

TEAM ID: PNT2022TMID14089

TEAM MEMBERS:

1. ALEN A (TEAM LEADER) 2. PRADEEP P

3. HEMKANTH P 4. KING DINAKARAN R.K

MACHINE LEARNING BASED VEHICLE PERFORMANCE ANALYZER

PROJECT REPORT

1. INTRODUCTION

1.1 Project Overview

The monitoring of car performance, especially gas consumption, has so far been approached only very superficially. A typical fuel gauge, when closely monitored, shows an extremely nonlinear relationship between needle movement and fuel consumption. Inaccuracies occur especially in the range of critical low fuel values of 5-10% or more. In the past, due to this limitation, some luxury cars had an audible and flashing light alarm function to indicate a low fuel condition. These systems, which added to the existing fuel level, have no more accuracy than the fuel level monitor alone. In recent years, with the availability of computer techniques and reliable and less expensive computer equipment, a number of systems have been developed to provide some what more accurate information about vehicle performance.

1.2 Purpose

The solution mainly aims on to predict the miles per gallon value based on the given input values that effect the performance of the vehicle.

2. LITERATURE SURVEY

2.1 Existing problem

predicting the performance level of cars is an important and interesting problem. The main goal of the current study is to predict the performance of the car to improve certain behavior of the vehicle. This can significantly help to improve the systems fuel consumption and increase the efficiency. The performance analysis of the car based on the engine type, no of engine cylinders, fuel type and horsepower etc. These are the factors on which the health of the car can be predicted. It is an on-going process of obtaining, researching, analyzing and recording the health based on the above three factors. The performance objectives like mileage, dependability, flexibility and cost can be grouped together to play a vital role in prediction engine and engine management system. This approach is the very important step towards understanding the vehicles performance.

2.2 References

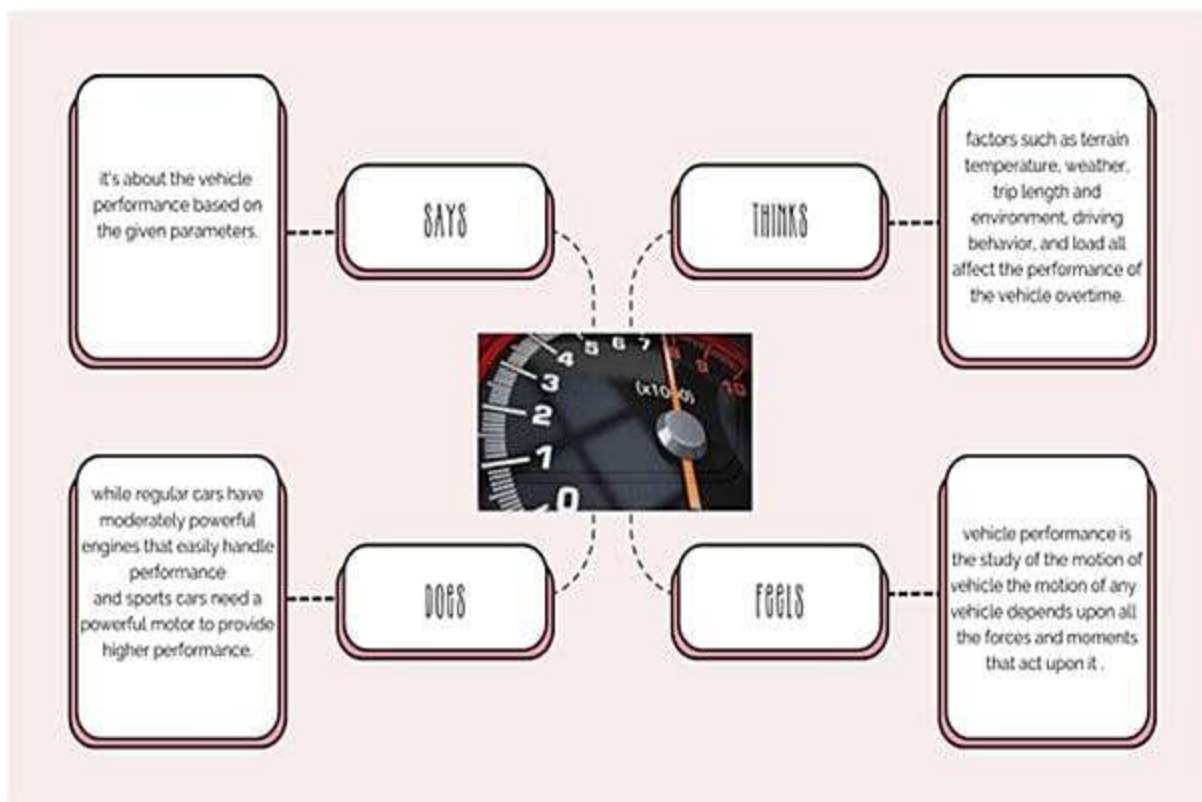
1. Singh D, Singh M., "Internet of Vehicles for Smart and Safe Driving", International Conference on Connected Vehicles and Expo (ICCVE), Shenzhen, 19 - 23 Oct., 2015. (This paper has discussed about smart transportation services in cloud (CloudSTS) for safety and convenience. STS provide driver centric board services in the cloud networks. STS composed of Vehicle to WiFi networks (V to WiFi), Vehicle to Cloud Network (V to CN), Vehicle to Vehicle (V to V), and Cloud Network to service provider (CN to SP). The idea is to utilize the (WiFi enabled) Smart Highways and 3D camera enabled dash board navigation device to enhance accident prevention / monitoring and control.)
2. Zhang, Y., Lin, W., and Chin, Y., "Data - Driven Driving Skill Characterization: Algorithm Comparison and Decision Fusion," SAE Technical Paper 2009-01-1286, 2009, <https://doi.org/10.4271/2009-01-1286>. Azevedo, C. L Cardoso. (By adapting vehicle control systems to the skill level of the driver, the overall vehicle active safety provided to the driver can be further enhanced for the existing active vehicle controls, such as ABS, Traction Control, Vehicle Stability Enhancement Systems. As a follow-up to the feasibility study in, this paper provides some recent results on data driven driving skill characterization. In particular, the paper presents an enhancement of discriminant features, the comparison of three different learning algorithms for recognizer design, and the performance enhancement with decision fusion.
3. The paper concludes with the discussions of the experimental results and some of the future work. J. E. Meseguer, C. T. Calafate, J. C. Cano and P. Manzoni, "Driving Styles: A smartphone application to assess driver behaviour," 2013 IEEE Symposium on Computers and Communications (ISCC), Split, 2013, pp. 000535 - 000540. doi:10.1109/ISCC.2013.6755001. (The Driving Styles architecture integrates both data mining techniques and neural networks to generate a classification of driving styles by analyzing the driver behavior along each route. In particular, based on parameters such as speed, acceleration, and revolutions per minute of the engine (rpm), we have implemented a neural network based algorithm that is able to characterize the type of road on which the vehicle is moving, as well as the degree of aggressiveness of each driver. The final goal is to assist drivers at correcting the bad habits in their driving behaviour, while offering helpful tips to improve fuel economy.

2.3 Problem Statement Definition

It is an important to analyse the factors using number of wellknown approaches of machine learning algorithms like linear regression, decision tree and random forest to improve the vehicle performance efficiency. The range, durability and longevity of automotive traction batteries are 'hot topics' in automotive engineering. And here we consider a performance in mileage. To solve this problem, we will develop the models, using the different algorithms and neural networks. We will then see which algorithm predicts car performance(Mileage) with higher accuracy.


3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas






3.2 Ideation & Brainstorming

Template




Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.


 10 minutes to prepare
 1 hour to collaborate
 2-8 people recommended


[Share template feedback](#)




Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.


 10 minutes

 **Team gathering**


Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.


 **Set the goal**

Think about the problem you'll be focusing on solving in the brainstorming session.

 **Learn how to use the facilitation tools**


Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) 




Define your problem statement

Predicting the performance level of cars is an important and interesting problem. This can significantly help to improve the system's fuel consumption and increase efficiency.

 5 minutes


PROBLEM


Predicting the performance level of cars is an important and interesting problem.





Key rules of brainstorming


To run an smooth and productive session


 Stay in topic.

 Encourage wild ideas.

 Defer judgment.

 Listen to others.

 Go for volume.

 If possible, be visual.

