PROJECT REPORT

Date	16 November 2022
Team ID	PNT2022TMID15686
Project Name	Project – Machine Learning based Vehicle Performance Analyzer
Team Leader	Dhinakaran R
Team Members	Ashok B Dhanush M Elankathir S

1. INTRODUCTION:

The rapidly expanding discipline of data science includes machine learning as a key element. Algorithms are trained to generate classifications or predictions using statistical techniques, revealing important insights in data mining operations. The decisions made as a result of these insights influence key growth indicators in applications and enterprises, ideally. Data scientists will be more in demand as big data develops and grows because they will be needed to help identify the most important business issues and then the data to answer them. Machine learning is a subfield of artificial intelligence (AI) and computer science that focuses on using data and algorithms to mimic human learning processes and progressively increase accuracy.

1.1 PROJECT OVERVIEW

This project implies applied data science concepts and concepts of machine learning to the dataset that is collected as a part of the preparation phase to analyze and predict vehicle performance using a machine learning providing high accuracy and effficiency. In this project mulitple regression models are trained on the obtained dataset and checked for accuracy and tested for the suitability of the problem. Among different models that were trained decision tree regression have proven to be most effective with higher accuracy. Hence this model is saved for future predictions on vehicle performance given by the user which is obtained using a web application. This machine learning model is integrated with the web application with the help of a flask app. The entire application is also deployed on IBM cloud as a part of this project. On the other hand the project planning and management is implemented on Agile methodology where after each and every phase testing process is done. Here we perform performance and user acceptance testing after each sprint to ensure that the project stays on track and achieving its objectives on time.

1.2 PURPOSE

The main purpose of this project is to enable users to analyze their vehicle condition based on its performance. Predicting a car's performance level is a significant and intriguing challenge. Predicting a car's performance in order to change a certain behaviour of the vehicle is the major objective of the current study. This can drastically reduce the fuel consumption of the system and boost efficiency. Analysis of the car's performance based on the horsepower, fuel type, engine type, and the number of cylinders. These are the variables that can be used to forecast the condition of the vehicle. The process of gathering, investigating, interpreting, and documenting health data based on the aforementioned three elements is ongoing. For the prediction engine and engine management system, performance goals like mileage, dependability, flexibility, and cost can be integrated together and are crucial.

This strategy is a crucial first step in comprehending how a vehicle performs. To increase the performance efficiency of the vehicle, it is crucial to examine the elements utilising a variety of well-known machine learning methodologies, including as linear regression, decision trees, and random forests. Automobile engineering's "hot subjects" right now revolve around the power, longevity, and range of automotive traction batteries. We also take a performance in mileage into account here. We will create the models, utilising various techniques and neural networks, to resolve this issue. Then, we'll compare which algorithm accurately forecasts car performance (Mileage).

2. LITERATURE SURVEY

Here are some of the literature survey done to analyse different exiting models and to identify the right data and the machine learning model to approach the problem.

[1] An approach of modeling on dynamic performance evaluation for off-road vehicle

Authors:

Junshu Han

Institute of Medical Equipment, Academy of Military Medical Sciences, Tianjin, China

Zhenhai Gao

Institute of Medical Equipment, Academy of Military Medical Sciences, Tianjin, China

Shulin Tan

Institute of Medical Equipment, Academy of Military Medical Sciences, Tianjin, China

Xiangdong Cui

Institute of Medical Equipment, Academy of Military Medical Sciences, Tianjin, China

Published in: 2010 8th World Congress on Intelligent Control and Automation

Abstract:

Automobile dynamic is one of the most important performance indexes, the key is whether the simulation accords with the real status, whether the vehicle performance under real driving conditions can be reflected more validly, and it

will be used to estimate automobile dynamic accurately, or to provide theory reference for vehicle design. On the point of view of using vehicle, considering the effects of external factors as air velocity, tyre slip, adhesion coefficient on vehicle dynamic performance, a novel method on dynamic performance evaluation is established, and the effectiveness of the dynamic performance model is vertified.

[2] Steering performance simulation of three-axle vehicle with multi-axle dynamic steering

Authors:

Shufeng Wang

College of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China

Junyou Zhang

College of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China

Huashi Li

College of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China

Published in: 2008 IEEE Vehicle Power and Propulsion Conference

Abstract:

Because three-axle heavy-vehicle with front-wheel steering has big radius at low speed and bad stability at high speed, in order to improve heavy vehicle steering performance at different speed, the multi-axle dynamic steering technology is put forward. Selecting zero side-slip angle of mass centre and proportional control strategy to control vehicle, Using MATLAB, the steering performance of the

three-axle vehicle with different steering modes are simulated. The result shows that multi-axle dynamic steering can decrease the steering radius at low speed and improve vehicle stability at high speed.

[3] Simulation study on synthetical performance of electric vehicles

Authors:

Sun Fengchun

Electric Vehicle Research and Development Center, Beijing Institute of Technology, Beijing, China

Sun Liqiug

Zhu Jiaguang

Electric Vehicle Research and Development Center, Beijing Institute of Technology, Beijing, China

Published in: Proceedings of the IEEE International Vehicle Electronics Conference (IVEC'99) (Cat. No.99EX257)

Abstract:

This paper presents the software development on the performance simulation of electric vehicles. Software verification is carried out via the comparison of simulation results with on-road test. Applications of the software in prototype design are also presented in terms of theoretical inference, modeling, software development and simulation of synthetical performance for EVs such as dynamic performance, economy performance as well as analysis of parameters' influences on EV performance. The commonly used European drive cycle is adopted for simulation in the paper. Simulation with the software proves an efficient and money-saving means for prototyping of EV or HEV systems with control units.

[4] Simulation and Analysis of Performance of a Pure Electric Vehicle with a Super-capacitor

Authors:

N Jinrui

School of mechanical and vehicular engineering, Beijing Institute of Technology, Beijing, China

W Zhif

School of mechanical and vehicular engineering, Beijing Institute of Technology, Beijing, China

Published in: 2006 IEEE Vehicle Power and Propulsion Conference

Abstract:

Energy storage and power boost are major problems in the development of electric vehicles (EV). Installing a super-capacitor as an auxiliary power source to improve the performance of electric vehicles is a feasible and realistic solution. In this paper, the structure of a multi-energy system and the principles of flow of the multi energy of electric vehicles were introduced first, explaining how different sources of energy work in different situations. A model of electric vehicle with a battery and a super- capacitor, based on Matlab/Simulink was built up. The model was validated by comparing the simulation results and the actual data from later field tests. The drive cycle used in the simulation was the CYC_CONST_45. Comparisons were made between the model of the vehicle with a super- capacitor and the model of a vehicle without a super-capacitor. Field tests of an electric vehicle were conduct and analyses were made. The analysis includes vehicle dynamic performance and economical performance in urban environments where the vehicle accelerated and decelerated frequently. The results showed that installing a super-capacitor improves the working conditions of the battery. The variation of the current drawn by the vehicle was smoothed due to the working of the super-capacitor, which provided better working conditions for the battery and increased the operating life of the battery.

[5] Steering feel study on the performance of EPS

Authors:

Xin. Zhang

School of Mechanical, Electric and Control Engineering, Beijing Jiaotong University, Beijing, China

Zhang Xin

School of Mechanical, Electric and Control Engineering, Beijing Jiaotong University, Beijing, China

Shi Guobiao

Electric Vehicle Center of Analysis and Technology, Beijing Institute of Technology, Beijing, China

Published in: 2008 IEEE Vehicle Power and Propulsion Conference

Abstract:

The steering feel study is very important in the development of electric power steering system (EPS). This paper describes a method about how to evaluate and get the suitable steering feel when driving a vehicle equipped with EPS. The EPS steering feel subjective tests were performed to obtain objective quality parameters that correlate with subjective evaluation. Afterthis, the paper briefly describes the statistical technique used to identify which parameters best correlate with vehicle steering qualities. As there was no correlation between a single partial rating and a single objective indicator, the principal component analysis (PCA) method was chosen and obtained objective indices. The objective evaluation parameters have been validated by drivers psila subjective evaluation. In the third part, the analytical method was applied to vehicle dynamic analysis to analyze vehicle steering feel characteristics, we established a closed-loop steering feel simulation model to analyze steering torque characteristics, vehicle dynamic response and assess steering feel performance for different settings of a

EPS system. The design of EPS was optimized and achieved more suitable driving feel by using the dynamic analysis model without plenty of real vehicle tests. This method make it possible to easily and accurately benchmark steering dynamic characteristics, set design targets, and is helpful to achieve good steering feel.

[6] Study on the performance and control of SR machine for vehicle regenerative braking

Authors:

Xiaoling Yuan

College of Electrical Engineering, Hohai University, HHU, Nanjing, Jiangsu, China

Yimin Gao

Department of Electrical Engineering, Texas A and M University, College Station, TX, USA

M. Ehsani

Department of Electrical Engineering, Texas A and M University, College Station, TX, USA8

Published in: 2008 IEEE Vehicle Power and Propulsion Conference

Abstract:

A regenerative braking system with simple structure, high efficiency, good performance and easy control is crucial for electric vehicle (EV), hybrid electric vehicle (HEV) and fuel cell vehicle (FCV). SR machine is one of the promising candidates. In this paper, the current and torque performance of a SR machine for application to vehicle regenerative braking has been studied. The relationship between the torque, speed, turn-on and turn-off angles has been established. The data obtained through simulation is very useful for vehicle control design.

[7] A method to analyze driver influence on the energy consumption and power needs of electric vehicles

Authors:

Rayad Kubaisi

Chair of Vehicle Technology, Karlsruhe, Germany Institute of Vehicle System Technology,

Frank Gauterin

Chair of Vehicle Technology, Karlsruhe, Germany Institute of Vehicle System Technology,

Martin Giessler

Chair of Vehicle Technology, Karlsruhe, German

Published in: 2009 IEEE Intelligent Symposium Institute of Vehicle System Technology,

Abstract:

The energy consumption and power needs of electric vehicles are evaluated on roller test benches according to test procedures defined by legal standards and by vehicle manufacturers. These test procedures are mainly defined by driving cycles and include tolerances to compensate for the human error during these tests. These tolerances may seem to make the tests easier but they can have a big effect on the appropriate dimensioning of the components, and also on the performance of the vehicle. Within this paper, a method is presented, which enables the quantification of these effects depending on the type of the test procedure, and the way the driving cycle is driven. The developed method has been tested in a simulation environment and several standard test procedures were analyzed.

