculated via sensors
Engine coolant temperature	°C	Coolant temperature
Activation of Air compressor	ON–OFF	Air compressor active or not
Vehicle speed	Km/hr	Speedometer reading of the vehicle
Indication of brake switch ON/OFF	ON–OFF	Event of pressing the brake paddle

$$A = \frac{1}{n}\sum_{i=1}^{n} ai \tag{1}$$

4.1.2 IDLE_ENGINE

Idle engine donates that given engine is in idle condition. If the vehicle is not used while the engine is idle for longer durations this means that there is a wastage of fuel that's why this negatively affects the driving score i.e.,

RPM > 100 AND gear position = 0. (Idle speed of this engine was 670 to 740 RPM).

4.1.3 HIGH_SPEED_BRAKING

High-speed braking means an instance when the brake was applied at the vehicle speed of more than 55 km/h. Brakes applied at high speed negatively affect driving scores and their effectiveness is also moderate.

4.1.4 REVV_ENGINE

Revving the engine again is the event when fuel is wasted without performing any useful work. In our proposed model this is considered to be negative in driving score calculation. Revving engine instance means that the accelerator pedal is pressed when the vehicle is stationary or not in any of the gears. This was calculated as follows:—Current gear=0 and IDLE_ENGINE not TRUE.

4.1.5 DEV_STR

Following a straight route as compared to a zig-zag is considered to be a good driving pattern in the proposed model. Deviation in the steering position means how much deviation was there in steering movement for a given driver. Calculated by the formula as mentioned, where μ=Mean, σ=Standard deviation, N=Total number of data values and x_i=Each value.

$$\sigma = \sqrt{\frac{\sum (xi - \mu)}{N}} \tag{2}$$

4.1.6 VS_DEV

A steady speed of the vehicle is considered to be better than rapid acceleration and deacceleration in the proposed model. Vehicle speed deviation means how much deviation was there in vehicle speed for a given driver. The formula is mentioned in Eq. (2).

4.1.7 AVG_SPEED

Good average speed is considered to be between 50 and 70 km/h in the proposed method. The average speed of the vehicle is calculated from the vehicle speed sensor's data. The formula used is mentioned in Eq. (1).

4.1.8 AVG_GEAR

Gear shifting should be situation dependent so that it should not put much pressure on the engine and gearbox. The average gear used by the driver is calculated as the formula mentioned in Eq. (1). Its value is between 0 to 6th gear.

4.1.9 IDLE_INSTANCE

Idle instances should be as low as possible in our model and it negatively affects driving score. The number of instances when the vehicle was kept idle, using condition.

IF IDLE_ENGINE = TRUE increment IDLE_INSTANCE.

4.1.10 HB_INSTANCES

High-speed braking instances should be as low as possible and its effect is negative on the driving score. High-speed braking instances were calculated as follows.

Table 4 Assignment of driving scores

Derived parameters name	Its effect on the driving score	Its influence on the driving score
AVG_FUEL	Negative	High
IDLE_ENGINE	Negative	Low
HIGH_SPEED_BRAKING	Negative	Moderate
REVV_ENGINE	Negative	Low
DEV_STR	Negative	Moderate
VS_DEV	Negative	Moderate
AVG_SPEED	Positive	Moderate
AVG_GEAR	Positive	High
IDLE_INSTANCE	Negative	Low
HB_INSTANCES	Negative	Moderate
REV_INSTANCES	Negative	Low

Table 5 Ten driver classes based on driving data

Driver rank	Driver class
10	Poorest
9	Poor
8	Bad
7	Below average
6	Average
5	Above average
4	Good
3	Very good
2	Extremely good
1	Excellent

IF HIGH_SPEED_BRAKING = TRUE increment HB_INSTANCES.

4.1.11 REV_INSTANCES

A good driver should not unnecessarily rev the engine in our model these instances also negatively affect the score. Revving engine instances are calculated as follows.

IF REVV_ENGINE = TRUE increment REV_INSTANCES.

