

MACHINE LEARNING BASED VEHICLE PERFORMANCE ANALYZER

Bonafede record of work done by

Team ID: PNT2022TMID21230

HARSHA J S (Team Leader) - (917719C032)

LINESH RAJ T - (917719C049)

SRIJITH B - (917719C101)

VASANTH KUMAR M - (917719C113)

THIAGARAJAR COLLEGE OF ENGINEERING

BACHELOR OF ENGINEERING

BRANCH: COMPUTER SCIENCE AND ENGINEERING

INTRODUCTION.....	1
1.1 Project Overview	1
1.2 Purpose	1
LITERATURE SURVEY	2
2.1 Existing Problem	2
2.2 Problem Definition	2
2.3 References	2
IDEATION AND PROPOSED SOLUTION	6
3.1 Empathy Map	6
3.2 Ideation and Brainstorming	7
3.3 Proposed Solution	8
3.4 Problem Solution Fit	10
REQUIREMENT ANALYSIS	11
4.1 Functional Requirements	11
4.2 Non-Functional Requirements	11
PROJECT DESIGN	12
5.1 Dataflow Diagram	12
5.2 Technical Architecture	12
PROJECT PLANNING AND SCHEDULING	15
6.1 Sprint Planning & Estimation	15
6.2 Sprint Delivery Schedule	15
6.3 Reports for JIRA	16
CODING AND SOLUTION.....	17
7.1 Feature 1.....	17
TESTING	18
8.1 Test Cases	18
8.2 User Acceptance Testing	18
RESULTS	19
9.1 Performance Metrics.....	19
PROS AND CONS	20
CONCLUSION	21
FUTURE WORKS	22
APPENDIX	23
13.1 Source Code	23
13.2 GitHub & Project Demo Link	29

INTRODUCTION

1.1 Project Overview

It's a significant and intriguing challenge to predict a vehicle's performance level. The primary objective of the current study is to forecast automobile performance to enhance specific vehicle behaviour. This can greatly reduce the fuel consumption of the system and boost its effectiveness. Analysis of the vehicle's performance depending on the kind of engine, number of cylinders, fuel type, and horsepower, among other factors. The health of the automobile may be predicted based on these variables. Obtaining, investigating, interpreting, and documenting health data based on the three elements is a continuous process. Prediction engines and engine management systems both heavily rely on performance metrics like mileage, reliability, flexibility, and cost that may be combined. To increase the performance efficiency of the vehicle, it is crucial to analyse the elements utilizing 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, lifespan, and range of automotive traction batteries. We also take a performance in mileage into account here. We will create the models, utilizing various techniques and neural networks, to resolve this issue. Then, we'll compare which algorithm accurately predicts vehicle performance (Mileage).

1.2 Purpose

The use of Machine Learning techniques (supervised and unsupervised) on automotive engine sensor data to discover drivers' usage patterns, and to perform classification through a distributed online sensing platform. Such platforms can be used in different domains, such as fleet management, insurance market, fuel consumption optimization, CO2 emission reduction, among others. Thus, the main purpose of the project is to predict the performance of the vehicle to improve certain behaviours of the vehicle using various machine learning algorithms.

LITERATURE SURVEY

2.1 Existing Problem

The potential for processing vehicle sensing data has increased in recent years due to the development of new technologies. Having this type of data is important, for instance, to analyse the way drivers behave when sitting behind the steering wheel. Very little has been done to analyse vehicle usage patterns based on vehicle engine sensor data, and, therefore, it has not been explored to its full potential by considering all sensors within a vehicle engine. Aiming to bridge this gap, the use of Machine Learning techniques (supervised and unsupervised) on automotive engine sensor data to discover drivers' usage patterns, and to perform classification through a distributed online sensing platform. Such platforms can be used in different domains, such as fleet management, insurance market, fuel consumption optimization, CO2 emission reduction, among others.

2.2 Problem Definition

Thus, by going through the existing problem and gaining knowledge from the various papers in the literature survey. We can frame the problem definition as follows:

“To predict the performance of the vehicle to improve certain behaviours of the vehicle using various machine learning algorithms”

2.3 References

2.3.1 ML Based Real-Time Vehicle Data Analysis for Safe Driving Modelling

In the paper “Machine Learning Based Real-Time Vehicle Data Analysis for Safe Driving Modelling” Machine learning approach to analyse and predict the vehicle performance in real time. The focus is on analysing the data which is collected from the vehicle using the OBD-II scanner and eventually providing the driver's safety solutions The meta features of the vehicle are analysed in the cloud and are then shared to the concerned parties. The proposed system consists of an OBD-II scanner and a mini dash cam which continuously send data to the cloud server where data analysis is done.

Multivariate Linear Regression Model:

It is used when we want to predict the value of a variable based on the value of two or more different variables. The variable we want to predict is called the Dependent Variable, while those used to calculate the dependent variable are termed as Independent Variables.

