INTRODUCTION

1.1 Overview

A typical passenger vehicle emits about 4.6 metric tons of carbon dioxide per year. This number can vary based on a vehicle's fuel, fuel economy, and the number of miles driven per year. And about 20% of the total Co2 Emissions are contributed by wheeled vehicles (including cars). It is high time this percentage should be reduced to control Global Warming. One of the ways of achieving this by increasing the Fuel Economy of vehicles. Fuel Economy of the vehicles can be improved by boosting engine efficiency and hybridization of cars. Furthermore, electric vehicles are another worthwhile solution for reducing the Co2 Emission from vehicles.

1.2 Purpose

The scope of this project is to check if a passenger vehicle - Car - is maintaining the standard for Co2 emissions. The vehicle's Co2 emissions must be lower than the given threshold value, and if it is, then the vehicle will be permitted to operate. If not, then the details of the vehicle will be sent over to the head of the particular region's RTA for the seizing of the vehicle.

LITERATURE SURVEY

2.1 Existing problem

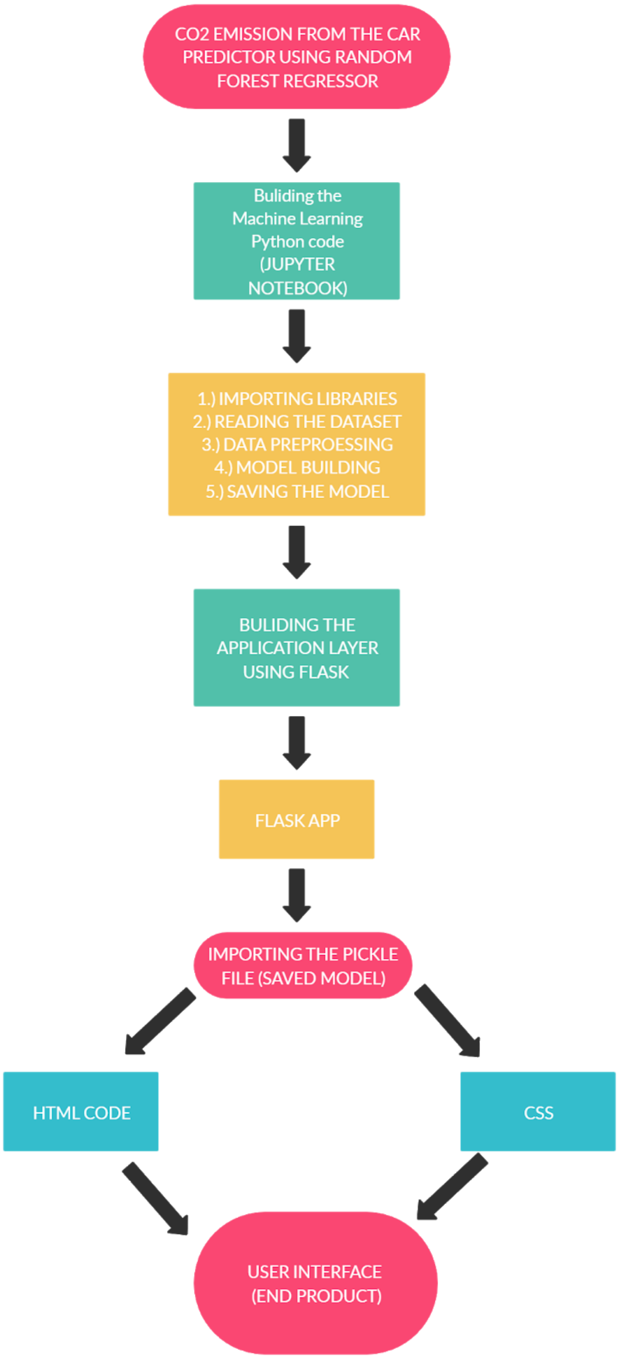
As of now, there are no viable means for testing how much Co2 a particular vehicle is emitting. The proposed solution will be suitable for real time testing of cars and will make sure only the vehicles emitting Co2 below the standardized values will be allowed, and the ones which do not will be seized.