4.2 Fitting derived parameters to make the model for classification

For the classification of drivers, driving scores [3] are assigned based on these derived parameters that are mentioned in Table 4 along with the effect and influence on the driving score. With the help of driving score classification of drivers was done in ten classes as mentioned in Table 5.

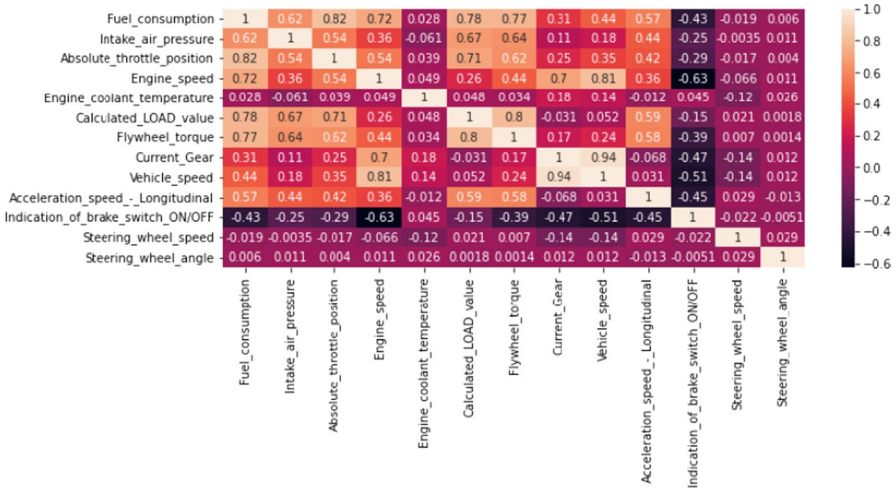

Fig. 2 Correlation matrices for different engine parameters

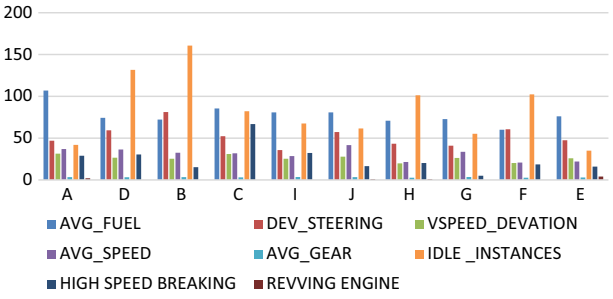

Fig. 3 Features of entrust for classification

5 Visualization of driving data

5.1 Correlation matrices for selected parameters

A correlation matrix Fig. 2 can be used to find the correlation between different parameters related to driving data for example fuel consumption is related to engine speed having a correlation value of 0.72. This gives an idea about derived parameters which can be used to assign scores to drivers and then the classification of drivers.

5.2 Features for classification

Feature classification is a graphical representation of derived paraments for different classes of drivers from Fig. 3. It can be understood that driver class A consumed maximum fuel while driver class B is having maximum idle instances. Similarly remaining parameters can provide clear visualization of data so that a model can be formed for the classification of drivers.

6 Implementation of the proposed method

The proposed model was fitted to a dataset from a CSV file, and was then put into different classification algorithms to verify the results, using a training-test ratio of 70:30. The implementation technique is mentioned in Fig. 4.

6.1 Support vector machine

A well-known supervised learning algorithm, SVM addresses both classification and regression issues [7, 8]. As the dataset taken in the proposed setup is multidimensional, linear, and time-dependent, which needs to be classified, SVM employs a subset of training points in support vectors, making SVM memory efficient. Additionally, SVM with a linear kernel was found to be suitable. In any case, it is mainly used to solve classification issues in machine learning. The goal of the SVM method is to produce the optimal line or decision boundary that can separate the classes in the n-layered space, enabling us to classify newly discovered data with certainty. A hyperplane is a restriction in this best-case scenario. The outrageous vectors and

