MLDL - APRIL 15 - TEAM 8

Project

On

Car Performance Prediction

Using

Machine Learning Algorithms

Team Members:

1. Boyina Padmaja
2. Chandika Ramya
3. Cheni Gopinadh
4. Jaya Sai Charan Sammeta
5. Hari Manikanta Pudi

**Table of Contents:**

1. **INTRODUCTION**
   1. **Overview & Purpose**
2. **LITERATURE SURVEY**
   1. **Existing Problem**
   2. **Proposed Solution**
3. **THEORETICAL ANALYSIS**
   1. **Block Diagram**
   2. **Hardware/Software Designing**
4. **EXPERIMENTAL INVESTIGATIONS**
5. **FLOWCHART**
6. **RESULT**
7. **ADVANTAGES & DISADVANTAGES**
8. **APPLICATIONS**
9. **CONCLUSION**
10. **APPENDIX**
    1. **Main Model Code**
    2. **App.py Code**
    3. **HTML Page Code**
11. Introduction:
    1. Overview & Purpose :

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 the 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.

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide**. The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.

**Some Machine Learning Methods:**

* Supervised machine learning algorithms
* Unsupervised machine learning algorithms
* Semi – Supervised machine learning algorithms
* Reinforcement machine learning algorithms

Machine learning enables analysis of massive quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources to train it properly. Combining machine learning with AI and cognitive technologies can make it even more effective in **processing large volumes of information.** And these ML methods have become popular as they allow researchers to improve the prediction accuracy.

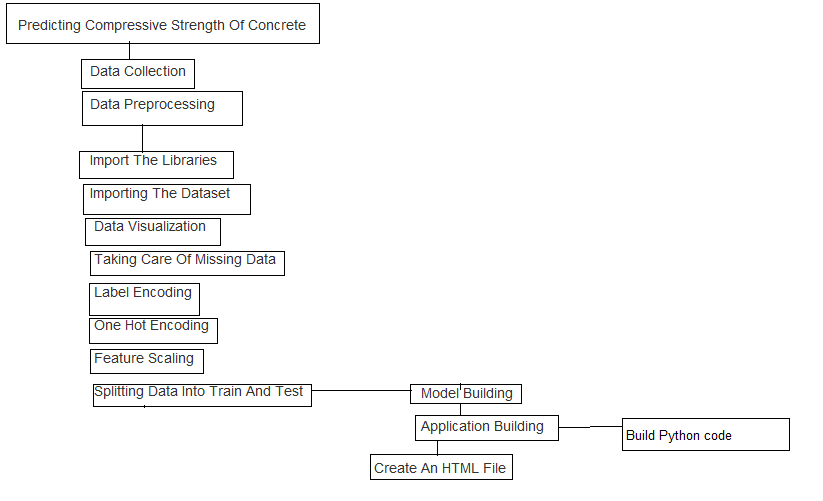
It is an important to analyse the factors using number of well-known approaches of machine learning algorithms like linear regression, decision tree and random forest to improve the vehicle performance efficiency. The range, durability and longevity of automotive traction batteries are ‘hot topics’ in 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.

1. Literature Survey:
   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 the 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.

* 1. Proposed Solution:

It is an important to analyse the factors using number of well-known 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.

1. Theoretical Analysis:
   1. Block Diagram: 
   2. Hardware/Software Designing:

* Model Building using python
* Create Frontend HTML code file
* Create Backend file using python
* Build Flask Web Application

1. Experimental Investigation:

To train the model we need a dataset containing information about Several Car models invented along with normal features since past hundred or more years . For this project this data used is a file named ‘Auto-mpg.csv’ which contains above mentioned data.

1. Flowchart for WebApplication:

Read Input “cylinders” input - 1

Read Input “displacement” input - 2

Read Input “Fuel-type” input - 8

Predict the Output using “Backend Application”

Display the “Output”



1. Result:

We build the Random Forest Regression model to predict the performance of a car in (mpg) using python language and got accuracy about ‘92.3’. And finally, implemented a Web Application to make use of the model. The model takes 8 parameters (cylinders, displacement, horsepower, weight, acceleration, model year, origin, Engine Fuel type) as inputs to predict the mileage (mpg) as output.

1. Advantages & Disadvantages:
   1. Advantages:

Uses less time for prediction

Fast method

* 1. Disadvantages:
* Sometimes digital output maybe error due to lack of information
* Only can be done by trained professionals

1. Applications:

* Automatic cars
* Predictions in testing
* Simulation of cars

1. Conclusion:

By using the Web Application we built one can determine whether a chosen site is suitable for the construction of the Wind power mill or not. And it helps energy suppliers can coordinate the collaborative production of different energy sources more efficiently to avoid costly overproduction.

