Machine Learning For Prediction of CO2 Emissions By Country To Carbon Footprint Analysis

SUBMITTED BY
ANJANA PRABHAKARAN

CONTENTS

INTRODUCTION

Carbon emissions and environmental protection issues have brought pressure from the international community during Chinese economic development. Recently, Chinese Government announced that carbon emissions per unit of GDP would fall by 60–65% compared with 2005 and non-fossil fuel energy would account for 20% of primary energy consumption by 2030.The Beijing-Tianjin-Hebei region is an important regional energy consumption center in China, and its energy structure is typically coal-based which is similar to the whole country.

Therefore, forecasting energy consumption related to carbon emissions is of great significance to emissions reduction and upgrading of energy supply in the Beijing-Tianjin-Hebei region. Thus, this study thoroughly analyzed the main energy sources of carbon emissions including coal, petrol, natural gas, and coal power in this region.

1.1 OVERVIEW

Predictive Machine Learning (ML) models and the big amount of available data can be very useful to analyze the development of climate change trends or relevant contributors. In theory, the country emissions of greenhouse gases such as CO2 over a year could depend on certain country-specific aspects. In this context, I have developed a ML project aiming to analyze and predict CO2 emissions from country-specific parameters such as economic indicators, population, energy use, land use, etc.

1.2 PURPOSE

A Machine Learning Model for calculating CO2 emission by country due to the increasingly deteriorating environment, it is time the government to upgrade the energy consumption structure.

LITERATURE SURVEY

2.1 Existing Problem

Due to the increasingly deteriorating environment, it is time for the government to upgrade the Energy Consumption structure by making use of Machine Learning prediction to analyze and control the CO_2 emissions in future.

2.2 Proposed Solution

A Machine Learning model for calculating CO_2 Emissions by country

THEORETICAL ANALYSIS

3.1 HARDWARE / SOFTWARE DESIGNING

HARDWARE DESIGNING

The hardware required for the development of this project is:

Processor : Intel CoreTM i5-9300H
Processor speed : 2.4GHz
RAM Size : 8 GB DDR
System Type : X64-based processor

SOFTWARE DESIGNING:

The software required for the development of this project is:

Desktop GUI - Anaconda Navigator
Operating system - Windows 10
Front end - HTML, CSS, JAVASCRIPT
Programming - PYTHON
Cloud Computing Service - IBM Cloud Services

EXPERIMENTAL INVESTIGATION

IMPORTING AND READING THE DATASET

Importing the Libraries

First step is usually importing the libraries that will be needed in the program.

- **Numpy-** It is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines. Pandas objects are very much dependent on NumPy objects.
- **Pandas-** It is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.
- **Counter:** Python Counter is a container that will hold the count of each of the elements present in the container.
- **Matplotlib and Seaborn:** Both are the data visualization library used for plotting graphs which will help us to understand the data.
- **Accuracy score:** used in classification type problems and for finding accuracy it is used.
- **Train_test_split:** used for splitting data arrays into training data and for testing data.
- **Pickle:** to serialize your machine learning algorithms and save the serialized format to a file.
- **Random Forest Regressor:** random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.
- **Label Encoding:** It is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

Reading the Dataset

For this project, we use a dataset named 'Indicators.csv'. We will be selecting the important features from these datasets that will help us in recommending the best results.

The next step is to read the dataset into a data structure that's compatible with pandas. Let's load a .csv data file into pandas. There is a function for it, called **read_csv().**We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program).If the dataset in same directory of your program, you can directly read it, without any path. After the next Steps we made following bellow:

1.Data visualization

2.Collabrative and filtering

3.Creating the Model

4.Test and save the model

5.Buil Python Code

6.Build HTML Code

7.Run the Application
We are the following above sections we did and investigate it.