Features such as fuel efficiency, average speed value, maximum speed value, fourth section speed value, interval driving distance, driving time value during green zone, traveling time value, emergency accelerated value, emergency decelerated value, fourth rpm time value and fifth rpm time value are used for training the model.

The real time data obtained is normalized using Min-Max normalization technique and they hypothesize an outcome called Economic Driving Index (ECN_DRVG_INDX) and another outcome called Safe Driving Index (SFTY_DRVG_INDX). The results have proven to be approximately 80% fitting the given features.

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

The paper “A Machine Learning Approach Based on Automotive Engine Data Clustering for

Driver Usage Profiling Classification” proposes the use of Machine Learning techniques (supervised and unsupervised) on automotive engine sensor data to discover drivers’ usage patterns, and to perform classification through a distributed online sensing platform and that such platform can be useful used in different domains, such as fleet management, insurance market, fuel consumption optimization, CO2 emission reduction, among others.

As automotive engine data has no class label, we use the following Machine Learning models used for clustering and class labels:

K means:

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabelled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on. It is a

centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

Hierarchical Clustering:

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabelled datasets into a cluster. In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the dendrogram.

Machine learning algorithms for Classification:

Decision Tree:

The decision tree and its variants are the other learning algorithms that divide the input space into regions and has separate parameters for each region. They are classified as a non- parametric supervised learning method which is widely used in classification and regression, as well as in representing decisions and decision making. The structure of a decision tree is a flowchart, in which each internal node represents a “test” on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label..

Random Forest

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map

The primary purpose of the empathy map is to bridge the understanding of the user and developer. Figure 3.1 represents the empathy map for the machine learning based vehicle performance analyser

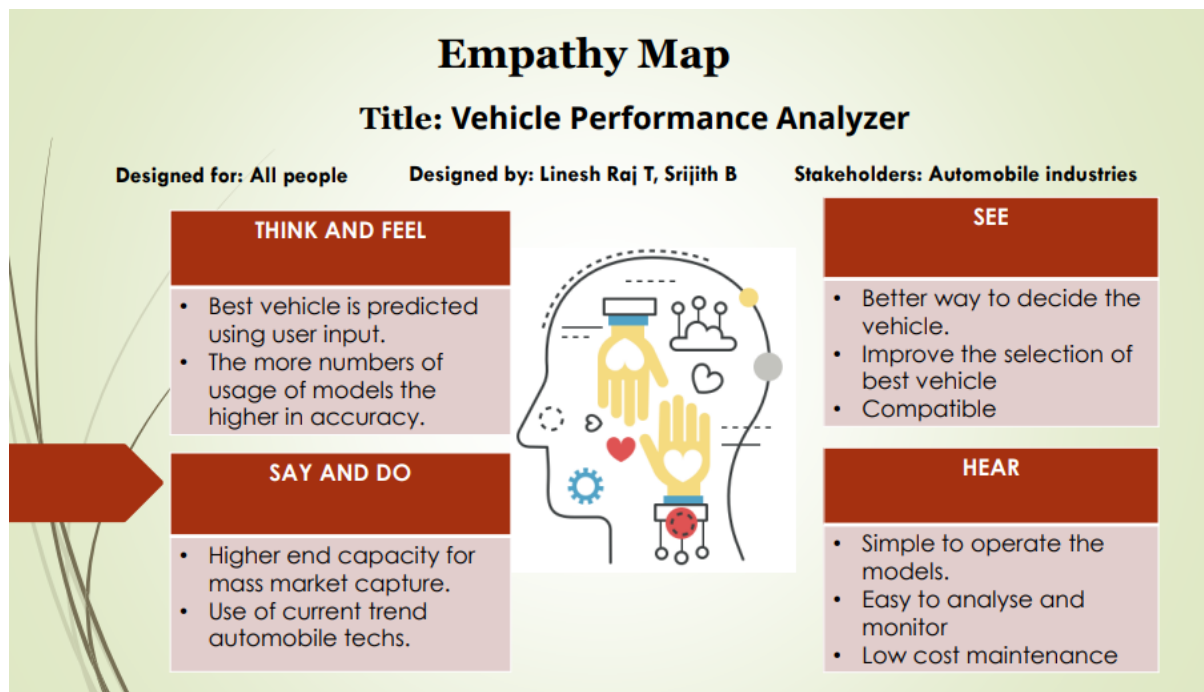


Figure 3.1 – Empathy Map

3.2 Ideation and Brainstorming

This is often the most exciting stage in a project, because during Ideation and brainstorming, the aim is to generate a large quantity of ideas that the team can then filter and cut down into the best, most practical, or most innovative ones to inspire new and better design solutions and products. Figure 3.2 shows the stages of ideation and brainstorming for the machine learning based vehicle performance analyser.