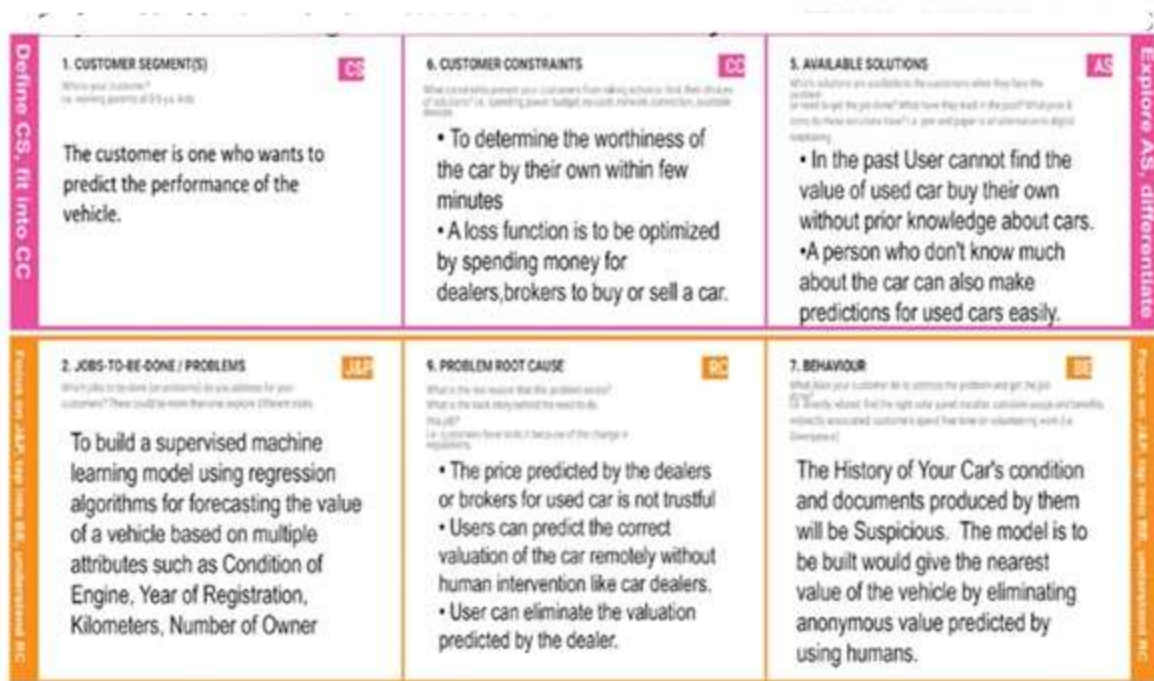
3.3 Proposed Solution

Proposed Solution Template:

The project team shall fill in the following information in the proposed solution template.

<u>S.No.</u>	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 into the web-based application where the user is notified of the status.
3.	Novelty / Uniqueness	By using the optimal regression model to predict the value in 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, and kilometers 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 a few seconds.
6.	Scalability of the Solution	Machine learning approaches, this project proposed a scalable framework for predicting values for a <u>different types of used cars</u> . The solution given by the trained system is efficient and is a nearly accurate value of the vehicle.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	<ol style="list-style-type: none"> 1. The system doesn't require any prior technical knowledge from the user, thus even a novice user can access it. 2. The user interface would prioritize recognition over recall. 3. User friendly 4. Pay attention to internal sources of control 5. It wouldn't take long for the content to load and show (30 seconds). 6. The fields in the site would be selfexpla

NFR-2	Security	<ul style="list-style-type: none"> • Only the authenticated user will be able to use the site's services. • The database should be backed up every hour.
-------	----------	--

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
--------	-------------------------------	------------------------------------

FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Details	No of cylinders, Displacement, Horsepower, Weight, Model year, and Origin
FR-4	User Requirements	<ul style="list-style-type: none"> • Upload all essential details to the website's appropriate. • The system would extract all essential data based on the uploads. • Based on the information that was scraped, a list of every potential potential results will be delivered.

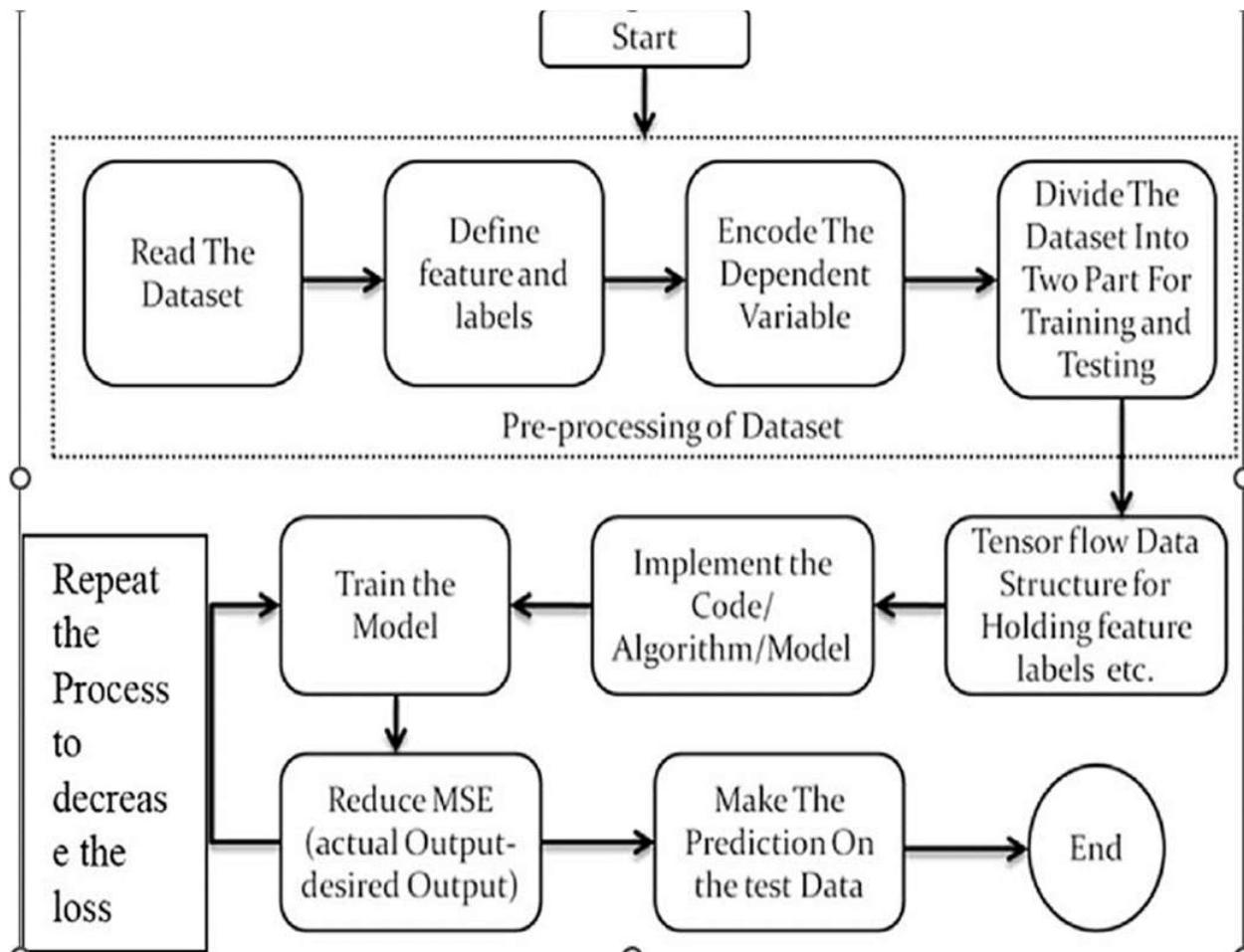
4.2 Non-Functional requirements

NFR-3	Reliability	<ol style="list-style-type: none"> 1. Due to the value of data and the potential harm that inaccurate or incomplete data could do, the system will always strive for optimum reliability. 2. The system will be operational every day of the week, 24 hours a day.
NFR-4	Performance	<ol style="list-style-type: none"> 1. The website can efficiently handle traffic by responding to requests right away. 2. A 64-kbps modem connection would take no longer than 30 seconds to see this webpage (quantitatively, the mean time)
NFR-5	Availability	<ol style="list-style-type: none"> 1. Low data redundancy 2. reduced error risk, quick and effective