FLOWCHART

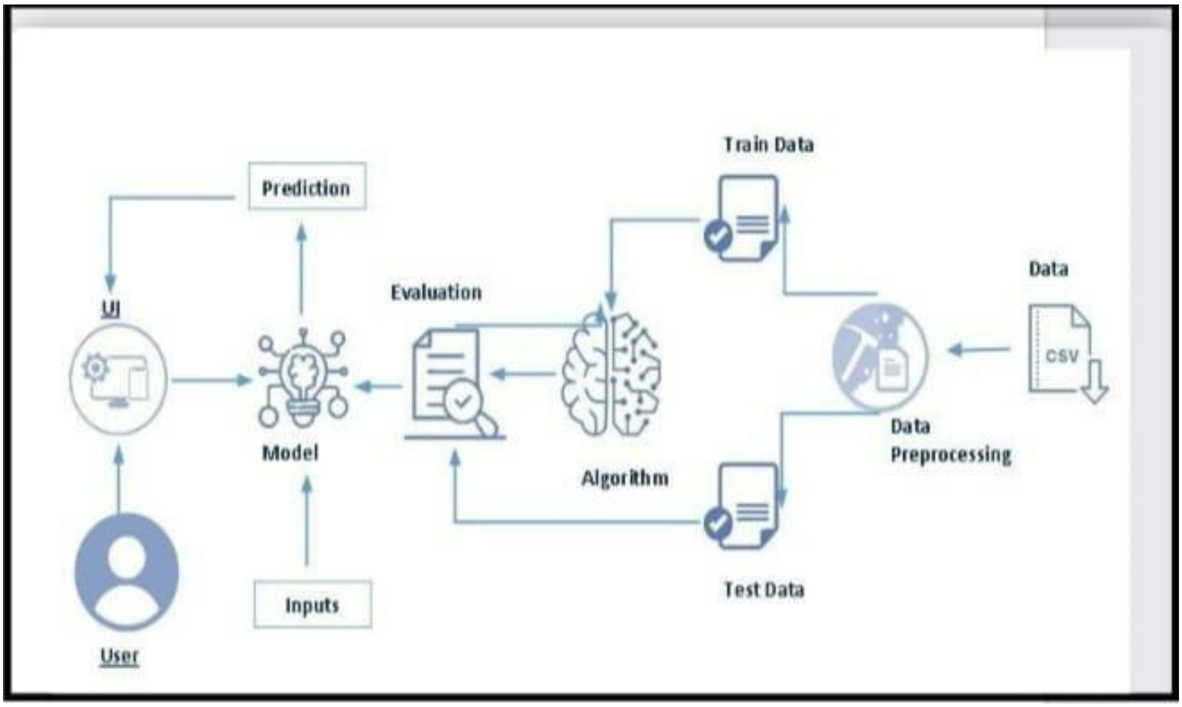

Fig 5.1 Flowchart of the project

Project Flow:

- User interacts with the UI (User Interface) to upload the input features.
- Uploaded features/input is analysed by the model which is integrated.

Once a model analyses the uploaded inputs, the prediction is showcased on the UI.

1. **Data Collection.**

   - Collect the dataset or Create the dataset

2. **Data Pre- processing.**

   - Import the Libraries.
   - Importing the dataset.
   - Exploratory Data Analysis
   - Data Visualization

3. **Collaborating Filtering**

   - Merging datasets
   - Creating the Model
   - Predicting the results
   - Saving our model and dataset