[8] Transmission system performance analysis of traditional power vehicle

Authors:

Feng Kang

Research Center of Advanced Powertrain Technology, State Key Laboratory of Advanced D&M for Vehicle Body, Hunan University, Changsha, China

Liu Jingping

Research Center of Advanced Powertrain Technology, State Key Laboratory of Advanced D&M for Vehicle Body, Hunan University, Changsha, China

Fu Jianqin

Research Center of Advanced Powertrain Technology, State Key Laboratory of Advanced D&M for Vehicle Body, Hunan University, Changsha, China

Yang Hanqian

Research Center of Advanced Powertrain Technology, State Key Laboratory of Advanced D&M for Vehicle Body, Hunan University, Changsha, China

Published in: 2011 International Conference on Electric Information and Control Engineering

Abstract:

Based on simulation software GT-drive, the author analyzed the transmission system performance of a passenger car with diesel engine and provided the appropriate research methods. Firstly, the numerical simulation model of a vehicle was built based on vehicle weight, frontal area, rolling, air-drag coefficient, etc. The different matching schemes were simulated and compared. The results show that, for a given engine, using different transmission systems, the matching efficiency is significantly different. In view of power and economy of the vehicle, it is important that selected suitable power transmission device. This method has provided a theoretical basis for studying traditional power

vehicle, also giving some information to study the new type vehicle power train system.

[9] Real-time performance of control allocation for actuator coordination in heavy vehicles

Authors:

Kristoffer Tagesson

Department Chassis Strategies & Vehicle Analysis, VOLVO 3PDepartment 26661, AB4S GOTEBORG, Sweden

Peter Sundstrom

MaDELON AB, Lund, Sweden

Leo Laine

Department Chassis Strategies & Vehicle Analysis, VOLVO 3PDepartment

26661, AB4S GOTEBORG, Sweden

Published in: 2009 IEEE Intelligent Vehicles Symposium

Abstract:

This paper shows how real-time optimisation for actuator coordination, known as control allocation, can be a viable choice for heavy vehicle motion control systems. For this purpose, a basic stability control system implementing the method is presented. The real-time performance of two different control allocation solvers is evaluated and the use of dynamic weighting is analysed. Results show that sufficient vehicle stability can be achieved when using control allocation for actuator coordination in heavy vehicle stability control. Furthermore, real-time simulations indicate that the optimisation can be performed with the computational capacity of today's standard electronic control units

2.1 EXISTING PROBLEM

The main problem in predicting machine learning is that different model suits for different situations but identifying which model suits overall with only some loss is crucial. So we need to train different regression models on the given data and identify which model suits our particular scenario and dataset so that it could be carried on for predicting purposes.

2.2 REFERENCES:

Byerly, A., Hendrix, B., Bagwe, R., Santos, E. and Ben-Miled, Z. (2019). A machine learning model for average fuel consumption in heavy vehicles, IEEE Transactions on Vehicular Technology, 68(7), 6343-6351, doi: 10.1109/TVT.2019.2916299 .

C¸ apraz, A. G., Ozel, P., S¸evkli, M. and Beyca, ¨ O. F. (2016). Fuel Consumption Models ¨Applied to Automobiles Using Real-time Data: A Comparison of Statistical Models, 83, pp. 774-781, doi: 10.1016/j.procs.2016.04.166.

Cortes, C. and Vapnik, V. (1995). Support-vector networks, Machine learning, 20(3), pp.273-297 . Fayyad, U. M., Haussler, D. and Stolorz, P. E. (1996). Kdd for science data analysis: Issues and examples., KDD pp. 50-56.

Freedman, D. A. (2009). Statistical models: theory and practice, cambridge university press. Fugiglando, U., Massaro, E., Santi, P., Milardo, S., Abida, K., Stahlmann, R., Netter, F. and Ratti, C. (2019). Driving behavior analysis through can bus data in an uncontrolled environment, IEEE Transactions on Intelligent Transportation Systems. 20(2), pp. 737- 748, doi: 10.1109/TITS.2018.2836308 .

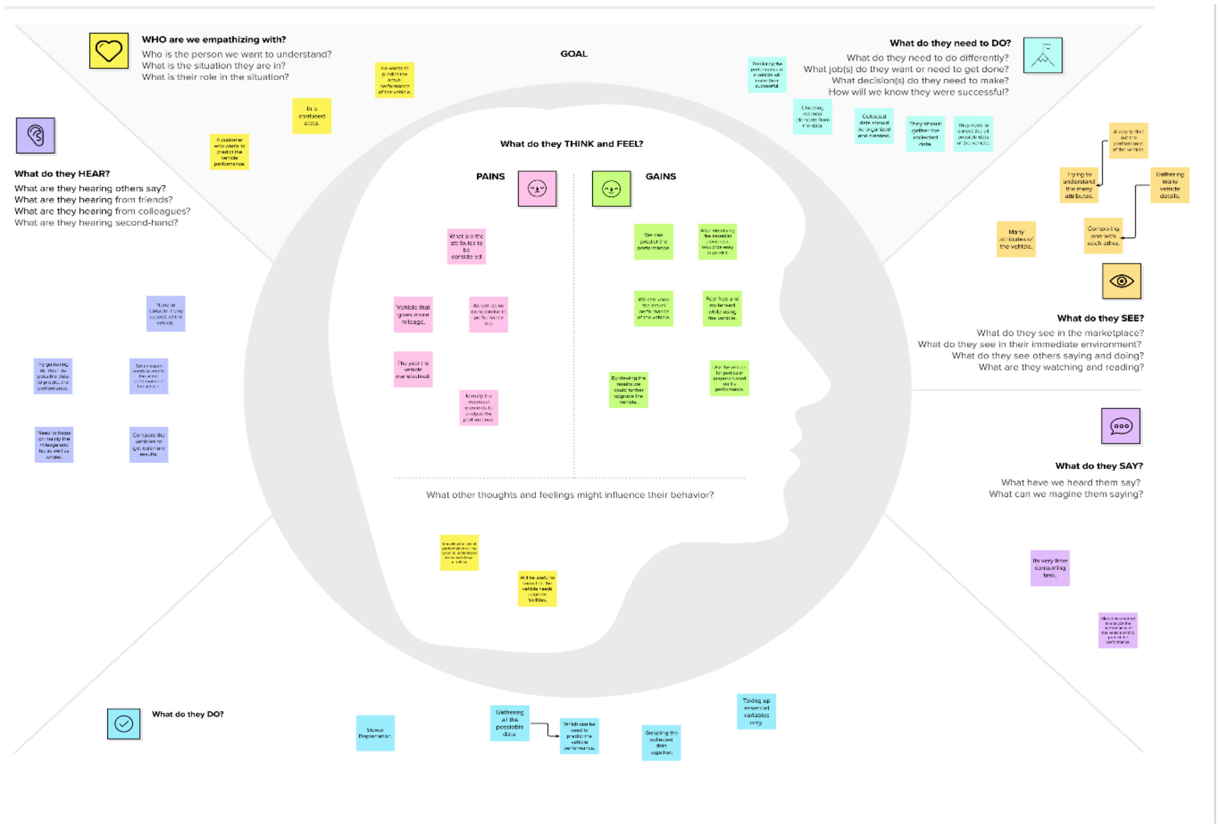
Gilman, E., Keskinarkaus, A., Tamminen, S., Pirttikangas, S., R¨oning, J. and Riekki, J. (2015). Personalised assistance for fuel-efficient driving, Transportation Research Part C: Emerging Technologies, 58, pp. 681-705 doi: 10.1016/j.trc.2015.02.007 .

2.3 PROBLEM STATEMENT DEFNITION

I am	**I am trying to**	**But**	**Because**	**Which makes me feel**
Owner of a vehicle.	Owner of a vehicle.	I don't know which aspect tofocus more.	A very large of factors impact performance.	Confused
Owner of a vehicle.	Analyze the performance of vehicle.	Can't focus on every single car.	There are a variety of cars at different prices	Little worried

