**Car Performance Prediction**

**1.INTRODUCTION**

1.1 Overview

Transportation is playing a seriously important role in one’s day-to-day life. Cars being a part of this are drawing the people’s attention. This fact has helped the company owners to lead a progressive business. Usually when a car is bought, the users usually look into its performance. Although performance can be determined by many factors in our dataset, we predict it by the parameter mileage. We use a machine learning model with a befitting algorithm and integrating it with the flask app.

1.2 Purpose

Often there will be an ambiguity in choosing one from many. Even though there will be other parameters like the cost, dependability, flexibility, etc our job which predicts the performance taking mileage into consideration can help the customers in buying the appropriate car they need. So basically, this prediction helps to compare performances among different cars.

**2.LITERATURE SURVEY**

2.1 EXISTING PROBLEM

Due to lack of proper idea on the performance of cars, they later face problems with their vehicles. Sometimes, there might be some inconvenience with respect to the car in the middle of a journey.

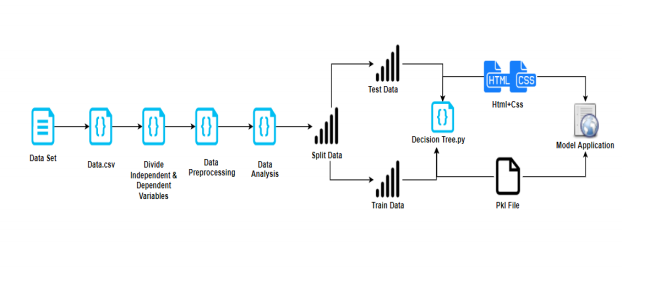
So, the users need to be more specific while buying their car.

2.2 PROPOSED SOLUTION

Taking necessarily, the prerequisite inputs from which performance can be predicted beforehand helps the user to learn a good option which can avoid any kind of interruptions in their journey later on and so. This performance prediction is done with the concept of machine learning that has highest accuracy infers the finest algorithm.

**3.THEORETICAL ANALYSIS**

3.1 BLOCK DIAGRAM



3.2 HARDWARE/SOFTWARE DESIGNING

Car performance prediction

* Dataset preparation
* Dataset pre-processing

1. Data visualisation
2. Taking care of the missing data
3. Label encoding and one-hot encoding(not indispensable)
4. Data transformation
5. Data splitting into train and test

* Model Building

1. Training and testing the model
2. Evaluation and the final predicting method should be such that the accuracy should probably be atmost.

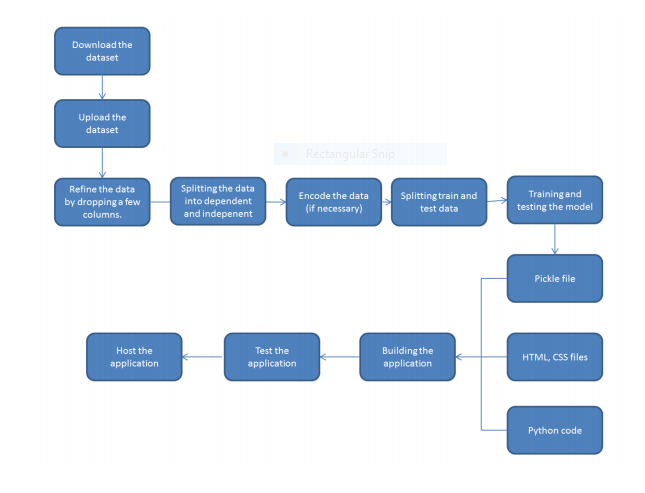
* Model Deployment

1. Creating an HTML file
2. Structuring a proper python code

**4.EXPERIMENTAL INVESTIGATIONS**

In our prediction we collect information i.e. inputs required for the prediction of mileage of the car. In our task we collect parameters like the horsepower, weight, cylinders, displacement and acceleration.

**5.FLOWCHART**



**6.RESULT**

The model can make the users aware of the performance of the car making predictions based on the processed inputs and displays the mileage .

**7.a. ADVANTAGES**

* Accurate predictions
* Easy usable interface
* Results shown are without any errors
* It gives information about mileage as well as performance

**b. DISADVANTAGES**

* For the prediction we need many inputs and sometimes it may lead to some uncertainty in the data or there might be any missing values which need to be taken care about.