4. **Application Building**

   - Create an HTML file
   - Build a Python Code

RESULT

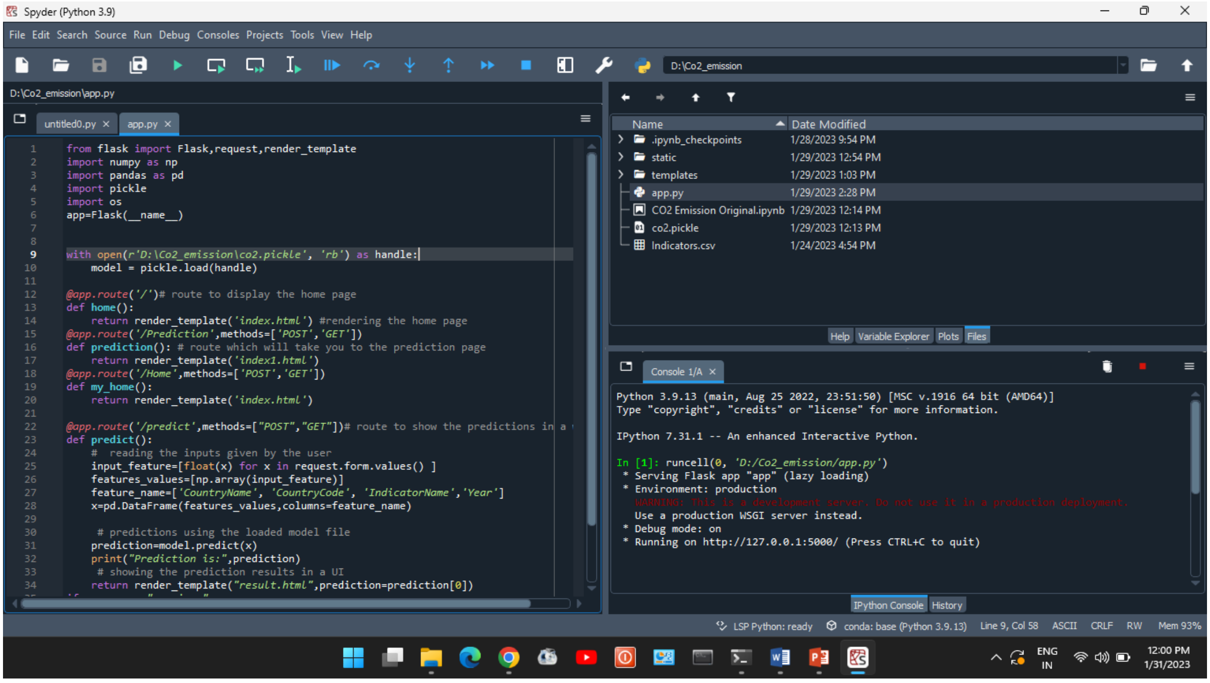

Fig 6.1 Flask App Code with Output Page

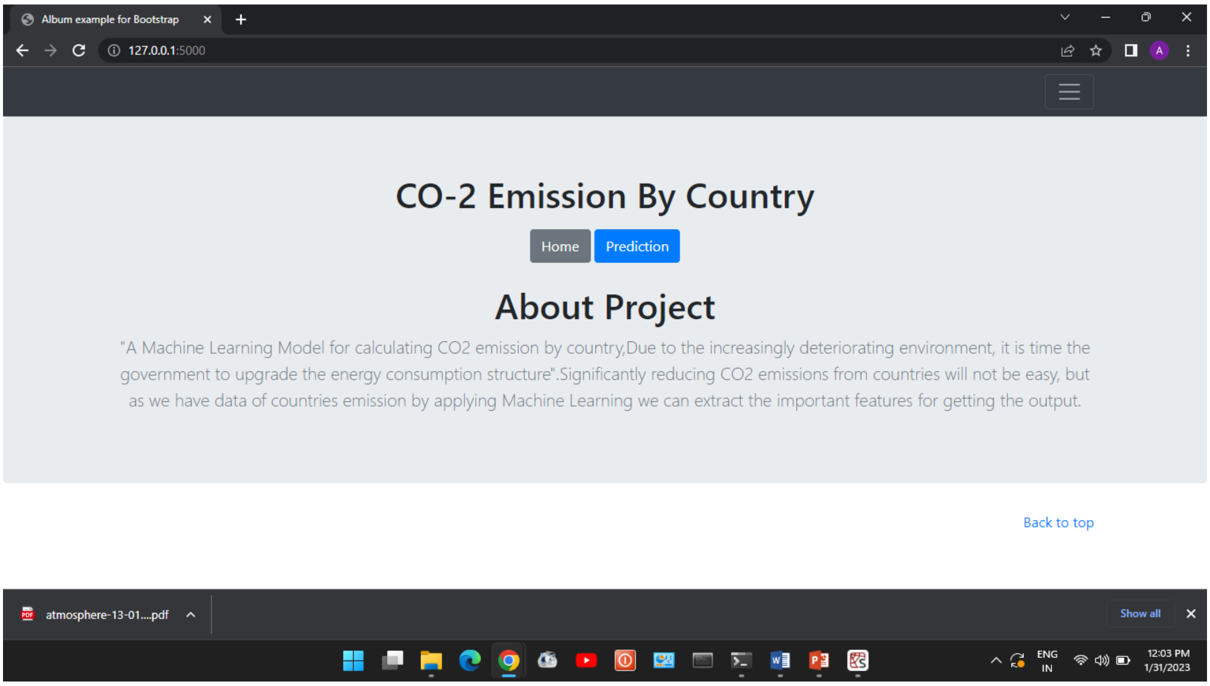

Fig 6.2 Home Page CO-2 Emission By Country using IBM Watson, In IBM

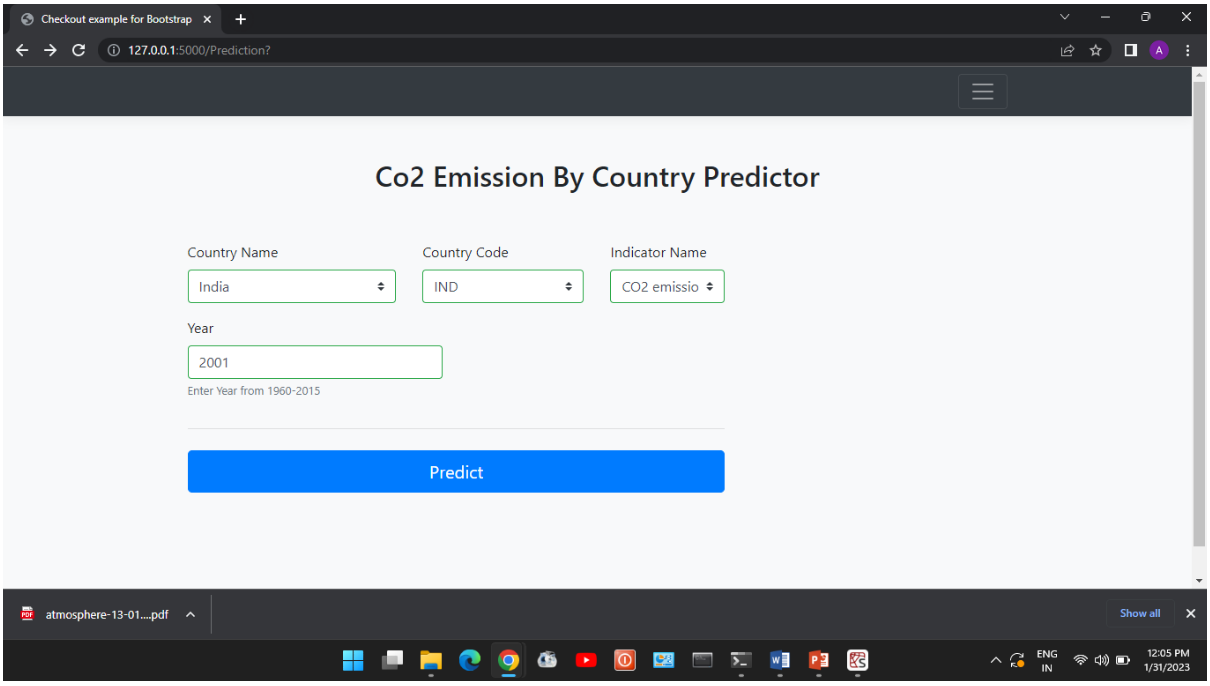

Fig 6.3 Prediction page of CO-2 Emission By Country System using IBM Watson

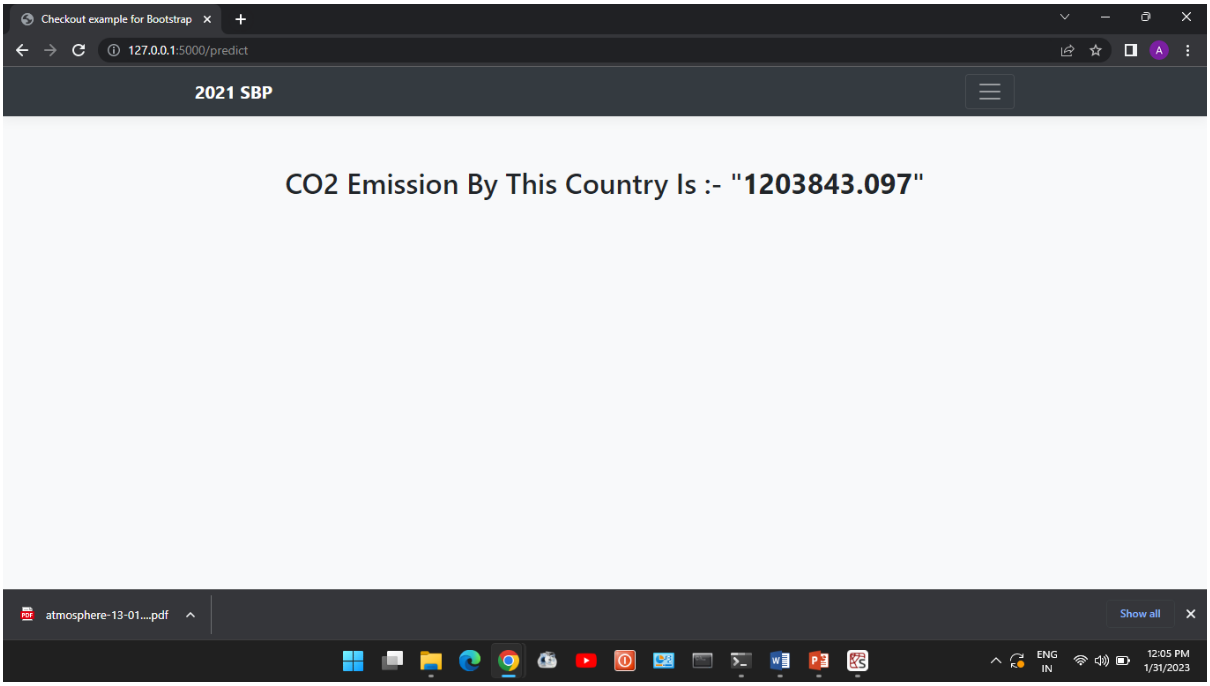

Fig 6.4 Output of Prediction page of CO-2 Emission By Country System using IBM Watson

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- Slow global warming
- Protection of species
- Fairer distribution of the costs of emissions
- Higher investments in R&D

DISADVANTAGES

- Increase in product prices
- Some companies may go out of business
- Higher unemployment rates
- Lower profits

APPLICATIONS

- The carbon in CO_2 can be used to produce fuels that are in use today, including methane, methanol, gasoline and aviation fuels.
- CO_2 is a versatile industrial material, used, for example, as an inert gas in welding and fire extinguishers, as a pressurizing gas in air guns and oil recovery, and as a supercritical fluid solvent in decaffeination of coffee and supercritical drying. It is also a feedstock for the synthesis of fuels and chemicals.
- Carbon dioxide is used as a refrigerant, in fire extinguishers, for inflating life rafts and life jackets, blasting coal, foaming rubber and plastics, promoting the growth of plants in greenhouses, immobilizing animals before slaughter, and in carbonated beverages.