3. IDEATION AND PROPOSED SOLUTION:

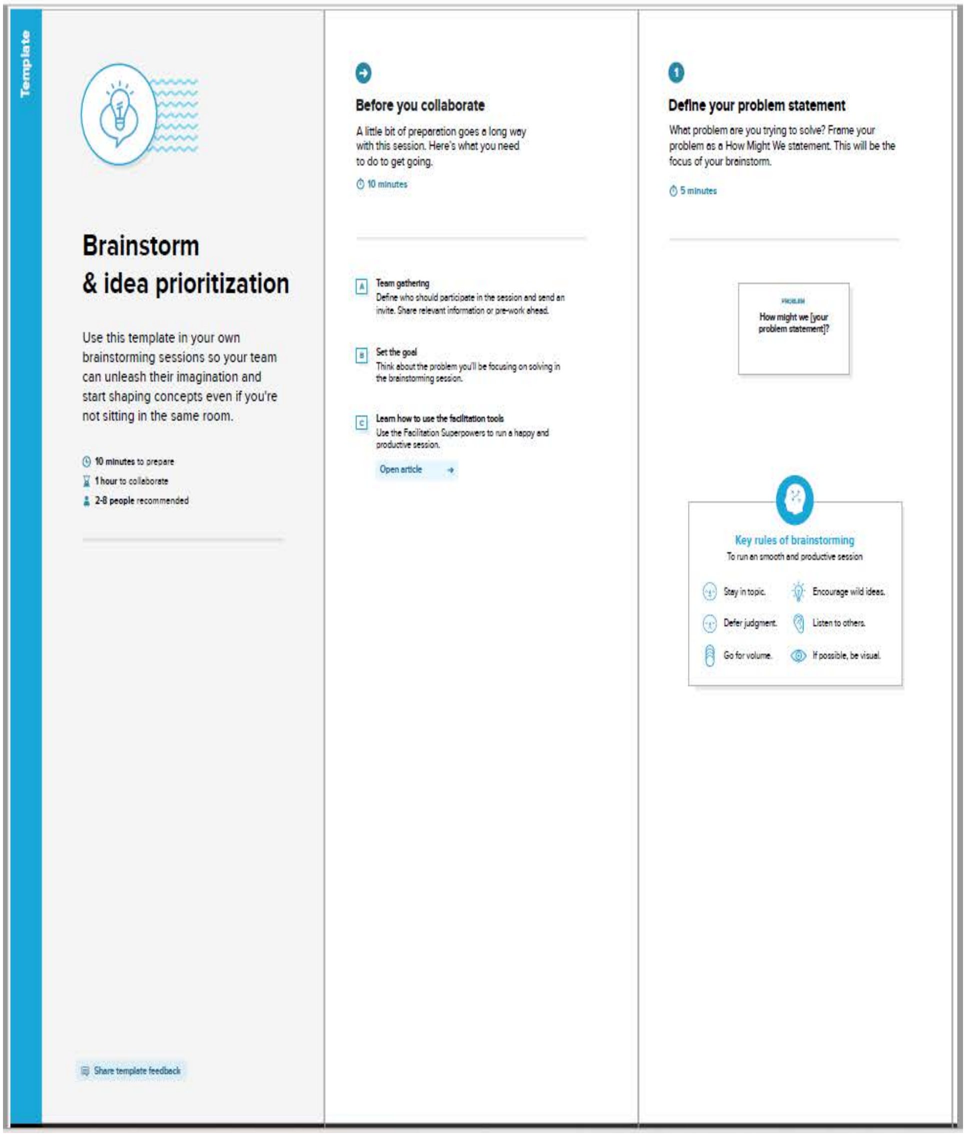
3.1 EMPATHY MAP CANVAS

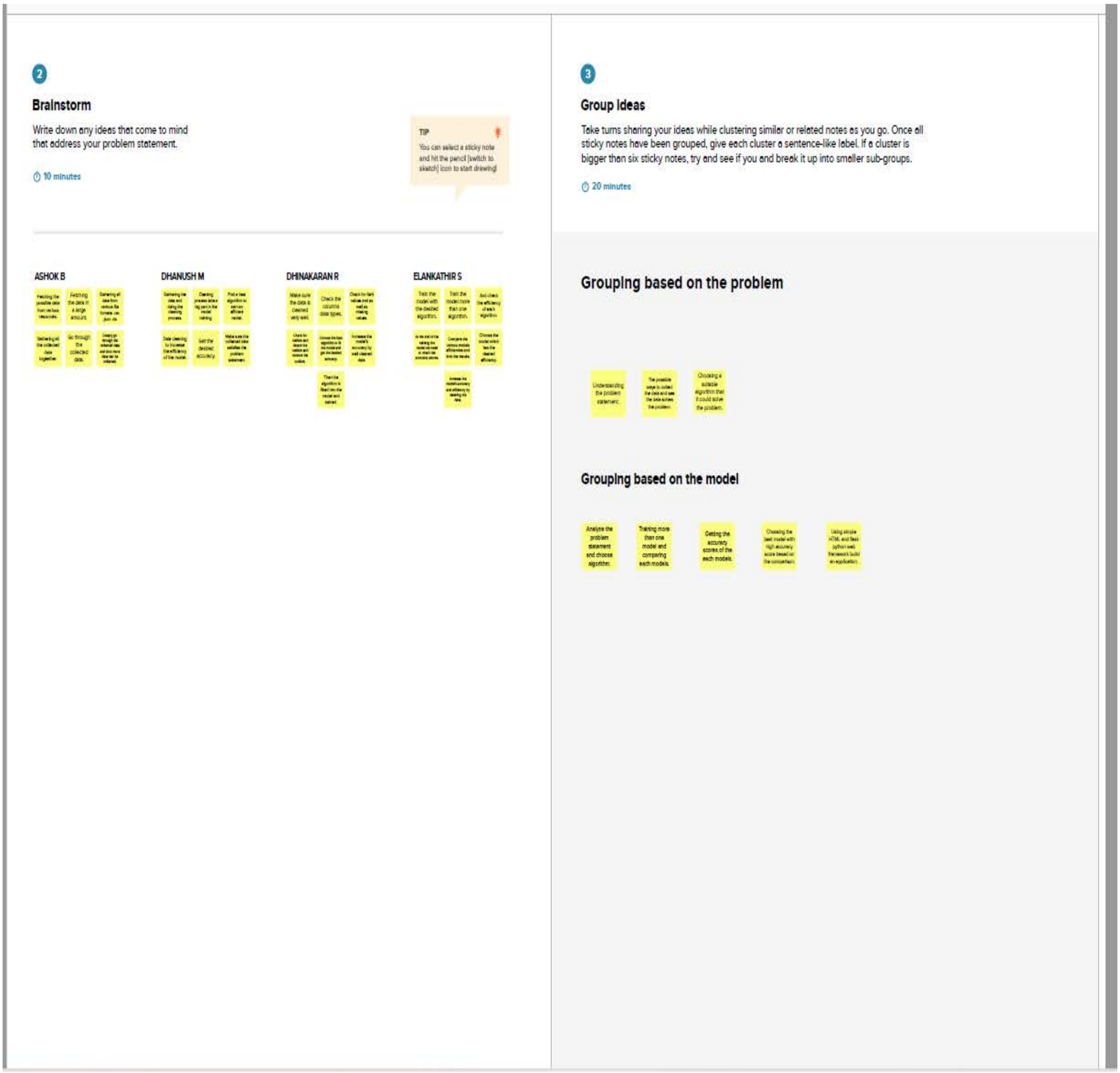


3.2 IDEATION & BRAINSTORMING:

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Step-2: Brainstorm, Idea Listing and Grouping





2
Brainstorm
Write down any ideas that come to mind
that address your problem statement.
10 minutes
TIP
You can select a sticky note
and hit the pencil [switch to
sketch] icon to start drawing!
ASHOK B
Fetching
the data in
a large
amount.
Go through
the
collected
data.
DHANUSH M
Get the
desired
accuracy.
DHINAKARAN R
Make sure
the data is
cleaned
very well.
Check the
column
data types.
ELANKATHIR S
Train the
model with
the desired
algorithm.
Train the
model more
than one
algorithm.
3
Group Ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all
sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is
bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.
20 minutes
Grouping based on the problem
Understanding
the problem
statement.
Choosing a
suitable
algorithm that
it could solve
the problem.
Grouping based on the model
Analyze the
problem
statement
and choose
algorithm.
Training more
than one
model and
comparing
each models.
Getting the
accuracy
scores of the
each models.

3

Brainstorm as a group

Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and..." and build on the ideas of other people along the way.

15 minutes

TIP

You can use the **Voting session** tool above to focus on the strongest ideas.

By increasing the frequency of the vehicle monitoring the performance of the vehicle can be analyzed and measures can be taken.

By enabling the vehicle to adapt to different regions, the high differentiation of vehicle performance can be stabilized.

By selecting fuel that works best in terms of chemical composition can increase acceleration and also reliablity

By implementing a light weight body that supports better aerodynamics can increase performance efficiency

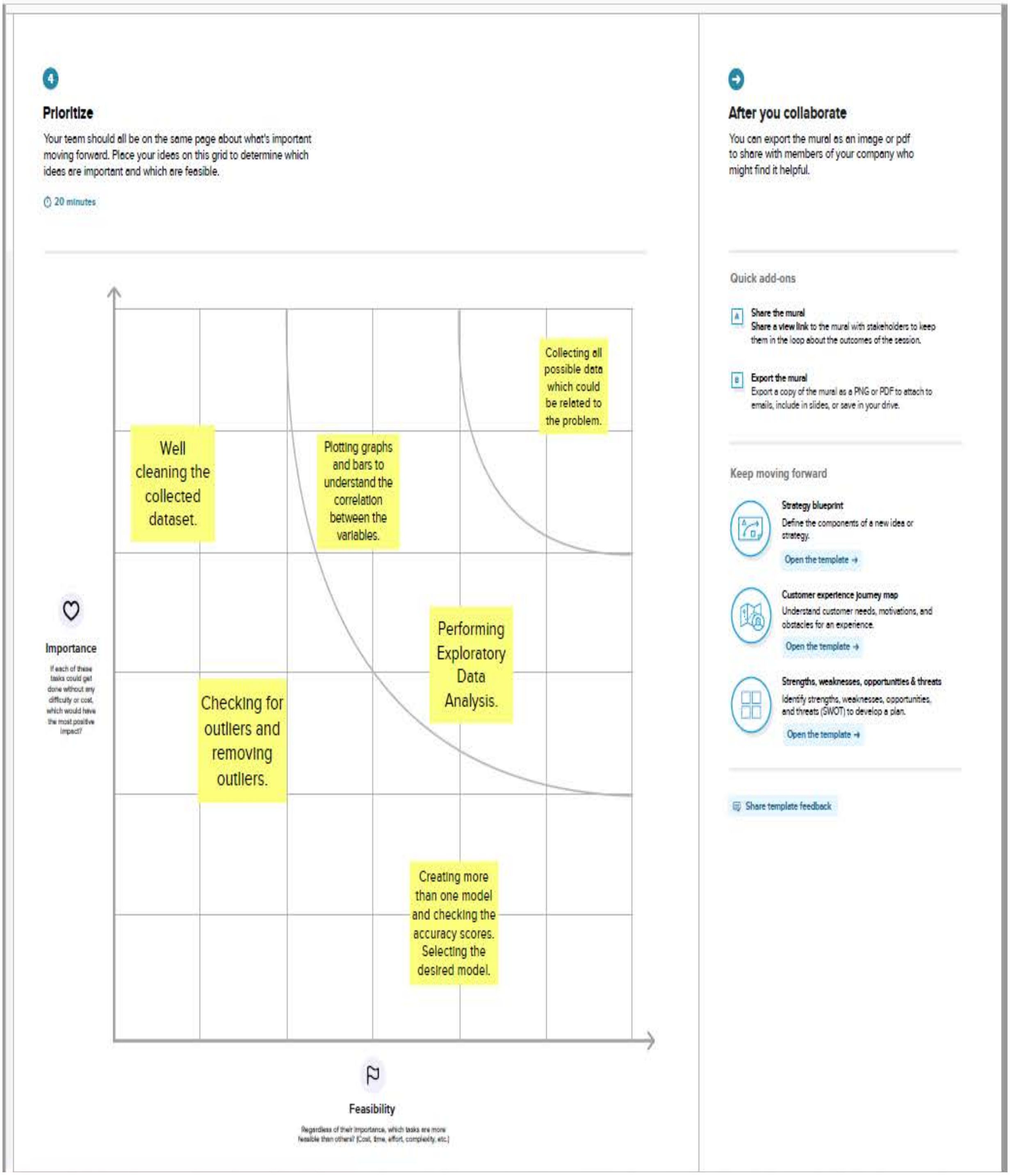
By using constant customer reviews we can identify the key area in which the customer has the problem with and can be resolved.