**8.APPLICATIONS**

* Checking the performance of a car
* Comparing different figures of performance to find a better car

**9.CONCLUSION**

For a good car, performance is the key factor to know its efficiency. And this is the model which helps in printing the mileage and performance.

**10.FUTURE SCOPE**

With the growing population, the demand for cars has also been increasing. So the people also keep searching for the best car in a suitable price. And this best car is determined by its performance and this is known with our machine learning algorithm.

**11.BIBLIOGRAPHY**

**Source code**

In [1]:

**import** **pandas** **as** **pd**  
**import** **numpy** **as** **np**

In [2]:

datas = pd.read\_csv(r"C:\Users\hp\Desktop\Data Sets\car performance-dataset.csv")

In [3]:

type(datas)

Out[3]:

pandas.core.frame.DataFrame

In [4]:

datas.isnull().any()

Out[4]:

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

from sklearn.preprocessing import LabelEncoder l = LabelEncoder() datas['car name'] = l.fit\_transform(datas['car name'])

In [5]:

datas.head(399)

Out[5]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **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 |
| **394** | 44.0 | 4 | 97.0 | 52 | 2130 | 24.6 | 82 | 2 | vw pickup |
| **395** | 32.0 | 4 | 135.0 | 84 | 2295 | 11.6 | 82 | 1 | dodge rampage |
| **396** | 28.0 | 4 | 120.0 | 79 | 2625 | 18.6 | 82 | 1 | ford ranger |
| **397** | 31.0 | 4 | 119.0 | 82 | 2720 | 19.4 | 82 | 1 | chevy s-10 |

398 rows × 9 columns

In [6]:

datas['car name'].unique()

Out[6]:

array(['chevrolet chevelle malibu', 'buick skylark 320',  
       'plymouth satellite', 'amc rebel sst', 'ford torino',  
       'ford galaxie 500', 'chevrolet impala', 'plymouth fury iii',  
       'pontiac catalina', 'amc ambassador dpl', 'dodge challenger se',  
       "plymouth 'cuda 340", 'chevrolet monte carlo',  
       'buick estate wagon (sw)', 'toyota corona mark ii',  
       'plymouth duster', 'amc hornet', 'ford maverick', 'datsun pl510',  
       'volkswagen 1131 deluxe sedan', 'peugeot 504', 'audi 100 ls',  
       'saab 99e', 'bmw 2002', 'amc gremlin', 'ford f250', 'chevy c20',  
       'dodge d200', 'hi 1200d', 'chevrolet vega 2300', 'toyota corona',  
       'ford pinto', 'plymouth satellite custom', 'ford torino 500',  
       'amc matador', 'pontiac catalina brougham', 'dodge monaco (sw)',  
       'ford country squire (sw)', 'pontiac safari (sw)',  
       'amc hornet sportabout (sw)', 'chevrolet vega (sw)',  
       'pontiac firebird', 'ford mustang', 'mercury capri 2000',  
       'opel 1900', 'peugeot 304', 'fiat 124b', 'toyota corolla 1200',  
       'datsun 1200', 'volkswagen model 111', 'plymouth cricket',  
       'toyota corona hardtop', 'dodge colt hardtop', 'volkswagen type 3',  
       'chevrolet vega', 'ford pinto runabout', 'amc ambassador sst',  
       'mercury marquis', 'buick lesabre custom',  
       'oldsmobile delta 88 royale', 'chrysler newport royal',  
       'mazda rx2 coupe', 'amc matador (sw)',  
       'chevrolet chevelle concours (sw)', 'ford gran torino (sw)',  
       'plymouth satellite custom (sw)', 'volvo 145e (sw)',  
       'volkswagen 411 (sw)', 'peugeot 504 (sw)', 'renault 12 (sw)',  
       'ford pinto (sw)', 'datsun 510 (sw)',  
       'toyouta corona mark ii (sw)', 'dodge colt (sw)',  
       'toyota corolla 1600 (sw)', 'buick century 350',  
       'chevrolet malibu', 'ford gran torino', 'dodge coronet custom',  
       'mercury marquis brougham', 'chevrolet caprice classic',  
       'ford ltd', 'plymouth fury gran sedan',  
       'chrysler new yorker brougham', 'buick electra 225 custom',  
       'amc ambassador brougham', 'plymouth valiant',  
       'chevrolet nova custom', 'volkswagen super beetle', 'ford country',  
       'plymouth custom suburb', 'oldsmobile vista cruiser',  
       'toyota carina', 'datsun 610', 'maxda rx3', 'mercury capri v6',  
       'fiat 124 sport coupe', 'chevrolet monte carlo s',  
       'pontiac grand prix', 'fiat 128', 'opel manta', 'audi 100ls',  
       'volvo 144ea', 'dodge dart custom', 'saab 99le', 'toyota mark ii',  
       'oldsmobile omega', 'chevrolet nova', 'datsun b210',  
       'chevrolet chevelle malibu classic', 'plymouth satellite sebring',  
       'buick century luxus (sw)', 'dodge coronet custom (sw)',  
       'audi fox', 'volkswagen dasher', 'datsun 710', 'dodge colt',  
       'fiat 124 tc', 'honda civic', 'subaru', 'fiat x1.9',  
       'plymouth valiant custom', 'mercury monarch', 'chevrolet bel air',  
       'plymouth grand fury', 'buick century',  
       'chevroelt chevelle malibu', 'plymouth fury', 'buick skyhawk',  
       'chevrolet monza 2+2', 'ford mustang ii', 'toyota corolla',  
       'pontiac astro', 'volkswagen rabbit', 'amc pacer', 'volvo 244dl',  
       'honda civic cvcc', 'fiat 131', 'capri ii', 'renault 12tl',  
       'dodge coronet brougham', 'chevrolet chevette', 'chevrolet woody',  
       'vw rabbit', 'dodge aspen se', 'ford granada ghia',  
       'pontiac ventura sj', 'amc pacer d/l', 'datsun b-210', 'volvo 245',  
       'plymouth volare premier v8', 'mercedes-benz 280s',  
       'cadillac seville', 'chevy c10', 'ford f108', 'dodge d100',  