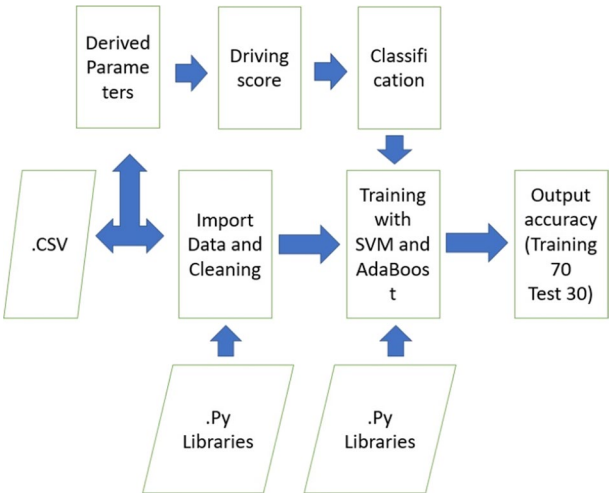

Fig. 4 Implementation technique

spots selected by SVM help create the hyperplane. These extreme points are referred to as support vectors, and the SVM technique is named after them.

Support Vector Machines for Multi-Class Problem Solving To use SVM on multi-class problems, we can create binary classifiers for each category of the data. Each classifier's two outcomes will be: The data point is either a member of that class OR it is not a member of that class. For instance, we can develop a binary classifier for each fruit in a group of fruits to conduct multi-class categorization. There will be a binary classifier to determine whether something belongs to the "Orange" class, for example, or not. The output of the SVM is the classifier with the highest score. Complex (nonlinearly separable) Fig. 5b data using SVM, SVM performs well for data that can be linearly separated without any alterations. If a set of data can be plotted on a graph and separated into classes by a straight line, it is said to be linearly separable Fig. 5a. We use kernelized SVM to separate data that cannot be separated linearly. Consider a set of data that is nonlinearly separable along a single axis. This data are linearly separable and modifiable in two dimensions. To do this, each 1-D data point is translated to a corresponding 2-D ordered pair. Therefore, for any nonlinearly separable data in any dimension, we can simply relocate the data to a higher dimension and make it linearly separable. This change is significant and has a broad impact. A kernel is nothing more than a comparison of the similarity between data points [8].

6.1.1 Implementation of SVM using scikit-learn

The dataset used in this study was taken from the car's ECU while ten drivers traveled the same 34-km path with mixed traffic. The dataset consists of 50 different parameters collected about the vehicle, all of which are available in the time domain with one sample collected per second. The data are stored in a CSV file, which is imported into Python using the read_csv method. The following methods were used for importing libraries for implementation:

"import pandas as pd
import numpy as np

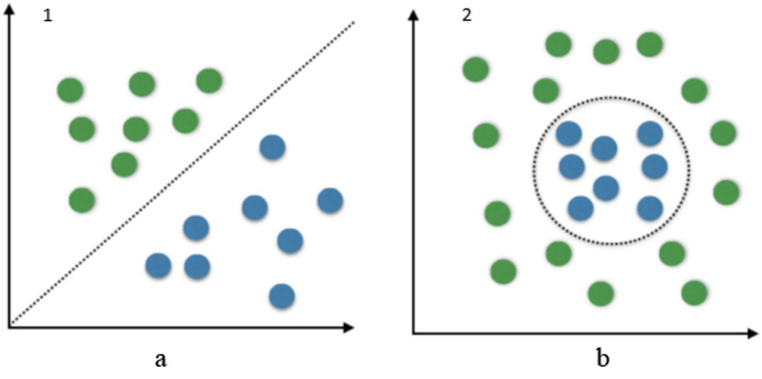

Fig. 5 **a** linear data, **b** nonlinear data [7]

import matplotlib.pyplot as plt
import seaborn as sns"

