MACHINE LEARNING BASED VEHICLE PERFORMANCE ANALYZER

Bonafede record of work done by

Kaviyarasu K (TL)	(732719104015)
Ashok Bharadhi K	(732719104004)
Akhil Kumar J	(732719104013)
Mahesh R	(732719104018)

BACHELOR OF ENGINEERING

BRANCH: COMPUTER SCIENCE ENGINEERING

**SRI SHANMUGHA COLLEGE OF ENGINEERING
AND TECHNOLOGY**

NOVEMBER 2022

CHAPTER 1

INTRODUCTION

1.1 Project Overview

It's a significant and intriguing challenge to predict a car's performance level. The primary objective of the current study is to forecast automobile performance to enhance specific vehicle behavior. 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 car 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, CO_2 emission reduction, among others. Thus, the main purpose of the project is to predict the performance of the car to improve certain behaviors of the vehicle using various machine learning algorithms.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem

The potential for processing car sensing data has increased in recent years due to the development of new technologies. Having this type of data is important, for instance, to analyze the way drivers behave when sitting behind the steering wheel. Very little has been done to analyze car usage patterns based on car engine sensor data, and, therefore, it has not been explored to its full potential by considering all sensors within a car 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, CO_2 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 car to improve certain behaviors of the vehicle using various machine learning algorithms"

2.3 References

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

In the paper "Machine Learning Based Real-Time Vehicle Data Analysis for Safe Driving Modeling" Machine learning approach to analyze and predict the vehicle performance in real time. The focus is on analyzing 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 analyzed 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, CO_2 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 unlabeled 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.

Expectation-Maximization:

The expectation-maximization algorithm is an approach for performing maximum likelihood estimation in the presence of latent variables. It does this by first estimating the values for the latent variables, then optimizing the model, then repeating these two steps until convergence. It is an effective and general approach and is most used for density estimation with missing data, such as clustering algorithms like the Gaussian Mixture Model.

Hierarchical Clustering:

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled 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. Besides, the paths from root to leaf represent classification rules.

KNN:

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

Multilayer Perceptron:

A multilayer perceptron is a fully connected class of feedforward artificial neural network. it uses proximity to make classifications or predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of

features given the value of the class variable. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

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 setR andom forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Support Vector Mechanism:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine.

2.3.3 Driving Behavior Analysis Using Machine and Deep Learning Methods for Continuous Streams of Vehicular Data:

The paper "Driving Behavior Analysis Using Machine and Deep Learning Methods for Continuous Streams of Vehicular Data" authored by Nikolaos Peppes, Theodoros Alexakis, Evgenia Adamopoulou, Konstantinos Demestichas aims to combine well-known machine and deep learning algorithms together with open-source-based tools to gather, store, process, analyze and correlate different data flows originating from vehicles

Machine Learning Algorithms for Classification:

Support Vector Mechanisms (SVM):

Support vector machines is a supervised machine learning algorithm used for both classification and regression. SVM classifies data points based on the hyperplane in an N – dimensional space. The separation function in support vector classification is a linear combination of kernels linked to the support vector.

Decision Tree-Based Algorithms:

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 nonparametric 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 treelike 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. Besides, the paths from root to leaf represent classification rules. Three decision

tree-based models, including decision tree (DT), extra trees (ExT), and random forest, were evaluated in relation to various learning methods.

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. R andom forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Deep Learning Model:

RNN-based algorithms:

RNN-based models have been used widely nowadays due to its robustness and capability to handle nonlinear data even with its typically structured, single hidden layer, or advanced structured, multiple hidden layers. RNN includes three layers: input, hidden, and output layers. In case of increasing complexity of the problem, the number of layers will rise, and the computational resources will consequently also rise. Here, both the mentioned structures of the RNN-based models were utilized for predicting the Driving Behavioral Analysis.