       'honda accord cvcc', 'buick opel isuzu deluxe', 'renault 5 gtl',  
       'plymouth arrow gs', 'datsun f-10 hatchback',  
       'oldsmobile cutlass supreme', 'dodge monaco brougham',  
       'mercury cougar brougham', 'chevrolet concours', 'buick skylark',  
       'plymouth volare custom', 'ford granada', 'pontiac grand prix lj',  
       'chevrolet monte carlo landau', 'chrysler cordoba',  
       'ford thunderbird', 'volkswagen rabbit custom',  
       'pontiac sunbird coupe', 'toyota corolla liftback',  
       'ford mustang ii 2+2', 'dodge colt m/m', 'subaru dl', 'datsun 810',  
       'bmw 320i', 'mazda rx-4', 'volkswagen rabbit custom diesel',  
       'ford fiesta', 'mazda glc deluxe', 'datsun b210 gx',  
       'oldsmobile cutlass salon brougham', 'dodge diplomat',  
       'mercury monarch ghia', 'pontiac phoenix lj',  
       'ford fairmont (auto)', 'ford fairmont (man)', 'plymouth volare',  
       'amc concord', 'buick century special', 'mercury zephyr',  
       'dodge aspen', 'amc concord d/l',  
       'buick regal sport coupe (turbo)', 'ford futura',  
       'dodge magnum xe', 'datsun 510', 'dodge omni',  
       'toyota celica gt liftback', 'plymouth sapporo',  
       'oldsmobile starfire sx', 'datsun 200-sx', 'audi 5000',  
       'volvo 264gl', 'saab 99gle', 'peugeot 604sl',  
       'volkswagen scirocco', 'honda accord lx', 'pontiac lemans v6',  
       'mercury zephyr 6', 'ford fairmont 4', 'amc concord dl 6',  
       'dodge aspen 6', 'ford ltd landau', 'mercury grand marquis',  
       'dodge st. regis', 'chevrolet malibu classic (sw)',  
       'chrysler lebaron town @ country (sw)', 'vw rabbit custom',  
       'maxda glc deluxe', 'dodge colt hatchback custom', 'amc spirit dl',  
       'mercedes benz 300d', 'cadillac eldorado', 'plymouth horizon',  
       'plymouth horizon tc3', 'datsun 210', 'fiat strada custom',  
       'buick skylark limited', 'chevrolet citation',  
       'oldsmobile omega brougham', 'pontiac phoenix',  
       'toyota corolla tercel', 'datsun 310', 'ford fairmont',  
       'audi 4000', 'toyota corona liftback', 'mazda 626',  
       'datsun 510 hatchback', 'mazda glc', 'vw rabbit c (diesel)',  
       'vw dasher (diesel)', 'audi 5000s (diesel)', 'mercedes-benz 240d',  
       'honda civic 1500 gl', 'renault lecar deluxe', 'vokswagen rabbit',  
       'datsun 280-zx', 'mazda rx-7 gs', 'triumph tr7 coupe',  
       'ford mustang cobra', 'honda accord', 'plymouth reliant',  
       'dodge aries wagon (sw)', 'toyota starlet', 'plymouth champ',  
       'honda civic 1300', 'datsun 210 mpg', 'toyota tercel',  
       'mazda glc 4', 'plymouth horizon 4', 'ford escort 4w',  
       'ford escort 2h', 'volkswagen jetta', 'renault 18i',  
       'honda prelude', 'datsun 200sx', 'peugeot 505s turbo diesel',  
       'volvo diesel', 'toyota cressida', 'datsun 810 maxima',  
       'oldsmobile cutlass ls', 'ford granada gl',  
       'chrysler lebaron salon', 'chevrolet cavalier',  
       'chevrolet cavalier wagon', 'chevrolet cavalier 2-door',  
       'pontiac j2000 se hatchback', 'dodge aries se',  
       'ford fairmont futura', 'amc concord dl', 'volkswagen rabbit l',  
       'mazda glc custom l', 'mazda glc custom', 'plymouth horizon miser',  
       'mercury lynx l', 'nissan stanza xe', 'honda civic (auto)',  
       'datsun 310 gx', 'buick century limited',  
       'oldsmobile cutlass ciera (diesel)', 'chrysler lebaron medallion',  
       'ford granada l', 'toyota celica gt', 'dodge charger 2.2',  
       'chevrolet camaro', 'ford mustang gl', 'vw pickup',  
       'dodge rampage', 'ford ranger', 'chevy s-10'], dtype=object)