NFR-6	Scalability	<ol style="list-style-type: none"> 1. A significant number of users must be able to access the system simultaneously because an academic portal is essential to the courses that use it. 2. The system will likely be most stressed during the admissions season. 3. Therefore, it must be able to handle several users at once.
-------	-------------	---

5. PROJECT DESIGN

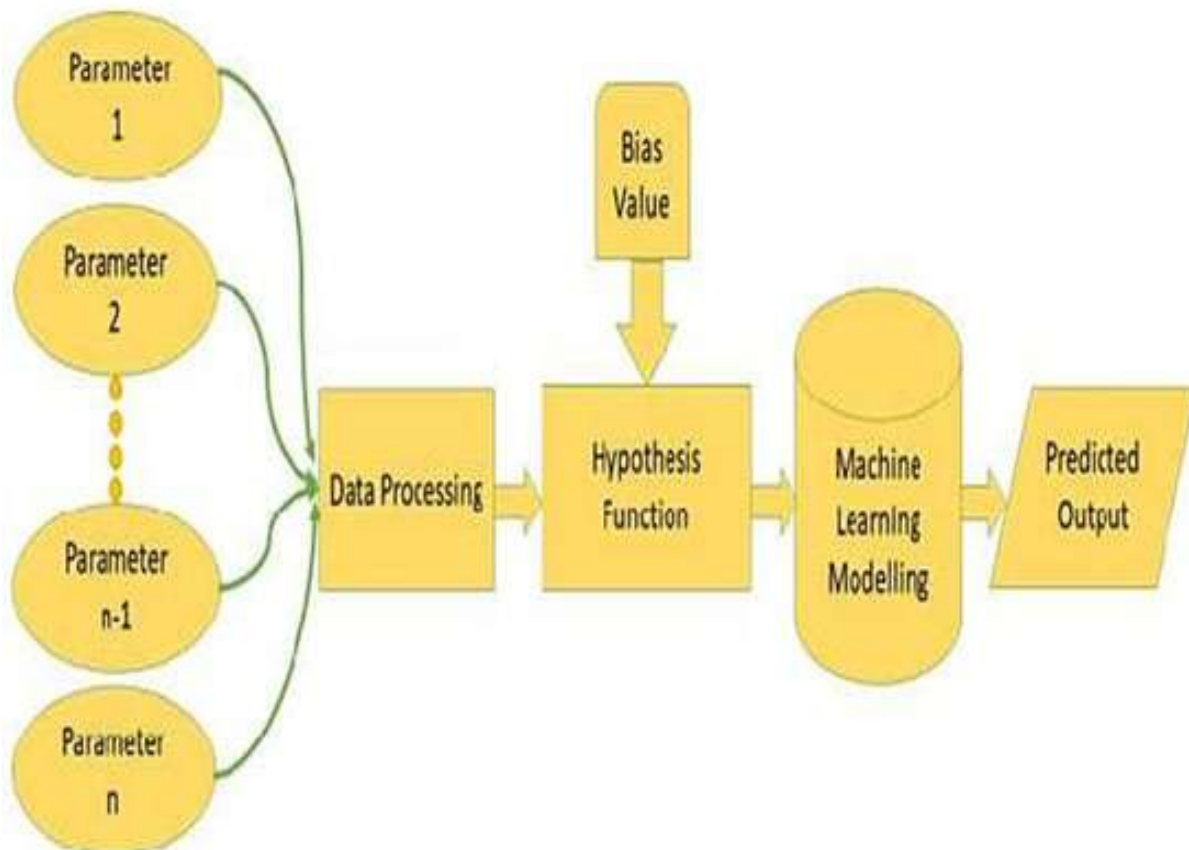
5.1 Data Flow Diagrams

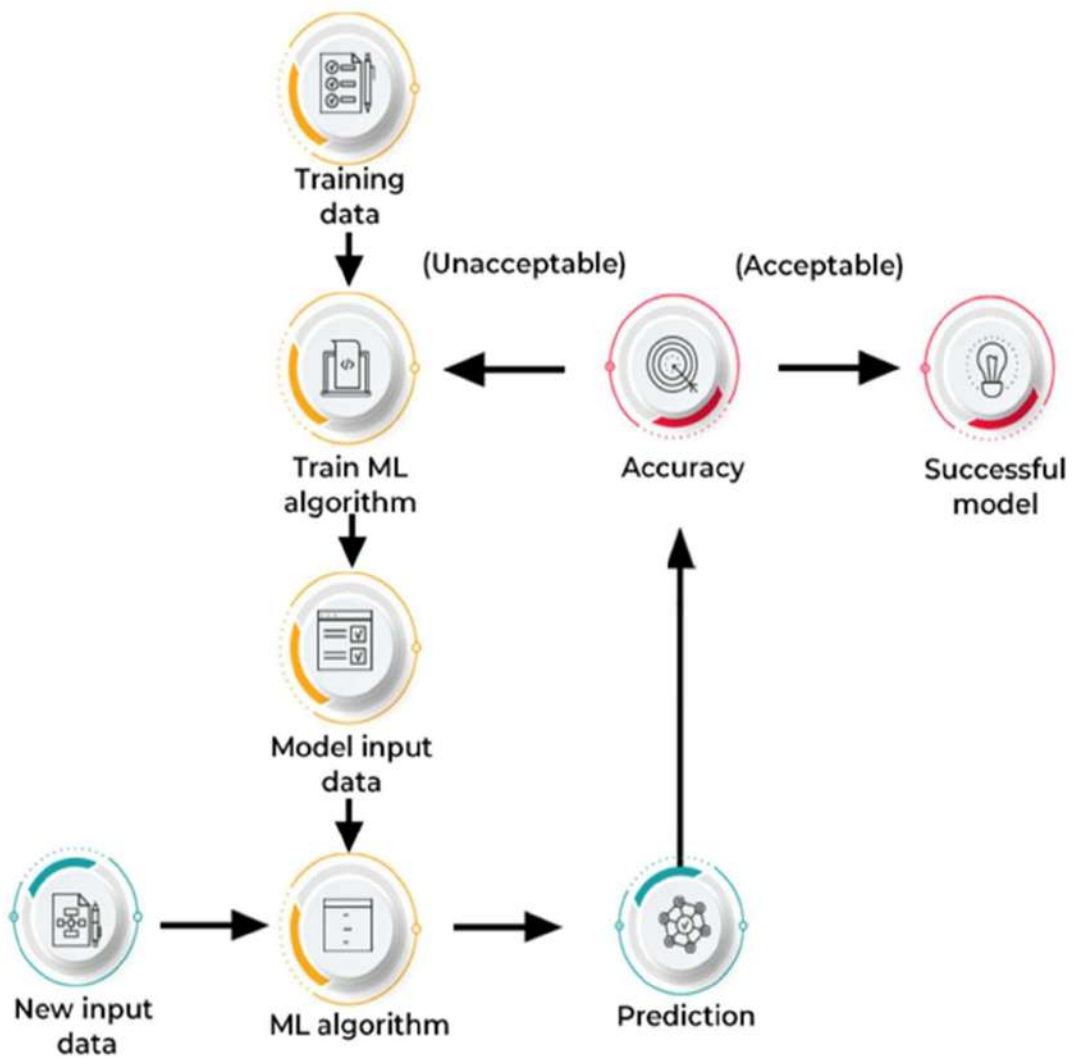


5.2 Solution & Technical Architecture

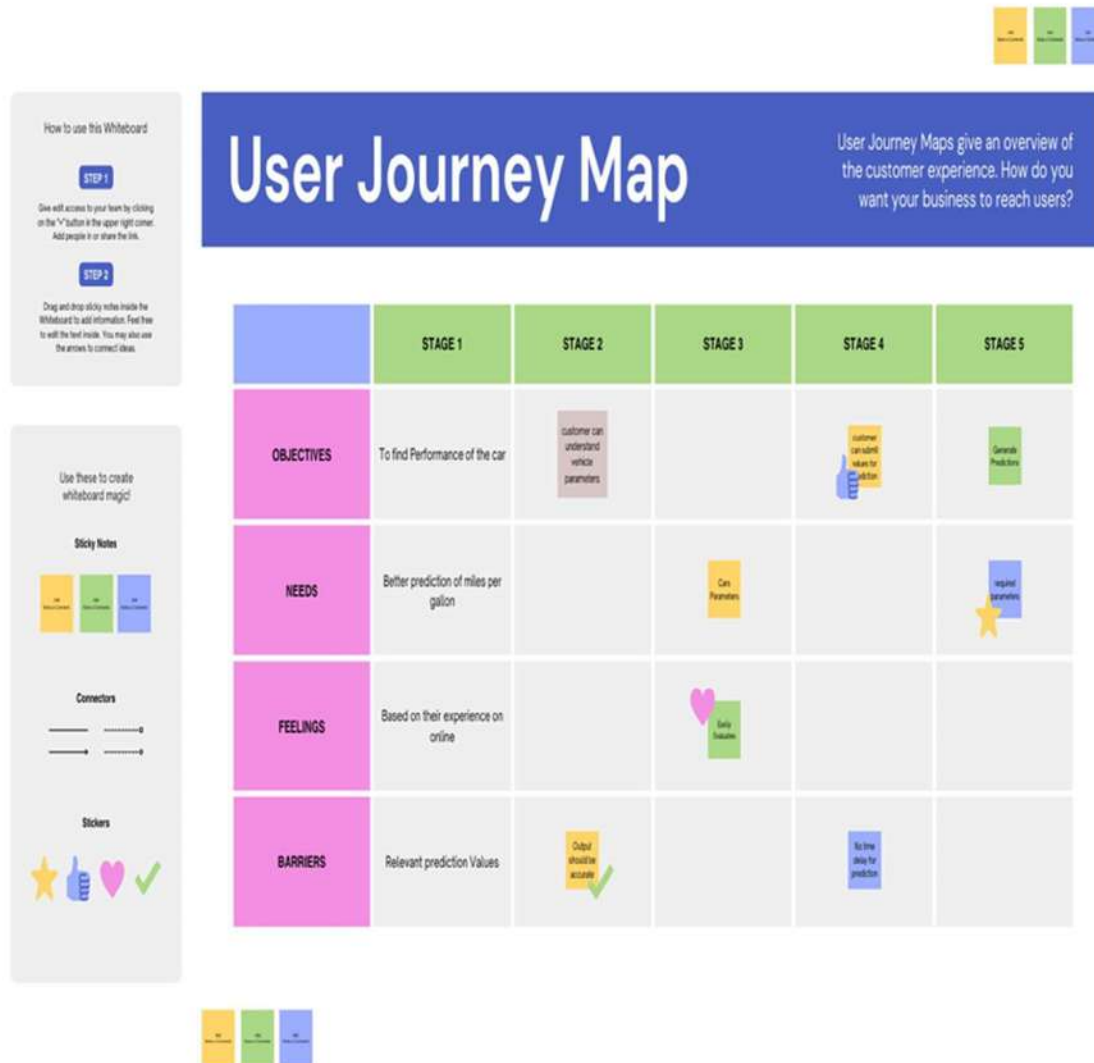
Solution:

predicting the performance level of cars is an important and interesting problem. The main goal of the current study is to predict the performance of the car to improve certain behavior of the vehicle. This can significantly help to improve the systems fuel consumption and increase the efficiency. The performance analysis of the car based on the engine type, no of engine cylinders, fuel type and horsepower etc. These are the factors on which the health of the car can be predicted. It is an on-going process of obtaining, researching, analyzing and recording the health based on the above three factors. The performance objectives like mileage, dependability, flexibility and cost can be grouped together to play a vital role in prediction engine and engine management system.





5.3 User Stories



6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Steps To Perform Predictive Analysis:

Some basic steps should be performed in order to perform predictive analysis.

Define Problem Statement:

Define the project outcomes, the scope of the effort, objectives, identify the data sets that are going to be used.

Data Collection:

Data collection involves gathering the necessary details required for the analysis. It involves the historical or past data from an authorized source over which predictive analysis is to be performed.

Data Cleaning:

Data Cleaning is the process in which we refine our data sets. In the process of data cleaning, we remove unnecessary and erroneous data. It involves removing the redundant data and duplicated data from our data sets.

Data Analysis:

It involves the exploration of data. We explore the data and analyze it thoroughly in order to identify some patterns or new outcomes from the data set. In this stage, we discover useful information and conclude by identifying some patterns or trends.

Build Predictive Model:

In this stage of predictive analysis, we use various algorithms to build predictive models based on the patterns observed. It requires knowledge of python, R, Statistics and MATLAB and so on. We also test our hypothesis using standard statistical models.

Validation:

It is a very important step in predictive analysis. In this step, we check the efficiency of our model by performing various tests. Here we provide sample input sets to check the validity of our model. The model needs to be evaluated for its accuracy in this stage.

Deployment:

In deployment we make our model work in a real environment and it helps in everyday decision making and make it available to use.

Model Monitoring:

Regularly monitor your models to check performance and ensure that we have proper results.

6.2 Sprint Delivery Schedule

Project Tracker, Velocity & Burn down Chart: (4 Marks)

Project Tracker:

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	19 Nov 2022

Velocity:

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

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

Average velocity of sprint-1: $AV = 17.8 \div 2.125$

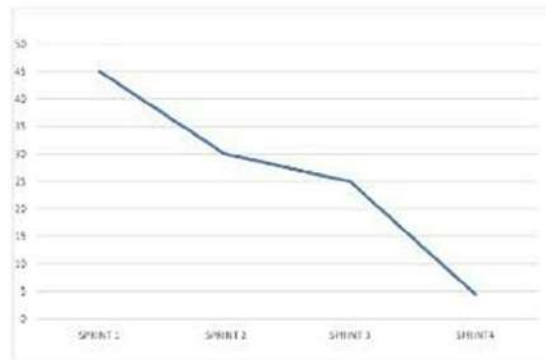
Average velocity of sprint-2: $AV = 11.4 \div 2.75$

Average velocity of sprint-3: $AV = 22.5 \div 5.5$

Average velocity of sprint-4: $AV = 15.4 \div 3.75$

Burn down chart:

A burn down 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.



6.3 Reports from JIRA

Test case ID	Feature Type	Component	Test Scenario	Pre-Requirement	Steps To Execute	Test Data	Expected Result
TC_001	Functional	Home Page	Verify user is able to see the homepage		1.Enter URL and click go	Homepage URL	Login/Signup popup sh
TC_002	UI	Home Page	Verify the UI output in submit page		1.Enter URL and click go 2.Click on submit button	Homepage URL	Application should show elements before click button : No. of Cylinder Displacement Horsepower Weight Model Year Origin
TC_003	Functional	Home page	Verify user is able to get output		1.Enter URL and click go 2.Enter the vehicle parameters for the prediction	No. of Cylinders : 8 Displacement : 110 Horsepower : 210 Weight : 3246 Model Year : 70 Origin : 3	User should navigate t
TC_004	Functional	Submit page	Verify user is able to get output with invalid credentials		1.Enter URL and click go 2.Enter the vehicle parameters for the prediction 3.Then click on the submit button	No. of Cylinders : 8 Displacement : 110 Model Year : 70 Origin : 3	Application should show and "horsepower" value
TC_004	Functional	Submit page	Verify user is able to get output with invalid credentials		1.Enter URL and click go 2.Enter the vehicle parameters for the prediction 3.Then click on the submit button 4. Output page is visible with predicted value of the performance	No. of Cylinders : 8 Displacement : 110 Horsepower : 210 Weight : 3246 Model Year : 70	Application should show validation message
					1.Enter URL and click go	Displacement : 110	Application should show

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

▼ 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
from scipy import stats
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import math
```

▼ Importing Dataset

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

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl

▼ Finding missing data

```
dataset.isnull().any()
```

```
mpg           False
cylinders     False
displacement  False
horsepower    False
```

```
weight      False
acceleration False
model year  False
origin      False
car name    False
dtype: bool
```

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

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

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

```
0
```

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

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