2.2 Proposed solution

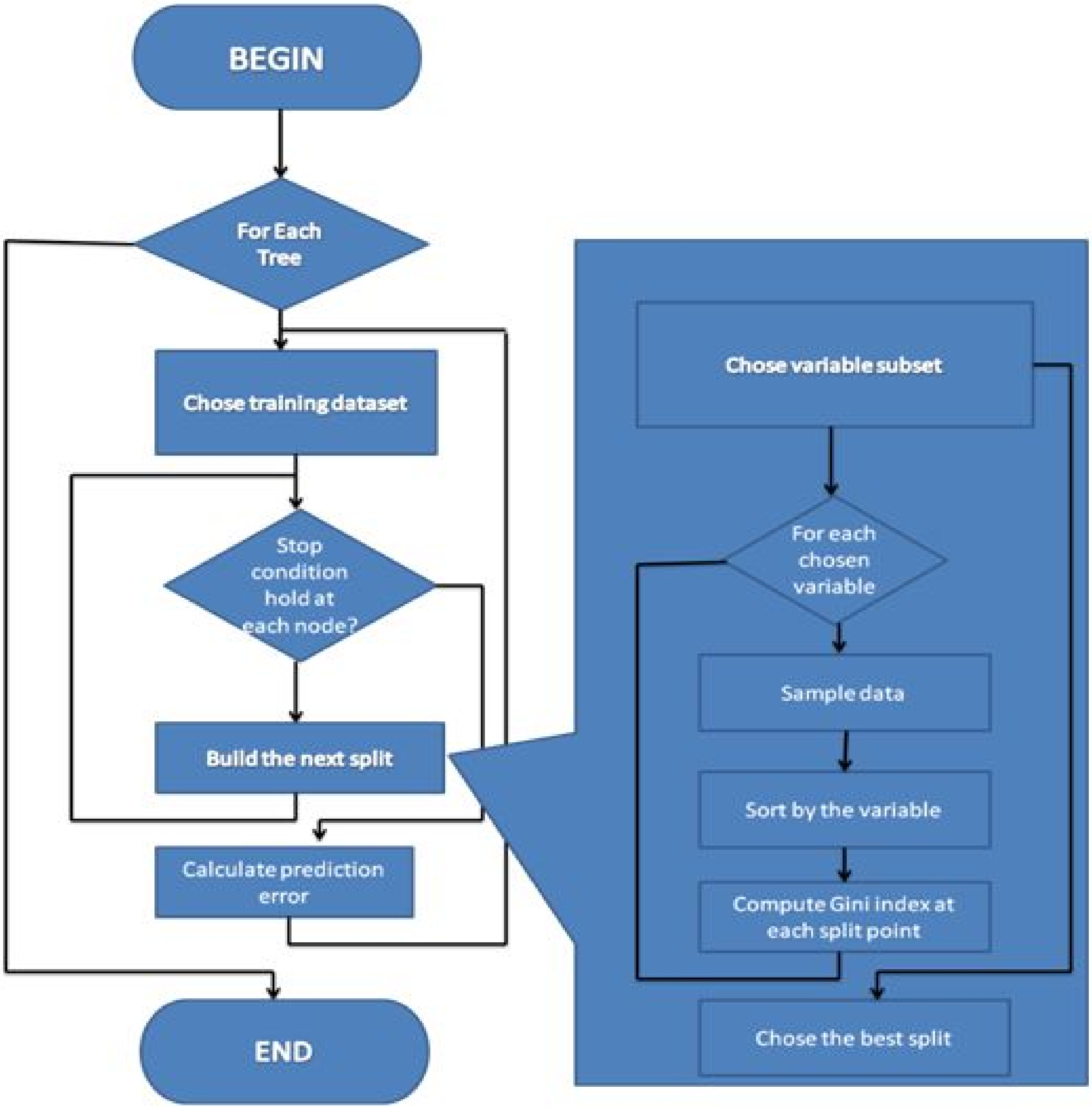
The solution is using a Machine Learning Algorithm called Random Forest Regression. A dataset containing the car details such as make, model, year of manufacture, class, fuel type used, engine size, number of cylinders, transmission, fuel consumption - city & highway, and fuel consumption - combined is collected. Then the machine learning algorithm - Random Forest Regression - is trained and tested for accuracy. The model is then exported as a .pkl file for use in Web Applications. The Web Applications will collect the above mentioned details of the car to be tested and if the Co2 emissions of the vehicle is less than the given threshold value, then the car will be permitted to remain operational. If not, it will be seized.