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

⌚ 5 minutes

PROBLEM

PREDICTING THE PERFORMANCE LEVEL OF VEHICLE

Key rules of brainstorming

To run a smooth and productive session

Stay in topic.

Defer judgment.

Go for volume.

Encourage wild ideas.

Listen to others.

If possible, be visual.

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

⌚ 10 minutes

Collect all Data from Various the Forum

Processing Personal data from various resources

When use the collected data enables the system understand

Learning Learning various and predict the performance

Display the prediction result using graphs.

Automation performance

System can built with in various storage of data.

Process which data with the data and are accessible

When use the system learning and reference by using the data.

Go through the collected data

Impact of disturbance on engine performance

Automation performance

Users access speed performance

When select algorithm for better comfort

Customer experience with vehicle

When use the system learning and reference by using the data.

Send an alert when safety level is not secure

Reduce chance of carbon more noise

Use fuel when the vehicle is idle

TIP

You can select a sticky note and hit the pencil (switch to sketch) icon to start drawing!

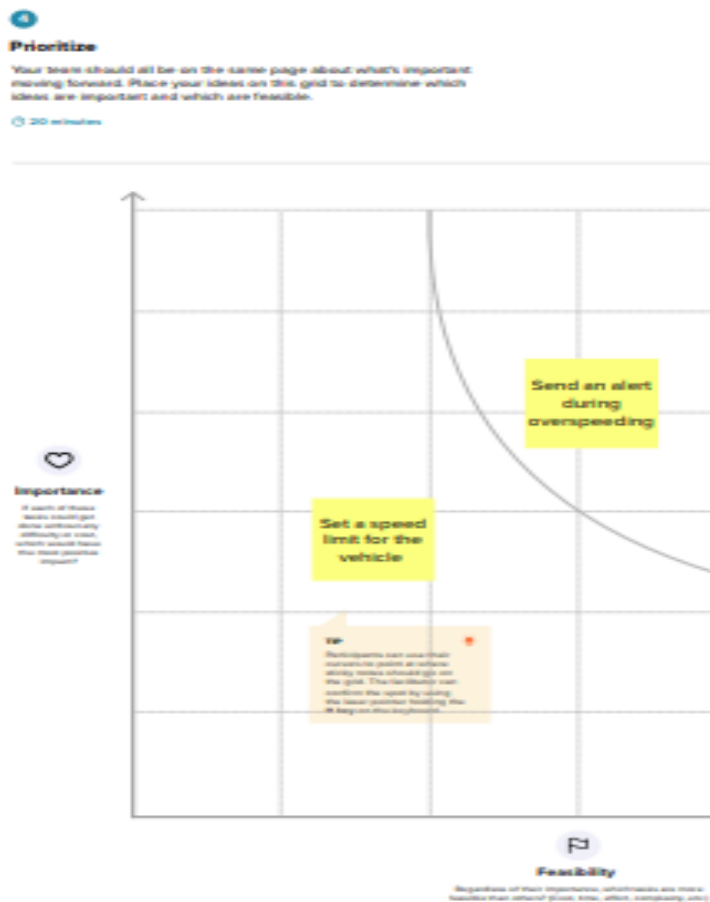


Figure 3.2 – Ideation & Brainstorming

3.3 Proposed Solution

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	The objective of this project is to predict the price of used cars using the various Machine Learning models by using User Interface (UI).
2	Idea / Solution description	To train the system with the dataset using a regression model and it will be integrated to the web-based application where the user is notified with the status.
3	Novelty / Uniqueness	By using the optimal decision tree model to predict the value in a less amount to time and predict its value.
4	Social Impact / Customer Satisfaction	The customer can get an idea about the resale value of their vehicle to predict the performance. By knowing the vehicle brand, fuel type, kilometres driven .
5	Business Model (Revenue Model)	The web-based application has a friendly UI for the customer to enter their vehicles detail and the system predicts the value within few seconds.
6	Scalability of the Solution	Machine learning approaches, this project proposed a scalable framework for predicting values for different type of used cars. The solution given by the trained system is efficient and is nearly accurate value of the vehicle.

3.4 Problem Solution Fit

The problem solution fit is the solution one has found to address the problem of the customer. Figure 3.4 depicts the solution fit for the machine learning based vehicle performance analyser.