By adapting dust and heat resisting technologies, the performance of the ev's in high temperature zones can be increased.

By making an analysis model that into account different independent variables and providing accurate analysis

By learning the customer requirements we can tweak the vehicle according to the customer needs

Step-3: Idea Prioritization

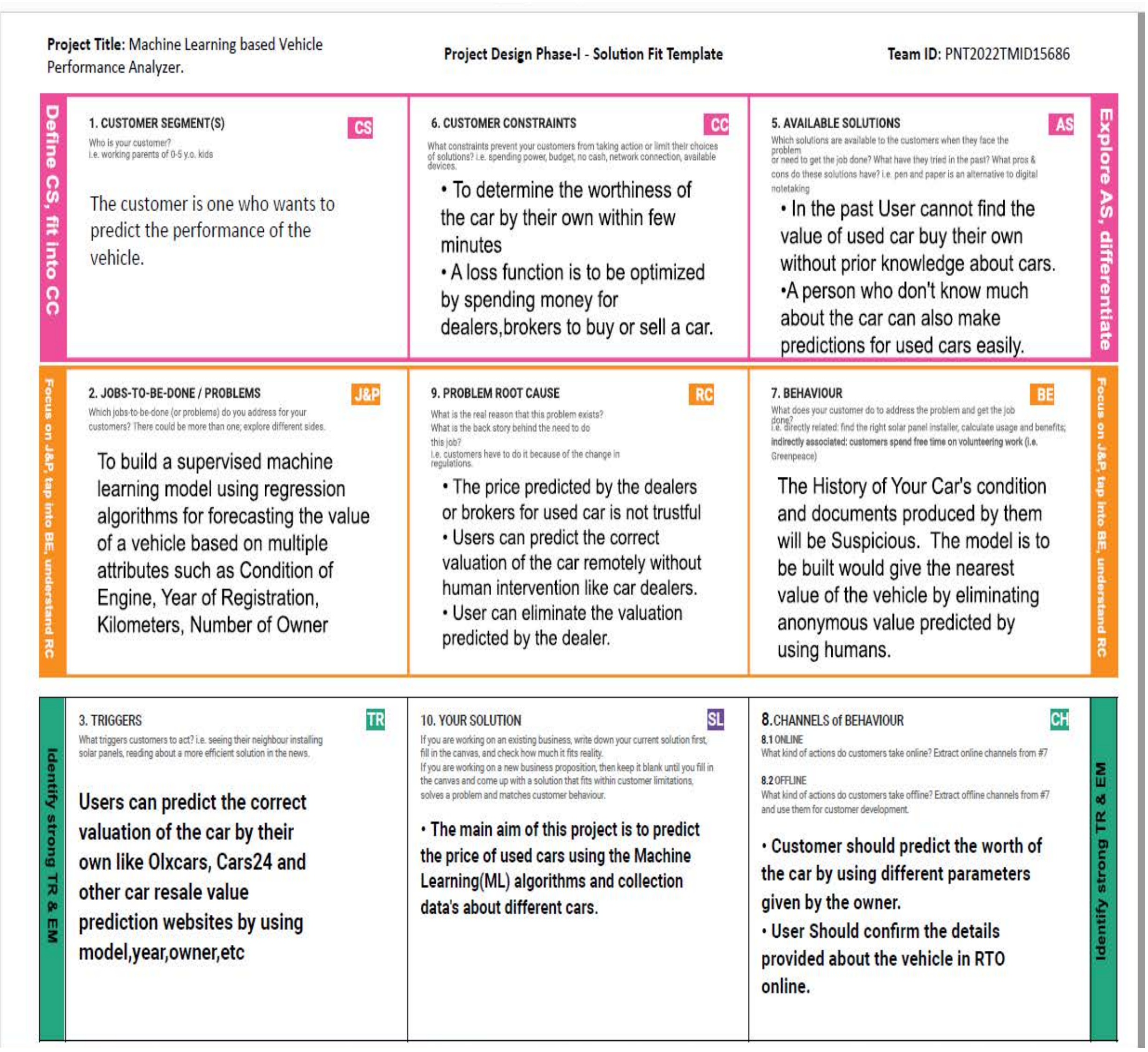


3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The main objective of this project is to enable user to analyse their vehicle performance without external assistance quickly and accurately using little to no time. This performance can then be used to estimate the price of the car which can be used for both buying and selling a customer's vehicle.
2.	Idea / Solution description	The solution includes identifying a machine learning algorithm in this case a model is integrated with a web application interface which help user to analyse vehicle performance anywhere and anytime using just the internet
3.	Novelty / Uniqueness	By selecting the optimal regression model that suits the application, the time that is taken to analyse performance is significantly reduced and the amount of effort put in by the user is also reduced significantly.
4.	Social Impact / Customer Satisfaction	By using this application power using a machine learning model, time and cost spend in analysing a vehicle's

		performance is reduced, and the process of setting a price for buying and selling second hand used vehicles becomes a straight forward process with customer satisfaction increasing on both the parties of transaction
5.	Business Model (Revenue Model)	The application can use two revenue model, subscription based in which the user pays a small amount of fee every month to get services or pay to use where customer's pay significantly less for each prediction according to their need.
6.	Scalability of the Solution	The project is highly scalable, as people of all ages are customer segments as almost everyone owns vehicle in today's world. Also the change in vehicle specification is very low from region to region, so this helps scaling the project quite easy

3.4 PROBLEM SOLUTION FIT:



Project Title: Machine Learning based Vehicle Performance Analyzer.

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID15686

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The customer is one who wants to predict the performance of the vehicle.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

- To determine the worthiness of the car by their own within few minutes
- A loss function is to be optimized by spending money for dealers,brokers to buy or sell a car.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

- In the past User cannot find the value of used car buy their own without prior knowledge about cars.
- A person who don't know much about the car can also make predictions for used cars easily.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

To build a supervised machine learning model using regression algorithms for forecasting the value of a vehicle based on multiple attributes such as Condition of Engine, Year of Registration, Kilometers, Number of Owner

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

- The price predicted by the dealers or brokers for used car is not trustful
- Users can predict the correct valuation of the car remotely without human intervention like car dealers.
- User can eliminate the valuation predicted by the dealer.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

The History of Your Car's condition and documents produced by them will be Suspicious. The model is to be built would give the nearest value of the vehicle by eliminating anonymous value predicted by using humans.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

Users can predict the correct valuation of the car by their own like Olxcars, Cars24 and other car resale value prediction websites by using model,year,owner,etc

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

- The main aim of this project is to predict the price of used cars using the Machine Learning(ML) algorithms and collection data's about different cars.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

- Customer should predict the worth of the car by using different parameters given by the owner.
- User Should confirm the details provided about the vehicle in RTO online.

Identify strong TR & EM

4. REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL REQUIREMENTS:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP.
FR-3	Add Vehicles to Account	Register vehicles through registration number and identify the model.
FR-4	Get attributes of vehicles as inputs	Use a form to obtain input.
FR-5	Predict the performance of the vehicle	Use techniques such as dimensionality reduction and feature selection to pre process data and handle missing values.
FR-6	Tracking the vehicle performance	Use a machine learning model to predict the performance of the vehicle
FR-7	Tracking the vehicle performance	Use storage techniques to store predicted performance from time to time and track them
FR-8	Interact with the user	Use a web application interface developed using HTML to interact with user.