THEORETICAL ANALYSIS

3.1 Block diagram

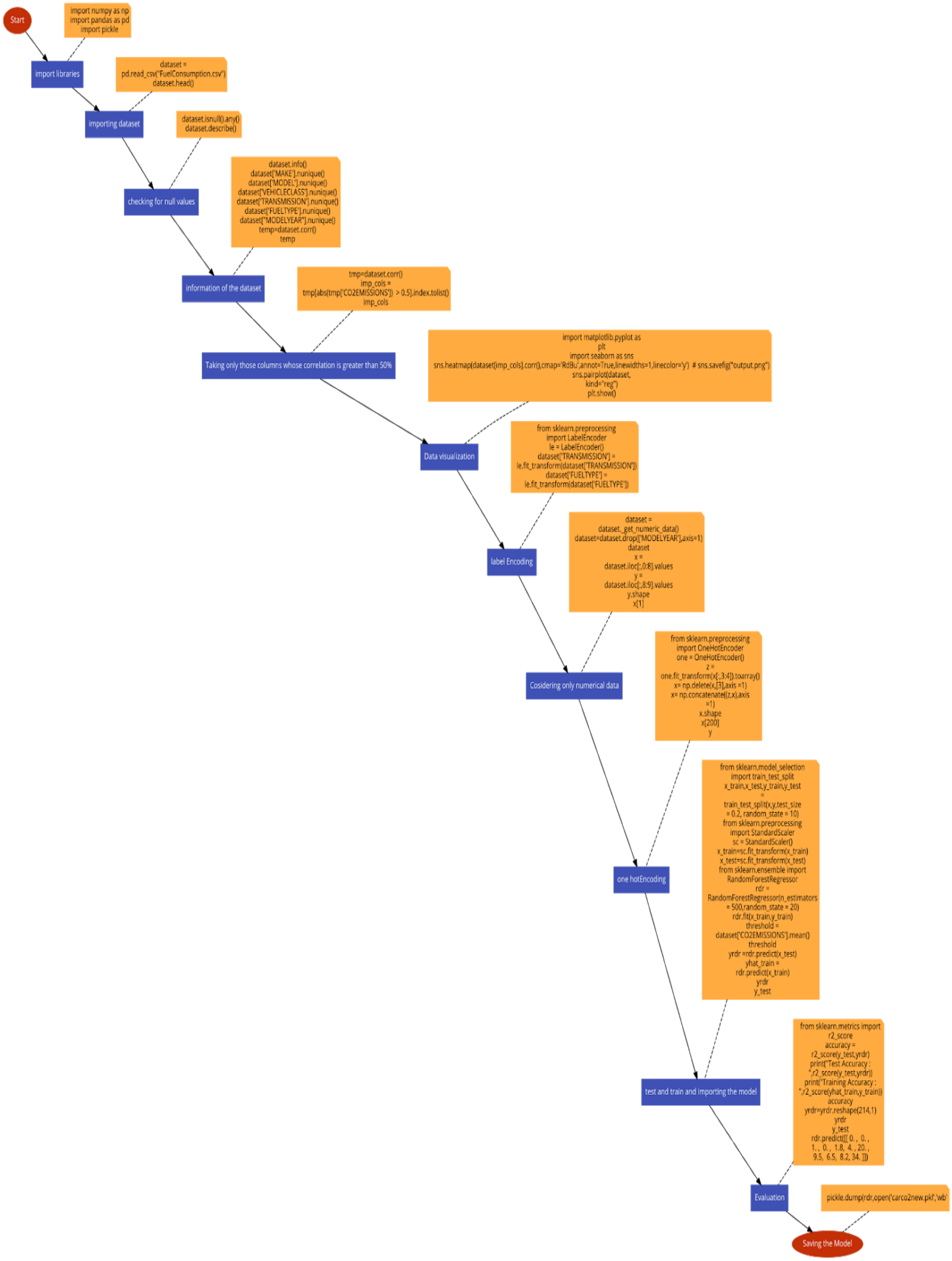

CO2 EMISSION FROM THE CAR PREDICTOR USING RANDOM FOREST REGRESSOR
Buliding the Machine Learning Python code (JUPYTER NOTEBOOK)
1.) IMPORTING LIBRARIES
2.) READING THE DATASET
3.) DATA PREPROESSING
4.) MODEL BUILDING
5.) SAVING THE MODEL
BULIDING THE APPLICATION LAYER USING FLASK
FLASK APP
IMPORTING THE PICKLE FILE (SAVED MODEL)
HTML CODE
CSS
USER INTERFACE (END PRODUCT)

3.2 Hardware / Software designing



EXPERIMENTAL INVESTIGATIONS

FLOWCHART


Start
import libraries
importing dataset
checking for null values
information of the dataset
Taking only those columns whose correlation is greater than 50%
Data visualization
label Encoding
Cosidering only numerical data
one hotEncoding
test and train and importing the model
Evaluation
Saving the Model

```
import numpy as np
import pandas as pd
import pickle
```

```
dataset =
pd.read_csv("FuelConsumption.csv")
dataset.head()
```