Multilayer Perceptron:

A multilayer perceptron is a fully connected class of feedforward artificial neural network. it uses proximity to make classifications or predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

CHAPTER 3

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 analyzer.
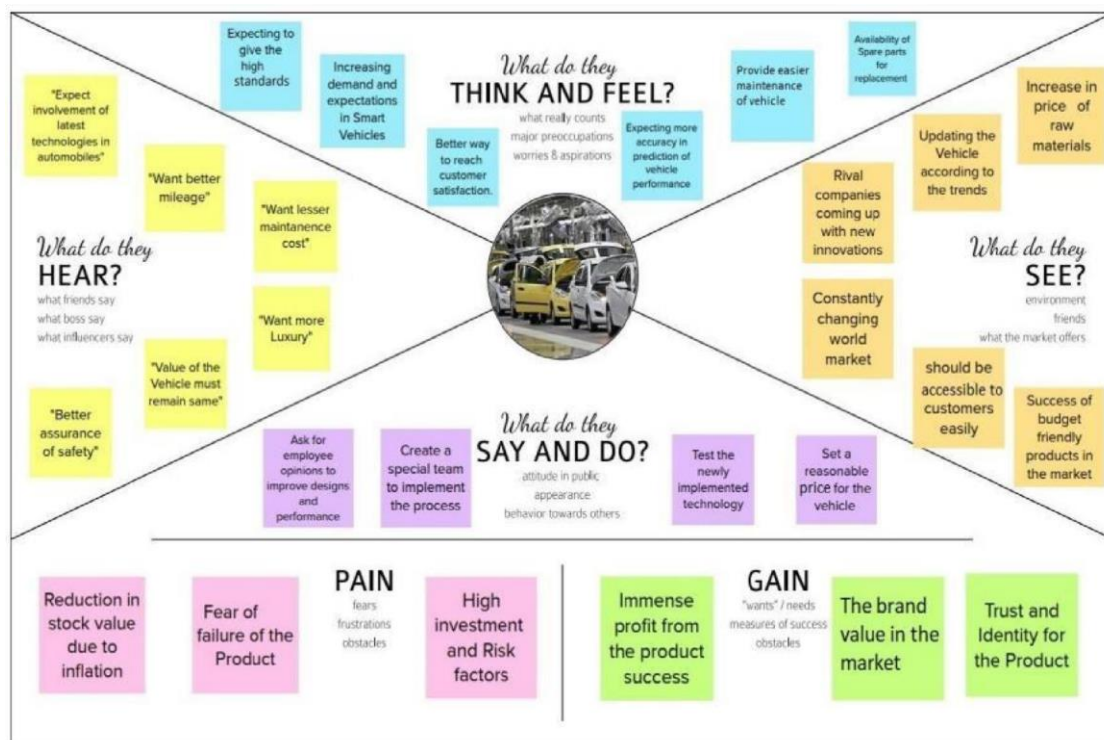

