PREDICTION OF CO2 EMISSIONS BY COUNTRIES

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

MYADARABOINA VAMSHI	**21UK1A6601**
JANGALA PRIYANKA	**21UK1A6603**
MAMIDI MALAVIKA	**21UK1A6618**
GUDISE SAGAR	**21UK1A6663**
KUNOORU RUSHIKESH	**21UK1A6621**

Under the guidance of

Dr.MOHAMMAD ABDUL WAJEED

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION
INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "PREDICTING CO2 EMISSIONS BY COUNTRIES USING MACHINE LEARNING" is being submitted by MYADARABOINA VAMSHI(21UK1A6601),JANGALA PRIYANKA(2UK1A6603),MAMIDI MALAVIKA(21UK1A6618),GUDISE SAGAR(21UK1A6663) , KUNOORU RUSHIKESH(21UK1A6621)in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2023- 2024.

Project Guide
Mrs.K.Sowjanya
(Assistant Professor)

HOD
Dr. K. Sharmila
(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.P.PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.K.SHARMILA**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **Dr.MOHAMMAD ABDUL WAJEED ,** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

MYADARABOINA VAMSHI(21UK1A6601),JANGALA PRIYANKA(2UK1A6603),MAMIDI MALAVIKA(21UK1A6618),GUDISE SAGAR(21UK1A6663),KUNOORU RUSHIKESH(21UK1A6621)

.

ABSTRACT

Telecom customer churn prediction is a crucial aspect of the telecommunications industry, aimed at identifying customers who are likely to terminate their services. This project focuses on developing a predictive model to forecast customer churn using machine learning techniques. The dataset encompasses various features such as customer demographics, account information, and usage patterns. By analyzing these attributes, the model aims to uncover patterns and trends indicative of potential churn. The objective is to enable proactive retention strategies by telecom companies, ultimately enhancing customer satisfaction and reducing turnover rates. This research highlights the importance of data-driven decision-making in maintaining a competitive edge in the telecom sector.

TABLE OF CONTENTS:-

1.INTRODUCTION

1.1.OVERVIEW

In this project, the goal is to predict CO2 emissions of countries using machine learning techniques. The process involves gathering and cleaning data from reliable sources such as international databases and national statistics, selecting relevant features like GDP, population, and energy consumption, and then conducting exploratory data analysis to understand relationships and patterns in the data. Various regression models such as linear regression, random forest will be trained and evaluated using metrics , and the final model will be deployed to make predictions on CO2 emissions for new data points. Ethical considerations, including biases in data and model fairness, will be addressed throughout the project, with documentation and reporting ensuring transparency and reproducibility of results. Global CO2 emissions contribute significantly to climate change, and there is a need for accurate prediction models to help policymakers and researchers understand future trends and plan mitigation strategies. The challenge is to create a model that can predict CO2 emissions with high accuracy using historical data and other relevant features.

1.2.PURPOSE

By pursuing these objectives step-by-step, the project aims to leverage machine learning as a transformative tool for tackling one of the most pressing challenges of our time—climate change—and contributing to a sustainable future for all.

1. Enhance Understanding of Emission Dynamics

By developing predictive models for CO2 emissions, the project aims to deepen our understanding of the complex interplay between socio-economic factors (such as GDP, population, industrial activity) and environmental outcomes. This understanding is crucial for identifying the primary drivers of emissions within and across countries.

2. Enable Proactive Policy Formulation

Armed with accurate predictions, policymakers can formulate proactive and targeted policies aimed at reducing carbon emissions. These policies may include regulatory measures, incentives for renewable energy adoption, and initiatives to promote energy efficiency and sustainable practices.

3. Support Climate Change Mitigation Efforts

The project seeks to support global climate change mitigation efforts by providing reliable tools for assessing current emission trends and forecasting future trajectories. This information is essential for countries to align with international agreements (e.g., Paris Agreement) and set ambitious emission reduction targets.

4. Facilitate Adaptation and Resilience Planning

Predictive models can also aid in adaptation planning by anticipating the impacts of climate change on different regions and sectors. This capability enables proactive measures to enhance resilience to climate-related challenges, such as extreme weather events and sea-level rise.

5. Empower Decision-Makers with Actionable Insights

By generating actionable insights from data, the project empowers decision-makers at various levels—national governments, local authorities, businesses, and civil society organizations—to make informed choices that balance economic development with environmental sustainability.

6. Drive Innovation and Research

The development and refinement of machine learning models for predicting CO2 emissions spur innovation in environmental science, data analytics, and policy research. This innovation contributes to the advancement of technologies and methodologies for measuring, monitoring, and mitigating greenhouse gas emissions.

7. Promote Global Collaboration and Accountability

By providing transparent and robust methodologies for estimating CO2 emissions, the project fosters international collaboration and accountability in addressing climate change. It encourages data sharing, best practices exchange, and peer-reviewed validation of emission estimates.

10. Continuous Improvement and Adaptation

The project is designed to evolve over time, incorporating new data sources, advancing modeling techniques, and responding to emerging environmental challenges. Continuous

improvement ensures that the predictive models remain relevant, reliable, and impactful in addressing the global climate crisis.

2.LITERATURE SURVEY

Existing Problem in Predicting CO2 Emissions of Countries

1. **Complexity of Factors**: CO2 emissions are influenced by a multitude of complex factors such as economic activities, population dynamics, industrial output, energy consumption patterns, and environmental policies. The challenge lies in effectively capturing and integrating these diverse variables into predictive models.
2. **Data Availability and Quality**: Obtaining reliable and consistent data across countries and over time is a significant hurdle. Variations in data collection methods, reporting standards, and data completeness can introduce biases and inaccuracies into the analysis.
3. **Non-linear Relationships**: The relationships between predictors (e.g., GDP, energy consumption) and CO2 emissions may not be linear. Identifying and modeling non-linearities requires sophisticated techniques beyond traditional linear regression approaches.
4. **Spatial and Temporal Variability**: CO2 emissions exhibit spatial variability across regions and countries, as well as temporal variability influenced by seasonal changes, economic fluctuations, and policy interventions. Capturing these dynamics accurately is essential for robust predictions.
5. **Model Generalization**: Ensuring that predictive models generalize well across different regions and time periods is a critical concern. Overfitting to specific datasets or underestimating uncertainties can compromise the reliability and applicability of the models.

Proposed Solutions in Predicting CO2 Emissions of Countries

1. **Advanced Machine Learning Models**: Utilize advanced machine learning techniques such as ensemble methods (e.g., random forest, gradient boosting), deep learning models (e.g., neural networks), and support vector machines to capture complex relationships and non-linearities in the data.
2. **Feature Engineering**: Develop innovative approaches for feature engineering to extract meaningful predictors from raw data. This may include creating interaction

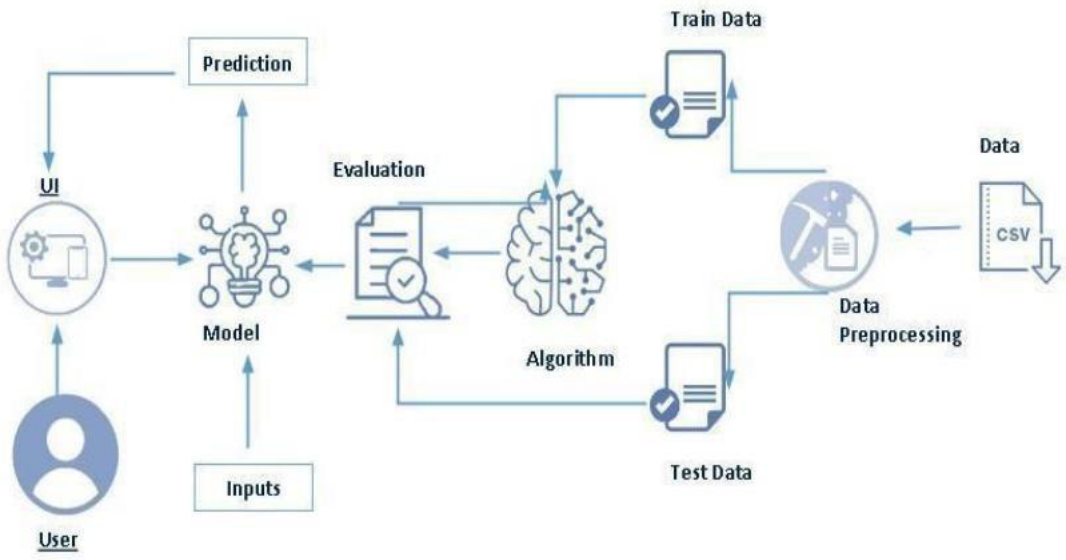
terms, transforming variables, and incorporating domain-specific knowledge to enhance model performance.

3. **Data Fusion and Integration**: Integrate diverse datasets from multiple sources (e.g., satellite imagery, remote sensing data) to enrich the feature set and improve the accuracy of emissions estimates. Employ data harmonization techniques to address discrepancies in reporting standards.
4. **Spatial and Temporal Modeling**: Implement spatial and temporal modeling techniques such as spatial autoregressive models, time series analysis with seasonality adjustments, and geospatial analysis to account for regional variations and temporal trends in emissions.
5. **Uncertainty Quantification**: Develop methodologies to quantify and propagate uncertainties throughout the modeling process. Use probabilistic modeling approaches, ensemble techniques, and sensitivity analyses to assess the robustness of predictions and provide decision-makers with uncertainty bounds.
6. **Validation and Benchmarking**: Validate predictive models using rigorous cross-validation techniques, out-of-sample testing, and benchmarking against established emission inventories or independent datasets. Incorporate feedback mechanisms to continuously refine and improve model performance.
7. **Open Data and Transparency**: Promote transparency in data sources, methodologies, and model outputs to facilitate peer review, replication, and stakeholder engagement. Advocate for open data initiatives to foster collaboration and knowledge sharing across research communities.