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

```
dataset.info()
dataset['MAKE'].nunique()
dataset['MODEL'].nunique()
dataset['VEHICLECLASS'].nunique()
dataset['TRANSMISSION'].nunique()
dataset['FUELTYPE'].nunique()
dataset["MODELYEAR"].nunique()
temp=dataset.corr()
temp
```

```
tmp=dataset.corr()
imp_cols =
tmp[abs(tmp['CO2EMISSIONS']) > 0.5].index.tolist()
imp_cols
```

```
import matplotlib.pyplot as
plt
import seaborn as sns
sns.heatmap(dataset[imp_cols].corr(),cmap='RdBu',annot=True,linewidths=1,linecolor='y')  # sns.savefig("output.png")
sns.pairplot(dataset,
kind="reg")
plt.show()
```

```
from sklearn.preprocessing
import LabelEncoder
le = LabelEncoder()
dataset['TRANSMISSION'] =
le.fit_transform(dataset['TRANSMISSION'])
dataset['FUELTYPE'] =
le.fit_transform(dataset['FUELTYPE'])
```

```
dataset =
dataset._get_numeric_data()
dataset=dataset.drop(['MODELYEAR'],axis=1)
dataset
x =
dataset.iloc[:,0:8].values
y =
dataset.iloc[:,8:9].values
y.shape
x[1]
```

```
from sklearn.preprocessing
import OneHotEncoder
one = OneHotEncoder()
z =
one.fit_transform(x[:,3:4]).toarray()
x= np.delete(x,[3],axis =1)
x= np.concatenate((z,x),axis
=1)
x.shape
x[200]
y
```

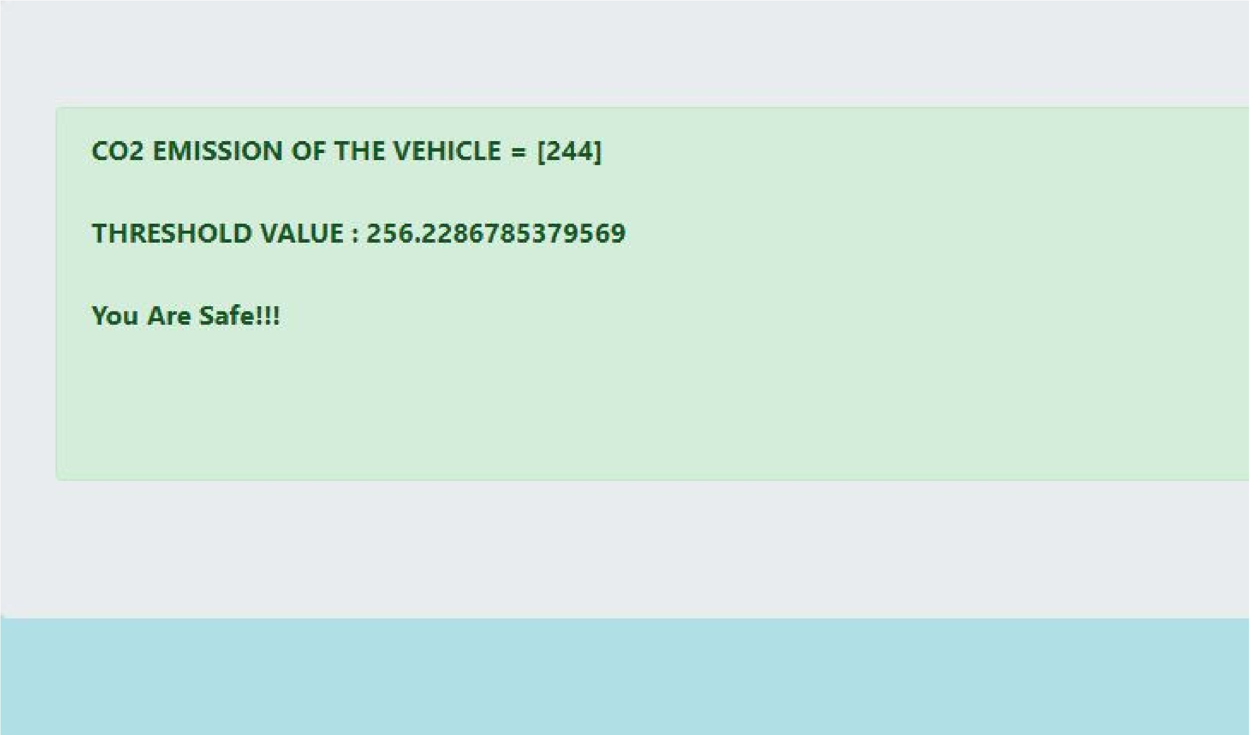
```
from sklearn.model_selection
import train_test_split
x_train,x_test,y_train,y_test
=
train_test_split(x,y,test_size
= 0.2, random_state = 10)
from sklearn.preprocessing
import StandardScaler
sc = StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
from sklearn.ensemble import
RandomForestRegressor
rdr =
RandomForestRegressor(n_estimators
= 500,random_state = 20)
rdr.fit(x_train,y_train)
threshold =
dataset['CO2EMISSIONS'].mean()
threshold
yrdr =rdr.predict(x_test)
yhat_train =
rdr.predict(x_train)
yrdr
y_test
```

```
from sklearn.metrics import
r2_score
accuracy =
r2_score(y_test,yrdr)
print("Test Accuracy :
",r2_score(y_test,yrdr))
print("Training Accuracy :
",r2_score(yhat_train,y_train))
accuracy
yrdr=yrdr.reshape(214,1)
yrdr
y_test
rdr.predict([[ 0. ,  0. ,
1. ,  0. ,  1.8,  4. , 20. ,
9.5,  6.5,  8.2, 34. ]])
```

```
pickle.dump(rdr,open('carco2new.pkl','wb'
```