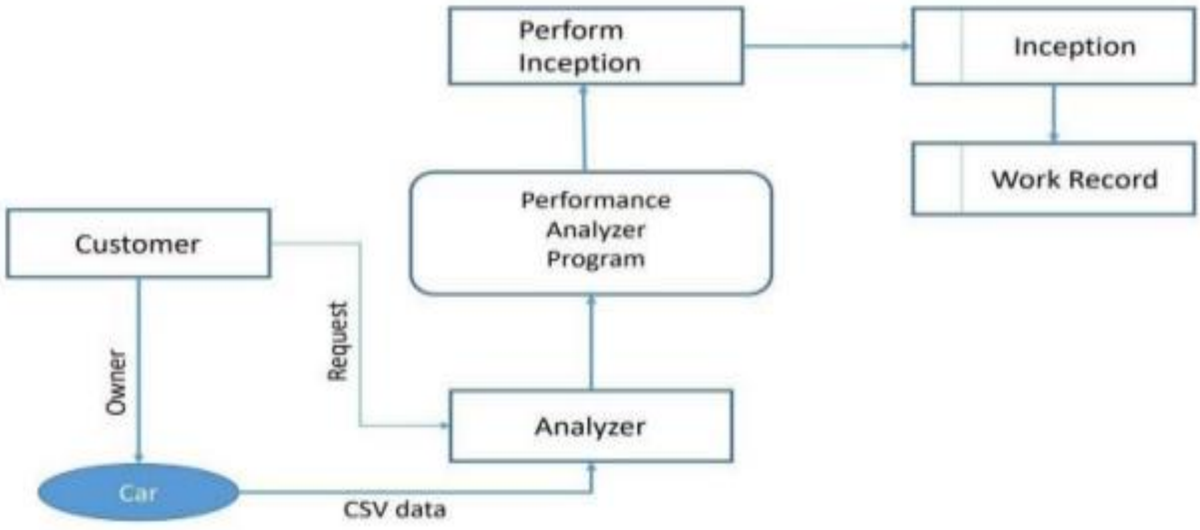
4.2 NON-FUNCTIONAL REQUIREMENTS:

Following are the non-functional requirements of the proposed solution

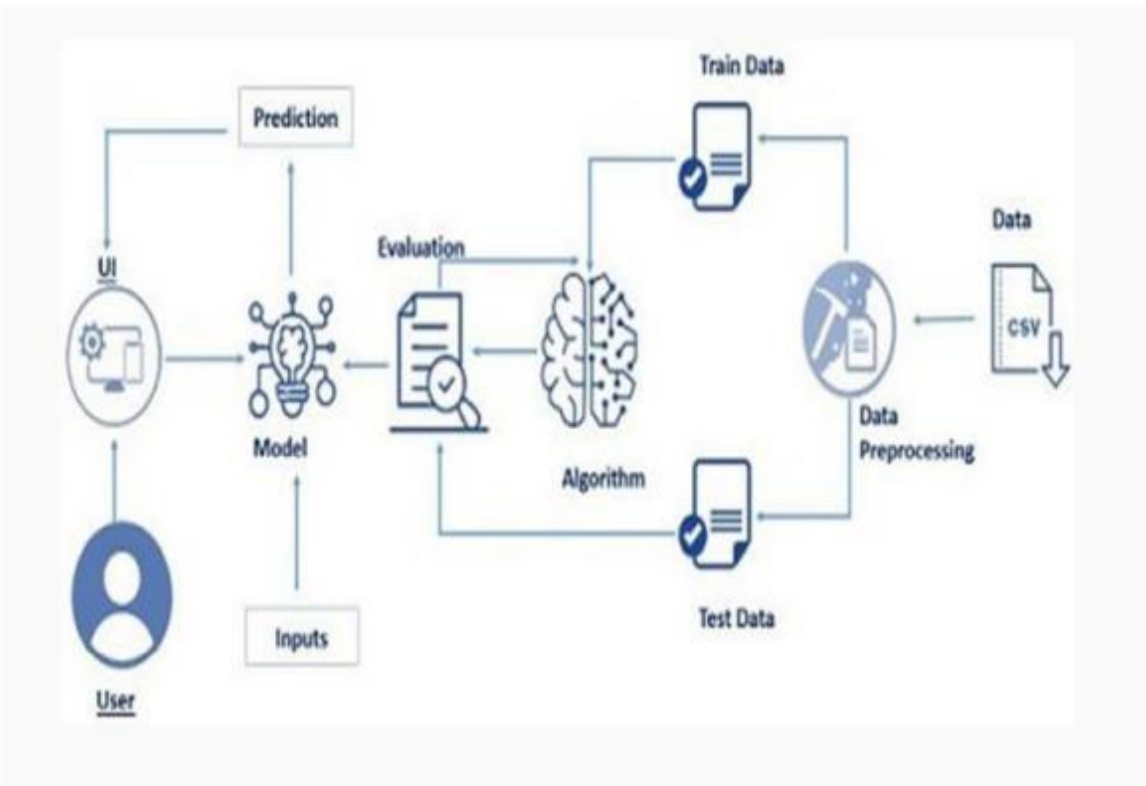
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The project Is made easy to use by any one using traditional method of web designing.
NFR-2	**Security**	Security is enabled by providing proper authentication strategies and incorporating some form of encryption.
NFR-3	**Reliability**	The machine learning model is chosen in such a way that it is most suitable for the field of application and makes accurate prediction almost all the time.
NFR-4	**Performance**	The performance will be improved by choosing efficient algorithms making predictions in less time
NFR-5	**Availability**	It provides an efficient outcome and has the ability to increase or decrease the performance of the system based on the datasets.
NFR-6	**Scalability**	The application is highly scalable as vehicle performance is measured in every country.

5. PROJECT DESIGN:

5.1 DATA FLOW DIAGRAMS:



5.2 SOLUTION & TECHNICAL ARCHITECTURE:



5.3 USER STORIES:

User type	Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority
Admin	Sprint-1	Data Preparation	USN-1	Collecting water dataset and pre-processing it	High
Admin	Sprint-1	Handling Missing values	USN-2	Handle all the missing values in the dataset	High
Admin	Sprint-1	Calculate the Water Quality Index	USN-3	Calculate the vehicle performance index using the collected dataset	High
Admin	Sprint-1	Data Visualization	USN-4	Visualize the data using the histogram and heatmaps.	Medium
Admin	Sprint-2	Model Building	USN-5	Create an ML model to predict Vehicle Performance	High
Admin	Sprint-3	Model Evaluation	USN-6	Calculate the performance, error rate, and complexity of the ML model and evaluate thedataset based on the parameter that the dataset consists of.	High
Admin	Sprint-3	Model Deployment	USN-7	As a user, I need to deploy the model and need to find the results.	Medium
User	Sprint-3	Web page (Form)	USN-8	As a user, I can use the application by entering the vehicle dataset to analyze or predict the results.	High
Admin	Sprint-4	Flask App	USM-9	Flask app should be created to act as an interface between the frontend and model	High

6. PROJECT PLANNING AND SCHEDULING:

6.1. SPRINT PLANNING &ESTIMATION:

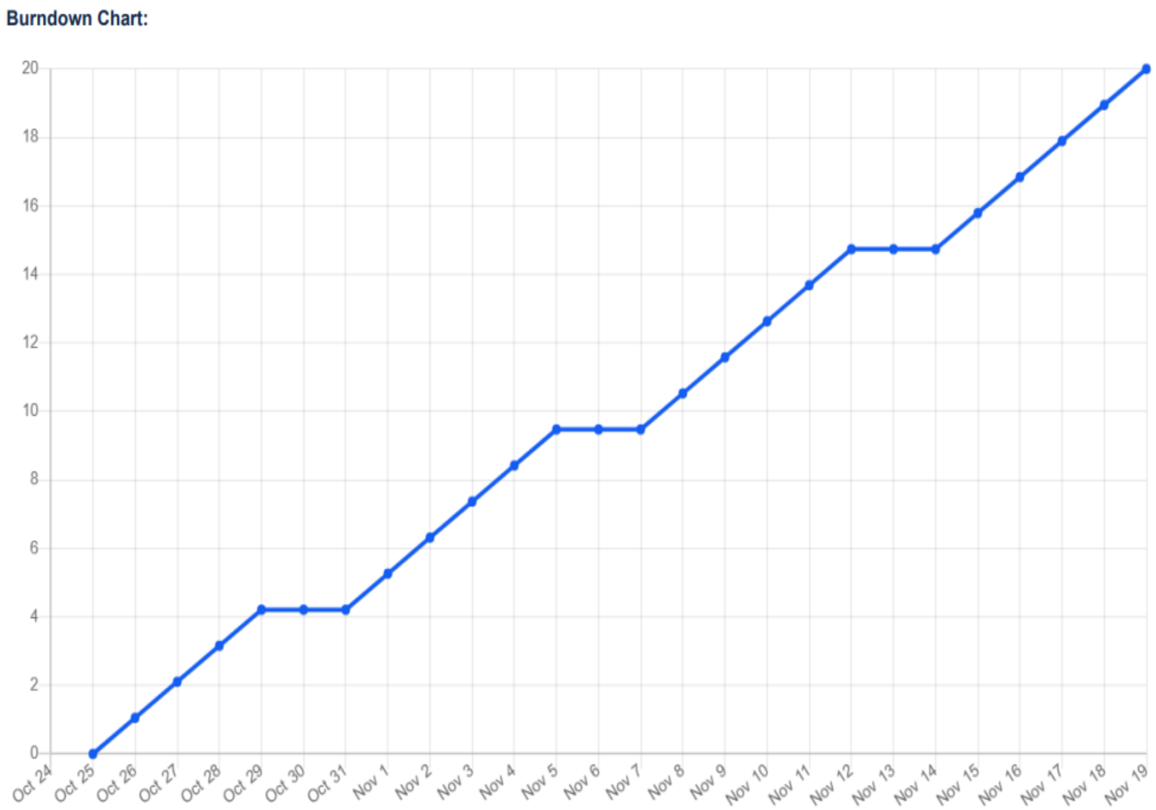
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Download the dataset	5	High	Dhanush .M Elankathir.S
Sprint-2	Data Pre-processing	USN-2	Import libraries and read the dataset	5	Medium	
Sprint-2		USN-3	Handle the missing value and label the encoding	5	High	Dhinakaran.R
Sprint-2		USN-4	Split the dataset into Dependent and independent variables	5	Medium	
Sprint-2		USN-5	Split the dataset into train and test data	20	High	Ashok.B Dhinakaran.R
Sprint-3	Model Building	USN-6	Train the datasets to run smoothly and see an incremental improvement in the prediction rate for the available Machine	5	High	

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
			Learning algorithms parameter that the dataset consists of.			
Sprint-3		USN-7	Build The Model With The Decision Tree Algorithm	10	High	Dhanush.M
Sprint-3		USN-8	Predict The Values	5	High	Elankathir.S Ashok.B
Sprint-3		USM-9	Flask app should be created to act as an interface between the frontend and model	20	High	Dhanush.M Elankathir.S
Sprint-4	Application Building	USM-10	Building An Index. Html File	5	Medium	Ashok.B Dhanush.M Dhinakaran.R Elankathir.S
Sprint-4		USM-11	Build Python Code	5	Medium	
Sprint-4		USM-12	Run the app using flask	5	Medium	
Sprint-4		USM-13	Output	5	High	

6.2. SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	14 Nov 2022

6.3 REPORTS FROM JIRA:



7.CODING & SOLUTIONING:

7.1 FEATURE 1:

The proposed system here is a machine learning based vehicle performance analyzer where users can enter different attributes associated with a vehicle and obtain its performance in this case mileage per gallon of fuel instantly. To choose a optimized machine learning algorithm we train most of the regression models to find the most suitable to approach this prediction. Among all the different models used it is inferred that decision tree regression seems to score high in performance metrics hence is chosen as the model to predict vehicle performance. Once the model is built and trained and when becomes ready for prediction, the model is dumped in a pickle file which can then be imported in application that requires it. Then create a flask application to use this pickled model to predict vehicle performance. On the other hand instead of pickling the model is deployed on IBM cloud and imported using the API key.