Figure 3.1 – Empathy Map

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
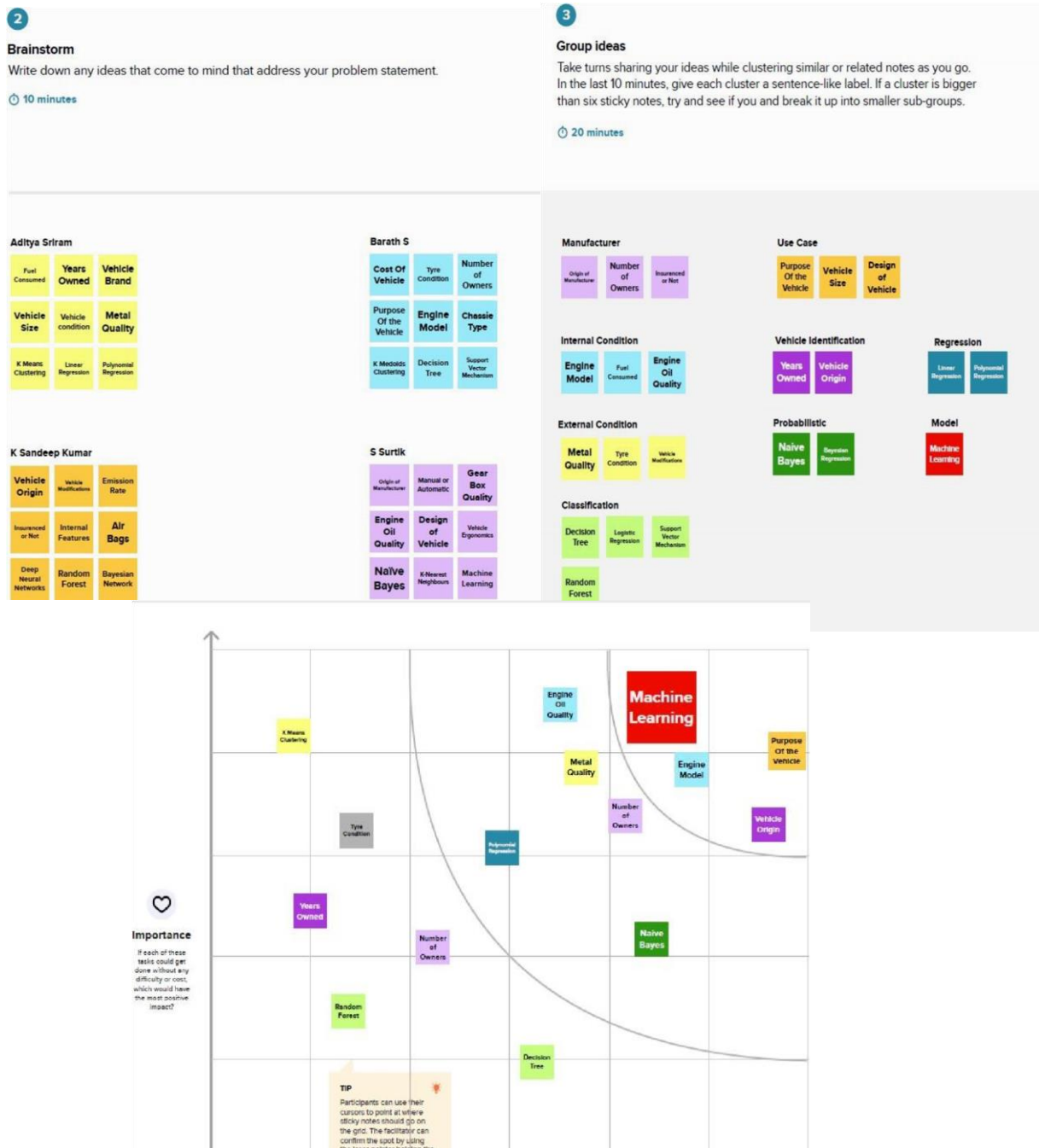
based vehicle performance analyzer.

Figure 3.2 – Ideation & Brainstorming

3.3 Proposed Solution

S.No.	Parameter	Description
1	**Problem Statement (Problem to be solved)**	The number of motor cars being registered for commercial and non-commercial purposes daily is enormous and yet continues to rise at an alarming rate due to the changing demographics of the population. This directly and unmistakably affects how much fossil fuel is used globally and the following environmental impacts, which are very concerning now. Promising outcomes are anticipated from several current initiatives from diverse research fields to address this worldwide challenge. One such effort is this initiative, which uses machine learning approaches to map the elements impacting vehicle performance in terms of fuel efficiency.

| 2 | **Idea / Solution description** | It is crucial to analyse the elements utilising a variety of well-known machine learning methods, such as Multiple Linear Regression, XGBoost, Support Vector Machine, Artificial Neural Network, and Random Forest. Here, we take a performance in terms of mileage. We will create the models, utilising various techniques and neural networks, to resolve this issue. Then, we'll compare which algorithm accurately forecasts automobile Fuel efficiency. |
| 3 | **Novelty / Uniqueness** | In our concept, we consider the number of cylinders, displacement, horsepower, weight, model year, country of origin, bore, stroke, compression ratio, wheelbase, etc. to assess vehicle performance. The sensitivity of our metric is expected to rise when additional data are added to the model to make it fit. Our model will be exposed to a wider range of potential outcomes, allowing it to identify more data that is comparable to the previously unknown ones. |