RESULT

If the CO2 emission from the car is less than the Threshold Value :

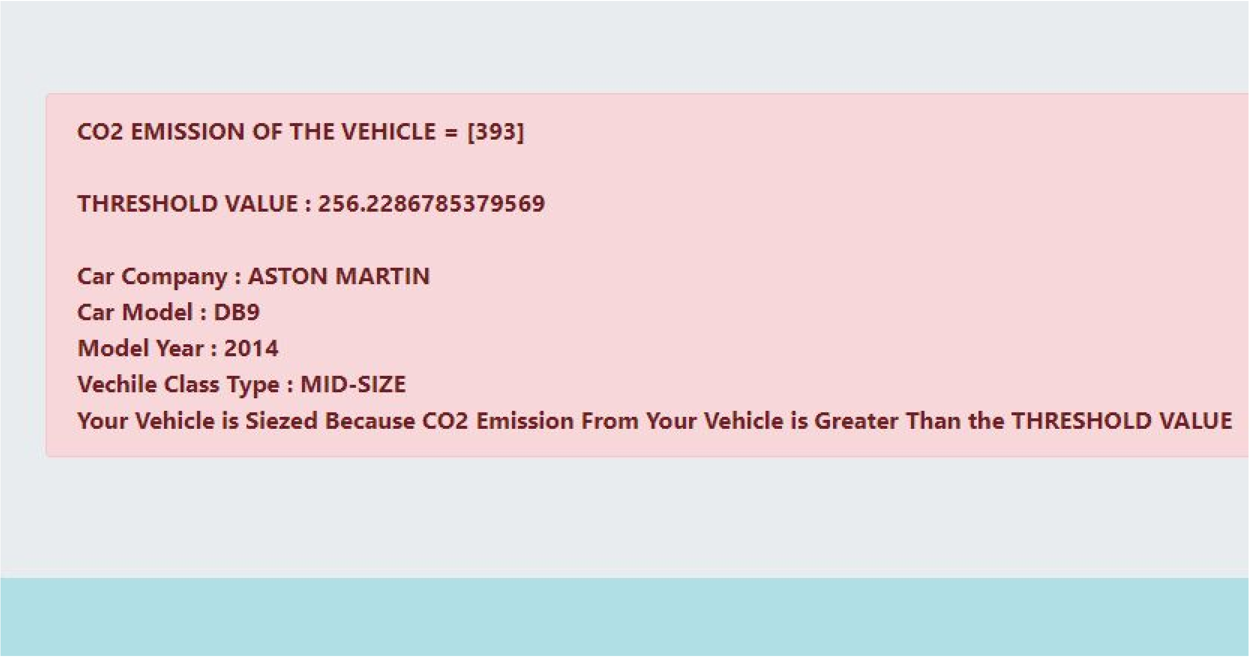


CO2 EMISSION OF THE VEHICLE = [244]

THRESHOLD VALUE : 256.2286785379569

You Are Safe!!!

If the CO2 emission from the car is more than the Threshold Value :



CO2 EMISSION OF THE VEHICLE = [393]

THRESHOLD VALUE : 256.2286785379569

Car Company : ASTON MARTIN
Car Model : DB9
Model Year : 2014
Vechile Class Type : MID-SIZE
Your Vehicle is Siezed Because CO2 Emission From Your Vehicle is Greater Than the THRESHOLD VALUE

ADVANTAGES & DISADVANTAGES

1. Officials can check immediately if the vehicle is following the standardized values for Co2 emissions. They can seize the vehicles that do not comply or even stop the production of the vehicle.