In [ ]:

In [7]:

x = datas.iloc[:,1:8].values

In [8]:

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

In [ ]:

In [9]:

**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,random\_state=0)

In [10]:

x.shape

Out[10]:

(398, 7)

In [11]:

x\_train

Out[11]:

array([[  8. , 318. , 150. , ...,  13.5,  72. ,   1. ],  
       [  4. ,  97. ,  60. , ...,  19. ,  71. ,   2. ],  
       [  4. ,  97. ,  78. , ...,  15.8,  80. ,   2. ],  
       ...,  
       [  4. ,  68. ,  49. , ...,  19.5,  73. ,   2. ],  
       [  6. , 250. , 100. , ...,  15. ,  71. ,   1. ],  
       [  4. ,  90. ,  71. , ...,  16.5,  75. ,   2. ]])

In [12]:

**from** **sklearn.preprocessing** **import** StandardScaler  
sd = StandardScaler()  
x\_train = sd.fit\_transform(x\_train)  
x\_test = sd.fit\_transform(x\_test)

In [13]:

x\_train

Out[13]:

array([[ 1.49526939,  1.22961301,  1.24359144, ..., -0.79520768,  
        -1.13752513, -0.73301171],  
       [-0.85285735, -0.92367663, -1.16092059, ...,  1.24411524,  
        -1.41177304,  0.5068698 ],  
       [-0.85285735, -0.92367663, -0.68001818, ...,  0.05760009,  
         1.05645814,  0.5068698 ],  
       ...,  
       [-0.85285735, -1.206235  , -1.45480539, ...,  1.42950823,  
        -0.86327722,  0.5068698 ],  
       [ 0.32120602,  0.56706235, -0.09224857, ..., -0.2390287 ,  
        -1.41177304, -0.73301171],  
       [-0.85285735, -0.99188037, -0.86703579, ...,  0.31715028,  
        -0.31478141,  0.5068698 ]])