By addressing these existing problems and implementing proposed solutions, the field of predicting CO2 emissions of countries using machine learning can advance towards more accurate, reliable, and actionable insights for sustainable development and climate change mitigation strategies.

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM



3.2. SOFTWARE DESIGNING

The following is the Software required to complete this project:

- **Google Colab**: Google Colab will serve as the development and execution environment for your predictive modeling, data preprocessing, and model training tasks. It provides a cloud-based Jupyter Notebook environment with access to Python libraries and hardware acceleration.

- **Dataset (CSV File)**: The dataset in CSV format is essential for training and testing your predictive model. It should include historical air quality data, weather information, pollutant levels, and other relevant features.
- **Data Preprocessing Tools**: Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling and data cleaning.
- **Feature Selection/Drop**: Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency.
- **Model Training Tools**: Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Regression or classification models can be considered, depending on the nature of the churn prediction task.
- **Model Accuracy Evaluation**: After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. You'll measure the model's ability to predict Telecom customer categories based on historical data.
- **UI Based on Flask Environment**: Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for users to input user data or view churn predictions
- Google Colab will be the central hub for model development and training, while Flask will facilitate user interaction and data presentation. The dataset, along with data preprocessing, will ensure the quality of the training data, and feature selection will optimize the model. Finally, model accuracy evaluation will confirm the system's predictive capabilities, allowing users to rely on the telecom customer churn prediction.

EXPERIMENTAL INVESTIGATION

In this project, we have used a dataset. This dataset is a CSV file consisting of labelled data and having the following columns:

1.country name: the name of the country.

2.country code: the unique code representing the country.

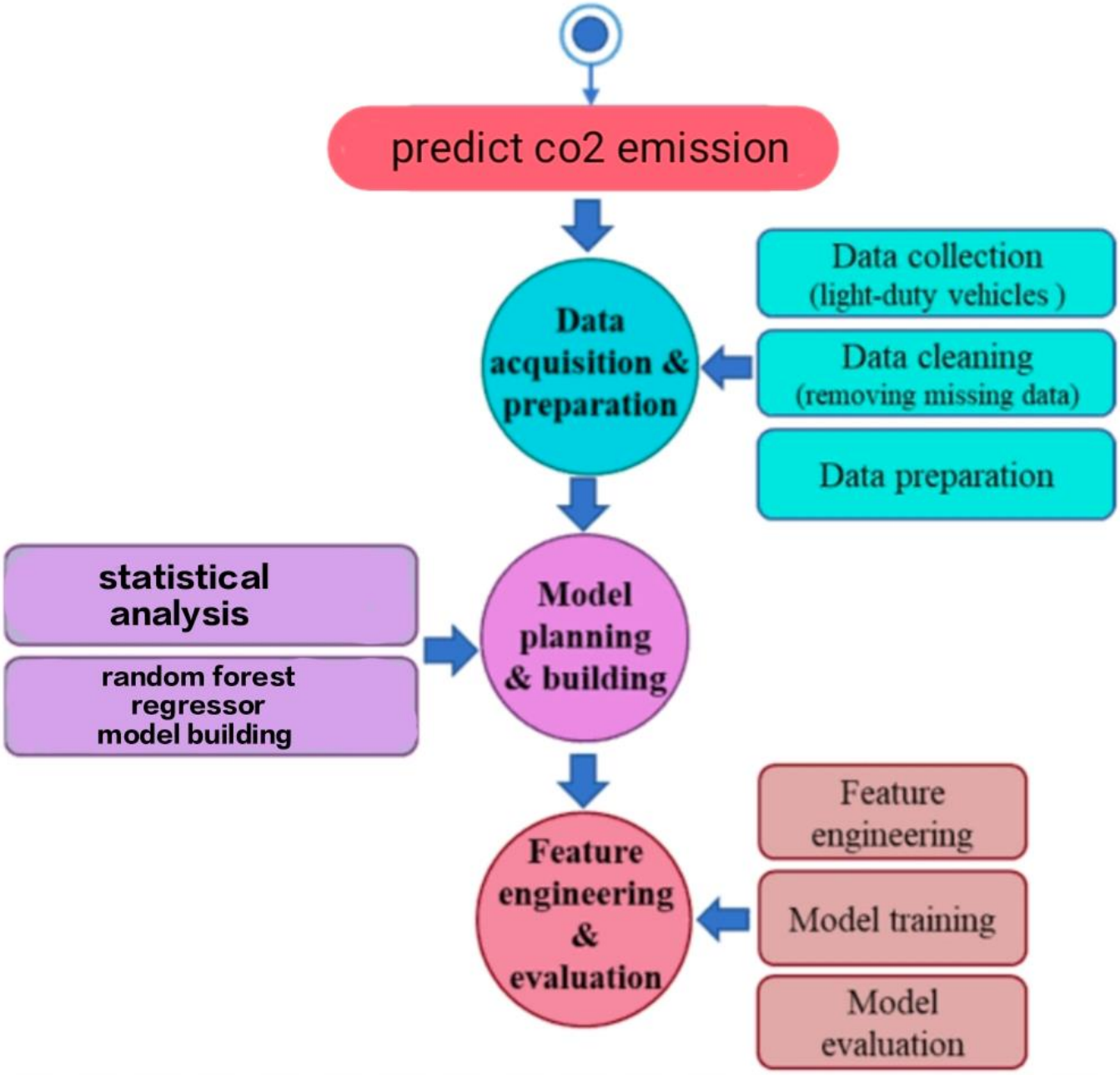
3.indicator name: the name of the indicator being measured (e.g., CO2 emissions).

4.indicator code: the code associated with the indicator.

5.year: the year of the data entry.

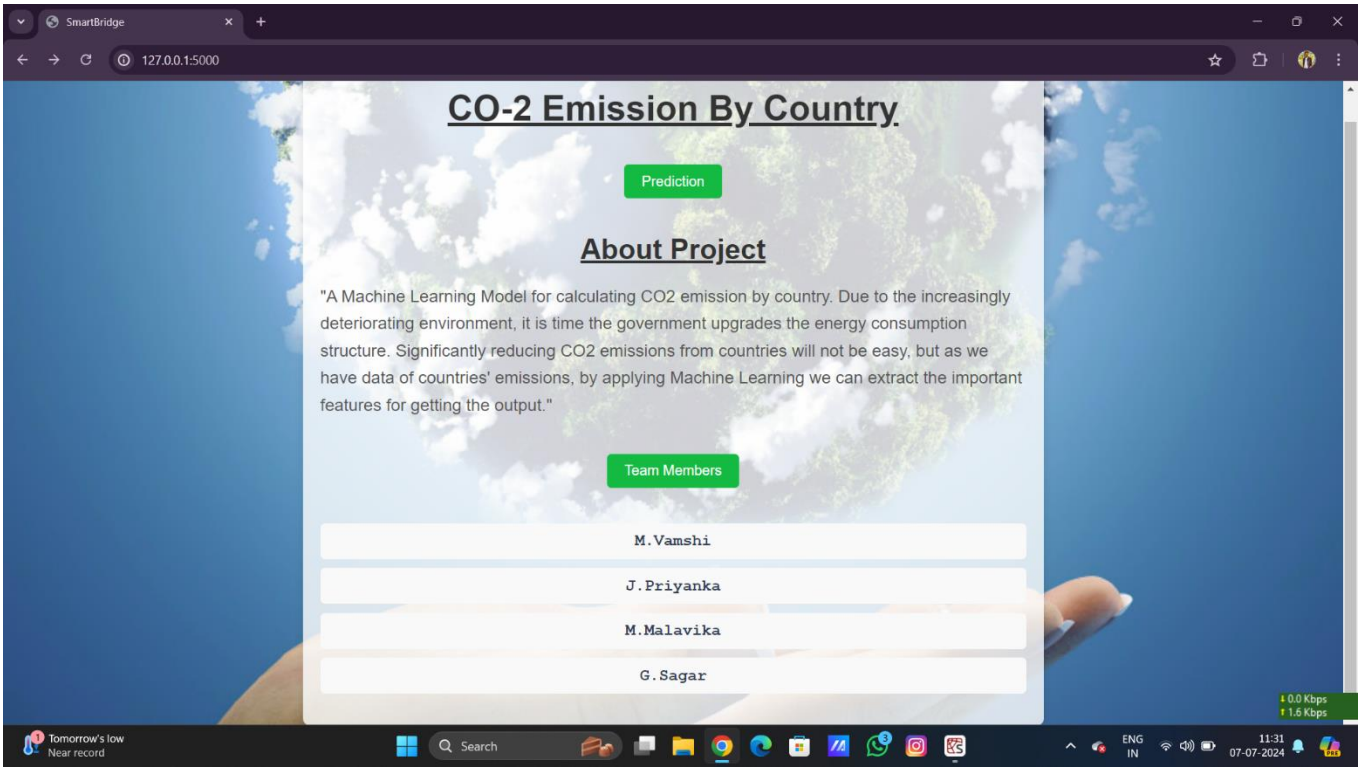
6.value: the value of the indicator for that year.