APPLICATIONS

- Random forest algorithms can be used for both classifications and regression tasks.
- It provides higher accuracy.
- Random forest classifiers will handle the missing values and maintain the accuracy of a large proportion of data.
- If there are more trees, it won't allow overfitting trees in the model.
- It has the power to handle a large data set with higher dimensionality

There are several applications where the random forest can be applied. We will discuss some of the sectors where random forest can be applied. We will also look closer when the random forest analysis comes into the role.

Banking Sector:

The banking sector consists of most users. There are many loyal customers and also fraud customers. To determine whether the customer is a loyal or fraud, Random forest analysis comes in. With the help of a random forest algorithm in machine learning, we can easily determine whether the customer is fraud or loyal. A system uses a set of a random algorithm which identifies the fraud transactions by a series of the pattern.

Medicines:

Medicine needs a complex combination of specific chemicals. Thus, to identify the great combination in the medicines, Random forest can be used. With the help of machine learning algorithms, it has become easier to detect and predict the drug sensitivity of a medicine. Also, it helps to identify the patient's disease by analyzing the patient's medical record.
Stock Market: Machine learning also plays a role in the stock market analysis. When you want to know the behavior of the stock market, with the help of Random forest algorithm, the behavior of the stock market can be analyzed. Also, it can show the expected loss or profit which can be produced while purchasing a particular stock.

E-Commerce:

When you will find it difficult to recommend or suggest what type of products your customer should see. This is where you can use a random forest algorithm. Using a machine learning system, you can suggest the products which will be more likely for a customer. Using a certain pattern and following the product's interest of a customer, you can suggest similar products to your customers.

CONCLUSION

Hence through our project we can predict whether the co2 Emission from the car is above the threshold value or not. If it is greater than the threshold value then a warning message will be displayed indicating that it will be seized or else it will display that your car is safe.

The more data we have, the better we can predict and coherently prevent failures and carbon emissions through additional energy use

FUTURE SCOPE

Looking at the current market situation in Germany, there is still a lot of potential for increasing the innovation and penetration level. Young and innovative companies like Evertracker (offering an AI based platform based control tower for multiple activities within the Supply Chain) have made a successful entry to the market, but the market share of digital native companies or services are yet extremely low - the logistic industry has to catch up with the current state of ML applications though in order to accomplish the ambitious carbon emission targets.

BIBLIOGRAPHY

1. https://www.kaggle.com/berhag/co2-emission-forecast-with-python-seasonal-arima/data - Using data from Carbon Emissions
2. https://www.epa.gov/greenvehicles/greenhouse-gas-emissions-typical-passenger-vehicle - Greenhouse Gas Emissions from a Typical Passenger Vehicle

APPENDIX

A. Source code of MACHINE LEARNING (Jupyter Notebook)

```
## import libraries
import numpy as np
import pandas as pd
import pickle

## importing dataset
dataset = pd.read_csv("FuelConsumption.csv")
dataset.head()
```

```
## checking for null values
dataset.isnull().any()
dataset.describe()

## information of the dataset
dataset.info()
dataset['MAKE'].nunique()
dataset['MODEL'].nunique()
dataset['VEHICLECLASS'].nunique()
dataset['TRANSMISSION'].nunique()
dataset['FUELTYPE'].nunique()
dataset["MODELYEAR"].nunique()
temp=dataset.corr()
Temp

#### Taking only those columns whose correlation is greater than 50%
tmp=dataset.corr()
imp_cols = tmp[abs(tmp['CO2EMISSIONS']) > 0.5].index.tolist()
Imp_cols

### Data visualization
import matplotlib.pyplot as plt
import seaborn as sns
sns.heatmap(dataset[imp_cols].corr(),cmap='RdBu',annot=True,linewidths=1,linecolor='y')

# sns.savefig("output.png")
sns.pairplot(dataset, kind="reg")
plt.show()

## Label Encoding
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
dataset['TRANSMISSION'] = le.fit_transform(dataset['TRANSMISSION'])
dataset['FUELTYPE'] = le.fit_transform(dataset['FUELTYPE'])

#### Cosidering only numerical data
dataset = dataset._get_numeric_data()
dataset=dataset.drop(['MODELYEAR'],axis=1)
dataset
x = dataset.iloc[:,0:8].values
y = dataset.iloc[:,8:9].values
y.shape
x[1]

#### one hotEncoding
from sklearn.preprocessing import OneHotEncoder
one = OneHotEncoder()
z = one.fit_transform(x[:,3:4]).toarray()
x= np.delete(x,[3],axis =1)
x= np.concatenate((z,x),axis =1)
x.shape
x[200]
y

### test and 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, random_state = 10)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)

### Using RandomForestRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
rdr = RandomForestRegressor(n_estimators = 500,random_state = 20)
rdr.fit(x_train,y_train)
threshold = dataset['CO2EMISSIONS'].mean()
threshold
yrdr =rdr.predict(x_test)
yhat_train = rdr.predict(x_train)
yrdr
y_test

### Evaluation
from sklearn.metrics import r2_score
accuracy = r2_score(y_test,yrdr)
print("Test Accuracy : ",r2_score(y_test,yrdr))
print("Training Accuracy : ",r2_score(yhat_train,y_train))
accuracy
yrdr=yrdr.reshape(214,1)
yrdr
y_test
rdr.predict([[ 0. ,  0. ,  1. ,  0. ,  1.8,  4. , 20. ,  9.5,  6.5,  8.2, 34. ]])

### Saving the Model
pickle.dump(rdr,open('carco2new.pkl','wb'))
```