6.1.2 Training the algorithm

Training and testing datasets are available for use. To train our SVM, we will use the Scikit-Learn SVM module, which includes built-in classes for different types of SVM algorithms. Since we will perform a classification task, we will use the Support Vector Classifier (SVC) class in the Scikit-Learn SVM package, which only accepts kernel-based input. For a simple SVM that only categorizes linearly separable data, we can set the kernel type parameter to 'linear.' It is important to note that this parameter setting is only appropriate for linearly separable data. The following method were used: -

"from sklearn.model_selection import train_test_split
training_set, test_set = train_test_split(data, test_size = 0.3, random_state = 1)
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 1)cla
ssifier.fit(X_train,Y_train)
The accuracy of the model was calculated as follows: -
"from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,Y_pred)
accuracy = float(cm.diagonal().sum())/len(Y_test)
print("\nAccuracy Of SVM For The Given Dataset: ", accuracy)"

6.2 AdaBoost

AdaBoost is one of the most promising, fast, and easy-to-implement machine learning algorithms. As it does not require any earlier information about the weak learner and can be effectively executed alone as well as combined with other algorithms to track down weak hypotheses. The AdaBoost technique preserves a set of weights across training data and adaptively modifies them after each cycle of weak learning to produce a group of weak learners. Training samples are weighted more strongly than categorically correct training samples based on the categorizations that the weak learner is currently using. Maintaining a distribution or set of weights over the training set utilized for training is the primary tenet of the AdaBoost approach. The symbol $D_t(i)$ stands for the weight of this distribution on training example i on round t. The weights of samples that were incorrectly identified are changed after each round, but all weights are initially set similarly. improved so that the weak learner needs to focus on the difficult trading set examples. The frail finding is a weak hypothesis [11]h_t: X ->$\{-1,+1\}$suitable for the distribution D_t is the learner's responsibility.

($x1$, $y1$), ……., (xn, yn) where $x_i \varepsilon X$, $y_i \varepsilon Y = \{-1,+1\}$ Working as $D_1(i) = 1/M$ for $t = 1,\ldots, T$

Training for week learner for distribution D_t.find week hypothesis h_t: X ->$\{-1,+1\}$ having errors $\varepsilon_t = \Pr_{i\sim Dt}[h_t(x_i)/=y_t]$

6.2.1 Implementation of Adaboost

The dataset used for this is taken from the car's ECU [6] for a set of ten divers on the same path having a distance of 34 km, having mixed traffic. The dataset heading consists of 50 different parameters collected about the vehicle; all data is available in the time domain at the rate of one sample per second. The dataset is available in the form of a CSV file, this file is imported using the read_csv method in Python. The following methods were used for importing libraries for implementation:

"import pandas as pd
import numpy as np
import matplotlib. pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)"

6.2.2 Training the algorithm

Data sets for training and testing have been created. With this the training data to train our AdaBoost classifier. In the context of machine learning, there are commonly two types of learners: those that learn the correlations well and produce accurate predictions, and those that are lazy and produce predictions that are only marginally more accurate than chance or guesswork. Strong learners are those algorithms that fit into the first group, and weak or lazy learners are those that fit into the second. Essentially, the goal of boosting is to recover the effectiveness or performance of weak learners to make them stronger. Boosting simply involves learning from the inaccurate predictions of weak classifiers or regressors to produce a strong classifier or regressor from a series of weak classifiers or regressors. The following method were used: -

"# Import train_test_split function
from sklearn.model_selection import train_test_split
 # Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)
 # Import train_test_split function
from sklearn.model_selection import train_test_split
 # Import the AdaBoost classifier
from sklearn.ensemble import AdaBoostClassifier
 # Create an adaboost classifer object
abc = AdaBoostClassifier(n_estimators = 50, learning_rate = 2, random_state = 1)
 # Train Adaboost Classifer
model1 = abc.fit(X_train, y_train)
#Predict the response for the test dataset
#import scikit-learn metrics module for accuracy calculation
from sklearn.metrics import accuracy_score
 # calculate and print model accuracy
print("AdaBoost Classifier Model Accuracy:", accuracy_score(y_test, y_pred))"

6.3 Random Forest Classifier

A machine learning approach that can carry out both classification and regression tasks. To increase classification accuracy and combat overfitting, a Random Forest classifier generates several decision trees that are trained on multiple parts of the same training set. Random Forest generates K numbers of trees with various attributes each time, without trimming, by selecting the attributes at random. In contrast to Random Forest, where the test data are evaluated on every produced tree, the most frequent output is then allocated to that instance, in Decision Tree, the test data is tested on just one constructed tree [16]. In general, a random forest with more trees will be more robust.