Flow chart

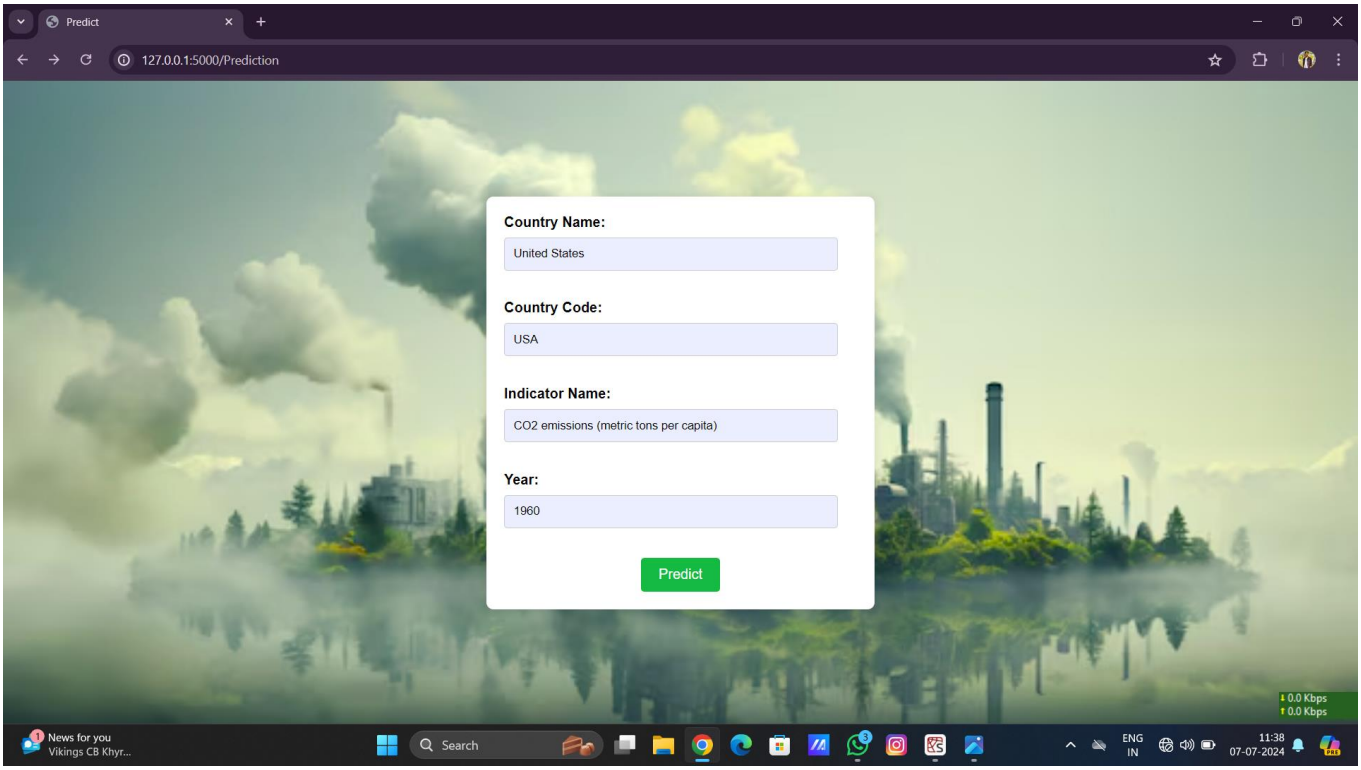

predict co2 emission
Data acquisition & preparation
Data collection (light-duty vehicles)
Data cleaning (removing missing data)
Data preparation
statistical analysis
random forest regressor model building
Model planning & building
Feature engineering & evaluation
Feature engineering
Model training
Model evaluation

6.RESULT

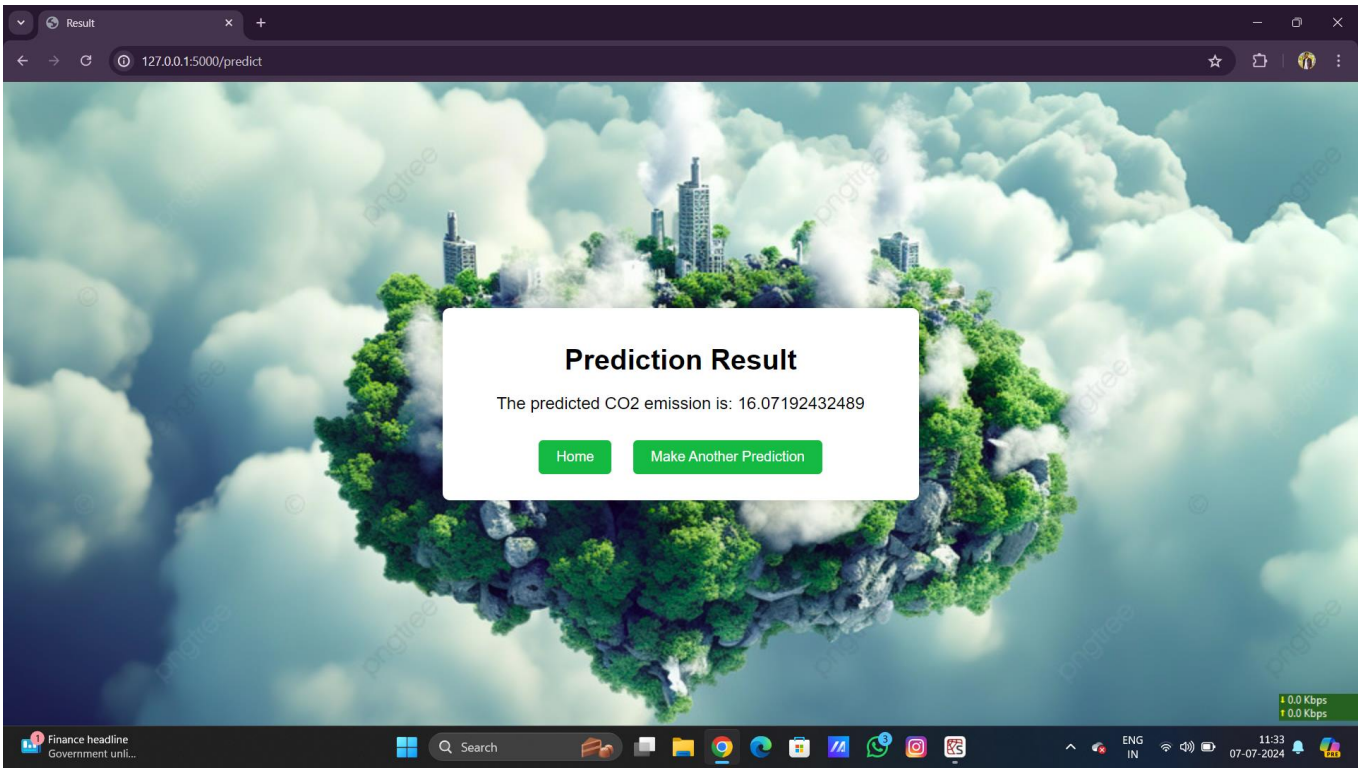
HOME PAGE



Index page


Predict
127.0.0.1:5000/Prediction
Country Name:
United States
Country Code:
USA
Indicator Name:
CO2 emissions (metric tons per capita)
Year:
1960
Predict
0.0 Kbps
0.0 Kbps
News for you
Vikings CB Khyr...
Search
ENG
IN
11:38
07-07-2024

PREDICTIONS



7.ADVANTAGES AND DISADVANTAGES

Advantages:

1. **Timely Insights**: Machine learning models can analyze data quickly and provide timely insights into current CO2 emission trends, enabling rapid responses and interventions by policymakers and stakeholders.
2. **Continuous Monitoring**: Machine learning can facilitate continuous monitoring of CO2 emissions in real-time, allowing for immediate detection of changes or anomalies that may require attention.
3. **Improved Accuracy**: ML algorithms can potentially offer more accurate predictions by identifying complex patterns and relationships in emissions data that may not be evident through traditional methods.
4. **Adaptability**: These models can adapt to new data as it becomes available, improving prediction accuracy over time and adjusting to changes in environmental, economic, or policy factors influencing emissions.
5. **Integration with IoT and Sensor Data**: Machine learning can integrate with IoT devices and sensor networks to gather real-time data on emissions from various sources (e.g., factories, transportation), enhancing the granularity and reliability of predictions.

Disadvantages:

1. **Data Quality Issues**: Real-time prediction accuracy heavily relies on the quality, reliability, and completeness of real-time data inputs. Inconsistent or incomplete data can lead to inaccurate predictions.
2. **Complexity and Interpretability**: Advanced machine learning models, such as deep learning, may lack transparency and interpretability. Understanding the reasoning behind predictions can be challenging, complicating decision-making based on model outputs.
3. **Computational Requirements**: Real-time prediction often requires significant computational resources, including processing power and memory, which may be costly and limit scalability in resource-constrained environments.
4. **Bias and Fairness**: Machine learning models can inherit biases from training data or algorithms, potentially leading to biased predictions that disproportionately impact certain regions, communities, or sectors.

5. **Privacy Concerns**: Real-time data collection and analysis raise privacy concerns, particularly when integrating personal or sensitive data from individuals or organizations contributing to emissions data.