B. Source code of FLASK APP (Spyder)

```
from flask import Flask , render_template, request
import pickle
app = Flask(__name__)
model = pickle.load(open('carco2new.pkl','rb'))

@app.route('/')
def hello_world():
    return render_template('index.html')
```

```
@app.route('/login', methods = ["POST"])
def login():
   BRAND=request.form["BRAND"]
   CM=request.form["CM"]
   MY=request.form["MY"]
   VC=request.form["VC"]
   FT=request.form["FT"]
   if(FT =="Z"):
      s1,s2,s3,s4 = 0,0,0,1
   if(FT =="D"):
      s1,s2,s3,s4 = 0,0,1,0
   if(FT =="E"):
      s1,s2,s3,s4 = 0,1,0,0
   if(FT =="X"):
      s1,s2,s3,s4 = 1,0,0,0
   ES=request.form["ES"]
   C=request.form["C"]
   T=request.form["T"]
   FCC=request.form["FCC"]
   FCHWY=request.form["FCHWY"]
   FCCC=request.form["FCCC"]
   FCCMPG=request.form["FCCMPG"]

                                                    total          =
[[s1,s2,s3,s4,float(ES),int(C),int(T),float(FCC),float(FCHWY),float(FCCC),int(FC
CMPG)]]

   p = model.predict(total)

   if(p>256.2286785379569):
          return render_template('details.html',label1 = "CO2 EMISSION OF THE
VEHICLE = "+str(p), label2 ="THRESHOLD VALUE : 256.2286785379569",
label3 ="Car Company : "+str(BRAND), label4 ="Car Model : "+str(CM), label5
="Model Year : "+str(MY), label6 ="Vechile Class Type : "+str(VC), label7 ="
Your Vehicle is Siezed Because CO2 Emission From Your Vehicle is Greater
Than the THRESHOLD VALUE",label8=1)
    else:
            return render_template('details.html',label1 = "CO2 EMISSION OF THE
VEHICLE = "+str(p), label2 ="THRESHOLD VALUE : 256.2286785379569",
label3 ="You Are Safe!!! ",label8=0)



if __name__=='__main__':
    app.run(debug=True,port=9000)
```

C. Source code of HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
                                        <link                rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5M
YYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">
    <!-- <script src='fms.js'></script> -->
     <script type = "text/javascript"  src = "{{ url_for('static', filename = 'fms.js') }}"
></script>
</head>
<body style="background-color:powderblue">
    <div class="container p-3 my-3 bg-dark text-white">
    <form action="http://localhost:9000/login" method="POST">
```

```
        <div class="form-group">
          <label for="BRAND">Enter the Car Company</label>
                           <select class="browser-default custom-select" name="BRAND"
id="BRANDSel" required>
                <option value="" selected="selected">Enter the Car Company</option>
              </select>
        </div>
        <div class="form-group">

            <label for="BRAND">Enter the Car Model</label>
                <select class="browser-default custom-select" name="CM" id="CMSel"
required>
                <option value="" selected="selected">Select BRAND FIRST </option>
              </select>
        </div>
        <div class="form-group">

            <label for="BRAND">Vehicle Class Type</label>
                     <select class="browser-default custom-select" name="VC" id="VC"
required>

                <option value ="COMPACT">COMPACT</option>
                <option value ="SUV - SMALL">SUV-SMALL</option>
                <option value ="MID-SIZE">MID-SIZE</option>
                <option value ="MINICOMPACT">MINICOMPACT</option>
                <option value ="SUBCOMPACT">SUBCOMPACT</option>
                <option value ="TWO-SEATER">TWO-SEATER</option>
                <option value ="FULL-SIZE">FULL-SIZE</option>
                                                    <option value ="STATION WAGON -
SMALL">STATION_WAGON-SMALL</option>
                <option value ="SUV - STANDARD">SUV-STANDARD</option>
                <option value ="VAN - CARGO">VAN-CARGO</option>
                <option value ="VAN - PASSENGER">VAN-PASSENGER</option>
```