1. CUSTOMER SEGMENTS The customer is one who wants to predict the performance of the vehicle	2. CUSTOMER CONSTRAINTS <ul style="list-style-type: none"> 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. 	3. AVAILABLE SOLUTIONS <ul style="list-style-type: none"> 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.
4. JOBS-TO-BE-DONE/PROBLEMS 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.	5. PROBLEM ROOT CAUSE <ul style="list-style-type: none"> 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. 	6. BEHAVIOUR 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.
7. TRIGGERS 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	8. YOUR SOLUTION <ul style="list-style-type: none"> 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. The project should take parameters related to used car as inputs and enable the customers to make decisions by their own. 	9. CHANNELS of BEHAVIOUR <ul style="list-style-type: none"> 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. User can decide by seeing the exterior and interior condition of the car. User can test the performance of the car and to buy it up in a affordable price based on its condition.
10. EMOTIONS: BEFORE / AFTER <u>Before:</u> User will be in fear about the biased values predicted by the humans based on the condition of the car. <u>After:</u> User can determine the worthiness of the car by their own without human intervention.		

REQUIREMENT ANALYSIS

4.1 Functional Requirements

Table 4.1 are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Enter the input	Get input through the form
FR-2	User Essential	Predict the performance of the vehicle
FR-3	Data pre-processing	Sample dataset for training purpose
FR-4	User input Evaluation	Evaluating the given user values
FR-5	Prediction	Fuel consumption and efficiency of the vehicle

4.2 Non-Functional Requirements

Table 4.2 – Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The analyser allows the user to improve performance based on the results provided. It is easy to use with just the data required.
NFR-2	Security	The security is improved by using vehicle alarm, wheel lock, vehicle lock and GPS tracker.
NFR-3	Reliability	The reliability rating is good due to best performance, less frequency of problem occurrence and cost for repairing is low.
NFR-4	Performance	The vehicle is upgraded in quality and infrastructure to provide better performance like good mileage, smooth travel.

NFR-5	Availability	The data required is collected by research persons and this data can be used to provide better results.
NFR-6	Scalability	Better scalability since our model analyzes all information and provides better refined solutions. With less change to the vehicle, we could achieve maximum performance.

PROJECT DESIGN

5.1 Dataflow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of how information flows within a system. A neat and clear DFD can thus depict the right amount of the system requirements graphically. It not only shows how data enters and leaves the system, but also what changes the information and where the data is stored. Figure 5.1 represents the DFD for the given project.