4	**Social Impact / Customer Satisfaction**	The primary goal of this vehicle performance analyzer is to significantly reduce vehicle emissions. The air quality in our area will undoubtedly improve because of the decreased emission of toxic gases. The ability for customers to understand more about their own automobiles The vehicle's relative environmental friendliness will reduce costs.
5	**Business Model (Revenue Model)**	A very user-friendly interface and thorough information regarding the vehicle performance will be provided by this system. The knowledge discovered from the system could be used by car manufacturers to design cars in future to mitigate the fuel consumption.
6	**Scalability of the Solution**	This technology will analyse the performance of the vehicle regardless of the kind or number of cars. The system allows simultaneous access from several users, and it processes the findings immediately

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 analyzer.
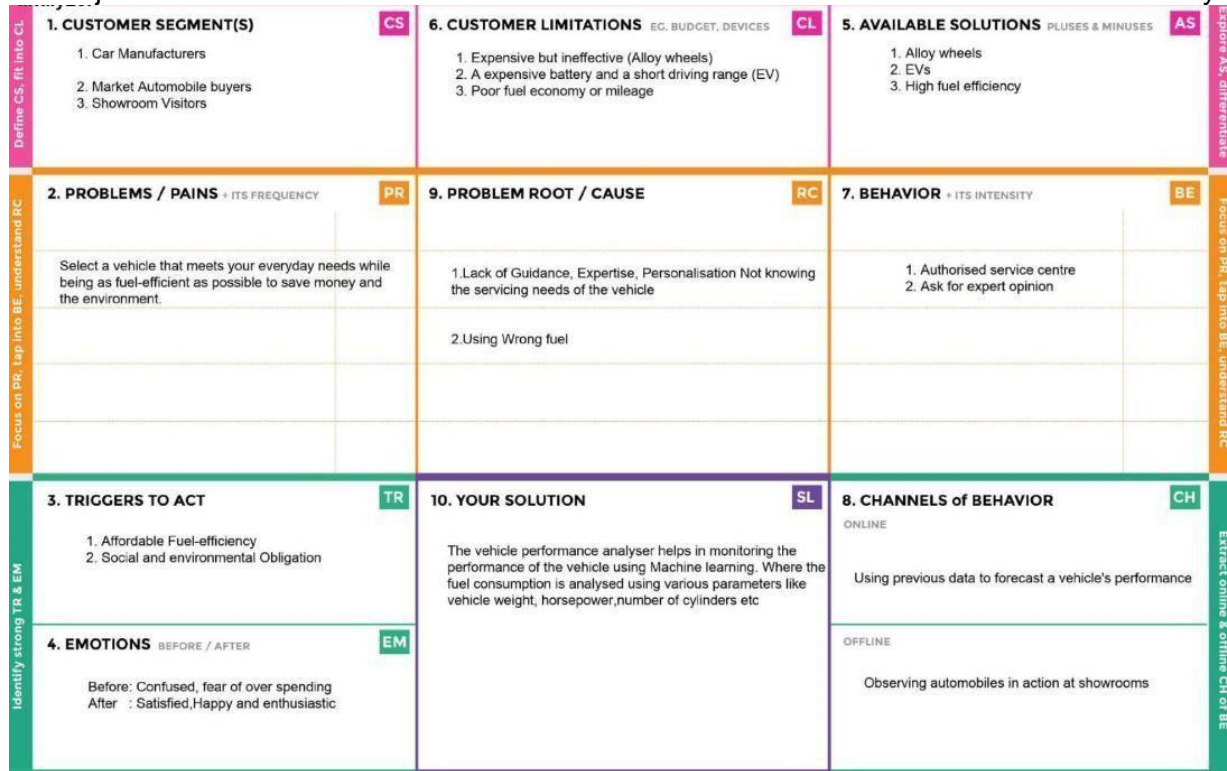
Figure 3.4 – Problem Solution Fit

CHAPTER 4

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 preprocessing	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 4.1 – Functional Requirements

4.2 Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The analyzer 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.