8.APPLICATIONS

1. **Climate Policy and Regulation**: Governments and international bodies can use machine learning models to predict future CO2 emissions based on socio-economic data, energy consumption patterns, and policy changes. This information helps in setting realistic emission reduction targets and designing effective climate policies.
2. **Energy Sector Planning**: Utilities and energy companies use CO2 emission predictions to plan for future energy demand and to optimize their energy generation mix. Machine learning models can forecast emissions associated with different energy sources (e.g., fossil fuels, renewables) under various scenarios, aiding in long-term planning and investment decisions.
3. **Transportation and Urban Planning**: Predicting CO2 emissions from transportation sectors (e.g., road, rail, aviation) helps urban planners and transportation authorities design more sustainable cities and transportation networks. Machine learning can analyze traffic patterns, vehicle types, and infrastructure data to estimate emissions and suggest ways to reduce them.
4. **Corporate Sustainability**: Companies use CO2 emission predictions to assess their carbon footprint and develop sustainability strategies. Machine learning can analyze operational data to identify emission hotspots, optimize supply chains, and support decisions on energy efficiency measures and renewable energy investments.
5. **Financial Risk Assessment**: Financial institutions and investors incorporate CO2 emission predictions into their risk assessment models to evaluate climate-related risks associated with investments. Predicting future emissions helps in assessing the long-term financial viability of projects and businesses, considering potential regulatory changes and market shifts towards sustainability.
6. **Environmental Impact Assessments**: Predictive models are used in environmental impact assessments (EIAs) for infrastructure projects, industrial facilities, and land-use changes. By estimating CO2 emissions, EIAs can evaluate the project's contribution to climate change and identify mitigation measures to minimize environmental impacts.
7. **International Climate Negotiations**: Machine learning predictions support international climate negotiations by providing data-driven insights into countries' emission trajectories and compliance with global climate agreements (e.g., Paris

Agreement). Accurate emission forecasts facilitate negotiations on emission reduction targets and the allocation of international climate finance.

9.CONCLUSION

In conclusion, leveraging machine learning to predict CO2 emissions of countries represents a pivotal advancement in addressing global climate challenges. By harnessing vast datasets and sophisticated models, this approach offers invaluable insights for policymakers to craft effective climate strategies and set ambitious emission reduction targets. Furthermore, it supports industries in optimizing resource use and fostering sustainable practices, while empowering communities with awareness and engagement in climate action. As we continue refining these predictive capabilities and enhancing data accessibility, collaboration across sectors will be essential to translating these insights into meaningful policy interventions and achieving collective global goals for a resilient and low-carbon future.

10.FUTURE SCOPE

Future Scopes predicting co2 emissions of countries

1. **Enhanced Accuracy and Granularity**: Future advancements in machine learning algorithms, coupled with access to high-resolution data (such as satellite imagery, IoT sensor data), will enable more accurate and granular predictions of CO2 emissions at regional and local levels. This will facilitate targeted interventions and policy adjustments tailored to specific geographical and sectoral contexts.
2. **Integration with Climate Models**: Integrating CO2 emission prediction models with climate models can provide a comprehensive understanding of how human activities contribute to climate change. This integration will allow for scenario analysis, projecting the impacts of different emission trajectories on climate variables like temperature, precipitation patterns, and sea level rise.
3. **Real-Time Monitoring and Early Warning Systems**: Advances in real-time data processing and machine learning techniques will enable the development of early warning systems for sudden changes in emissions patterns. This capability can assist in proactive response measures and disaster risk reduction related to climate events and extreme weather phenomena.

4. **Interdisciplinary Applications**: Collaborations between machine learning experts, climate scientists, economists, and policy analysts will foster interdisciplinary research to address complex environmental challenges. This includes exploring the interconnectedness of factors influencing emissions, such as economic trends, technological advancements, and societal behaviors.
5. **Global Collaboration and Data Sharing**: Encouraging international collaboration and data sharing initiatives will be crucial for improving the robustness and inclusiveness of CO2 emission prediction models. This will enhance transparency, accountability, and mutual learning among countries striving towards common climate objectives.

11.BIBILOGRAPHY

[1] Climate Action Tracker. (2021). Global update: When will China peak its CO2 emissions? Retrieved from https://climateactiontracker.org/publications/global-update-when-will-china-peak-its-co2-emissions/

[2] *He, K., & Sun, Y. (2020)*. Machine learning applications in climate science. Nature Climate Change, 10(6), 475-485. doi:10.1038/s41558-020-0788-0

[3] International Energy Agency. (2020). CO2 emissions from fuel combustion: Overview. Paris: IEA Publications.

[4] IPCC. (2018). Global warming of 1.5°C. An IPCC Special Report on the impacts of global warming of 1.5°C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty. Geneva: IPCC.

[5] *Le Quéré, C., Jackson, R. B., Jones, M. W., Smith, A. J., Abernethy, S., Andrew, R. M., ... & Peters, G. P. (2020).* Temporary reduction in daily global CO2 emissions during the COVID-19 forced confinement. Nature Climate Change, 10(7), 647-653. doi:10.1038/s41558-020-0797-z

[6*] Liu, Z., Ciais, P., Deng, Z., Lei, R., Davis, S. J., Feng, S., ... & Zhou, F. (2018).* Near-real-time monitoring of global CO2 emissions reveals the effects of the COVID-19 pandemic. Nature Communications, 11(1), 5172. doi:10.1038/s41467-020-18922-7

[7] *Smith, A. J., Dauwe, S., Davis, S. J., Feng, S., Zheng, B., Keesstra, S., ... & Zeng, Z. (2021).* Machine learning for estimating global CO2 emissions from satellite-observed CO2 concentration data. Earth System Science Data, 13(5), 2089-2113. doi:10.5194/essd-13-2089-2021

[8] *Wang, Z., Chen, Y., & Sun, Y. (2019).* Deep learning for global CO2 concentration estimation from satellite observations. Remote Sensing of Environment, 231, 111220. doi:10.1016/j.rse.2019.111220

[9] World Bank. (2021). World Development Indicators: CO2 emissions (metric tons per capita). Retrieved from https://databank.worldbank.org/source/world-development-indicators

12.APPENDIX

Model building :

1)Dataset
2)jupyter Notebook and Spyder Application Building
1. HTML file (home file,index file,predict file)
1. CSS file
2. Models in pickle format

SOURCE CODE:

FIRST.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>SmartBridge</title>
    <link rel="stylesheet" href="{{url_for('static', filename='styles.css')}}">
    <style>
        body {
            font-family: Arial, sans-serif;
        }

        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
            text-align: center;
        }

        h1, h2 {
            color: #333;
```

```
}

p {
    text-align: left;
    color: #555;
}

button {
    background-color: #4CAF50;
    color: white;
    padding: 10px 20px;
    border: none;
    cursor: pointer;
    font-size: 16px;
    margin-top: 20px;
}

button:hover {
    background-color: #45a049;
}

button:hover:active{
     background-color: deepskyblue;
}

.team-members {
    display: none;
    margin-top: 20px;
}

.team-members ul {
    list-style-type: none;
    padding: 0;
}

.team-members li {
    background-color: #f9f9f9;
```

```
            margin: 10px 0;
            padding: 10px;
            border-radius: 5px;
            font-size: 18px;
            font-family: 'Courier New', Courier, monospace;
            font-weight: bold;
            color: #2c3e50;
        }
    </style>
    <script>
        function toggleTeamMembers() {
            var x = document.getElementById("team-members");
            if (x.style.display === "none") {
                x.style.display = "block";
            } else {
                x.style.display = "none";
            }
        }
    </script>
</head>
<body>
    <div class="container">
        <h1>CO-2 Emission By Country</h1>
        <a href="{{ url_for('prediction') }}"><button>Prediction</button></a>
        <h2>About Project</h2>
        <p>
            "A Machine Learning Model for calculating CO2 emission by country. Due to the
increasingly deteriorating environment,
            it is time the government upgrades the energy consumption structure. Significantly
reducing CO2 emissions from countries
            will not be easy, but as we have data of countries' emissions, by applying Machine
Learning we can extract the important features
            for getting the output."
        </p>
        <button onclick="toggleTeamMembers()">Team Members</button>
        <div id="team-members" class="team-members">
            <ul>
```

```
            <li>M.Vamshi</li>
            <li>J.Priyanka</li>
            <li>M.Malavika</li>
            <li>G.Sagar</li>
        </ul>
    </div>
  </div>
</body>
</html>
```

styles.css

```
/* styles.css */
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  min-height: 100vh;
  background-color: #f0f0f0;
  text-align: center;
  background-image: url("https://www.enigma-is.com/wp-content/uploads/carbon-offsetting-environmental-responsiblity-enigma-industrial-services-climate-care-co2-queens-award.jpg");
  background-size: cover;
  background-position: center;
}

.container {
  background-color: rgba(255, 255, 255, 0.8);
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  max-width: 800px;
```

```
    width: 100%;
    text-align: center;
}

h1, h2, p, form {
    color: #333;
    margin: 20px 0;
}

h1 {
    font-size: 2.5em;
    text-decoration: underline;

}

h2 {
    text-decoration: underline;
    font-size: 2em;
}

p {
    font-size: 1.2em;
    line-height: 1.6;
}

button {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 10px 20px;
    margin: 20px 0;
    border-radius: 5px;
    cursor: pointer;
    font-size: 1em;
    transition: background-color 0.3s ease;
```