```
                                <option value ="PICKUP TRUCK -
STANDARD">PICKUP_TRUCK-STANDARD</option>
        <option value ="MINIVAN">MINIVAN</option>
                                <option value ="SPECIAL PURPOSE
VEHICLE">SPECIAL_PURPOSE_VEHICLE</option>
       </select>
    </div>

  <div class="form-group">
      <label for="EngineSize"> Model Year</label>
         <input class="form-control" type="text" placeholder=" Enter the Model
YEAR" name="MY" required>
    </div>

    <div class="form-group">

      <label for="BRAND">Fuel Type</label>
           <select class="browser-default custom-select" name="FT" id="FT"
required>
        <option value ="">Select Fule Type</option>
        <option value ="Z">Z</option>
        <option value ="X">X</option>
        <option value ="E">E</option>
        <option value ="D">D</option>
       </select>
    </div>

    <div class="form-group">
      <label for="EngineSize"> ENGINE SIZE</label>
        <input class="form-control" type="text" placeholder=" Enter the Engine
Size" name="ES" required>
    </div>
    <div class="form-group">
```

```
          <label for="EngineSize">CYLINDERS</label>
                    <input class="form-control" type="text" placeholder="Enter the
CYLINDERS" name="C" required>
       </div>
       <div class="form-group">
          <label for="EngineSize">TRANSMISSION</label>
                    <input class="form-control" type="text" placeholder="Enter the
TRANSMISSION" name="T" required>
       </div>
       <div class="form-group">
          <label for="EngineSize">FUEL CONSUMPTION CITY</label>
                    <input class="form-control" type="text" placeholder="Enter the
FUELCONSUMPTION_CITY" name="FCC" required>

       </div>
       <div class="form-group">
          <label for="EngineSize">FUEL CONSUMPTION HWY</label>
                    <input class="form-control" type="text" placeholder="Enter the
FUELCONSUMPTION_HWY" name="FCHWY" required>

       </div>
       <div class="form-group">
          <label for="EngineSize">FUEL CONSUMPTION COMB</label>
                    <input class="form-control" type="text" placeholder="Enter the
FUELCONSUMPTION_COMB<" name="FCCC" required>

       </div>
       <div class="form-group">

          <label for="EngineSize">FUEL CONSUMPTION COMB MPG</label>
                <input class="form-control" type="text" placeholder="Enter the FUEL
CONSUMPTION COMB MPG" name="FCCMPG" required>
       </div>
```

```
    <button type="submit" class="btn btn-primary">Submit</button>
   </form>



  </div>
   <b>{{label1}}</b>
    <br>
    <br>
         <b>{{label2}}</b>
    <br>
    <br>
        <b>{{label3}}</b>
    <br>
        <b>{{label4}}</b>
    <br>
        <b>{{label5}}</b>
    <br>
         <b>{{label6}}</b>
    <br>
        <b>{{label7}}</b>


<script                    src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+Ib
bVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3U
ksdQRVvoxMfooAo" crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3I
pu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>
```

```
</body>
</html>
```

D. Source code of DETAILS (CSS)

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
                                                    <link                rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5M
YYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">

</head>
<body style="background-color:powderblue">
   <div class="jumbotron">
   {%if label8==0%}
      <div class="alert alert-success" role="alert">
       <b>{{label1}}</b>
      <br>
      <br>
            <b>{{label2}}</b>
      <br>
      <br>
           <b>{{label3}}</b>
      <br>
           <b>{{label4}}</b>
      <br>
           <b>{{label5}}</b>
      <br>
```

```
            <b>{{label6}}</b>
        <br>
            <b>{{label7}}</b>




        </div>
{% endif %}
    {%if label8==1 %}
        <div class="alert alert-danger" role="alert">
        <b>{{label1}}</b>
        <br>
        <br>
            <b>{{label2}}</b>
        <br>
        <br>
            <b>{{label3}}</b>
        <br>
            <b>{{label4}}</b>
        <br>
            <b>{{label5}}</b>
        <br>
            <b>{{label6}}</b>
        <br>
            <b>{{label7}}</b>
        </div>
        {% endif %}


      </div>
```

```
                              <script     src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+Ib
bVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3U
ksdQRVvoxMfooAo" crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3I
pu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>


</body>
</html>
```