The below code is the model for Decision tree Algorithm.

```
from sklearn.tree import DecisionTreeRegressor
model=RandomForestRegressor(n_estimators=30,random_state=0)
model.fit(X_train,y_train)
prediction=model.predict(X_test)
```

The model is dumped into pickle and can be used as show below

```
import pickle
pickle.dump(dt,open('model.pkl','wb'))
```

The flask app is created to act as an interface to predict the quality from the details the user is giving,

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
```

```
app = Flask(__name__)

model = pickle.load(open('model.pkl', 'rb'))

@app.route("/")
def home():
   return render_template('index.html')

@app.route("/submit",methods=["POST"])
def prediction():
    if request.method == "POST":
        cyl = request.form["cylinder"]
        dis = request.form["displacement"]
        hp = request.form["horsepower"]
        w = request.form["weight"]
        a = request.form["a"]
        my = request.form["my"]
        ori = request.form["ori"]
        total = [[int(cyl), int(dis), int(hp), int(w),
 int(a), int(my), int(ori)]]
       pred="Worst performance with mileage " + str(prediction[0]) +". Carry extra
fuel"
   if(output>9 and output<=17.5):
        pred="Low performance with mileage " +str(prediction[0]) +". Don't go to
long distance"
   if(output>17.5 and output<=29):
       pred="Medium performance with mileage " +str(prediction[0]) +". Go for a
ride nearby."
   if(output>29 and output<=46):
```

```
        pred="High performance with mileage " +str(prediction[0]) +". Go for a healthy ride"
    if(output>46):
        pred="Very high performance with mileage " +str(prediction[0])+". You can plan for a Tour"


   return render_template('result.html', prediction_text='{}'.format(pred))

   @app.route('/predict_api',methods=['POST'])
   def predict_api():
    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])

  output = prediction[0]
  return jsonify(output)


  if __name__ == "__main__":
   app.run(debug=True)
```

7.2 FEATURE 2:

On the other hand instead of dumping into a pickle file, the model can be deployed in the IBM Cloud and can be used using the API key.

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    'apikey' : "XMGyyPuQidd9AY75a3wePvLGeJ8Wyck7wmts7XiJVK4",
   "url" : "https://us-south.ml.cloud.ibm.com"
```

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
   "url" : "https://us-south.ml.cloud.ibm.com",
   "apikey" : "HmgCq5mXkUrQ8MMvc6xKUsqqw2wspE_vP25rKLuyPnG5"
}
client = APIClient(wml_credentials)
def guid_from_space_name(client, space_name):
   space = client.spaces.get_details()
   return(next(item for item in space['resources'] if item['entity']["name"] ==
space_name)['metadata']['id'])
```

```
model_details =
client.repository.store_model(model=forest_reg,meta_props={
   client.repository.ModelMetaNames.NAME:"vehicle_performance",
   client.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software
_spec_uid
})
model_id = client.repository.get_model_uid(model_details)
```

```
API_KEY = "YIJAXb1Vp23FVn6FxaWNfEECIbjRwptpHaaL7jNGzuTE"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
#from joblib import load
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/y_predict',methods=['POST'])
def y_predict():
'''
For rendering results on HTML GUI
'''
x_test = [[int(x) for x in request.form.values()]]
print(x_test)
#sc = load('scalar.save')

     payload_scoring = {"input_data": [{"field": [["cylinder",
 "displacement", "horsepower",
                                   "weight", "a", "my", "ori"]],
                            "values": [[0.31188164, 0.07178791, -0.51345822,
 -0.00839082, 0.07769265,
                                   0.51815083, -0.72739454]]}]}

    print("Scoring response")
```

```
    print(response_scoring.json())
    pred=response_scoring.json()
    output=pred['predictions'][0]['values'][0][0]
    print(output)
    if(output<=9):
       ped="Worst performance with mileage " + str(output) +". Carry extra fuel"
    if(output>9 and output<=17.5):
        ped="Low performance with mileage " +str(output) +". Don't go to long
distance"
    if(output>17.5 and output<=29):
        ped="Medium performance with mileage " +str(output) +". Go for a ride
nearby."
    if(output>29 and output<=46):
        ped="High performance with mileage " +str(output) +". Go for a healthy
ride"
    if(output>46):
        ped="Very high performance with mileage " +str(output)+". You can plan
for a Tour"


    return render_template('index.html', prediction_text='{}'.format(ped))

if __name__ == "__main__":
   app.run(debug=true)
```

8. TESTING:

8.1. TEST CASES:

- Verify that the user could able to use that web page.
- Verify that the user could able to enter the value.
- Verify that the values entered by the user are computed.
- Verify that the user could able to see the predicted value.

8.2.USER ACCEPTANCE TESTING:

1. **Defect Analysis**

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	7	4	2	3	17
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	10	2	4	15	31
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	3	2	1	6
Totals	20	12	13	21	66

2.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	6	0	0	6
Client Application	45	0	0	45
Security	2	0	0	2
Outsource Shipping	2	0	0	2
Exception Reporting	7	0	0	7
Final Report Output	3	0	0	3
Redirecting	1	0	0	1

9. RESULTS:

9.1. PERFORMANCE METRICS:

Here to predict the performance of the above model two main measures are used. Model Accuracy and the r-square value. Then the mean squared error for the value is also checked. R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. The accuracy of the value is: 0.8948289556923962.

10. ADVANTAGES AND DISADVANTAGES:

Advantages:

- The model enables an user to immediately analyze a vehicle's performance and provide results instantly.
- The model uses decision tree regression which is proved to be more suitable for such cases.
- The model takes into account various error factors and acts upon them to produce almost accurate results.
- It automates the tedious and repetitive tasks.

Disadvantages:

- When dimension of the data is high the model tends to take little more time.
- This model is only suitable for measuring performance in terms of miles per gallons, and might not be suitable for other performance measure such as comfort etc.

11. CONCLUSION:

Vehicle performance prediction by using this model becomes easy and simple. It enables users of all category to predict their vehicle's performance without needing a deeper knowledge of know how about the vehicle. By employing this customers can also decide to sell or buy vehicles and it makes this transaction easier and clearer. The above model that is decision tree regression used is very much suitable to this scenarios and has an accuracy of about 89.48289556923962. It is on an overall scale doing good keeping prediction closer to accurate values.

12. FUTURE SCOPE:

The scope for this project is quite high due to high scalable nature. As almost everyone in the world owns a vehicle and everyone wants to know how their vehicle performing. This is a global scale and task which can be fulfilled using this model. The scalable and reliable nature based on its accuracy provides the clearance for the model to be employed everywhere for vehicle performance prediction.

13. APPENDIX:

SOURCE CODE:

Vehicle_performance_analysis.ipynb

```
#!/usr/bin/env python

# coding: utf-8


# # Importing Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import statsmodels.formula.api as smf

# # Importing Dataset

dataset=pd.read_csv('car performance.csv')

dataset

# # Finding missing data

dataset.isnull().any()

# There are no null characters in the columns but there is a special character '?'
in the 'horsepower' column. So we we replaced '?' with nan and replaced nan
values with mean of the column.

dataset['horsepower']=dataset['horsepower'].replace('?',np.nan)

dataset['horsepower'].isnull().sum()

dataset['horsepower']=dataset['horsepower'].astype('float64')
```

```
dataset['horsepower'].fillna((dataset['horsepower'].mean()),inplace=True)

dataset.isnull().any()

dataset.info() #Pandas dataframe.info() function is used to get a quick overview
of the dataset.

dataset.describe() #Pandas describe() is used to view some basic statistical
details of a data frame or a series of numeric values.

# There is no use with car name attribute so drop it

dataset=dataset.drop('car name',axis=1) #dropping the unwanted column.

corr_table=dataset.corr()#Pandas dataframe.corr() is used to find the pairwise
correlation of all columns in the dataframe.

corr_table


# # Data Visualizations

# Heatmap : which represents correlation between attributes

sns.heatmap(dataset.corr(),annot=True,linecolor ='black', linewidths =
1)#Heatmap is a way to show some sort of matrix plot,annot is used for
correlation.

fig=plt.gcf()

fig.set_size_inches(8,8)

# Visualizations of each attributes w.r.t rest of all attributes

sns.pairplot(dataset,diag_kind='kde') #pairplot represents pairwise relation
across the entire dataframe.

plt.show()
```

Regression plots(regplot()) creates a regression line between 2 parameters and helps to visualize their linear relationships.

sns.regplot(x="cylinders", y="mpg", data=dataset)

sns.regplot(x="displacement", y="mpg", data=dataset)

sns.regplot(x="horsepower", y="mpg", data=dataset)

sns.regplot(x="weight", y="mpg", data=dataset)

sns.regplot(x="acceleration", y="mpg", data=dataset)

sns.regplot(x="model year", y="mpg", data=dataset)

sns.regplot(x="origin", y="mpg", data=dataset)

sns.set(style="whitegrid")

sns.boxplot(x=dataset["mpg"])

Finding quartiles for mgp

The P-value is the probability value that the correlation between these two variables is statistically significant.

Normally, we choose a significance level of 0.05, which means that we are 95% confident that the correlation between

the variables is significant.

#

By convention, when the

p-value is $<$ 0.001: we say there is strong evidence that the correlation is significant.