Table 4.2 – Non-Functional Requirements

CHAPTER 5

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.
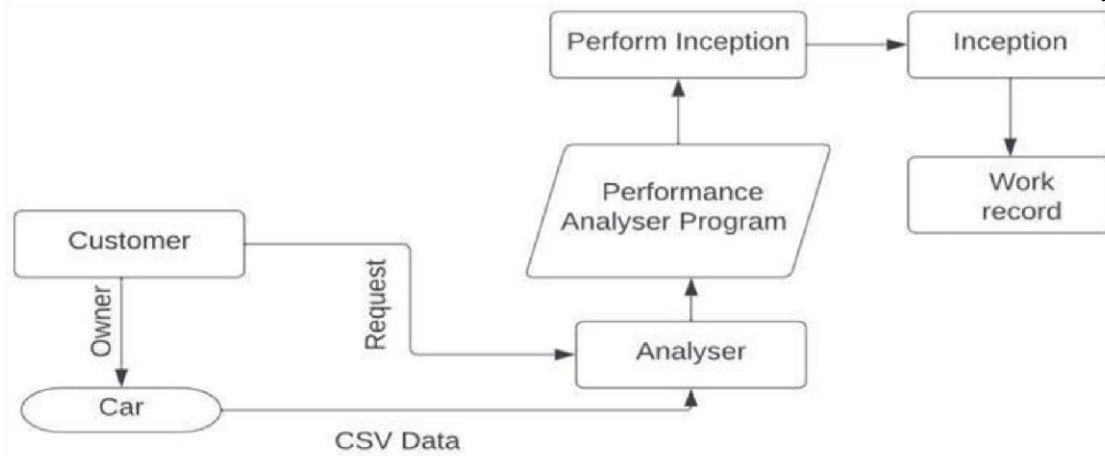
Figure 5.1 – Dataflow Diagram
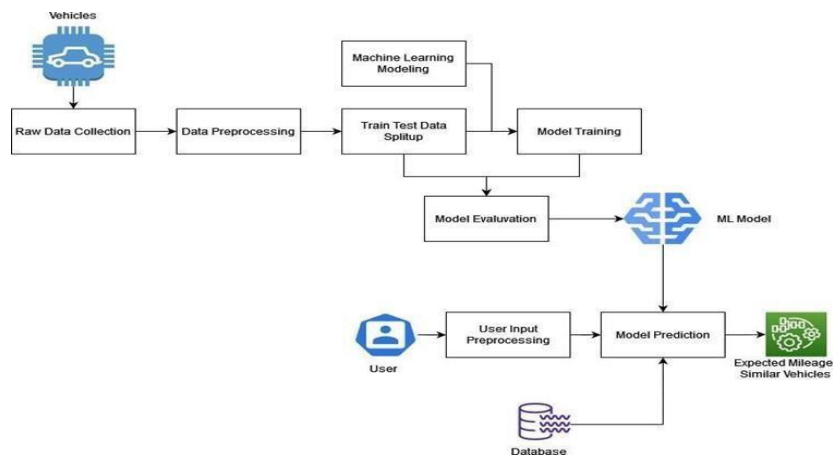
5.2 Technical Architecture

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.	React Js
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.	Database	Databases contain user information such as name, email, vehicle basic information, mileage predicted over time.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2
7.	External API-1	Vehicle Details Database	https://api.auto-data.net/
8.	Machine Learning Model	The machine learning model is used to predict mileage from the user inputs	Regression Modelling.
9.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Core i5, 8GB RAM Cloud Server Configuration:	Local, Docker

Table 5.2.1 – Components and Technologies

5.2.2 Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	React Js, Flask, Sci-kit Learn	Javascript, Python
2	Security Implementations	Identity and Access Management, OAUTH, WAF	IBM Cloud
3	Scalable Architecture	3 Tier Architecture, Model-View-Controller implementation	Model - SQL DB, View - ReactJS, Controller - Flask Server
4	Availability	Proxy servers, Load Balancers to help balance traffic among servers to help improve uptime	IBM Cloud load balancers
5	Performance	The frontend is detached from the Business logic server reducing requests sent to the server.	Nginx proxy

Table 5.2.2 – Application Characteristics

5.3 User Stories

User Type	Functional Requirements	User Story Number	User Story/Task	Acceptance Criteria	Priority	Release
Customer	Access the Webpage	USN -1	Anyone can access the webpage to check the specifications of the vehicle	I can access my webpage online at any time	High	Sprint-1

Customer	Performance of the Vechicle	USN - 2	As per the usage of the user, the performance of the vehicle should be predictable.	Prediction can be done in an easy way.	High	Sprint-2
Customer	Accuracy to check the performance and health of the car	USN -3	By using our prediction, it helps to check the health of the car.	The efficiency of the car can be predicted.	High	Sprint-3