CONCLUSION AND FUTURESCOPE

CONCLUSION

Carbon emissions, the greenhouse effect, climate change and catastrophic environmental issues have become the most crucial issues in the contemporary world. Application of AI and ML have a significant impact in terms of solving these issues. This work focuses on using AI to develop an ML model for global total CO_2 emissions to forecast CO_2 emissions for the near or far future. Building ML models considering reduced CO_2 emissions during the COVID-19 pandemic, we found some noticeable outcomes which can help in understanding CO_2 emissions across the world.

FUTURESCOPE

New IEA analysis of the latest data from around the world shows that these CO_2 emissions are on course to increase by close to 300 million tonnes in 2022 to 33.8 billion tonnes – a far smaller rise than their jump of nearly 2 billion tonnes in 2021, which resulted from the rapid global recovery from the economic crisis.

BIBILOGRAPHY

1. Jayant Tikmani, Sudhanshu Tiwari, Sujata Khedkar “Telecom customer segmentation based on
2. cluster analysis An Approach to Customer Classification using k-means”, IJIRCCE,Year: 2015.
3. [6]Yogita Rani and Dr. Harish Rohil “A Study of Hierarchical Clustering Algorithm”, IJICT, Year:
4. 2013.
5. [7]Omar Kettani, Faycal Ramdani, Benaissa Tadili “An Agglomerative Clustering Method for Large
6. Data Sets”, IJCA, Year: 2014.
7. [8]Snekha, Chetna Sachdeva, Rajesh Birok “Real Time Object Tracking Using Different Mean Shift