```
}

button:hover {
   background-color: #45a049;
}

button:active {
   background-color: #3e8e41;
}

button a {
   color: white;
   text-decoration: none;
}

@media (max-width: 600px) {
   h1 {
      font-size: 2em;
   }

   h2 {
      font-size: 1.5em;
   }

   p {
      font-size: 1em;
   }

   button {
      padding: 8px 16px;
      font-size: 0
```

SECOND.HTML

```
<!DOCTYPE html>
```

```
<html>
<head>
   <title>Predict</title>
   <link rel="stylesheet" href="{{url_for('static', filename='styles2.css')}}">
</head>
<body>
   <form action="/predict" method="post">
      <label for="cname">Country Name:</label>
      <input type="text" id="cname" name="cname" required><br><br>
      <label for="Ccode">Country Code:</label>
      <input type="text" id="Ccode" name="Ccode" required><br><br>
      <label for="Iname">Indicator Name:</label>
      <input type="text" id="Iname" name="Iname" required><br><br>
      <label for="year">Year:</label>
      <input type="text" id="year" name="year" required><br><br>
      <div class="submit-container">
         <input type="submit" href="{{url_for('predict')}}" value="Predict">
      </div>
   </form>
</body>
</html>
```

style2.css

```
/* styles2.css */
body {
   font-family: Arial, sans-serif;
   background-color: #f0f8ff;
   display: flex;
   justify-content: center;
   align-items: center;
   height: 100vh;
   margin: 0;
   background-image: url("https://img.freepik.com/free-photo/factory-producing-co2-
pollution_23-2150858349.jpg");
   background-size: cover;
```

```
}

form {
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    max-width: 400px;
    width: 100%;
}

label {
    display: block;
    font-weight: bold;
    margin-bottom: 8px;
}

input[type="text"] {
    width: calc(100% - 22px); /* Adjusted for padding and border */
    padding: 10px;
    margin-bottom: 15px;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}

.submit-container {
    display: flex;
    justify-content: center;
}

input[type="submit"] {
    background-color: #4CAF50;
```

```
    color: white;
    padding: 10px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 16px;
}

input[type="submit"]:hover {
    background-color: #45a049;
}

input[type="submit"]:hover:active {
    background-color: #3baedf;
}
```

RESULT.HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Result</title>
    <link rel="stylesheet" href="{{url_for('static', filename='styles3.css')}}">
    <style>
        /* styles3.css */

body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
```

```
    min-height: 100vh;
    background-color: #f0f0f0;
    background-image:
url("https://png.pngtree.com/thumb_back/fw800/background/20230720/pngtree-3d-
illustration-of-co2-emissions-from-carbon-dioxide-image_3720214.jpg");
    background-size: cover;
}

div {
    text-align: center;
    background-color: white;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    width: 80%;
    max-width: 500px;
    margin: 20px;
}

h1 {
    font-size: 2em;
    margin-bottom: 20px;
}

p {
    font-size: 1.2em;
    margin-bottom: 20px;
}

button {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 10px 20px;
    margin: 10px;
    border-radius: 5px;
```

```
    cursor: pointer;
    font-size: 1em;
    transition: background-color 0.3s ease;
}
button:hover {
    background-color: #45a049;
}

button a {
    color: white;
    text-decoration: none;
}

@media (max-width: 600px) {
    h1 {
        font-size: 1.5em;
    }

    p {
        font-size: 1em;
    }

    button {
        padding: 8px 16px;
        font-size: 0.9em;
    }
}

    </style>
</head>
<body>
    <div>
    <h1>Prediction Result</h1>
    <p>The predicted CO2 emission is: {{ prediction }}</p>
    <button><a href="{{ url_for('home') }}">Home</a></button>
```

```
    <button><a href="{{ url_for('prediction') }}">Make Another Prediction</a></button>
    </div>
    </body>
</html>
```

APP.PY

```
import pandas as pd

import numpy as np

import pickle

import joblib  # Import joblib for loading LabelEncoder

from flask import Flask, request, render_template


app = Flask(_name_)


# Load the machine learning model

with open('CO2(1).pickle', 'rb') as handle:

    model = pickle.load(handle)


# Load the LabelEncoders (assuming you have separate encoders for each categorical feature)

labelencoder_country = joblib.load('labelencoder_country.joblib')

labelencoder_code = joblib.load('labelencoder_code.joblib')

labelencoder_indicator = joblib.load('labelencoder_indicator.joblib')


@app.route('/')

def home():

    return render_template('first.html')
```

```
@app.route('/Prediction', methods=['POST', 'GET'])
def prediction():
    return render_template('second.html')

@app.route('/Home', methods=['POST', 'GET'])
def my_home():
    return render_template('first.html')

@app.route('/predict', methods=["POST", "GET"])
def predict():
    try:
        # Extract input features from the form
        country_name = request.form['cname']
        country_code = request.form['Ccode']
        indicator_name = request.form['Iname']
        year = float(request.form['year'])  # Convert year to float

        # Transform categorical variables using LabelEncoders
        country_name_encoded = labelencoder_country.transform([country_name])[0]
        country_code_encoded = labelencoder_code.transform([country_code])[0]
        indicator_name_encoded = labelencoder_indicator.transform([indicator_name])[0]

        # Create a DataFrame with the input features
```

```
        features_values = [[country_name_encoded, country_code_encoded, indicator_name_encoded, year]]
        feature_names = ['CountryName', 'CountryCode', 'IndicatorName', 'Year']
        x = pd.DataFrame(features_values, columns=feature_names)

        # Make prediction
        prediction = model.predict(x)
        print("Prediction is:", prediction)

        # Ensure prediction is not empty or None
        if prediction is not None and len(prediction) > 0:
            # Format the prediction value to desired format
            formatted_prediction = "{:.11f}".format(prediction[0])  # Adjust precision as needed

            # Render result template with the formatted prediction
            return render_template("result.html", prediction=formatted_prediction)
        else:
            return "Prediction could not be made."

    except Exception as e:
        print("Error:", str(e))
        return str(e)

if _name_ == "_main_":
```

app.run(debug=True, use_reloader=False)

CODE SNIPPETS

MODEL BUILDING

```
#Importing the required Libraries
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
from collections import Counter as c#return counts
import seaborn as sns #used for data Visualization
from sklearn.model_selection import train_test_split #splits data in random train and test array
from sklearn.metrics import accuracy_score #model performance
import pickle #python object hierarchy is converted into a byte stream
from sklearn.ensemble import RandomForestRegressor #Regression ML algorithm
from sklearn.preprocessing import LabelEncoder #importing the Labelencoding from sklearn
```

READING THE DATASET

```
[ ] #Reading the dataset

    data=pd.read_csv("/content/Indicators.csv")
```

DATASET

```
[ ] data.shape
```

```
(5656458, 6)
```

```
[ ] #Representing first 5 values from the dataset

    data.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
0	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT	1960	1.335609e+02
1	Arab World	ARB	Age dependency ratio (% of working-age populat...	SP.POP.DPND	1960	8.779760e+01
2	Arab World	ARB	Age dependency ratio, old (% of working-age po...	SP.POP.DPND.OL	1960	6.634579e+00
3	Arab World	ARB	Age dependency ratio, young (% of working-age ...	SP.POP.DPND.YG	1960	8.102333e+01
4	Arab World	ARB	Arms exports (SIPRI trend indicator values)	MS.MIL.XPRT.KD	1960	3.000000e+06

```
[ ] #Representing last 5 values from the dataset

    data.tail()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
5656453	Zimbabwe	ZWE	Time required to register property (days)	IC.PRP.DURS	2015	36.0
5656454	Zimbabwe	ZWE	Time required to start a business (days)	IC.REG.DURS	2015	90.0
5656455	Zimbabwe	ZWE	Time to prepare and pay taxes (hours)	IC.TAX.DURS	2015	242.0
5656456	Zimbabwe	ZWE	Time to resolve insolvency (years)	IC.ISV.DURS	2015	3.3
5656457	Zimbabwe	ZWE	Total tax rate (% of commercial profits)	IC.TAX.TOTL.CP.ZS	2015	32.8

CHECK UNIQUE VALUES IN **DATASET**

```
countries=data['CountryName'].unique().tolist()
len(countries)
```

```
247
```

```
#How many unique country code are there ? (should be the same #)


countryCodes=data['CountryCode'].unique().tolist()
len(countryCodes)
```

```
247
```

```
#How many unique Indicators are there ? (should be the same #)

indicators=data['IndicatorName'].unique().tolist()
len(indicators)
```

```
1344
```

```
#How many years of data do we have ?

years=data['Year'].unique().tolist()
len(years)
```

```
56
```

```
print(min(years),"to",max(years))
```

```
1960 to 2015
```

CO-2 EMISSIONS OF THE COUNTRIES

```
# select CO2 emissions for the country Arab

hist_indicator1='CO2 emissions \(metric'
hist_country1= 'ARB'

mask11 = data['IndicatorName'].str.contains(hist_indicator1)
mask22 = data['CountryCode'].str.contains(hist_country1)

stage1 = data[mask11 & mask22]
```

```
stage1.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
8	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	0.643964
23204	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	0.685501
49821	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1962	0.761148
78260	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1963	0.875124
106885	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1964	0.999248