```python
#     <li>the p-value is $<$ 0.05: there is moderate evidence that the correlation is significant.</li>

#     <li>the p-value is $<$ 0.1: there is weak evidence that the correlation is significant.</li>

#     <li>the p-value is $>$ 0.1: there is no evidence that the correlation is significant.</li>

# </ul>


from scipy import stats


# <h3>Cylinders vs mpg</h3>

#

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Cylinders' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['cylinders'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between cylinders and mpg is statistically significant, and the coefficient of ~ -0.775 shows that the relationship is negative and moderately strong.

# <h3>Displacement vs mpg</h3>

#

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Displacement' and 'mpg'.
```

```python
pearson_coef, p_value = stats.pearsonr(dataset['displacement'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.1, the correlation between displacement and mpg is statistically significant, and the linear negative relationship is quite strong (~-0.809, close to -1)</p>

# <h3>Horsepower vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'horsepower' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['horsepower'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between horsepower and mpg is statistically significant, and the coefficient of ~ -0.771 shows that the relationship is negative and moderately strong.

# <h3>Weght vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'weight' and 'mpg'

pearson_coef, p_value = stats.pearsonr(dataset['weight'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>
```

```
# <p>Since the p-value is $<$ 0.001, the correlation between weight and mpg is statistically significant, and the linear negative relationship is quite strong (~-0.831, close to -1)</p>

# <h3>Acceleration vs mpg</h3>

#

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Acceleration' and 'mpg'

pearson_coef, p_value = stats.pearsonr(dataset['acceleration'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $>$ 0.1, the correlation between acceleration and mpg is statistically significant, but the linear relationship is weak (~0.420).</p>

# <h3>Model year vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Model year' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['model year'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between model year and mpg is statistically significant, but the linear relationship is only moderate (~0.579).</p>

# <h3>Origin vs mpg</h3>

#
```

```
# Let's calculate the Pearson Correlation Coefficient and P-value of 'Origin' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['origin'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <>Since the p-value is $<$ 0.001, the correlation between origin and mpg is statistically significant, but the linear relationship is only moderate (~0.563).</p>

# <b>Ordinary Least Squares</b>  Statistics

test=smf.ols('mpg~cylinders+displacement+horsepower+weight+acceleration+origin',dataset).fit()

test.summary()

# Inference as in the above summary the p value of the accelaration is maximum(i.e 0.972) so we can remove the acc variable from the dataset

# # Seperating into Dependent and Independent variables

# <b>Independent variables</b>

x=dataset[['cylinders','displacement','horsepower','weight','model year','origin']].values

x

# <b>Dependent variables</b>

y=dataset.iloc[:,0:1].values

y
```

```
# # Splitting into train and test data.
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
# we are splitting as 75% train data and 25% test data
# # random forest regressor
from sklearn.tree import RandomForestRegressor
model=RandomForestRegressor(n_estimators=30, random_state=0)
model.fit(x_train,y_train)
import pickle
pickle.dump(dt,open('model.pkl','wb'))
prediction=model.predict(x_test)
prediction
y_test
import os
os.environ['PATH'] =
os.environ['PATH']+';'+os.environ['CONDA_PREFIX']+r"\Library\bin\graphviz
"
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(dt, out_file=dot_data,
```

```
                filled=True, rounded=True,
                special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png()
ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
sns.distplot(y_pred, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for mpg')
plt.xlabel('mpg')
plt.ylabel('Proportion of Cars')
plt.show()
plt.close()
# We can see that the fitted values are reasonably close to the actual values, since the two distributions overlap a bit. However, there is definitely some room for improvement.
# <b>R-squared</b>
# <p>R-squared is a statistical measure of how close the data are to the fitted regression line.
# It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.</p>
#
# R-squared = Explained variation / Total variation
# <b>Mean Squared Error (MSE)</b>
# <p>The Mean Squared Error measures the average of the squares of errors, that is, the difference between actual value (y) and the estimated value (ŷ).</p>
```

```
from sklearn.metrics import r2_score,mean_squared_error
r2_score(y_test,y_pred)
mean_squared_error(y_test,y_pred)
np.sqrt(mean_squared_error(y_test,y_pred))
# # random forest regressor
from sklearn.ensemble import RandomForestRegressor
rf= RandomForestRegressor(n_estimators=10,random_state=0,criterion='mae')
rf.fit(x_train,y_train)
y_pred2=rf.predict(x_test)
y_pred2
ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
sns.distplot(y_pred2, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for mpg')
plt.xlabel('mpg')
plt.ylabel('Proportion of Cars')
plt.show()
plt.close()
# We can see that the fitted values are reasonably close to the actual values, since the two distributions overlap a bit. However, there is definitely some room for improvement.
from sklearn.metrics import r2_score,mean_squared_error
r2_score(y_test,y_pred2)
```

```
mean_squared_error(y_test,y_pred2)

np.sqrt(mean_squared_error(y_test,y_pred2))

# # linear regression

from sklearn.linear_model import LinearRegression

mr=LinearRegression()

mr.fit(x_train,y_train)

y_pred3=mr.predict(x_test)

y_pred3

ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")

sns.distplot(y_pred3, hist=False, color="b", label="Fitted Values" , ax=ax1)

plt.title('Actual vs Fitted Values for mpg')

plt.xlabel('mpg')

plt.ylabel('Proportion of Cars')

plt.show()

plt.close()

# We can see that the fitted values are not as close to the actual values, since the two distributions overlap a bit. However, there is definitely some room for improvement.

from sklearn.metrics import r2_score,mean_squared_error

r2_score(y_test,y_pred3)

mean_squared_error(y_test,y_pred3)

np.sqrt(mean_squared_error(y_test,y_pred3))

# <b>Conclusion:</b>
```

<p>When comparing models, the model with the higher R-squared value is a better fit for the data.</p>

<p>When comparing models, the model with the smallest MSE value is a better fit for the data.</p>

#

Comparing these three models, we conclude that the DecisionTree model is the best model to be able to predict mpg from our dataset.

#

Vehicle_performance_Analysis_IBM_Deployment.ipynb

Importing Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import statsmodels.formula.api as smf

Importing Dataset

dataset=pd.read_csv('car performance.csv')

dataset

Finding missing data

dataset.isnull().any()

There are no null characters in the columns but there is a special character '?' in the 'horsepower' column. So we we replaced '?' with nan and replaced nan values with mean of the column.

dataset['horsepower'].isnull().sum()

dataset['horsepower']=dataset['horsepower'].astype('float64')

dataset['horsepower'].fillna((dataset['horsepower'].mean()),inplace=True)

dataset.isnull().any()

dataset.info() #Pandas dataframe.info() function is used to get a quick overview of the dataset.

dataset.describe() #Pandas describe() is used to view some basic statistical details of a data frame or a series of numeric values.

There is no use with car name attribute so drop it

dataset=dataset.drop('car name',axis=1) #dropping the unwanted column.

corr_table=dataset.corr()#Pandas dataframe.corr() is used to find the pairwise correlation of all columns in the dataframe.

corr_table

Data Visualizations

Heatmap : which represents correlation between attributes

sns.heatmap(dataset.corr(),annot=True,linecolor ='black', linewidths = 1)#Heatmap is a way to show some sort of matrix plot,annot is used for correlation.

fig=plt.gcf()

fig.set_size_inches(8,8)

Visualizations of each attributes w.r.t rest of all attributes

```
sns.pairplot(dataset,diag_kind='kde') #pairplot represents pairwise relation across the entire dataframe.

plt.show()


# Regression plots(regplot()) creates a regression line between 2 parameters and helps to visualize their linear relationships.

sns.regplot(x="cylinders", y="mpg", data=dataset)

sns.regplot(x="displacement", y="mpg", data=dataset)

sns.regplot(x="horsepower", y="mpg", data=dataset)

sns.regplot(x="weight", y="mpg", data=dataset)

sns.regplot(x="acceleration", y="mpg", data=dataset)

sns.regplot(x="model year", y="mpg", data=dataset)

sns.regplot(x="origin", y="mpg", data=dataset)

sns.set(style="whitegrid")

sns.boxplot(x=dataset["mpg"])

# Finding quartiles for mgp

# # The P-value is the probability value that the correlation between these two variables is statistically significant.

# Normally, we choose a significance level of 0.05, which means that we are 95% confident that the correlation between

# the variables is significant.

#

# By convention, when the

# <ul>
```

```
#     <li>p-value is $<$ 0.001: we say there is strong evidence that the correlation is significant.</li>

#     <li>the p-value is $<$ 0.05: there is moderate evidence that the correlation is significant.</li>

#     <li>the p-value is $<$ 0.1: there is weak evidence that the correlation is significant.</li>

#     <li>the p-value is $>$ 0.1: there is no evidence that the correlation is significant.</li>

# </ul>


from scipy import stats


# <h3>Cylinders vs mpg</h3>

#

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Cylinders' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['cylinders'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between cylinders and mpg is statistically significant, and the coefficient of ~ -0.775 shows that the relationship is negative and moderately strong.

# <h3>Displacement vs mpg</h3>

#
```

```
# Let's calculate the Pearson Correlation Coefficient and P-value of 'Displacement' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['displacement'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.1, the correlation between displacement and mpg is statistically significant, and the linear negative relationship is quite strong (~-0.809, close to -1)</p>

# <h3>Horsepower vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'horsepower' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['horsepower'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between horsepower and mpg is statistically significant, and the coefficient of ~ -0.771 shows that the relationship is negative and moderately strong.