Table 5.3 – User Stories

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirements	User Story Number	User Story	Story Points	Priority	Team Members
Sprint 1	Data preprocessing	USN -1	As a user, I can process raw data and perform manual analysis	30	Low	Team Lead
Sprint 2	Model Building	USN – 2	As a user, I get to predict performance of the vehicle using the given data	20	Low	Team Member
Sprint 3	Web Page Design	USN – 3	As a user I can view the website and I can get the predicted performance of the vehicle using the given data	30	High	Team Lead
Sprint 4	Result	USN -4	As a user, I expect the prediction is accurate	20	High	Team Member

Table 6.1 – Sprint Planning

6.2 Sprint Delivery Schedule

Sprint	Story Points	Duration (days)	Sprint Start Date	Story Points Completed	Sprint Release Date
Sprint 1	30	6	24 Oct 2022	30	29 Oct 2022
Sprint 2	20	6	31 Oct 2022	20	05 Nov 2022
Sprint 3	20	6	07 Nov 2022	20	12 Nov 2022
Sprint 4	30	6	14 Nov 2022	30	19 Nov 2022

Table 6.2 – Sprint Delivery Schedule

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)

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

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.
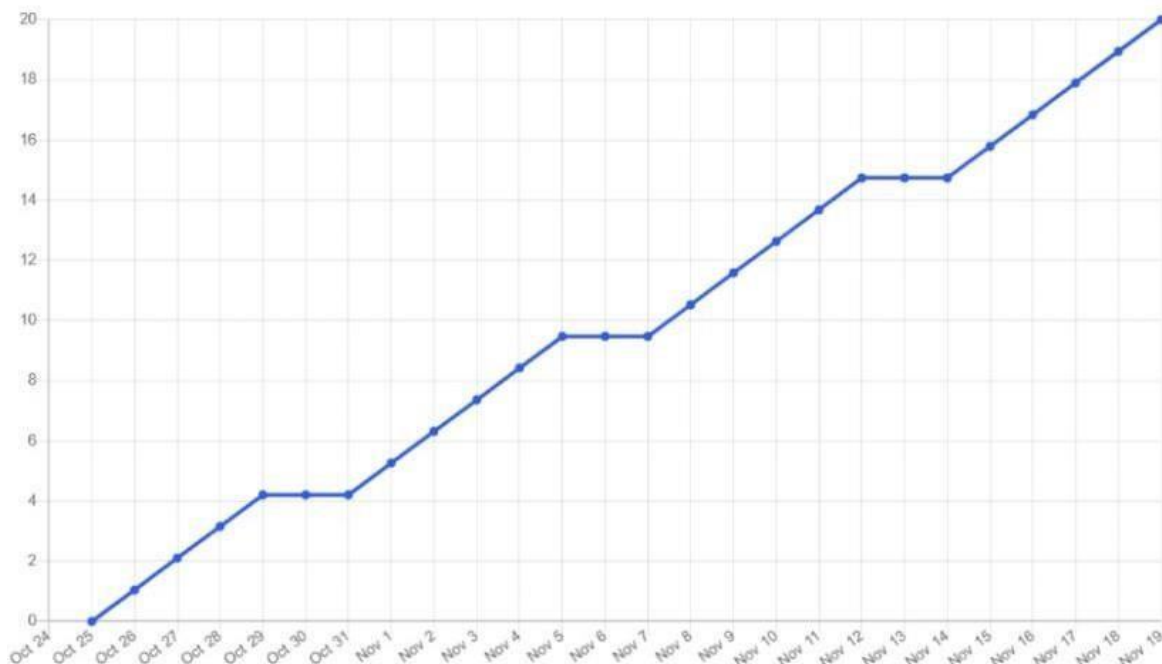
CHAPTER 7