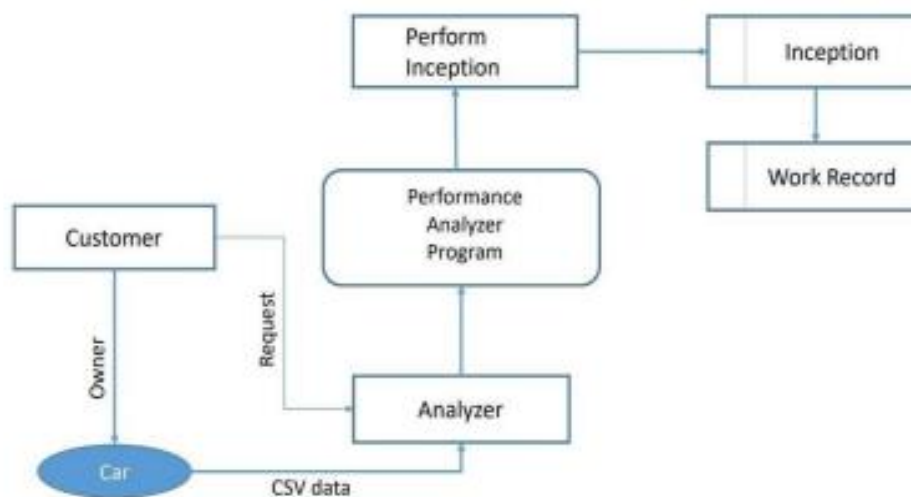


Figure 5.1 – Dataflow Diagram

5.2 Technical Architecture

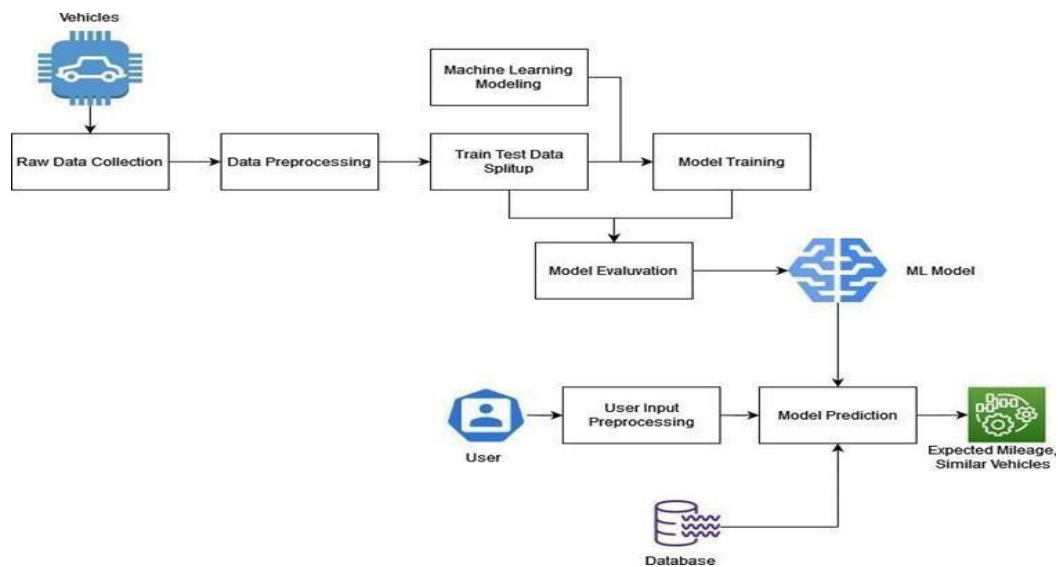


Figure 5.2 Technical Architecture

5.2.1 Component and Technologies

S.No	Component	Description	Technology
1.	User Interface	The user interacts with the application through a Web Application that is responsive to the device that is being used.	HTML, CSS, JAVASCRIPT
2.	Get User Data	The process collects the user input data that is collected via a form to the server as a JSON Object	REST API

3.	Model Prediction	Use the data collected from the user to make predictions on the mileage expected.	IBM Watson ML
4.	Send User Report	Send the predictions along with suggestions to the user as JSON Object	REST API
5.	Cloud Database	Database Service on Cloud	IBM DB2
6.	External API-1	Vehicle Details Database	https://api.auto-data.net/
7.	Machine Learning Model	The machine learning model is used to predict mileage from the user inputs	Decision tree
8.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Core i5, 8GB RAM Cloud Server Configuration:	Local

PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 1	Data Preparation	USN-1	Collecting Car dataset and pre-processing it	10	High	Harsha J S Linesh Raj T
Sprint – 1	Data Modelling	USN-2	create an ml model to	5	Medium	Srijith B

			predict the car performance			
Sprint – 1	Model Evaluation	USN-3	Calculate the performance, error rate and complexity of ML model	5	Medium	Vasanthkumar M
Sprint - 2	Model Building	USN-4	Build the model using suitable machine algorithm to check the performance of a car or any other vehicle	20	High	Harsha J S Linesh Raj T Srijith B Vasanthkumar M
Sprint – 3	Application Building	USN-5	Vehicle Creating the web page using HTML, CSS, JS and connecting it with flask.	10	High	Harsha J S Linesh Raj T
Sprint – 3	Dashboard	USN-6	As a user, I can use the application by entering Car data	10	High	Srijith B Vasanthkumar M
Sprint - 4	Deploying the model on IBM Cloud	USN-7	Using flask and deploy model finally in IBM cloud using IBM storage and Watson Studio	20	High	Harsha J S Linesh Raj T Srijith B Vasanthkumar M

6.2 Sprint Delivery Schedule

Project tracker:

Sprint	Total story point	Duration	Sprint start date	Sprint end date	Story points completed	Sprint release date
Sprint-1	20	6 days	27-oct-2022	28-oct-2022	20	04-nov-2022
Sprint-2	20	6 days	02-nov-2022	05-nov-2022	20	07-nov-2022
Sprint-3	20	6 days	08-nov-2022	12-nov-2022	20	12-nov-2022
Sprint-4	20	6 days	14-nov-2022	19-nov-2022	20	19-nov-2022

Velocity:

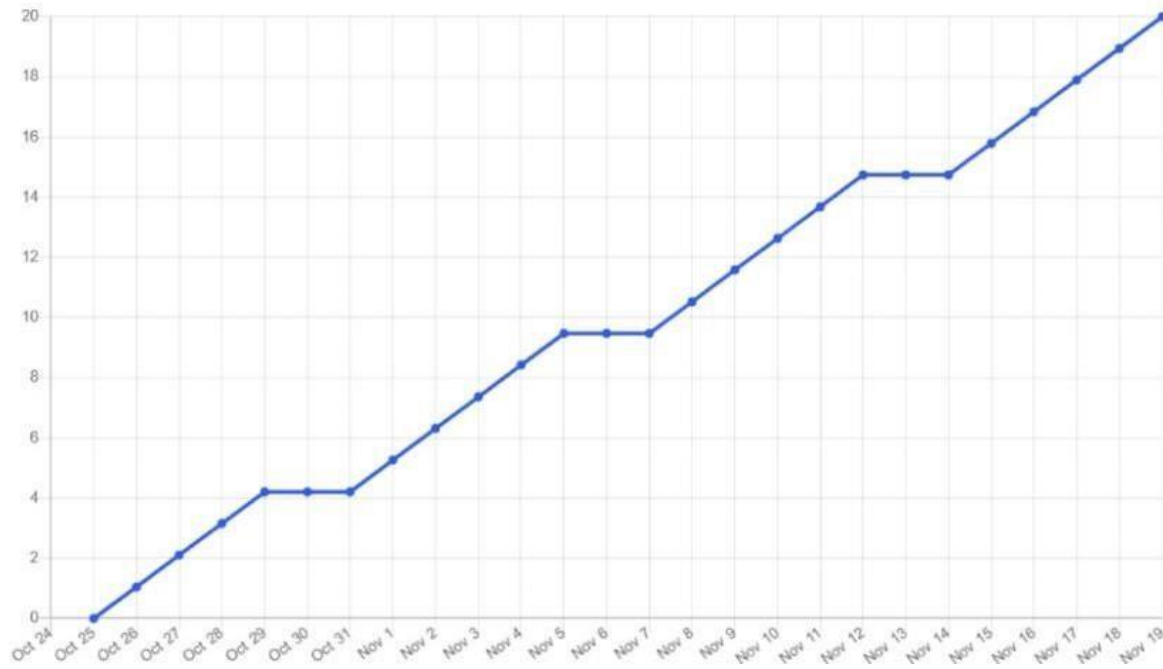
Average velocity = $80/20 = 4$ story points per day