APPENDIX

A Source Code of Flask:

```
from flask import Flask,request,render_template
import numpy as np
import pandas as pd
import pickle
import os
app=Flask(__name__)



with open(r'D:\Co2_emission\co2.pickle', 'rb') as handle:
    model = pickle.load(handle)

@app.route('/')# route to display the home page
def home():
    return render_template('index.html') #rendering the home page
@app.route('/Prediction',methods=['POST','GET'])
def prediction(): # route which will take you to the prediction page
    return render_template('index1.html')
@app.route('/Home',methods=['POST','GET'])
def my_home():
    return render_template('index.html')

@app.route('/predict',methods=["POST","GET"])# route to show the predictions in a web UI
def predict():
    #  reading the inputs given by the user
    input_feature=[float(x) for x in request.form.values() ]
    features_values=[np.array(input_feature)]
    feature_name=['CountryName', 'CountryCode', 'IndicatorName','Year']
```

```
    x=pd.DataFrame(features_values,columns=feature_name)

    # predictions using the loaded model file
    prediction=model.predict(x)
    print("Prediction is:",prediction)
    # showing the prediction results in a UI
    return render_template("result.html",prediction=prediction[0])
if __name__=="__main__":

    # app.run(host='0.0.0.0', port=8000,debug=True)    # running the app
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,debug=True,use_reloader=False)
```