In [14]:

**from** **sklearn.ensemble** **import** RandomForestRegressor  
d = RandomForestRegressor (n\_estimators=30,random\_state = 0)  
d.fit(x\_train,y\_train)

Out[14]:

RandomForestRegressor(n\_estimators=30, random\_state=0)

In [15]:

**import** **pickle**  
pickle.dump(d,open('regression.pkl','wb'))

In [16]:

r = d.predict(x\_test)

In [17]:

r

Out[17]:

array([14.38333333, 24.25666667, 14.21666667, 20.56666667, 18.47333333,  
       30.21666667, 34.63333333, 21.15      , 16.30333333, 25.76      ,  
       36.60333333, 36.27      , 19.53666667, 27.32333333, 16.54333333,  
       32.99333333, 28.32333333, 27.49666667, 17.03      , 35.82      ,  
       16.47333333, 23.54      , 23.16666667, 20.7       , 33.69666667,  
       26.45      , 33.79666667, 30.37333333, 31.93666667, 16.57333333,  
       20.26666667, 32.99      , 19.79666667, 34.08333333, 20.85666667,  
       25.02      , 19.65333333, 17.14      , 34.78333333, 12.76666667,  
       13.73333333, 15.2       , 28.32      , 32.76666667, 28.74333333,  
       22.68666667, 20.54333333, 16.50666667, 23.38      , 29.88333333,  
       34.31666667, 26.5       , 17.63      , 27.78333333, 15.96666667,  
       12.96666667, 18.86666667, 26.91666667, 31.95666667, 15.68      ,  
       20.81      , 25.97      , 19.84666667, 21.6       , 13.46666667,  
       15.33333333, 14.2       , 18.90333333, 24.72666667, 14.21666667,  
       34.87666667, 13.25      , 22.96666667, 18.77666667, 23.83333333,  
       32.16666667, 28.17666667, 31.23666667, 31.94      , 14.35      ])

In [18]:

**from** **sklearn.metrics** **import** r2\_score  
accuracy = r2\_score(y\_test,r)

In [19]:

accuracy

Out[19]:

0.8914224071232417

from sklearn.svm import SVR svm = SVR(kernel = 'linear') svm.fit(x\_train,y\_train)y\_pre = svm.predict(x\_test)from sklearn.metrics import r2\_score accurac = r2\_score(y\_test,y\_pre)accurac

In [20]:

d.predict([[1,6,17,34,11,123,299]])

Out[20]:

array([17.62333333])

In [ ]:

In [ ]:

In [ ]:

**app.py:**

from flask import Flask , render\_template , request

import pickle

app = Flask(\_\_name\_\_)

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

@app.route('/')

def intro():

    return render\_template('index.html')

@app.route('/login', methods = ["POST"])

def login():

    cyl = request.form["cyl"]

    dis = request.form["dis"]

    hp = request.form["hp"]

    w = request.form["w"]

    a = request.form["a"]

    my = request.form["my"]

    ori = request.form["ori"]

    total = [[int(cyl),int(dis),int(hp),int(w),int(a),int(my),int(ori)]]

    p = model.predict(total)

    p =p[0]

    return render\_template('index.html',label = "The performance of the car is  "+str(p))

if \_\_name\_\_=='\_\_main\_\_':

    app.run(debug = True,port = 9000

**index**.**html**:

<html>

<style>

.background-image {

  position: fixed;

  left: 0;

  right: 0;

  z-index: 1;

  display: block;

  background-image: url('https://www.mercedes-benz.com/en/vehicles/passenger-cars/\_jcr\_content/root/grid\_copy/grid-par/griditem\_copy/image/MQ6-8-image-20200129142634/00-mercedes-benz-passenger-cars-art-calender-2019-2560x1440.jpeg');

  width: 12000px;

  height: 200000px;

  -webkit-filter: blur(3px);

  -moz-filter: blur(3px);

  -o-filter: blur(3px);

  -ms-filter: blur(3px);

  filter: blur(3px);

}

.content {

  position: absolute;

  top: 50%

  left: 50%;

  right: 50%;

  z-index: 2;

  width: 20%;

  content: center;