6.3 Reports for JIRA

Velocity: Imagine we have a 10-day sprint duration, and the velocity of team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

Burndown Chart: A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



CODING AND SOLUTION

7.1 Feature 1

FR No.	Feature	Description
FR-1	Enter the input	Get input through the form
FR-2	User Essential	Predict the performance of the vehicle
FR-3	Data pre-processing	Sample dataset for training purpose
FR-4	User input Evaluation	Evaluating the given user values
FR-5	Prediction	Fuel consumption and efficiency of the vehicle

Table 7.1 – Description for Feature 1

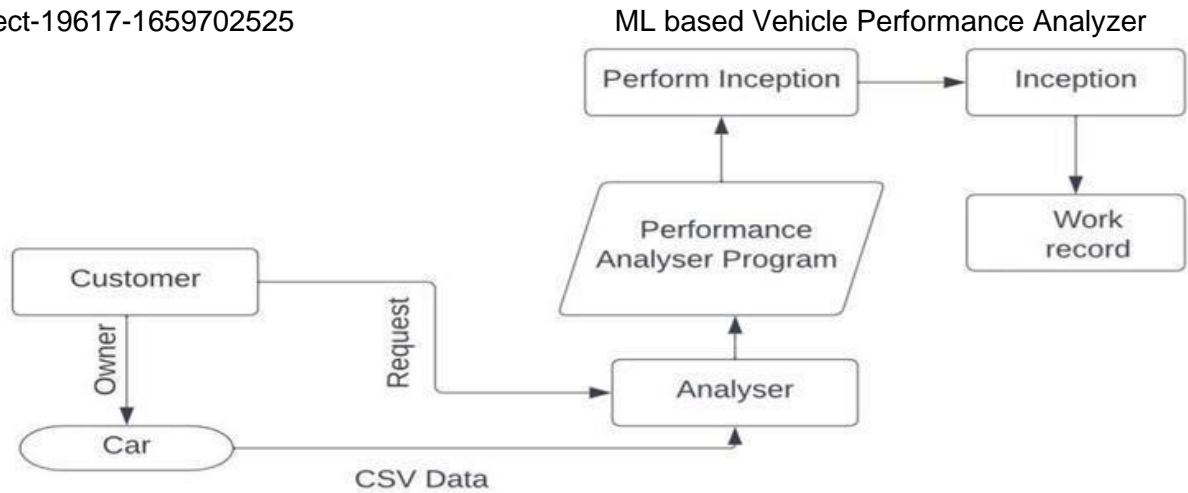


Figure 7.1 – Dataflow Diagram for Feature 1

The classification happens in the “apps.py” file where if the output is lesser than 9 the vehicle is said to have the worst performance. If the output is between 9 and 17.5 it is said to have low performance. If it is between 17.5 and 29 it has a medium performance. If the output is between 29 and 46 the vehicle is said to have a high performance and finally if it is above 46 the vehicle has a very high performance.

TESTING

8.1 Feature 1

				Date	14-Nov-22								
				Team ID	PNT2022TMD12635								
				Project Name	Project - Machine Learning based Vehicle Performance Analyzer								
				Maximum Marks	4 marks								
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
HomePage_TC_001	Functional	Home Page	Verify if the user is able to enter the data into the text field in the webpage and click the button		1. Enter the URL 2. Enter the values	[0,107,130,1504,70,1]	Page refresh	Working as expected	Pass				Sandeep
HomePage_TC_002	Functional	Home page	Verify if the user is able to view the output after the submit button has been clicked		1. Click the submit button		Low performance with mileage 17.1	Working as expected	Pass				Sandeep

Figure 8.1 – Test Cases Run

8.2 User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. 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	1	1	0	0	2
Duplicate	1	0	0	0	1
External	1	0	0	0	1
Fixed	1	1	1	1	4
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	4	2	1	1	13