6.3.1 Implementation of Random Forest Classifier

```
"from sklearn.ensemble import RandomForestClassifier
  X = data
  from sklearn.model_selection import train_test_split
  X = data.drop(columns = 'behavior', axis = 1)
  y = data['behavior']
  X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3)
  from sklearn.ensemble import RandomForestClassifier
  rf = RandomForestClassifier()
  rf.fit(X_train,y_train)
  predictions = rf.predict(X_test)f
  from sklearn.metrics import classification_report, confusion_matrix
  print("\nAccuracy For The Given Dataset: ")
  print(rf.score(X_train, y_train))"
```

7 Experimentation and results

In this paper, a model was formed with the help of some derived parameters Table 4. Which can directly or indirectly impact the driver's behavior. Based on which different classes were assigned as mentioned in Table 5, the assigned classes were then validated using machine learning algorithms. As the dataset taken in the proposed arrangement is multiple dimensional, linear and time-dependent information which is to be classified, SVM uses a subset of training points called support vectors, which makes SVM memory efficient SVM with a linear kernel was chosen as a reasonable and simple technique for classification. Another algorithm is AdaBoost. As this algorithm does not require any prior information about weak learners and can be efficiently executed alone as well as together with different algorithms to find weak hypotheses. A third method, i.e., the random forest classifier effectively handles missing data, it also addresses the problem of overfitting in decision trees and at the node's splitting point in every random forest tree, a subset of features is chosen at random. The data were pre-processed using SVM, AdaBoost, and Random Forest algorithms, and the derived

characteristics are shown in Table 4. Classification assignments are given in Table 5. The model was fitted using these algorithms, and the accuracy of driving behavior analysis was evaluated using train and test data. The accuracy of SVM and AdaBoost was 99%, while the overall accuracy of Random Forest was 100%.

8 Conclusion and future scope

Based on OBD data of vehicle operation, this research develops a classification model for driver's driving behaviors. To develop a classification model for driver behavior, over 50 vehicle-related operating parameters were recorded and analyzed, and eleven additional driving metrics or behavior-related events were calculated. Then, eleven additional driving metrics or behavior-related events were calculated. Additional or derived parameters (in Sect. 4.) were found to be very helpful and accurate for developing a behavior classification model. A model for categorizing driving behavior was created and verified using SVM, ADAboost and Random Forest. Ten classes of corresponding driving behaviors were offered. The behavior classification model created by this study offers a systematic and logical way to assess how a driver's actions can have a direct or indirect impact. The proposed methods were straightforward and accurate, with accuracy rates of 99%, 99%, and 100% for SVM, ADAboost, and Random Forest, respectively. This method is yet useful and can be helpful for traffic police, insurance companies, and for the processing of remote claims, the results of this work might be directly applied to the creation of driving support and classification devices. The major goal of this study was to demonstrate and evaluate the functionality of a real-world platform that had been implemented and was receiving real data streams from operating cars that belonged to a person, maintenance personnel, or any other type of administration. In this research work yet, modern tools and algorithms are used but still, there is large scope for implementation of other modern software tools and algorithms according to the futuristic requirements from time to time. The presented method is not limited to internal combustion engine-based vehicles and can also be incorporated into hybrid and electric vehicles in the future. The suggested solution is practical and open to new technologies. The outcome of the proposed method is affected by data acquisition and storage delays, due to this behavior classification cannot be done for small road trips (30 km or less) especially if the route or vehicle is different. Driver behavior analysis remains an active area of study, even as autonomous vehicles gain momentum.