  }

  </style>

<body>

<form action ="http://localhost:9000/login" method="post">

<div class="background-image">  </div>

<div class="content">

<header>

<h1 style="font-family:verdana;font-size:200%;content-align:center;color:white;"> CAR PERFORMANCE PREDICTION </h1>

<h2 style="font-family:courier;font-size:90%;content-align:center;color:white;"> ENTER THE BELOW DATA TO FIND THE CAR'S PERFORMANCE </h2>

<br>

<p style="text-align:center;color:white;"> Enter no:of cylinders </p>

<p style="text-align:center;color:white;"><input type= "text" name="cyl"/></p>

<br>

<p style="text-align:center;color:white;"> Enter the Displacement</p>

<p style="text-align:center;color:white;"> <input type= "text" name="dis"/></p>

<br>

<p style="text-align:center;color:white;"> Enter Horsepower</p>

<p style="text-align:center;color:white;"> <input type= "text" name="hp"/></p>

<br>

<p style="text-align:center;color:white;"> Enter Weight</p>

<p style="text-align:center;color:white;"><input type= "text" name="w"/></p>

<br>

<p style="text-align:center;color:white;"> Enter Acceleration</p>

<p style="text-align:center;color:white;"> <input type= "text" name="a"/></p>

<br>

<p style="text-align:center;color:white;"> Enter Model Year</p>

<p style="text-align:center;color:white;"> <input type= "text" name="my"/></p>

<br>

<p style="text-align:center;color:white;"> Enter Origin</p>

<p style="text-align:center;color:white;"> <input type= "text" name="ori"/></p>

<br>

<p style="text-align:center;color:white;"> <input type="submit" value ="check the performance"/></p>

</form>

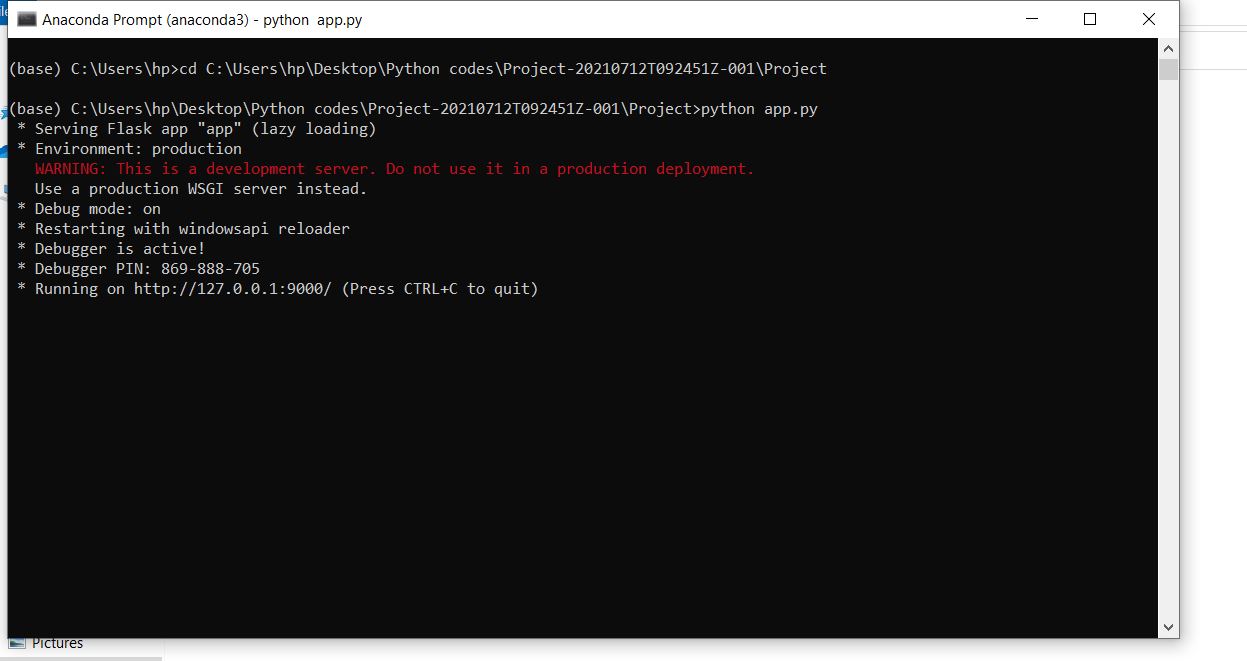
<b style="text-align:center;color:white;"> {{label}} </b>