3. 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	4	0	0	4
Client Application	4	0	0	4
Security	1	0	0	1

Outsource Shipping	0	0	0	0
Exception Reporting	1	0	0	1
Final Report Output	4	0	0	4
Version Control	1	0	0	1

Figure 8.2 – User Acceptance Testing

RESULTS

9.1 Performance Metrics

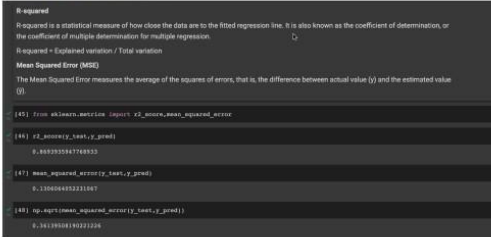

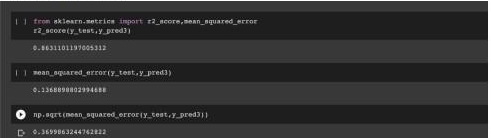
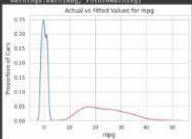
S.No.	Parameter	Values	Screenshot
1.	Metrics	Regression Model: MAE - , MSE - , RMSE - , R2 score - Classification Model: Confusion Matrix - , Accuracy Score - & Classification Report -	<p>Decision tree regressor</p>  <pre> R-squared R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. R-squared = Explained variation / Total variation Mean Squared Error (MSE) 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) (45) from sklearn.metrics import r2_score, mean_squared_error (46) r2_score(y_test, y_pred) 0.863101197005312 (47) mean_squared_error(y_test, y_pred) 0.10606402231007 (48) np.sqrt(mean_squared_error(y_test, y_pred)) 0.32139308190222206 </pre> <p>Random forest regressor</p>  <pre> (49) from sklearn.metrics import r2_score, mean_squared_error (50) r2_score(y_test, y_pred) 0.863101197005312 (51) mean_squared_error(y_test, y_pred) 0.10606402231007 (52) np.sqrt(mean_squared_error(y_test, y_pred)) 0.32139308190222206 </pre> <p>Linear regression</p>  <pre> () from sklearn.metrics import r2_score, mean_squared_error r2_score(y_test, y_pred) 0.863101197005312 () mean_squared_error(y_test, y_pred) 0.10606402231007 () np.sqrt(mean_squared_error(y_test, y_pred)) 0.32139308190222206 </pre> <p>Conclusion: When comparing models, the model with the higher R-squared value is a better fit for the data. When comparing models, the model with the smallest MSE value is a better fit for the data. Comparing these three models, we conclude that the DecisionTree model is the best model to be able to predict mpg from our dataset.</p>
Accuracy	Training Accuracy - 0.91724492094		 <pre> (53) plt.figure(figsize=(10, 5)) plt.scatter(data['mpg'], data['mpg_hat'], label='Actual Value') plt.scatter(data['mpg_hat'], data['mpg_hat'], label='Fitted Value', s=100) plt.title('Actual vs Fitted Values for mpg') plt.xlabel('mpg') plt.ylabel('mpg_hat') plt.legend() plt.show() </pre> <p>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.</p> <pre> (54) from sklearn.metrics import r2_score, mean_squared_error (55) r2_score(y_test, y_pred) 0.863101197005312 </pre>

Figure 9.1 – Performance Metrics

PROS AND CONS

10.1 Pros

- Using the Decision tree Algorithm in the model helps to perform both classification as well as regression tasks.
- A Decision tree produces good predictions that can be easily understood
- It can handle large datasets easily
- Decision tree Algorithm provides a higher-level accuracy in predicting outcomes.

10.2 Cons

- The main limitation of using Decision tree algorithm in the model is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions.
- The Decision tree algorithm is quite slow to create predictions once it is trained.

CONCLUSION

The ability to estimate a car's performance level presents a big and fascinating challenge. Forecasting vehicle performance in order to improve particular vehicle behavior was our main goal. performance evaluation of the car considering its horsepower, cylinder count, fuel type, and engine type, among other things. Based on the factors, like horsepower, cylinder count, fuel type, and engine type, the health of the car is forecasted. We analyzed the components using a number of well-known machine learning approaches, like linear regression, decision trees, and random forests, in order to optimize the performance efficiency of the vehicle. The power, longevity, and range of automobile traction batteries are now the "hot topics" in automotive engineering. In this case, we additionally consider mileage performance. To answer this problem, we have built the models using a variety of methods and neural networks. We've then compared which algorithm is most accurate in forecasting car performance (Mileage). A front-end webpage was designed to help give the user an attractive front while they input the values required by the developed machine learning model. The IBM cloud platform was used to develop the model.