Author contributions Raman Kumar A, Anuj Jain Ba wrote the main manuscript textb reviewed the manuscript.

Declarations

Conflict of interest The authors declare no competing interest.

References

1. Peppes N, Alexakis T, Adamopoulou E, Demestichas K (2021) Driving behaviour analysis using machine and deep learning methods for continuous streams of vehicular data. Sensors 21(14):4704
2. Jin L, Guo B, Jiang Y, Hua Q (2021) Analysis on the influencing factors of driving behaviours based on the theory of planned behaviour. Adv Civil Eng. https://doi.org/10.1155/2021/6687674
3. Chen C, Zhao X, Yao Y, Zhang Y, Rong J, Liu X (2018) Driver's eco-driving behaviour evaluation modeling based on driving events. J Adv Transp. https://doi.org/10.1155/2018/9530470
4. Massoud R, Bellotti F, Berta R, De Gloria A, Poslad S (2019) Eco-driving profiling and behavioral shifts using iot vehicular sensors combined with serious games. In: 2019 IEEE Conference on Games (CoG), pp. 1–8. IEEE
5. Al-Hussein WA, Kiah MLM, Yee L, Zaidan BB (2021) A systematic review on sensor-based driver behaviour studies: coherent taxonomy, motivations, challenges, recommendations, substantial analysis and future directions. PeerJ Comput Sci 7:e632
6. Chen, S-H, Pan J-S, Lu K (2015) Driving behavior analysis based on vehicle OBD information and adaboost algorithms. In: Proceedings of the International Multiconference of Engineers and Computer Scientists. 1:18-20
7. Deris AM, Zain AM, Sallehuddin R (2011) Overview of support vector machine in modeling machining performances. Procedia Eng 24:308–312
8. Villavicencio N, Charlyn JHJ, Hsieh J-G (2021) Support vector machine modelling for COVID-19 prediction based on symptoms using R programming language. In 2021 The 4th International Conference on Machine Learning and Machine Intelligence, pp. 65–70. 2021. https://doi.org/10.1145/3490725.3490735.
9. Kumar R, Jain A (2022) Monitoring and remote data logging of engine operation via on board diagnostic port. In: 2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT), pp. 550–555. IEEE
10. Il BK, Woo JY, Kim HK (2016) Know your master: driver profiling-based anti-theft method. In: 2016 14th Annual Conference on Privacy, Security and Trust (PST), pp. 211–218. IEEE
11. Wang R (2012) AdaBoost for feature selection, classification and its relation with SVM, a review. Phys Procedia 25:800–807
12. Chan TK, Chin CS, Chen H, Zhong X (2020) A comprehensive review of driver behavior analysis utilizing smartphones. IEEE Trans Intell Transp Syst 21(10):4444–4475. https://doi.org/10.1109/TITS.2019.2940481
13. Kashevnik A, Lashkov I, Gurtov A (2020) Methodology and mobile application for driver behavior analysis and accident prevention. IEEE Trans Intell Transp Syst 21(6):2427–2436. https://doi.org/10.1109/TITS.2019.2918328
14. Fabio M, Marulli F, Mercaldo F, Santone A (2021) Neural networks for driver behavior analysis. Electronics 10(3):342. https://doi.org/10.3390/electronics10030342
15. Uvarov K, Ponomarev A (2021) Driver identification with OBD-II public data. In: 2021 28th Conference of Open Innovations Association (FRUCT), pp 495–501. IEEE
16. Ahmed NS, Sadiq MH (2018) Clarify of the random forest algorithm in an educational field. In: 2018 International Conference on Advanced Science and Engineering (ICOASE), Duhok, Iraq. pp 179–184. https://doi.org/10.1109/ICOASE.2018.8548804

Authors and Affiliations

Raman kumar[1] · Anuj Jain[1]

✉ Raman kumar
ramankanythia@gmail.com

Anuj Jain
a1978jain@gmail.com

[1] Lovely Professional University, Phagwara, India