1. Appendix:

* Source Code:

# -\*- coding: utf-8 -\*-

"""Complete Project.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1\_AweTyUgJGzO\_EemYDD6Kfo-qhOUBVwJ

# Importing Libraries

"""

import os

os.chdir("C:/Users/Abhi Matlapudi/Desktop/ML-Project Cars")

# Commented out IPython magic to ensure Python compatibility.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# %matplotlib inline

from sklearn import preprocessing

import statsmodels.formula.api as smf

"""# Importing Dataset"""

cars = pd.read\_csv("auto-mpg.csv")

cars

cars.describe()

cars.cov()

cars.corr()

cars.head()

cars.tail()

cars.columns

cars.dtypes

cars.shape

"""# Missing values treatment"""

cars.isnull().any()

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

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

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

mean\_hp=cars['horsepower'].mean()

cars['horsepower'].fillna(mean\_hp,inplace=True)

mean\_weight = cars['weight'].mean()

mean\_weight

cars['weight'].fillna(mean\_weight,inplace=True)

cars.isnull().sum()

cars

"""# Data Visualization"""

cars['horsepower'].hist(bins=10)

cars['cylinders'].hist(bins=10)

cars.boxplot(column='%mpg')

cars.boxplot(column='%mpg',by='cylinders')

cars.boxplot(column='%mpg',by='Engine Fuel Type')

"""There is no use with car name attribute so we can drop it"""

cars=cars.drop('car name',axis=1)

cars

import seaborn as sns

sns.heatmap(cars.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(cars,diag\_kind='kde') #pairplot represents pairwise relation across the entire dataframe.

plt.show()

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

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

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

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

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

sns.set(style="whitegrid")

sns.boxplot(x=cars["%mpg"])

"""# Label Encoding"""

from sklearn.preprocessing import LabelEncoder

labelencoder\_y=LabelEncoder()

cars['Engine Fuel Type']=labelencoder\_y.fit\_transform(cars['Engine Fuel Type'])

cars

"""# One Hot Encoding"""

from sklearn.preprocessing import OneHotEncoder

oh=OneHotEncoder(categorical\_features=[1])

x=oh.fit\_transform(x).toarray()

x

x=x[:,8:]

x

"""# Outlier"""

from scipy import stats

import numpy as np

z = np.abs(stats.zscore(cars))

print(z)

threshold = 3

print(np.where(z > 3))

print(z[6][3])

Q1 = cars.quantile(0.25)

Q3 = cars.quantile(0.75)

IQR = Q3 - Q1

print(IQR)

print(cars < (Q1 - 1.5 \* IQR)) |(cars > (Q3 + 1.5 \* IQR))

cars\_o = cars[(z < 3).all(axis=1)]

cars\_o

cars.shape

cars\_o.shape

cars\_out = cars[~((cars < (Q1 - 1.5 \* IQR)) |(cars > (Q3 + 1.5 \* IQR))).any(axis=1)]

cars\_out.shape

"""# Feature Scaling"""

x=cars.iloc[:,1:9].values

x

y=cars.iloc[:,0].values

y

"""# Splitting into Train set and Test set"""

from sklearn.model\_selection import train\_test\_split #train\_test\_split splits arrays or matrices into random train and test subsets.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(x, y,test\_size=0.4,random\_state =1)

X\_train

y\_train

"""# Linear Regression"""

X = cars['weight']

Y = cars['%mpg']

X

Y

from sklearn.model\_selection import train\_test\_split #train\_test\_split splits arrays or matrices into random train and test subsets.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y,test\_size=0.1,random\_state =1)

X\_train

X\_train = X\_train.values.reshape((-1,1))

X\_train

from sklearn import linear\_model as lm

model=lm.LinearRegression()

results=model.fit(X\_train,y\_train)

accuracy = model.score(X\_train,y\_train)

print("Accuracy of the model: ",accuracy)

print('intercept:', model.intercept\_)

print('slope:', model.coef\_)

model

X\_test

y\_test

"""# Multiple Linear Regression"""

X = cars[['displacement','weight']]

Y = cars['%mpg']

X

Y

x=cars.iloc[:,1:9].values

x

y=cars.iloc[:,0].values

y

from sklearn.model\_selection import train\_test\_split #train\_test\_split splits arrays or matrices into random train and test subsets.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(x, y,test\_size=0.4,random\_state =1)

X\_train

from sklearn import linear\_model as lm

model=lm.LinearRegression()

results=model.fit(X\_train,y\_train)

predictions = model.predict(x)

accuracy=model.score(x,y)

print('Accuracy of the model:', accuracy)

plt.scatter(y, predictions)

"""# Decision Tree Regressor"""

from sklearn.tree import DecisionTreeRegressor

#Create the Decision Tree regressor object

regressor = DecisionTreeRegressor(random\_state=0)

regressor.fit(X\_train,y\_train)

y\_pred = regressor.predict([[4,120,98,2380,20,70,3,3]])

y\_pred

accuracy = regressor.score(X\_train,y\_train)

print("Accuracy of the model : ", accuracy)

"""# Random Forest Regressor"""

from sklearn.ensemble import RandomForestRegressor

regressor = RandomForestRegressor(n\_estimators = 100, random\_state = 0)

regressor.fit(X\_train, y\_train)

y\_pred = regressor.predict([[4,120,98,2380,20,70,3,3]])

y\_pred

accuracy = regressor.score(X\_train,y\_train)

print("Accuracy of the model : ", accuracy)