```
#select CO2 emissions for the country Singapore

hist_indicator4='CO2 emissions \(metric'
hist_country4= 'SGP'

mask42 = data['IndicatorName'].str.contains(hist_indicator4)
mask52 = data['CountryCode'].str.contains(hist_country4)

stage4 = data[mask42 & mask52]
stage4.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
18930	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	0.846368
44969	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	1.229944
73101	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1962	1.472918
101695	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1963	1.893765
130742	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1964	2.011115


```
#select CO2 emissions for the country Singapore

hist_indicator4='CO2 emissions \(metric'
hist_country4= 'SGP'

mask42 = data['IndicatorName'].str.contains(hist_indicator4)
mask52 = data['CountryCode'].str.contains(hist_country4)

stage4 = data[mask42 & mask52]
stage4.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
18930	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	0.846368
44969	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	1.229944
73101	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1962	1.472918
101695	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1963	1.893765
130742	Singapore	SGP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1964	2.011115


```
# select CO2 emissions for the country India

hist_indicator2='CO2 emissions \(metric'
hist_country2= 'IND'

mask22 = data['IndicatorName'].str.contains(hist_indicator2)
mask32 = data['CountryCode'].str.contains(hist_country2)

stage2 = data[mask22 & mask32]
```

```
stage2.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
11577	India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	0.268161
36513	India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	0.284292
64049	India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1962	0.306519
92493	India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1963	0.322533
121290	India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1964	0.308900

```
[ ] # select CO2 emissions for the United States

    hist_indicator='CO2 emissions \(metric'
    hist_country= 'USA'

    mask1 = data['IndicatorName'].str.contains(hist_indicator)
    mask2 = data['CountryCode'].str.contains(hist_country)

    stage = data[mask1 & mask2]
```

```
stage.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
22232	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	15.999779
48708	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	15.681256
77087	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1962	16.013937
105704	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1963	16.482762
134742	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1964	16.968119

UNDERSTANDING DATA TYPE AND SUMMARY OF FEATURES

```
#info will give summary of dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5656458 entries, 0 to 5656457
Data columns (total 6 columns):
 #   Column         Dtype
---  ------         -----
 0   CountryName    object
 1   CountryCode    object
 2   IndicatorName  object
 3   IndicatorCode  object
 4   Year           int64
 5   Value          float64
dtypes: float64(1), int64(1), object(4)
memory usage: 258.9+ MB
```

```
[ ] #returns important values for continuous column data
    data.describe()
```

	Year	Value
count	5.656458e+06	5.656458e+06
mean	1.994464e+03	1.070501e+12
std	1.387895e+01	4.842469e+13
min	1.960000e+03	-9.824821e+15
25%	1.984000e+03	5.566242e+00
50%	1.997000e+03	6.357450e+01
75%	2.006000e+03	1.346722e+07
max	2.015000e+03	1.103367e+16

OBSERVING TARGET, NUMERICAL AND CATEGORICAL COLUMNS


```
[ ] np.unique(data.dtypes,return_counts=True)
```

```
(array([dtype('int64'), dtype('float64'), dtype('O')], dtype=object),
 array([1, 1, 4]))
```

CATEGORICAL COLUMNS

```
[ ] cat=data.dtypes[data.dtypes!='0'].index.values
    cat
```

```
array(['CountryName', 'CountryCode', 'IndicatorName', 'IndicatorCode',
       'Year', 'Value'], dtype=object)
```

NUMERICAL COLUMNS

```
data.select_dtypes(include='number')
```

	Year	Value
0	1960	1.335609e+02
1	1960	8.779760e+01
2	1960	6.634579e+00
3	1960	8.102333e+01
4	1960	3.000000e+06
...	...	...
5656453	2015	3.600000e+01
5656454	2015	9.000000e+01
5656455	2015	2.420000e+02
5656456	2015	3.300000e+00
5656457	2015	3.280000e+01

5656458 rows × 2 columns

```
data.select_dtypes(include='object')
```

	CountryName	CountryCode	IndicatorName	IndicatorCode
0	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT
1	Arab World	ARB	Age dependency ratio (% of working-age populat...	SP.POP.DPND
2	Arab World	ARB	Age dependency ratio, old (% of working-age po...	SP.POP.DPND.OL
3	Arab World	ARB	Age dependency ratio, young (% of working-age ...	SP.POP.DPND.YG
4	Arab World	ARB	Arms exports (SIPRI trend indicator values)	MS.MIL.XPRT.KD
...	...	...	...	...
5656453	Zimbabwe	ZWE	Time required to register property (days)	IC.PRP.DURS
5656454	Zimbabwe	ZWE	Time required to start a business (days)	IC.REG.DURS
5656455	Zimbabwe	ZWE	Time to prepare and pay taxes (hours)	IC.TAX.DURS
5656456	Zimbabwe	ZWE	Time to resolve insolvency (years)	IC.ISV.DURS
5656457	Zimbabwe	ZWE	Total tax rate (% of commercial profits)	IC.TAX.TOTL.CP.ZS

5656458 rows × 4 columns

HANDLING MISSING DATA

```
[ ] #Returns true if any columns having null values

    data.isnull().any()
```

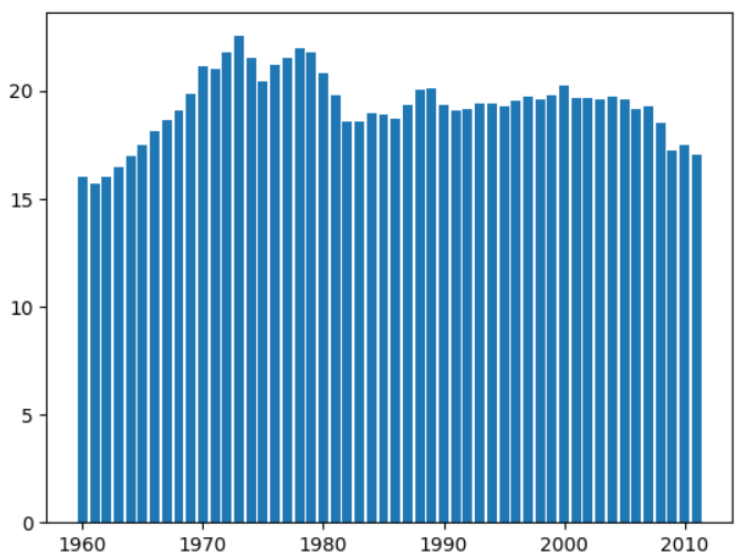
```
CountryName      False
CountryCode      False
IndicatorName    False
IndicatorCode    False
Year             False
Value            False
dtype: bool
```

```
    #Used for finding the null values

    data.isnull().sum()
```

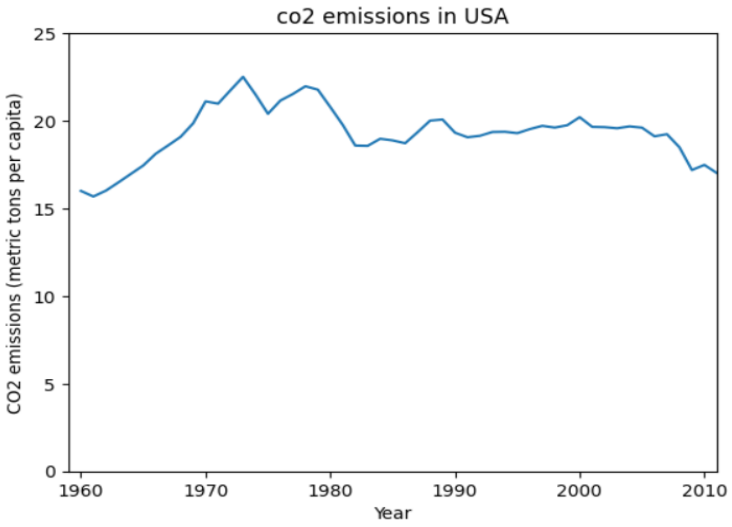
```
CountryName      0
CountryCode      0
IndicatorName    0
IndicatorCode    0
Year             0
Value            0
dtype: int64
```

DATA VISUALIZATION

```
[ ] #get the yearS
    years=stage['Year'].values

    #get the values
    co2=stage['Value'].values

    #create
    plt.bar(years,co2)
    plt.show()
```

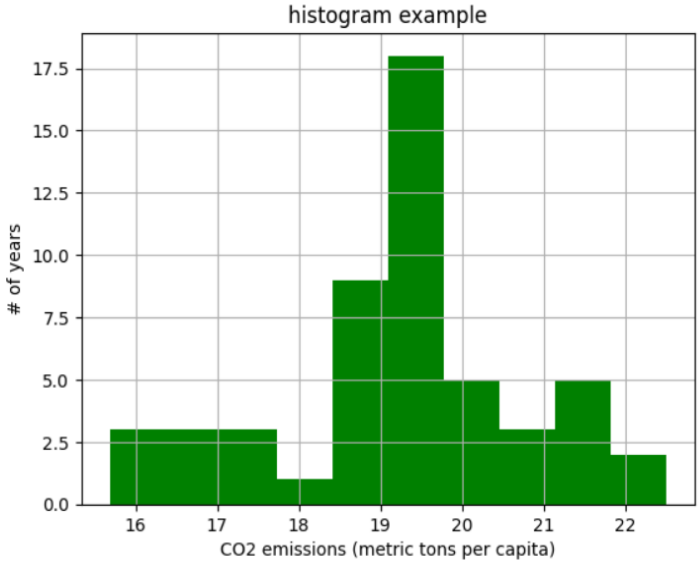


```
#switch to a line plot
plt.plot(stage['Year'].values,stage['Value'].values)
#label the axes
plt.xlabel('Year')
plt.ylabel(stage['IndicatorName'].iloc[0])
#label the figure
plt.title('co2 emissions in USA')
#to make more honest,start the y axis at 0
plt.axis([1959,2011,0,25])
plt.show()
```