```
dataset.isnull().any()
```

```
mpg          False
cylinders     False
displacement  False
horsepower    False
weight        False
acceleration  False
model year    False
origin        False
car name      False
dtype: bool
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   mpg              398 non-null   float64
1   cylinders         398 non-null   int64
2   displacement      398 non-null   float64
3   horsepower        398 non-null   float64
4   weight            398 non-null   int64
5   acceleration      398 non-null   float64
6   model year        398 non-null   int64
7   origin            398 non-null   int64
8   car name          398 non-null   object
dtypes: float64(4), int64(4), object(1)
memory usage: 28.1+ KB
```

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


	mpg	cylinders	displacement	horsepower	weight	acceleration	model year
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000
mean	23.514573	5.454774	193.425879	104.165829	2970.424623	15.568090	76.010050
std	7.815984	1.701004	104.269838	38.298676	846.841774	2.757689	3.697627
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.000000
25%	17.500000	4.000000	104.250000	75.000000	2223.750000	13.825000	73.000000
50%	23.000000	4.000000	148.500000	92.000000	2803.500000	15.500000	76.000000
75%	29.000000	8.000000	262.000000	125.000000	3608.000000	17.175000	79.000000

There is no use with car name attribute so drop it

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

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

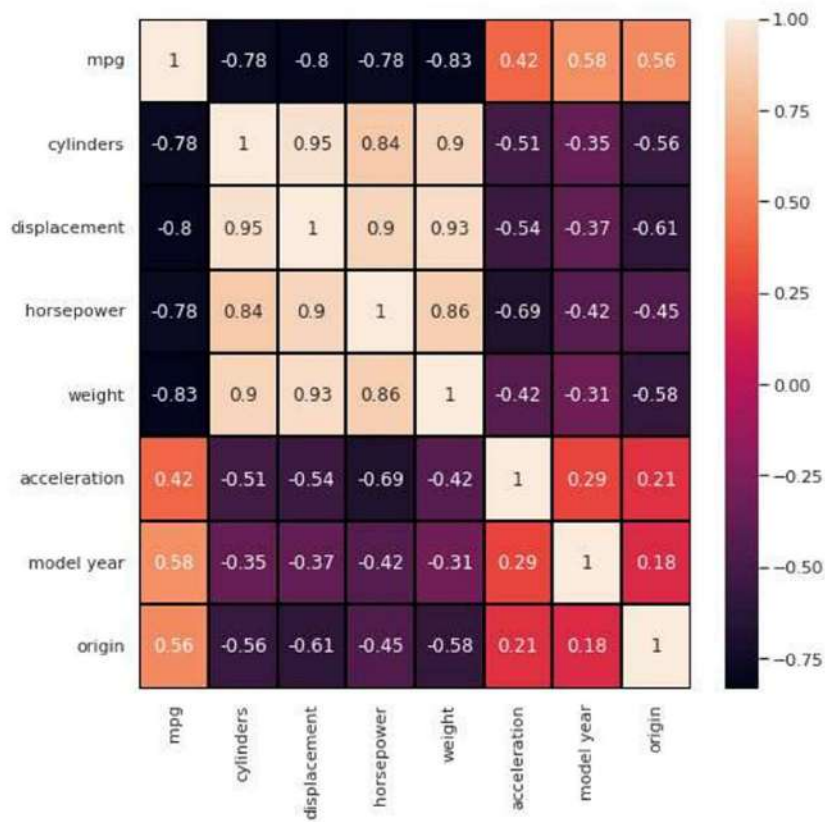
	mpg	cylinders	displacement	horsepower	weight	acceleration	model year
mpg	1.000000	-0.775396	-0.804203	-0.777501	-0.831741	0.420289	0.579267
cylinders	-0.775396	1.000000	0.950721	0.842437	0.896017	-0.505419	-0.348746
displacement	-0.804203	0.950721	1.000000	0.897082	0.932824	-0.543684	-0.370164
horsepower	-0.777501	0.842437	0.897082	1.000000	0.863990	-0.686436	-0.417081
weight	-0.831741	0.896017	0.932824	0.863990	1.000000	-0.417457	-0.306564
acceleration	0.420289	-0.505419	-0.543684	-0.686436	-0.417457	1.000000	0.288137
model year	0.579267	-0.348746	-0.370164	-0.417081	-0.306564	0.288137	1.000000

▼ Data Visualizations

Heatmap : which represents correlation between attributes

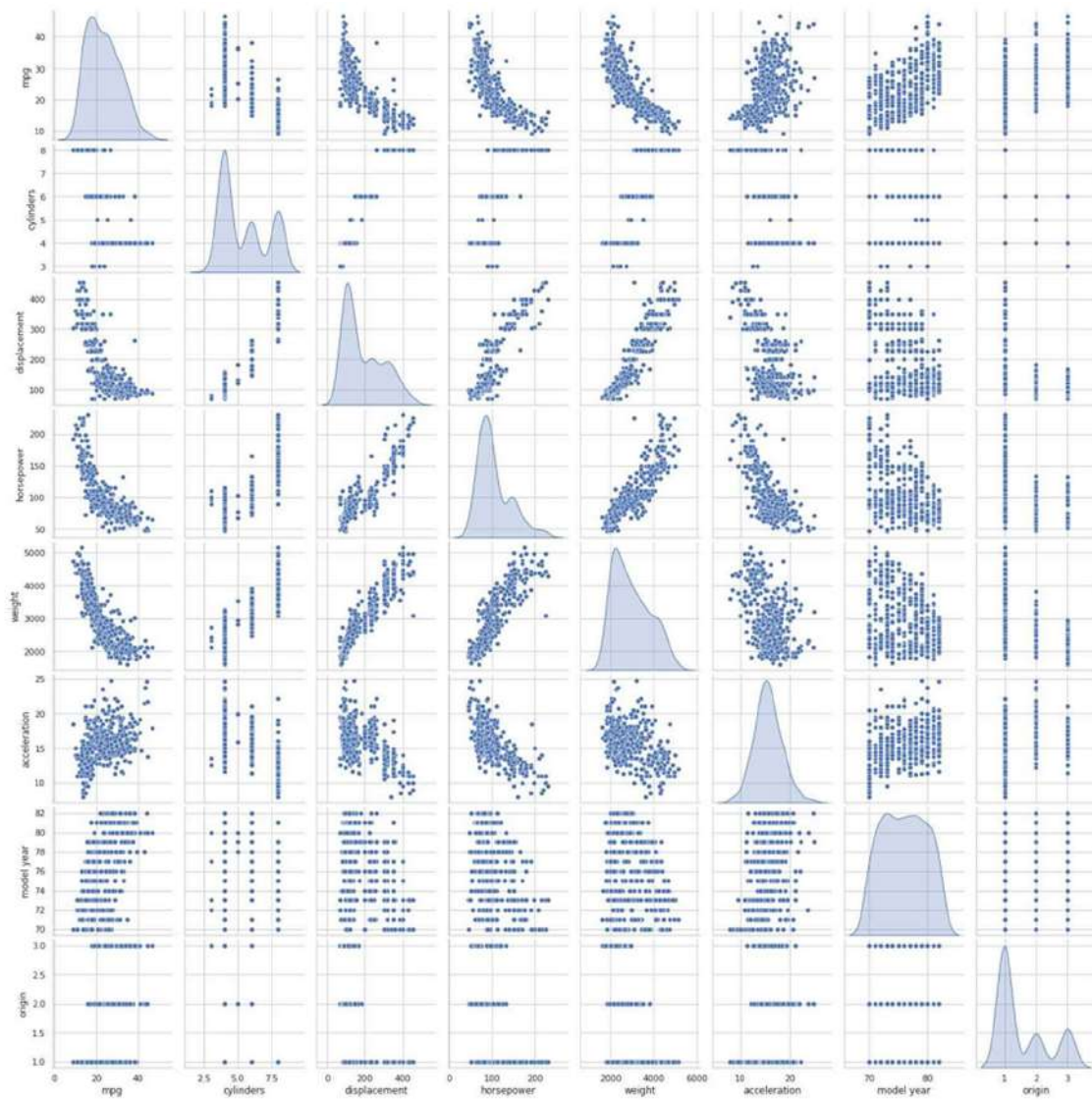
Double-click (or enter) to edit