# <h3>Weght vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'weight' and 'mpg'

pearson_coef, p_value = stats.pearsonr(dataset['weight'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between weight and mpg is statistically significant, and the linear negative relationship is quite strong (~-0.831, close to -1)</p>

# <h3>Acceleration vs mpg</h3>

#

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Acceleration' and 'mpg'

pearson_coef, p_value = stats.pearsonr(dataset['acceleration'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $>$ 0.1, the correlation between acceleration and mpg is statistically significant, but the linear relationship is weak (~0.420).</p>

# <h3>Model year vs mpg</h3>

# Let's calculate the Pearson Correlation Coefficient and P-value of 'Model year' and 'mpg'.

pearson_coef, p_value = stats.pearsonr(dataset['model year'], dataset['mpg'])

print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

# <h5>Conclusion:</h5>

# <p>Since the p-value is $<$ 0.001, the correlation between model year and mpg is statistically significant, but the linear relationship is only moderate (~0.579).</p>

# <h3>Origin vs mpg</h3>
```

```
#
# Let's calculate the Pearson Correlation Coefficient and P-value of 'Origin' and 'mpg'.
pearson_coef, p_value = stats.pearsonr(dataset['origin'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
# <h5>Conclusion:</h5>
# <>Since the p-value is $<$ 0.001, the correlation between origin and mpg is statistically significant, but the linear relationship is only moderate (~0.563).</p>
# <b>Ordinary Least Squares</b>  Statistics
test=smf.ols('mpg~cylinders+displacement+horsepower+weight+acceleration+origin',dataset).fit()
test.summary()
# Inference as in the above summary the p value of the accelaration is maximum(i.e 0.972) so we can remove the acc variable from the dataset
# # Seperating into Dependent and Independent variables
# <b>Independent variables</b>
x=dataset[['cylinders','displacement','horsepower','weight','model year','origin']].values
x
# <b>Dependent variables</b>
y=dataset.iloc[:,0:1].values
y
```

```
# # Splitting into train and test data.
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
# we are splitting as 75% train data and 25% test data
# # random forest regressor
from sklearn.ensemble import RandomForestRegressor
model=RandomForestRegressor(n_estimators=30,random_state=0)
model.fit(X_train,y_train)
y_pred=dt.predict(X_test)
y_pred
y_test
import os
os.environ['PATH'] =
os.environ['PATH']+';'+os.environ['CONDA_PREFIX']+r"\Library\bin\graphviz
"
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(dt, out_file=dot_data,
                filled=True, rounded=True,
```

```
                special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png()
ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
sns.distplot(y_pred, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for mpg')
plt.xlabel('mpg')
plt.ylabel('Proportion of Cars')
plt.show()
plt.close()
# We can see that the fitted values are reasonably close to the actual values,
since the two distributions overlap a bit. However, there is definitely some room
for improvement.
# <b>R-squared</b>
# <p>R-squared is a statistical measure of how close the data are to the fitted
regression line.
# It is also known as the coefficient of determination, or the coefficient of
multiple determination for multiple regression.</p>
#
# R-squared = Explained variation / Total variation
# <b>Mean Squared Error (MSE)</b>
# <p>The Mean Squared Error measures the average of the squares of errors,
that is, the difference between actual value (y) and the estimated value (ŷ).</p>
```

```
from sklearn.metrics import r2_score,mean_squared_error

r2_score(y_test,y_pred)

mean_squared_error(y_test,y_pred)

np.sqrt(mean_squared_error(y_test,y_pred))

# # random forest regressor

from sklearn.ensemble import RandomForestRegressor

rf= RandomForestRegressor(n_estimators=10,random_state=0,criterion='mae')

rf.fit(x_train,y_train)

y_pred2=rf.predict(x_test)

y_pred2

ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")

sns.distplot(y_pred2, hist=False, color="b", label="Fitted Values" , ax=ax1)

plt.title('Actual vs Fitted Values for mpg')

plt.xlabel('mpg')

plt.ylabel('Proportion of Cars')

plt.show()

plt.close()

# We can see that the fitted values are reasonably close to the actual values,
since the two distributions overlap a bit. However, there is definitely some room
for improvement.

from sklearn.metrics import r2_score,mean_squared_error

r2_score(y_test,y_pred2)

mean_squared_error(y_test,y_pred2)
```

```
np.sqrt(mean_squared_error(y_test,y_pred2))

# # linear regression

from sklearn.linear_model import LinearRegression

mr=LinearRegression()

mr.fit(x_train,y_train)

y_pred3=mr.predict(x_test)

y_pred3

ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")

sns.distplot(y_pred3, hist=False, color="b", label="Fitted Values" , ax=ax1)

plt.title('Actual vs Fitted Values for mpg')

plt.xlabel('mpg')

plt.ylabel('Proportion of Cars')

plt.show()

plt.close()

# We can see that the fitted values are not as close to the actual values, since the two distributions overlap a bit. However, there is definitely some room for improvement.

from sklearn.metrics import r2_score,mean_squared_error

r2_score(y_test,y_pred3)

mean_squared_error(y_test,y_pred3)

np.sqrt(mean_squared_error(y_test,y_pred3))

from ibm_watson_machine_learning import APIClient
wml_credentials = {
  'apikey' : "XMGyyPuQidd9AY75a3we-PvLGeJ8Wyck7wmts7XiJVK4",
```

```
    "url" : "https://us-south.ml.cloud.ibm.com"
}
wml_client=APIClient(wml_credentials)
wml_client.spaces.list()
space_id="5a3b74bd-d5c8-4f21-a5cc-b823b4345a14"
wml_client.set.default_space(space_id)
wml_client.software_specifications.list()
model_name="analysis_model"
deployment_name="analysis_deploy_model"
model_deploy=dt
software_spec_uid=wml_client.software_specifications.get_uid_by_name("runt
ime-22.1-py3.9")
model_props={
    wml_client.repository.ModelMetaNames.NAME:model_name,
    wml_client.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
     wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software
_spec_uid
}
model_details=wml_client.repository.store_model(
    model=model_deploy,
    meta_props=model_prop
   s,  training_data=X_train,
   training_target=y_train)
model_details
```

Index.html

```
<link    href="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
<link  href="https://fonts.googleapis.com/css2?family=Girassol&display=swap"
rel="stylesheet">
```

```
<script
src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<script
src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<div class="navbar">
     <section class="title">
      <h1><p style="font-family: 'cursive', cursive ;">PREDICT YOUR CAR'S
PERFORMANCE</p></h1>
    </section>
</div>

<div class="wrapper fadeInDown">
  <div id="formContent">
    <!-- Tabs Titles -->
   <section class="date">
    <!-- Icon -->
    <div class="fadeIn first">
        <script src="https://unpkg.com/@lottiefiles/lottie-player@latest/dist/lottie-
player.js"></script>
    </div>
    <div class="Contanier">
      <div class="card"></div>


      </div>
    <div class="fadeInDown">
    <form action="{{ url_for('y_predict')}}"method="post">
      <input type="text" name="Cylinders" placeholder="No.of cylinders (count)"
required="required" />
```

```html
        <input type="text" name="Displacement" placeholder="Displacement (in miles)" required="required" />
        <input type="text" name="Horsepower" placeholder="Horsepower (per sec)" required="required" />
        <input type="text" name="Weight" placeholder="Weight (in pounds)" required="required" />
        <input type="text" name="Model Year" placeholder="Model Year (YY)" required="required" />
        <input type="text" name="Origin" placeholder="Origin" required="required" />
        <br>
        <input type="submit" class="fadeIn fourth" value="Predict">
    </form>
    </section>

    <div id="formFooter">
      <a class="underlineHover" href="#">
        <strong>{{ prediction_text }}</strong></a>
    </div>
   </div>
  </div>
</div>
```

app.py

```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
#from joblib import load
app = Flask(__name__)
model = pickle.load(open('decision_model.pkl', 'rb'))
```

```
@app.route('/')
def home():
    return render_template('index.html')

@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''
    For rendering results on HTML GUI
    '''
    x_test = [[int(x) for x in request.form.values()]]
    print(x_test)
    #sc = load('scalar.save')
    prediction = model.predict(x_test)
    print(prediction)
    output=prediction[0]
    if(output<=9):
        pred="Worst performance with mileage " + str(prediction[0]) +". Carry extra fuel"
    if(output>9 and output<=17.5):
        pred="Low performance with mileage " +str(prediction[0]) +". Don't go to long distance"
    if(output>17.5 and output<=29):
        pred="Medium performance with mileage " +str(prediction[0]) +". Go for a ride nearby."
    if(output>29 and output<=46):
        pred="High performance with mileage " +str(prediction[0]) +". Go for a healthy ride"
    if(output>46):
        pred="Very high performance with mileage " +str(prediction[0])+". You can plan for a Tour"
```

```
    return render_template('index.html', prediction_text='{}'.format(pred))

@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

if __name__ == "__main__":
    app.run(debug=True)
```

IBM_app.py

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
import requests

# NOTE: you must manually set API_KEY below using information retrieved
from your IBM Cloud account.
API_KEY = "HmgCq5mXkUrQ8MMvc6xKUsqqw2wspE_vP25rKLuyPnG5"
token_response = requests.post('https://iam.cloud.ibm.com/identity/
token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
```

```
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
#from joblib import load
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''
    For rendering results on HTML GUI
    '''
    x_test = [[int(x) for x in request.form.values()]]
    print(x_test)
    #sc = load('scalar.save')
    payload_scoring = {"input_data": [{"field": [["cylinder", "displacement",
"horsepower","weight", "a", "my", "ori"]],"values": total}]}

response_scoring = requests.post('https://eu-
gb.ml.cloud.ibm.com/ml/v4/deployments/f4aecc62-cd58-47a3-af62-
6a940301a611/predictions?version=2022-11-15', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
    pred=response_scoring.json()
    output=pred['predictions'][0]['values'][0][0]
    print(output)
```

```
    if(output<=9):
        ped="Worst performance with mileage " + str(output) +". Carry extra fuel"
    if(output>9 and output<=17.5):
        ped="Low performance with mileage " +str(output) +". Don't go to long distance"
    if(output>17.5 and output<=29):
        ped="Medium performance with mileage " +str(output) +". Go for a ride nearby."
    if(output>29 and output<=46):
        ped="High performance with mileage " +str(output) +". Go for a healthy ride"
    if(output>46):
        ped="Very high performance with mileage " +str(output)+". You can plan for a Tour"
    return render_template('index.html', prediction_text='{}'.format(ped))

if __name__ == "__main__":
    app.run(debug=True)
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-12804-1659493638

PROJECT DEMONSTRATION LINK

https://drive.google.com/drive/folders/1eczG-rJ9N6E7fFKZnnYlxZ75BviTcb5_?usp=sharing