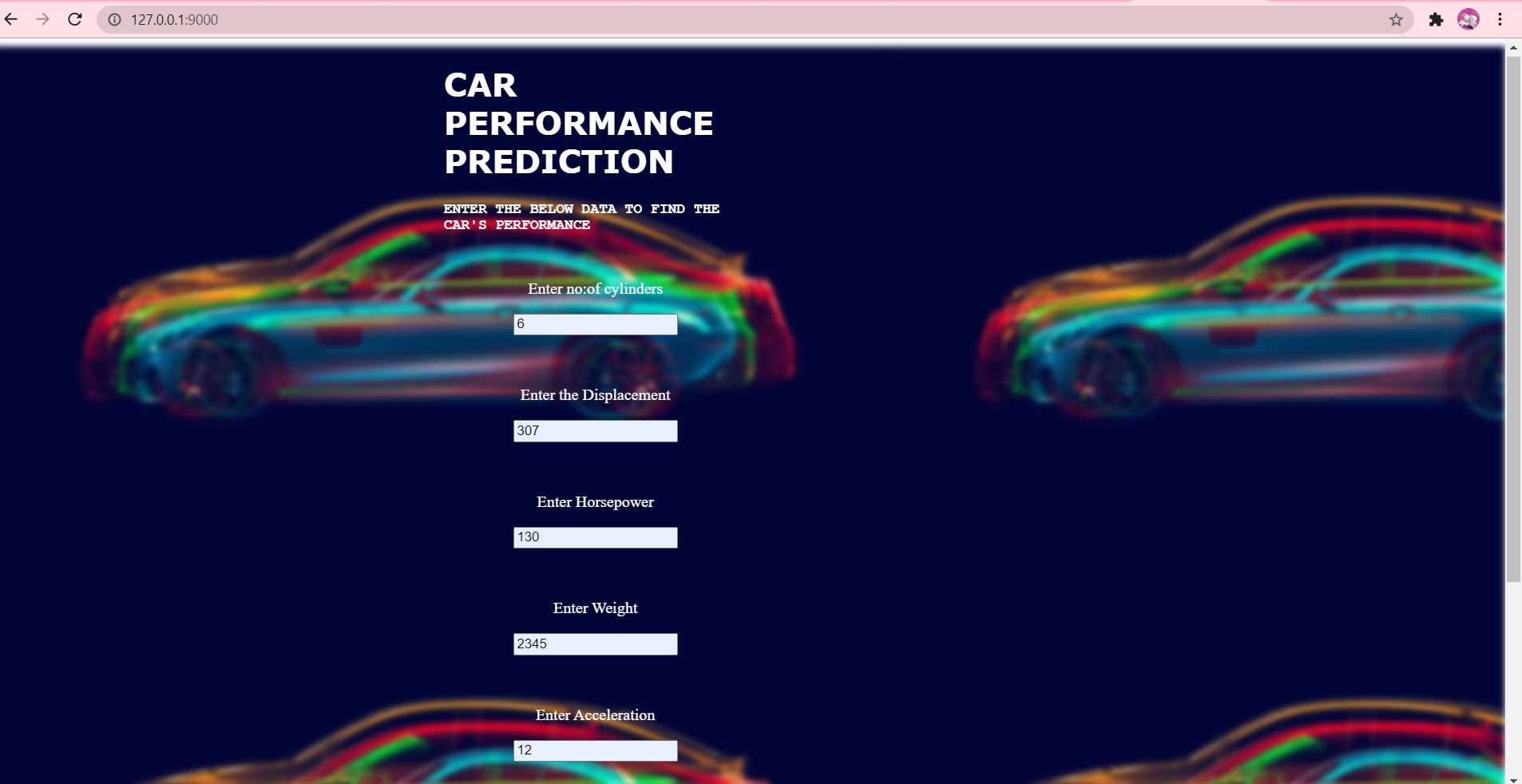
</div>

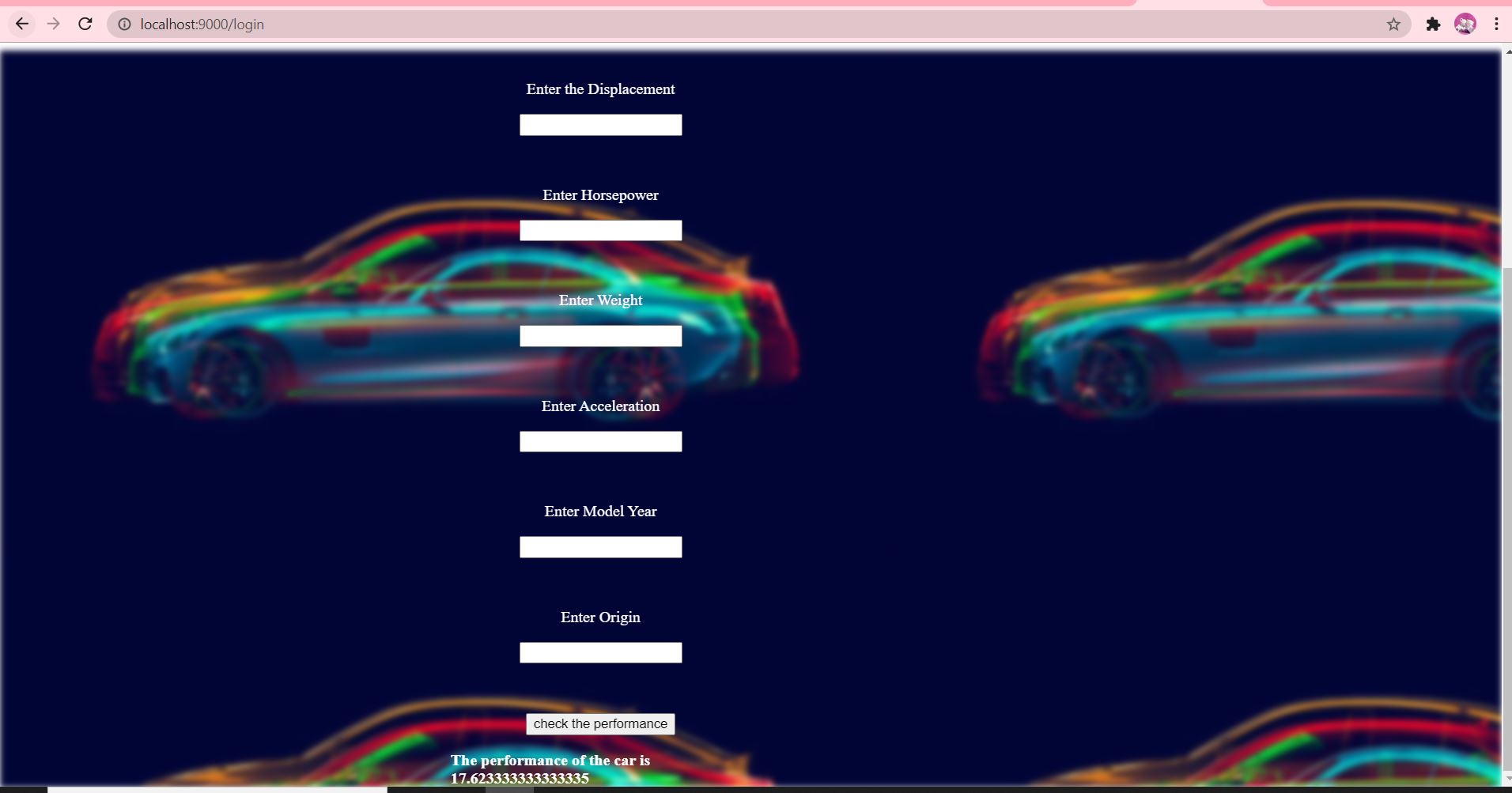
</body>

</html>

OutPut Screenshots:







APPENDIX

<https://www.sciencedirect.com/science/article/abs/pii/S0031320399001211>

https://www.sae.org/publications/technical-papers/content/650623/