```
#Heatmap is a way to show some sort of matrix plot,annot is used for correlation.
sns.heatmap(dataset.corr(),annot=True,linecolor='black', linewidths = 1)
fig=plt.gcf()
fig.set_size_inches(8,8)
```



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

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



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

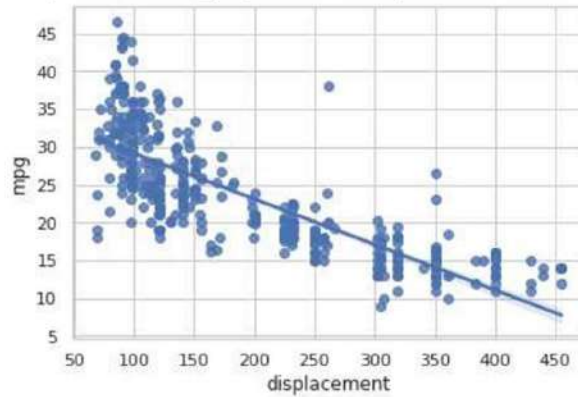
```
sns.regplot(x="cylinders", y="mpg", data=dataset)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5bc037c3d0>
```



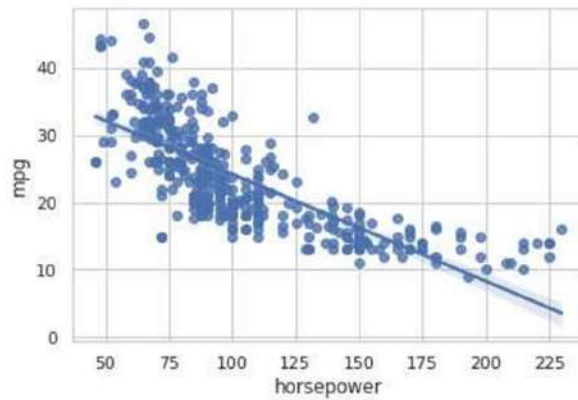
```
sns.regplot(x="displacement", y="mpg", data=dataset)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5bc03b2490>
```



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

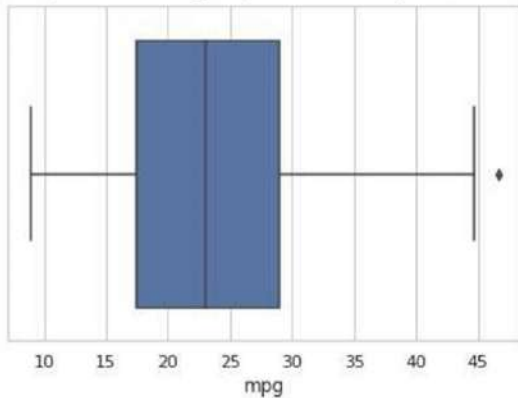
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5bbfedd590>
```



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

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5bc0ad5e10>
```

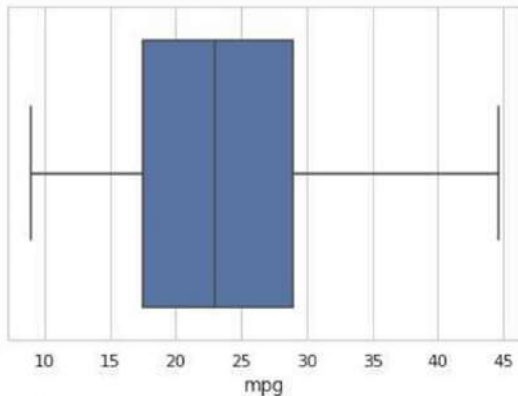


```
#finding the interquartilerange of mpg column and replacing the outliers with mean
q1=dataset['mpg'].quantile(0.25)
q3=dataset['mpg'].quantile(0.75)
iqr=q3-q1
lower_bound=q1-1.5*iqr
upper_bound=q3+1.5*iqr
dataset['mpg']=np.where(dataset['mpg']>upper_bound,dataset['mpg'].mean(),np.where(dataset['mpg']<lo
```

```
#boxplot for mpg column
sns.boxplot(dataset['mpg'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the foll
FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5bbd23b050>
```



Finding quartiles for mpg

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

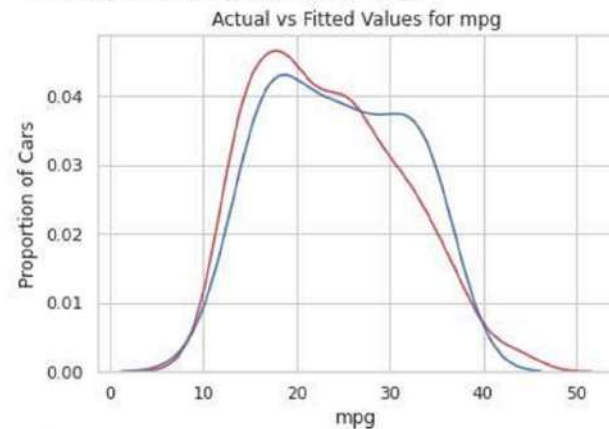
```
25.12 , 35.9 , 32.57 , 30.36 , 16.35 ,
19.87 , 31. , 17.54 , 33.42 , 20.85 ,
23.8 , 19.17 , 16.15 , 35.7 , 11.5 ])
```

```
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()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot`
warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot`
warnings.warn(msg, FutureWarning)
```



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.

Modal Evaluation

```
#importing necessary libraries to find evaluation of the model
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import math
```

```
#mean squared error
MSE=mean_squared_error(y_test,y_pred)
print("MSE:",MSE)
```

```
MSE: 6.505788848703318
```

```
#Root mean squared error
```



```
RMSE=math.sqrt(MSE)
print("RMSE:",RMSE)
```

```
RMSE: 2.550644790774152
```

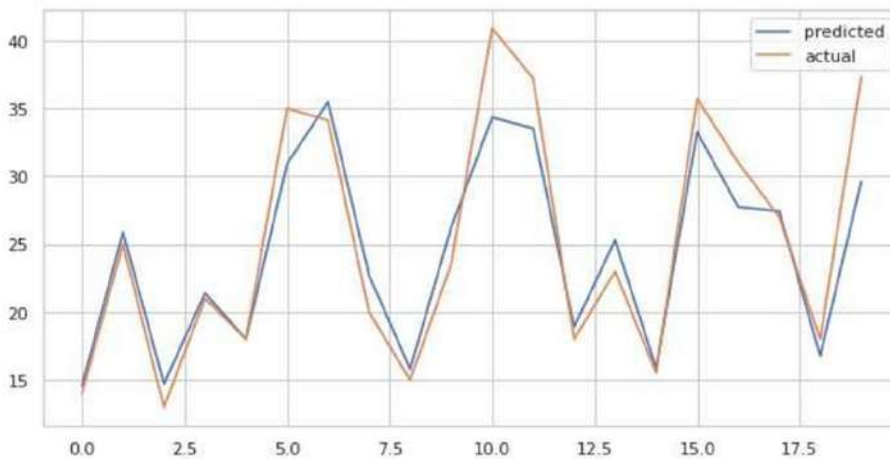
```
#checking the performance of the model using r2_score
r2=r2_score(y_test,y_pred)
print("R2_score:",r2)
```

```
R2_score: 0.9058760463516443
```

```
#Adjusted R square
Adjusted_R2=1-(1-r2*((x_test.shape[0]-1)/(x_test.shape[0]-x_test.shape[1]-1)))
print("Adjusted R2:",Adjusted_R2)
```

```
Adjusted R2: 1.0705807820519433
```

```
#plot for predicted and actual performance
plt.figure(figsize=(10,5))
plt.plot(y_pred[0:20])
plt.plot(np.array(y_test[0:20]))
plt.legend(["predicted","actual"])
plt.show()
```