CODING AND SOLUTION

7.1 Feature 1

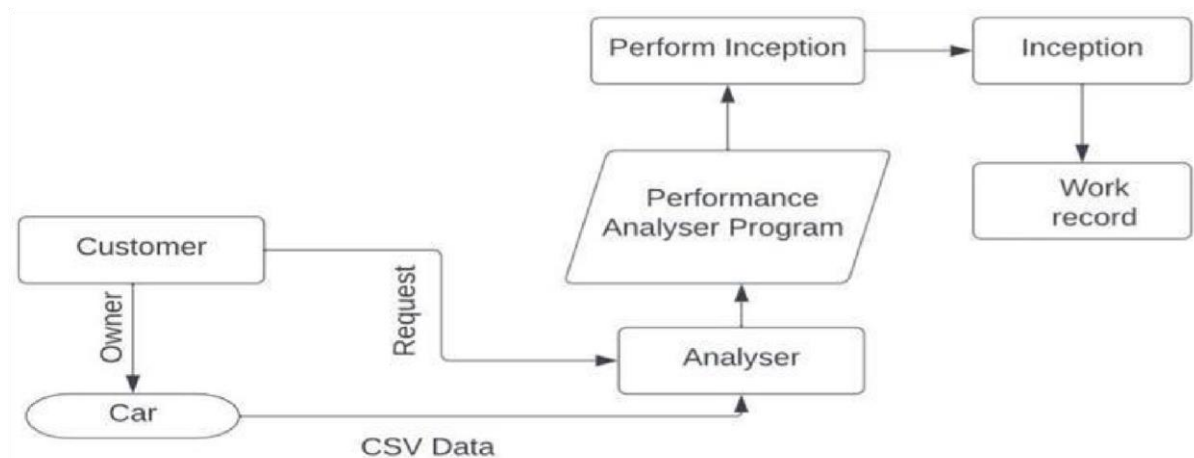| FR No.	Feature	Description
FR-1	Enter the input	Get input through the form
FR-2	User Essential	Predict the performance of the vehicle
FR-3	Data preprocessing	Sample dataset for training purpose
FR-4	User input Evaluation	Evaluating the given user values
FR-5	Prediction	Fuel consumption and efficiency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

Table 7.1 – Description for Feature 1

Figure 7.1 – Dataflow Diagram for Feature 1

The classification happens is 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 is 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.

CHAPTER 8

TESTING

8.1 Test Cases

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
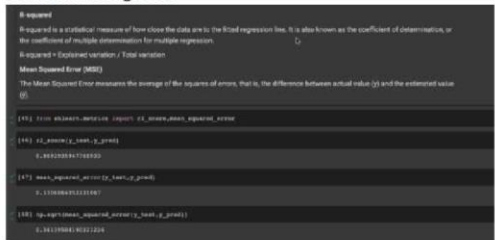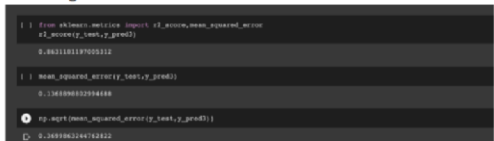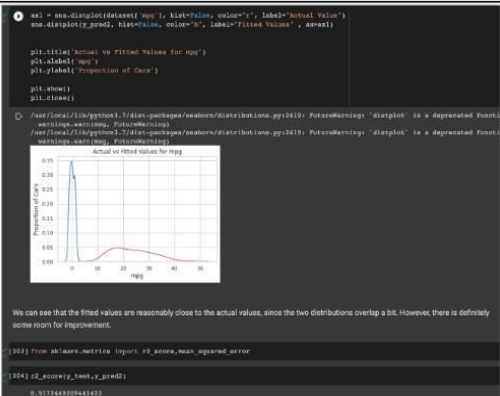
S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -	Decision tree regressor
			Random forest regressor Linear regression
	Accuracy	Training Accuracy - 0.91724492094	

Figure 9.1 – Performance Metrics

CHAPTER 10

PROS AND CONS

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

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

CHAPTER 11

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.

CHAPTER 12

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.