FUTURE WORKS

The dataset used for this model is an old vehicle dataset, thus the model's accuracy would drop when the details of vehicles released in recent times are given as input. Thus, in the future we propose to use the latest dataset set containing vehicle information to help train the model. We also plan to use other classification algorithms such as SVM and Decision Tress instead of Random Forest and measure if any accuracy gain occurs. Finally, we propose to scale the machine learning model to also analyse the performance of a larger range of vehicles.

APPENDIX

13.1 Source Code

13.1.2 car performance prediction.ipynb

```
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

import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object
# Storage. It includes your credentials.
# You might want to remove those credentials before you share the
# notebook.
cos_client = ibm_boto3.client(service_name='s3',

ibm_api_key_id='rZU37akB7V0FzI_8dLJOwQTVUOArVuSLsI6Tem7xIPQW',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-
storage.appdomain.cloud')

bucket = 'mlprojectdeployment-donotdelete-pr-rc2axqos7hwqmu'
object_key = 'car performance.csv'

body =
cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-
like object
if not hasattr(body, "__iter__"): body.__iter__ =
types.MethodType( __iter__, body )

dataset = pd.read_csv(body)
dataset.head()
# Importing Libraries
type(dataset)
```

```

# Finding missing data

dataset.isnull().any()
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.
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
# Separating 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=0.1,
random_state=0)
we are splitting as 90% train data and 10% test data
# decision tree regressor

from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor(random_state=0,criterion="absolute_error
")
dt.fit(x_train,y_train)
y_pred=dt.predict(x_test)
y_pred
y_test
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))
!pip install ibm_watson_machine_learning
from ibm_watson_machine_learning import APIClient

```



```

wml_credentials={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"vrRTRoUfZtNiDow3UtMXzdbcplUJWLy0NAh7osrM6S2U"
}
client=APIClient(wml_credentials)
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    #print(space)
    return(next(item for item in space['resources'] if
item['entity']['name'] == space_name)['metadata']['id'])
space_uid = guid_from_space_name(client, 'models')
print("Space UID = " + space_uid)
client.set.default_space(space_uid)
client.software_specifications.list()
software_spec_uid=client.software_specifications.get_uid_by_name(
"runtime-22.1-py3.9")
software_spec_uid
MODEL_NAME="Car performance Predictor"
DEPLOYMENT_NAME="Model Deployment"
DEMO_MODEL=dt
model_props={
    client.repository.ModelMetaNames.NAME:MODEL_NAME,
    client.repository.ModelMetaNames.TYPE:'scikit-learn_1.0',

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_
uid

}
model_details=client.repository.store_model(
model=DEMO_MODEL,
meta_props=model_props,
training_data=x_train,
training_target=y_train
)
model_id=client.repository.get_model_uid(model_details)
model_id
x_train[0]
dt.predict([[8.000e+00, 4.540e+02, 2.200e+02, 4.354e+03,
7.000e+01, 1.000e+00]])

```

13.1.2 Source end point.py

```

import requests
import json

```

NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.

```

API_KEY = "vrRTRoUfZtNiDow3UtMXzdbcplUJWLy0NAh7osrM6S2U"
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}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"field": [['cylinders', 'displacement', 'horsepower', 'weight', 'model
year', 'origin']], "values": [[8,330,130,3504,70,1]]}]

response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/b2e45034-d7b4-4ce0-a75a-
d936e0a01576/predictions?version=2022-11-17', json=payload_scoring,
    headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
print(response_scoring.json())

```

13.1.3 app.py

```

import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
from gevent.pywsgi import WSGIServer
import os
import requests
#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))

port = os.getenv('VCAP_APP_PORT','8080')

@app.route('/predict_api',methods=['POST'])
def predict_api():
    """
    For direct API calls through 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.secret_key = os.urandom(12)
    app.run(debug=True,host='0.0.0.0',port=port)

```

13.1.4 index.py

```

<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">
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<link rel="shortcut icon" href="{{ url_for('static', filename='css/IMG5.JPG') }}">

<div class="navbar">
    <h1><p style="font-family: 'Girassol', cursive ;">PREDICT YOUR VEHICLE'S
PERFORMANCE</p></h1>
</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="fadeInDown">
                <form action="{{ url_for('y_predict') }}" method="post" style="padding: 10px;">
                    <input type="text" name="Cylinders" placeholder="No.of cylinders (count)" required="required"
/>
                    <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>
            </div>
        </section>
    </div>
</div>

```

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

13.2 GitHub & Project demo Link

<https://github.com/IBM-EPBL/IBM-Project-25695-1659970915>

flask deploy: python-flask-app-cpp-boring-bongo-fa.mybluemix.net

Project demo link:

https://drive.google.com/file/d/1CeRstsIN8rYLKATtQcBQ_slGFq4QExH0/view?usp=share_link