```
accuracy = r2_score(y_test,y_pred)
accuracy
```

```
0.9058760463516443
```

```
import joblib
```

```
joblib.dump(model,'car performance')
```

```
['car performance']
```

```
siva=joblib.load('car performance')
```

EXPLORER



✓ DEPLOYMENT


> __pycache__


> .vscode


✓ TEMPLATES

<> ibm.html


<> submit.html

 app.py

 car performance

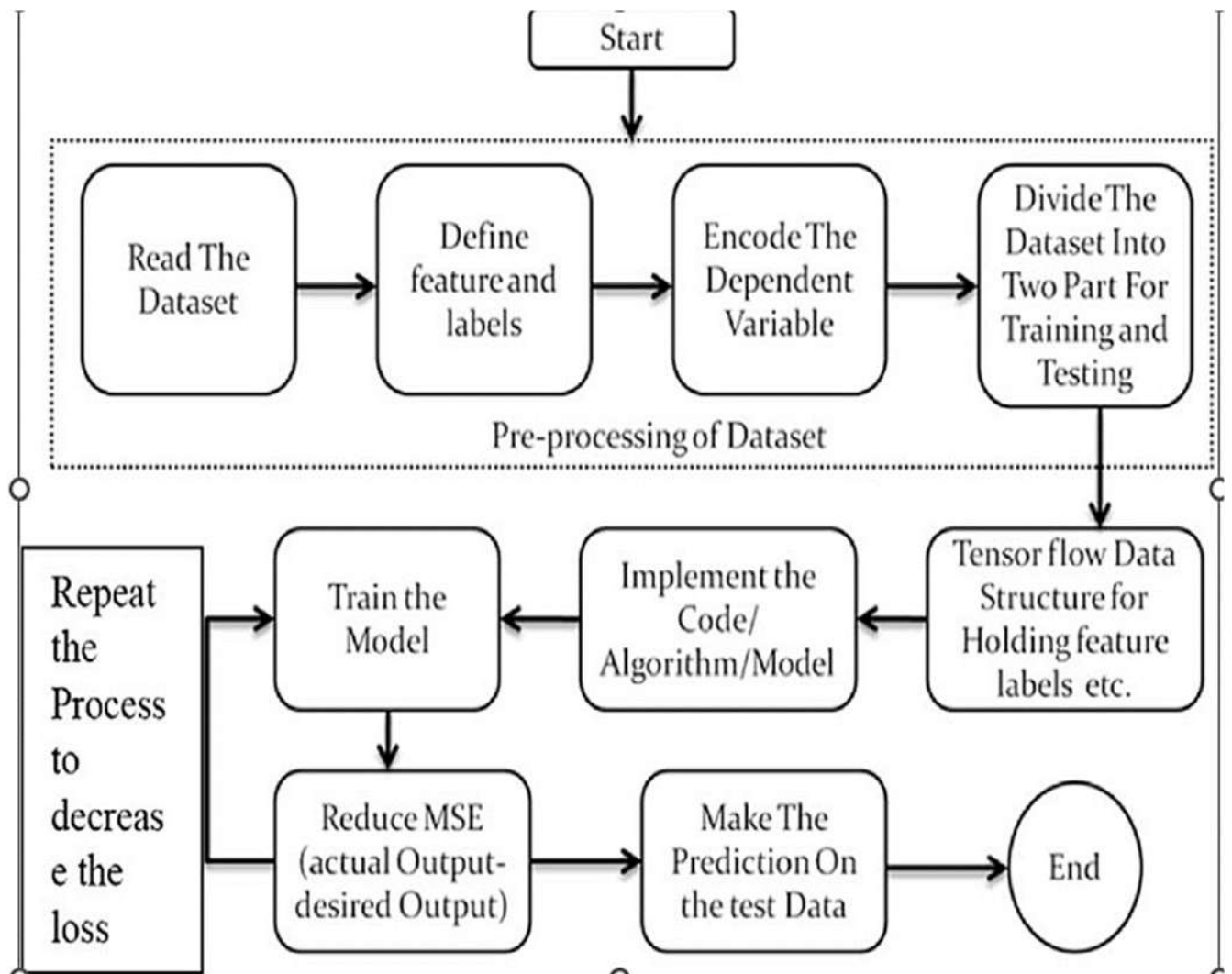
 car performance.csv

 CAR_PERFORMANCE_PREDICTION.ipynb

 ibm_app.py

7.3 Database Schema (if Applicable)

Homepage HTML code :



```

    <div class="inp7">
      <h3>Enter Origin</h3>
      <input type="search" name="origin" id="bs" placeholder="1,2,3" required>
    </div>
    <div class="btn"><button class="btn1"><a href="./submit.html"></a> Submit</button></div></span>
  </form><span>
</div>
</div>
<style>
  body {
    background-image: url('https://wallpapercave.com/wp/wp6676529.jpg');
    background-size: cover;
    opacity: 1;
  }
  .main {
    position: relative;
    text-align: center;
    padding: 1% 0% 1% 0%;
    -webkit-box-shadow: 15px 20px 125px 45px rgba(0, 0, 0, 1);
    -moz-box-shadow: 15px 20px 125px 45px rgba(0, 0, 0, 1);
    box-shadow: 15px 20px 125px 45px rgba(0, 0, 0, 1);
  }
  .main2 {
    position: relative;
    padding-top: 1% 0% 0% 0%;
  }
  .btn {
    position: relative;
    padding-top: 4%;
  }
  .btn1 {
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    -ms-transform: translate(-50%, -50%);
    background-color: #f1f1f1;
  }

```

```
background-color: #f1f1f1;
color: black;
font-size: 20px;
padding: 16px 30px;
border: none;
cursor: pointer;
border-radius: 10px;
text-align: center;
}
.btn1:hover {
background-color: rgb(47, 175, 197);
color: white;
}
.main3 {
padding: 2% 10% 3% 0%;
}
.head1 {
color: antiquewhite;
font-size: 400%;
text-shadow: 4px 4px 2px rgba(107, 187, 213, 0.7);
-webkit-box-shadow: 9px 7px 63px 14px rgba(0, 0, 0, 0.75);
-moz-box-shadow: 9px 7px 63px 14px rgba(0, 0, 0, 0.75);
box-shadow: 9px 7px 63px 14px rgba(0, 0, 0, 0.75);
}
input {
width: 15%;
padding: 9px 1px;
box-sizing: border-box;
border: 1px solid rgb(126, 132, 133);
outline: none;
}
input:focus {
background-color: lightblue;
}
h2 {
color: #BDE0FE;
```

Output page HTML code :

```
.btn1 {
  transform: translate(-50%, -50%);
  -ms-transform: translate(-50%, -50%);
  background-color: ■rgb(170, 224, 226);
  color: □black;
  font-size: 20px;
  padding: 8px 30px 8px 30px;
  border: none;
  cursor: pointer;
  border-radius: 10px;
  text-align: center;
}

.btn1:hover {
  background-color: ■rgb(255, 255, 255);
  color: □rgb(0, 0, 0);
}

h2 {
  margin: 10% 0% 0% 6%;
  font-size: 52px;
  color: ■#FFB808;
  text-shadow: -1px -1px 30px ■rgba(255, 68, 61, 1);
}
</style>

</html>
```

Integrate Flask Code

```

import flask
from flask import request, render_template, Flask
from flask_cors import CORS
import requests
# you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "hAaNpiuYlDodbW1xwrpmxZTycN5gMBJW_06m9m2fU2r1"
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}
app = Flask(__name__) # initialising flask app
# model = joblib.load('car performance') # load machine learning model
@app.route('/', methods=['GET'])
def home():
    return render_template('ibm.html')
@app.route('/ibm.html')
def formpg():
    return render_template('ibm.html')
@app.route('/predict', methods=['POST', 'GET'])
def predict():
    if request.method == 'POST':
        CYLINDERS = int(request.form['cylinders'])
        DISPLACEMENT = int(request.form['displacement'])
        HOESEPOWER = int(request.form['horsepower'])
        WEIGHT = int(request.form['weight'])
        MODEL_YEAR = int(request.form['model_year'])
        ORIGIN = int(request.form['origin'])

        X = [[CYLINDERS, DISPLACEMENT, HOESEPOWER, WEIGHT, MODEL_YEAR, ORIGIN]]

        # NOTE: manually define and pass the array(s) of values to be scored in the next line
        payload_scoring = {"input_data": [{"fields": [['cylinders', 'displacement', 'horsepower', 'weight',
            'model_year', 'origin']], "values": X}]}

        response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/
            da27f9c6-0faf-452a-952c-c400d4115758/predictions?version=2022-11-09', json=payload_scoring,
            headers={'Authorization': 'Bearer ' + mltoken})
        # print("Scoring response")
        predictions = response_scoring.json()

        predict = predictions['predictions'][0]['values'][0][0]
        output=predict
        if(output<=9):
            return render_template('submit.html', prediction_text="Worst performance with mileage " + str(predict) + ".
                Carry extra fuel")
        if(output>9 and output<=17.5):
            return render_template('submit.html', prediction_text="Low performance with mileage " + str(predict) + ".
                Don't go to long distance")
        if(output>17.5 and output<=29):
            return render_template('submit.html', prediction_text="Medium performance with mileage " + str(predict) + ".
                Go for a ride nearby.")
        if(output>29 and output<=46):
            return render_template('submit.html', prediction_text="High performance with mileage " + str(predict) + ".
                Go for a healthy ride")
        if(output>46):
            return render_template('submit.html', prediction_text="Very high performance with mileage " + str(predict) + ".
                You can plan for a Tour")
        else:
            return render_template('ibm.html')