CHAPTER 13

APPENDIX

13.1 Source Code

13.1.1 car performance prediction.ipbyn

```python
import pandas as pd import
numpy as np import
matplotlib.pyplot as plt import
seaborn as sns

import statsmodels.formula.api as smf

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='Uede0uog_DlYeHmTn0uQW3GYQhYgYHbY1kvWgQybaYM1',
      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 = 'vehicleperformanceprediction-donotdelete-pr-kj6qz2159y6996'
  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()

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=Tr
ue) 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
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)
sns.pairplot(dataset,diag_kind='kde') #pairplot represents pairwise relation
across the entire dataframe.
plt.show()
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)
from scipy import stats
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) 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) 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) 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) 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) 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) 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)
test=smf.ols('mpg~cylinders+displacement+horsepower+weight+acceleration+or
igi n',dataset).fit() test.summary()



x=dataset[['cylinders','displacement','horsepower','weight','mo
del year','origin']].values x
y=dataset.iloc[:,0:1].values
y
from sklearn.preprocessing import StandardScaler
sd = StandardScaler() x_train =
sd.fit_transform(x_train) x_test =
sd.fit_transform(x_test) y_train =
sd.fit_transform(y_train) y_test =
sd.fit_transform(y_test)
```

```
 x_train from sklearn.model_selection import train_test_split
 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0
 .2)
 from sklearn.ensemble import RandomForestRegressor
 rf= RandomForestRegressor(n_estimators=10)
 rf.fit(x_train,np.ravel(y_train)) x_train.shape


 !pip  install  ibm-watson-machine-learning  from
 ibm_watson_machine_learning  import  APIClient
 wml_credentials = {
     "url":"https://us-south.ml.cloud.ibm.com",
     "apikey":"k0ToNjB4fREMsVxEr0C3pjHT0bNJzgZvVt1S0SikVpMJ
",  }
 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("default_py3.8")
 software_spec_uid


 !pip install -U pyspark==2.1.2.
 sofware_spec_uid = client.software_specifications.get_id_by_name("runtime-
 22.1-py3.9") metadata = { client.repository.ModelMetaNames.NAME:
 'Gradient', client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
 client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:
 sofware_spec_uid
 }
 published_model = client.repository.store_model(
 model=rf, meta_props=metadata)
 published_model x_train[0] rf.predict([[4.000e+00, 1.210e+02,
 7.600e+01, 2.511e+03, 7.200e+01,
 2.000e+00]])
```

13.1.2 sourcing end point.py

```
  import requests


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

        API_KEY = "k0ToNjB4fREMsVxEr0C3pjHT0bNJzgZvVt1S0SikVpMJ" 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": [[8,307,130,3504,70,1]]}]}


response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/3950d430-efb8-43ea-b408-

28233df071d7/predictions?version=2022-11-13', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
print(response_scoring.json())
```

13.1.3 app.py

```
from flask import
Flask, request,
render_template

        import requests


        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("xtest= ", x_test)
            # sc = load('scalar.save')
```

```python
    # NOTE: you must manually set API_KEY below using information
retrieved from your IBM Cloud account.
    API_KEY = "k0ToNjB4fREMsVxEr0C3pjHT0bNJzgZvVt1S0SikVpMJ"
    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": x_test}]}


    response_scoring = requests.post(
        'https://us-south.ml.cloud.ibm.com/ml/v4/deployments/3950d430-
efb8-43ea-b408-28233df071d7/predictions?version=2022-11-13',
json=payload_scoring, headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring      response")      prediction     =
    response_scoring.json() print(prediction) output =
    prediction['predictions'][0]['values'][0] output =
    output[0] print(output) if (output <= 9):
        pred = "Worst performance with mileage " + str(output)
    if (output > 9 and output <= 17.5): pred = "Low
        performance with mileage " + str(output)
    if (output > 17.5 and output <= 29): pred = "Medium
        performance with mileage " + str(output)
    if (output > 29 and output <= 46):
        pred = "High performance with mileage " + str(output)
    if (output > 46):
        pred = "Very high performance with mileage " + str(output)
```

26

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



        if_name_== "__main_": app.run(debug=True)
```

13.1.4 index.html

```html
<div
class="wrapper
fadeInDown">
                <div id="formContent">
                  <!-- Tabs Titles -->
                 <section class="date">
                  <!-- Icon -->


                  <div class="fadeInDown">
                  <form action="{{ url_for('y_predict')}}"method="post">
                      <label style="font-size:30px;">INPUT</label>
                      <br>
                    <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 id="formFooter">
```

27

```
                        <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-52339-1660996981