```
#the histogram of the data
hist_data=stage['Value'].values
print(len(hist_data))
plt.hist(hist_data,10,density=False,facecolor='green')

plt.xlabel(stage['IndicatorName'].iloc[0])
plt.ylabel('# of years')
plt.title('histogram example')
plt.grid(True)
plt.show()
```

```
52
```



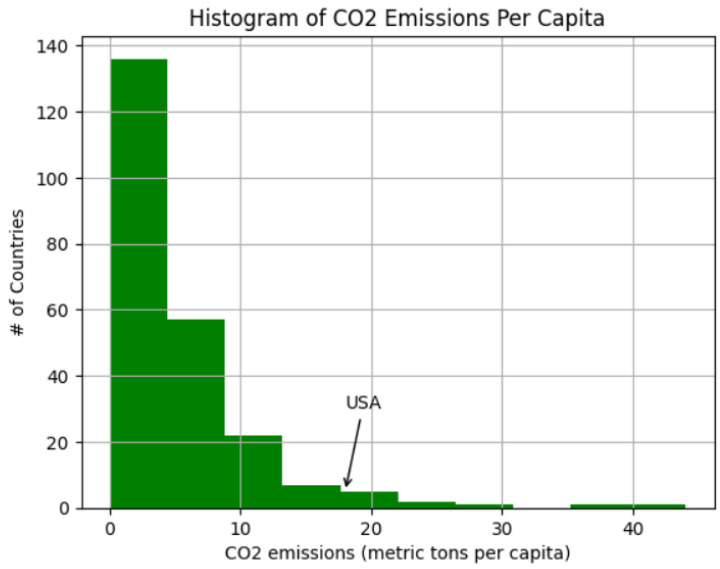
```
hist_indicator='CO2 emissions \(metric'
hist_year=2011
# Now try the original code again
mask1 = (data['IndicatorName'].str.contains(hist_indicator))
mask2 = (data['Year'].isin([hist_year]))
co2_2011 = data[mask1 & mask2]
co2_2011.head()
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
5026275	Arab World	ARB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2011	4.724500
5026788	Caribbean small states	CSS	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2011	9.692960
5027295	Central Europe and the Baltics	CEB	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2011	6.911131
5027870	East Asia & Pacific (all income levels)	EAS	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2011	5.859548
5028456	East Asia & Pacific (developing only)	EAP	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2011	5.302499

```
[ ] print(len(co2_2011))
```

```
232
```

```
[ ] # let's plot a histogram of the emmissions per capita by country

    # subplots returns a touple with the figure, axis attributes.
    fig, ax = plt.subplots()

    ax.annotate("USA",
                xy=(18, 5), xycoords='data',
                xytext=(18, 30), textcoords='data',
                arrowprops=dict(arrowstyle="->",
                                connectionstyle="arc3"),
                )

    plt.hist(co2_2011['Value'], 10, density=False, facecolor='green')

    plt.xlabel(stage['IndicatorName'].iloc[0])
    plt.ylabel('# of Countries')
    plt.title('Histogram of CO2 Emissions Per Capita')

    plt.grid(True)
    plt.show()
```



RELATION BETWEEN GDP AND CO2 EMISSIONS IN USA

```
# select GDP Per capita emissions for the United States
hist_indicator = 'GDP per capita \(constant 2005'
hist_country = 'USA'

mask1 = data['IndicatorName'].str.contains(hist_indicator)
mask2 = data['CountryCode'].str.contains(hist_country)

# stage is just those indicators matching the USA for country code and CO2 emissions over time.
gdp_stage = data[mask1 & mask2]

gdp_stage.head(2)
```

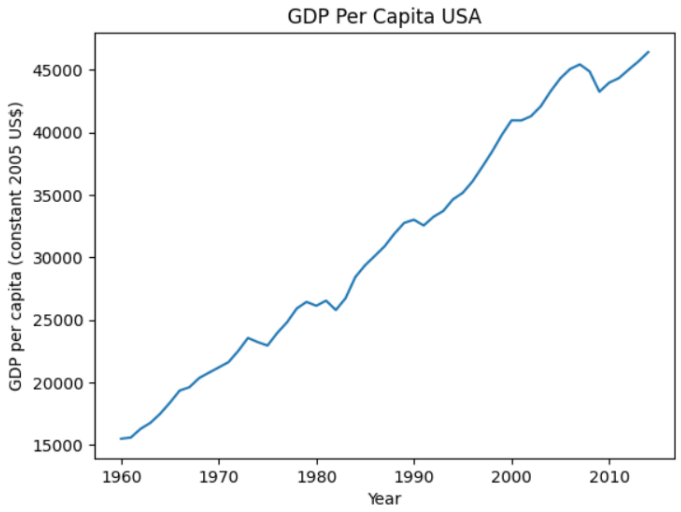
	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
22282	United States	USA	GDP per capita (constant 2005 US$)	NY.GDP.PCAP.KD	1960	15482.707760
48759	United States	USA	GDP per capita (constant 2005 US$)	NY.GDP.PCAP.KD	1961	15578.409657

```
[ ] stage.head(2)
```

	CountryName	CountryCode	IndicatorName	IndicatorCode	Year	Value
22232	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1960	15.999779
48708	United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1961	15.681256

```
# switch to a line plot
plt.plot(gdp_stage['Year'].values, gdp_stage['Value'].values)
```

```
# Label the axes
plt.xlabel('Year')
plt.ylabel(gdp_stage['IndicatorName'].iloc[0])

#label the figure
plt.title('GDP Per Capita USA')

# to make more honest, start they y axis at 0
#plt.axis([1959, 2011,0,25])
plt.show()
```



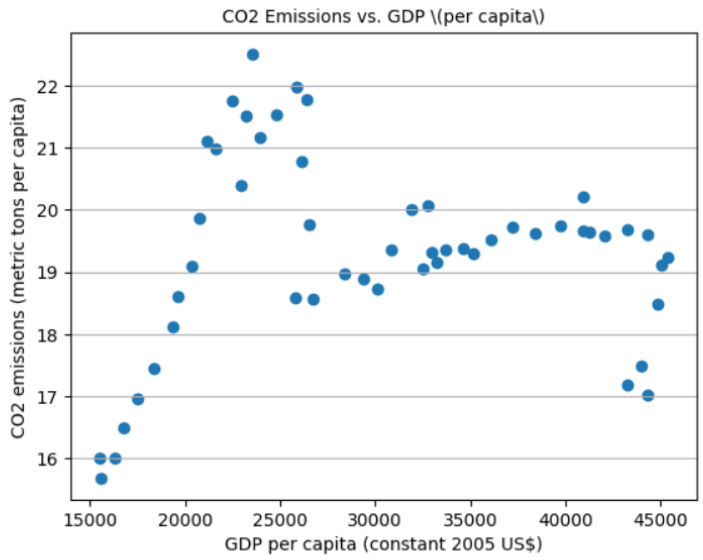
```
[ ] print("GDP Min Year = ", gdp_stage['Year'].min(), "max: ", gdp_stage['Year'].max())
    print("CO2 Min Year = ", stage['Year'].min(), "max: ", stage['Year'].max())
```

```
GDP Min Year =  1960 max:  2014
CO2 Min Year =  1960 max:  2011
```

```
[ ] gdp_stage_trunc = gdp_stage[gdp_stage['Year'] < 2012]
    print(len(gdp_stage_trunc))
    print(len(stage))
```

```
52
52
```

```
%matplotlib inline
import matplotlib.pyplot as plt

fig, axis = plt.subplots()
# Grid lines, Xticks, Xlabel, Ylabel

axis.yaxis.grid(True)
axis.set_title('CO2 Emissions vs. GDP \(per capita\)',fontsize=10)
axis.set_xlabel(gdp_stage_trunc['IndicatorName'].iloc[10],fontsize=10)
axis.set_ylabel(stage['IndicatorName'].iloc[0],fontsize=10)

X = gdp_stage_trunc['Value']
Y = stage['Value']

axis.scatter(X, Y)
plt.show()
```



CORRELATION BETWEEN THE INDEPENDENT COLUMNS

```
[ ] np.corrcoef(gdp_stage_trunc['Value'],stage['Value'])
```

```
array([[1.        , 0.07676005],
       [0.07676005, 1.        ]])
```

```
categorical_cols=pd.DataFrame(data.select_dtypes(include='object'))
categorical_cols
```

	CountryName	CountryCode	IndicatorName	IndicatorCode
0	Arab World	ARB	Adolescent fertility rate (births per 1,000 wo...	SP.ADO.TFRT
1	Arab World	ARB	Age dependency ratio (% of working-age populat...	SP.POP.DPND
2	Arab World	ARB	Age dependency ratio, old (% of working-age po...	SP.POP.DPND.OL
3	Arab World	ARB	Age dependency ratio, young (% of working-age ...	SP.POP.DPND.YG
4	Arab World	ARB	Arms exports (SIPRI trend indicator values)	MS.MIL.XPRT.KD
...	...	...	...	...
5656453	Zimbabwe	ZWE	Time required to register property (days)	IC.PRP.DURS
5656454	Zimbabwe	ZWE	Time required to start a business (days)	IC.REG.DURS
5656455	Zimbabwe	ZWE	Time to prepare and pay taxes (hours)	IC.TAX.DURS
5656456	Zimbabwe	ZWE	Time to resolve insolvency (years)	IC.ISV.DURS
5656457	Zimbabwe	ZWE	Total tax rate (% of commercial profits)	IC.TAX.TOTL.CP.ZS