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

```

TESTING REPORT

Testing of an individual software component or module is termed as Unit Testing. It is typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code.

The Code was developed in 3 separate parts-

1. AI Model developed using Jupyter Notebook
2. Web Front end was developed using VS Code
3. Backend Database was developed using MongoDB

PROJECT NAME	MACHINE LEARNING VEHICLE PERFORMANCE ANALYZER
PROJECT TYPE	APPLIED DATA SCIENCE

DEVELOPER	KUMMITHA SIVA MOHAN REDDY
LANGUAGE	PYTHON,HTML,CSS,JAVA SCRIPT
TOTAL NUMBER OF TEST CASES	50
NUMBER OF TEST CASES EXECUTED	49
NUMBER OF TEST CASES PASSED	45
NUMBER OF TEST CASES FAILED	4-DUE TO TECHNICAL ISSUES

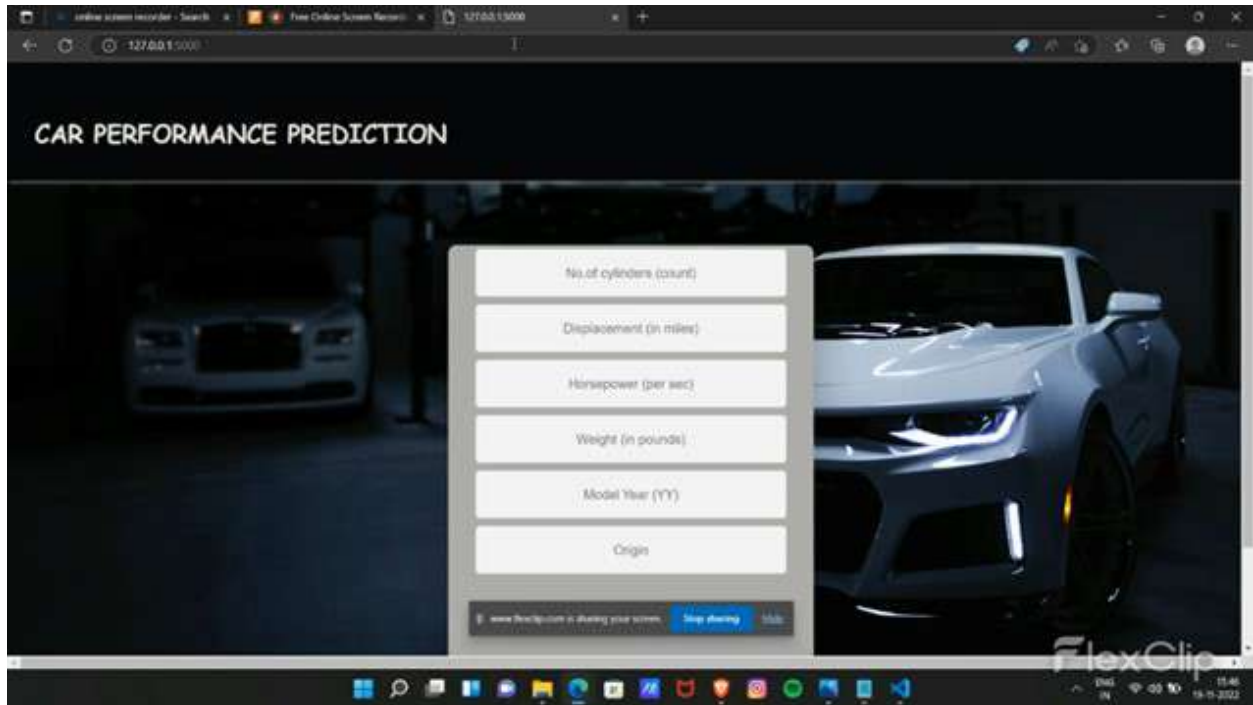
UNIT TESTING:

Unit testing is carried out screen-wise, each screen being identified as an object. Attention is diverted to individual modules, independently to one another to locate errors. This has enabled the detection of errors in coding and logic. This is the first level of testing. In this, codes are written such that from one module, we can move onto the next module according to the choice we enter.



8.1 Test Cases

TEST CASE	No of Cylinders	Displacement	HP	Weight	Year	Origin	Predicted Value
1	8	307	130	3504	70	1	18.1
2	8	350	165	3693	70	1	15.2
3	4	130	95	2372	70	3	24.2
4	6	198	95	2833	70	1	22.3
5	4	104	95	2375	70	2	24.2



8.2 User Acceptance Testing

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Machine Learning-based Vehicle Performance Analyzer project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	9	0	0	9
Client Application	44	0	0	44
Security	2	0	0	2

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

1. Test Case Analysis

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	15	6	2	3	26
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	12	3	5	22	42
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	4	2	1	7
Totals	30	16	14	28	88

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

Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. RESULTS

9.1 Performance Metrics

Performance test:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
-------	-----------	--------	------------

1	Metric	Regression Model:	<div> <div>✓ 0s</div> <pre>[63] #importing necessary libraries to find evaluation metrics from sklearn.metrics import mean_absolute_error from sklearn.metrics import r2_score from sklearn.metrics import mean_squared_error import math</pre> </div> <div> <div>✓ 0s</div> <div>▶</div> <pre># Mean Absolute Error MAE = mae(y_test, y_pred) print("MAE:",MAE)</pre> <div>MAE: 1.7841771356783922</div> </div> <div> <div>✓ 0s</div> <pre>[65] #mean squared error MSE=mean_squared_error(y_test,y_pred) print("MSE:",MSE)</pre> <div>MSE: 6.505788848703318</div> </div> <div> <div>✓ 0s</div> <pre>[66] #Root mean squared error RMSE=math.sqrt(MSE) print("RMSE:",RMSE)</pre> <div>RMSE: 2.550644790774152</div> </div> <div> <div>✓ 0s</div> <pre>[67] #checking the performance of the model using r2 score r2=r2_score(y_test,y_pred) print("R2_score:",r2)</pre> <div>R2_score: 0.9058760463516443</div> </div>
---	--------	-------------------	--

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- ★ It helps users for predicting the vehicle performance.
- ★ Here the chance of occurrence of error is less when compared with the existing system.
- ★ It is fast, efficient and reliable.
- ★ Avoids data redundancy and inconsistency.
- ★ It is Very user-friendly.
- ★ Easy accessibility of data

DISADVANTAGES:

- ★ computer literacy and network access
- ★ Low Computer Literacy
- ★ Security Concerns
- ★ Authenticity Infrastructural Requirement

11. CONCLUSION

The monitoring of car performance, especially gas consumption, has so far been approached only very superficially. A typical fuel gauge, when closely monitored, shows an extremely non-linear relationship between needle movement and fuel consumption. Inaccuracies occur especially in the range of critical low fuel values of 5-10% or more. In the past, due to this limitation, some luxury cars had an audible and flashing light alarm function to indicate a low fuel condition. These systems, which add to the existing fuel level, have no more accuracy than the fuel level monitor alone. In recent years, with the availability of computer techniques and reliable and less expensive computer equipment, a number of systems have been developed to provide somewhat more accurate information about vehicle performance.

12. FUTURE SCOPE

This merits exploratory methods based on actual failures to deduct likely failure modes. This thesis presents two methods for data mining the vehicle maintenance records and vehicle usage data to learn usage or wear patterns indicative of failures. This requires detailed maintenance records where the failure root cause can be deducted with accurate date or mileages of the repair.

Further, more wide-spread adoption of predictive maintenance calls for automatic and less human-resource demanding methods, e.g. unsupervised algorithms with lifelong learning. Such methods are easier to scale up and they can thus be ubiquitously applied since much of the modelling is automated and requires little or no human interaction. Maintenance predictions can be enhanced by combining the deviations in onboard data with off-board data sources such as

maintenance records and failure statistics. This is exclusive product knowledge, only accessible to the vehicle manufacturers, which gives them an advantage in predicting maintenance. Still, data mining has yet to become a core competence of vehicle manufacturers, which makes the road to industrialisation long.

The aim of this thesis is to investigate how on-board data streams and off-board data can be used to predict the vehicle maintenance. More specifically, how on-board data streams can be represented and compressed into a transmittable size and still be relevant for maintenance predictions. Further, the most interesting deviations must be found for a given repair which requires new ways of combining semantic maintenance records with deviations based on compressed on-board data.

13. APPENDIX

Source Code

GitHub ID

<https://github.com/IBM-EPBL/IBM-Project-40378-1660628844>

Project Demo Link

<https://youtu.be/StKz8v3855M>