5656458 rows × 4 columns

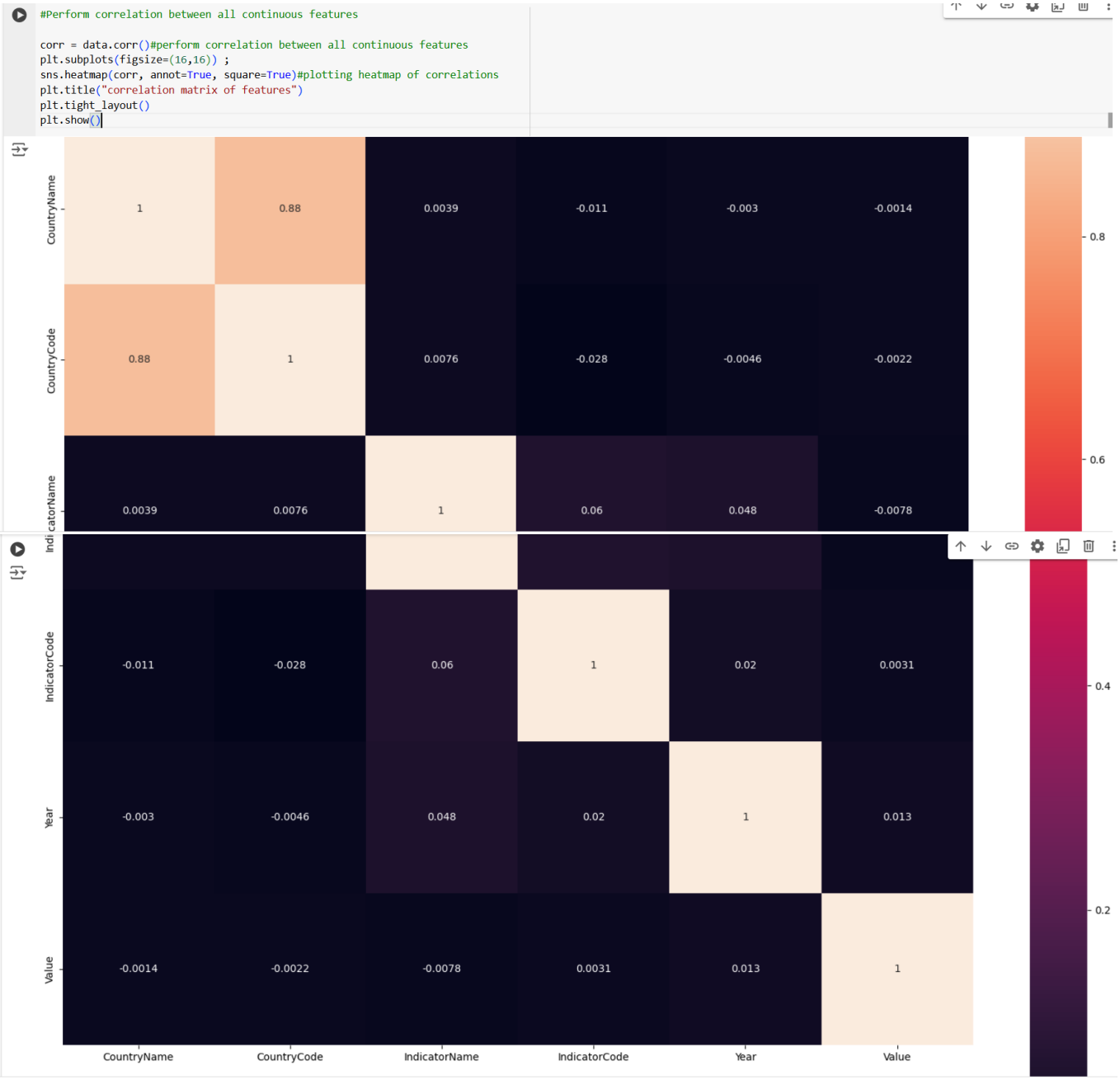
LABEL ENCODING

```
[ ]  #Label coding

     for col in categorical_cols:
         print(f"LABEL ENCODING OF: {col}")
         le = LabelEncoder()
         print("Before Encoding:", data[col])
         data[col] = le.fit_transform(data[col])
         print("After Encoding:", data[col])
         print("*" * 100)

     # Display final dataset
     print("Final Dataset:\n", data)
```

```
LABEL ENCODING OF: CountryName
Before Encoding: 0          Arab World
1          Arab World
2          Arab World
3          Arab World
4          Arab World
              ...
5656453      Zimbabwe
5656454      Zimbabwe
5656455      Zimbabwe
5656456      Zimbabwe
5656457      Zimbabwe
Name: CountryName, Length: 5656458, dtype: object
After Encoding: 0           7
1           7
2           7
3           7
4           7
          ...
5656453    246
5656454    246
```

```
5656453            Time required to register property (days)
5656454             Time required to start a business (days)
5656455                Time to prepare and pay taxes (hours)
5656456                   Time to resolve insolvency (years)
5656457             Total tax rate (% of commercial profits)
Name: IndicatorName, Length: 5656458, dtype: object
After Encoding: 0           44
1           48
2           49
3           50
4           90
          ...
5656453    1258
5656454    1259
5656455    1263
5656456    1264
5656457    1274
Name: IndicatorName, Length: 5656458, dtype: int64
**********************************************************************************************
LABEL ENCODING OF: IndicatorCode
Before Encoding: 0               SP.ADO.TFRT
1               SP.POP.DPND
2            SP.POP.DPND.OL
3            SP.POP.DPND.YG
4            MS.MIL.XPRT.KD
                ...
5656453         IC.PRP.DURS
5656454         IC.REG.DURS
5656455         IC.TAX.DURS
5656456         IC.ISV.DURS
5656457    IC.TAX.TOTL.CP.ZS
Name: IndicatorCode, Length: 5656458, dtype: object
After Encoding: 0          1195
1          1218
5656457    246
Name: CountryName, Length: 5656458, dtype: int64
**********************************************************************************************
LABEL ENCODING OF: CountryCode
Before Encoding: 0          ARB
1          ARB
2          ARB
3          ARB
4          ARB
          ...
5656453    ZWE
5656454    ZWE
5656455    ZWE
5656456    ZWE
5656457    ZWE
Name: CountryCode, Length: 5656458, dtype: object
After Encoding: 0            5
1            5
2            5
3            5
4            5
          ...
5656453    246
5656454    246
5656455    246
5656456    246
5656457    246
Name: CountryCode, Length: 5656458, dtype: int64
**********************************************************************************************
LABEL ENCODING OF: IndicatorName
Before Encoding: 0          Adolescent fertility rate (births per 1,000 wo...
1          Age dependency ratio (% of working-age populat...
2          Age dependency ratio, old (% of working-age po...
3          Age dependency ratio, young (% of working-age ...
4                Arms exports (SIPRI trend indicator values)
```

```
#Perform correlation between all continuous features

corr = data.corr()#perform correlation between all continuous features
plt.subplots(figsize=(16,16)) ;
sns.heatmap(corr, annot=True, square=True)#plotting heatmap of correlations
plt.title("correlation matrix of features")
plt.tight_layout()
plt.show()
```




```
#Creating the Dependent and Independent Variable

x=data.drop(['Value','IndicatorCode'],axis=1)
x=pd.DataFrame(x)
y=data['Value'] #Dependent feature
y=pd.DataFrame(y)
```

SPLITTING DATA

```
[ ] #Splitting dataset into train and test

    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
    print(x_train.shape)
    print(x_test.shape)
    print(y_train.shape)
    print(y_test.shape)
```

```
(4525166, 4)
(1131292, 4)
(4525166, 1)
(1131292, 1)
```

TRAINING THE MODEL

```
#Training the model

from sklearn.ensemble import RandomForestRegressor
rand = RandomForestRegressor(n_estimators=10,random_state=52,n_jobs=-1)
rand.fit(x_train,y_train)
```

```
<ipython-input-53-6c838af2cded>:5: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example us
  rand.fit(x_train,y_train)
RandomForestRegressor(n_estimators=10, n_jobs=-1, random_state=52)
```

```
[ ] ypred = rand.predict(x_test)
    print(ypred)
```

```
[2.23526022e+00 7.92900024e+01 4.63113569e+01 ... 9.33333333e+00
 3.45749686e+01 6.00578821e+09]
```

```
#Accuracy score To check how well our model is performing on the test data
rand.score(x_train,y_train)
```

```
0.9829119449040941
```

```
[ ] x_train
```

	CountryName	CountryCode	IndicatorName	Year
1937930	100	99	771	1990
2056226	102	101	104	1991
4291514	232	6	832	2006
1272651	211	202	1036	1983
4348108	34	40	382	2007
...	...	...	...	...
5030793	70	60	665	2011
491263	217	214	443	1972
3937352	237	233	750	2004
4686059	6	10	489	2009
4322341	0	2	263	2007

4525166 rows × 4 columns

```
[ ] x_pred=['7','5','44','1961']
```

```
[ ] rand.predict([x_pred])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
array([134.18303659])
```

SAVING THE MODEL

```
[ ] #Saving our model by importing pickle file

    import pickle
    pickle.dump(rand, open('CO2.pickle', 'wb'))
```