**Normal Delivery vs C-Section Delivery:  
A Geographical and Parity-Based Analysis**

**Index:**

1. Problem Statement (Abstract)
2. Description (Detailed)
3. Data Overview
4. Project Plan
5. Design (Visual Diagrams)
6. Implementation
7. Code & Explanation
8. Output (Screenshots)
9. Closure (Conclusion)
10. Bibliography

**Problem Statement:**

The rising global rate of caesarean (C-section) deliveries has sparked concerns regarding maternal and neonatal outcomes, healthcare costs, and resource allocation. Despite medical guidelines promoting vaginal births, especially for women with previous caesareans (VBAC), there exists significant variation in C-section rates based on geography and parity. This project aims to analyze and visualize the disparities in normal versus C-section delivery rates across different regions and among women with varying birth histories to identify patterns, potential causes, and policy implications.

**Project Description:**

This project analyzes the distribution and frequency of normal (vaginal) vs. C-section deliveries, using two key lenses:

1. **Geographical Rate Comparison (Choropleth Maps):**  
   Visualize and compare regional differences in C-section and normal delivery rates using choropleth maps. This will help identify areas with unusually high or low C-section rates, potentially indicating systemic healthcare issues, cultural preferences, or policy-driven practices.
2. **Parity-Based Comparison (VBAC Focus):**  
   Investigate how parity—particularly in women with a previous C-section—impacts the likelihood of undergoing a repeat C-section vs. attempting a Vaginal Birth After Caesarean (VBAC). This section will explore VBAC success rates, and regional trends influencing delivery decisions.

Through these analyses, the project aims to uncover hidden trends and guide better decision-making in maternal healthcare policies, patient counselling, and resource planning.

**Data Overview:**

**Dataset Includes:**

* State
* Year
* Total Births
* Caesarean Births
* VBAC Attempts
* VBAC Successes
* Primiparous Births
* Multiparous Births
* Previous C-Section Mothers
* Average Maternal Age

**Output Goals:**

* Choropleth maps showing C-section rate by region
* Bar graphs comparing VBAC success rates across parity levels and regions
* Identify regions with high/low VBAC uptake
* Line graph comparing the yearly trends of caesarean vs VBAC success

**Purpose & Outcome:**

* Discover geographic delivery trends
* Identify opportunities to improve VBAC awareness
* Suggest where C-section rates may be unnecessarily high
* Aid in maternal healthcare planning and policymaking

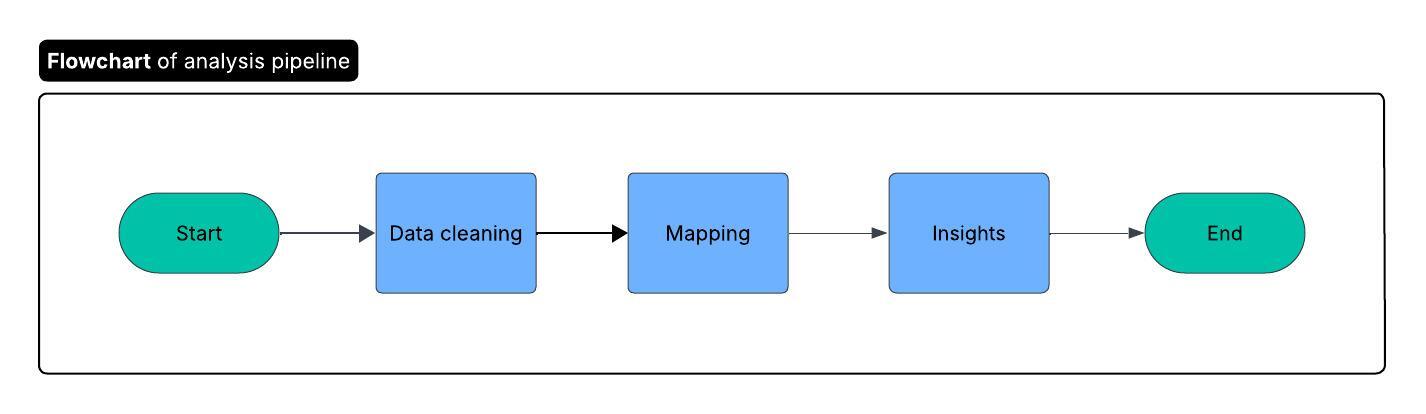
**Benefits:**

* Insights for healthcare providers and policy-makers
* Promotes patient-centric, informed birth choices
* Supports sustainable maternal healthcare systems

**Project Plan:**

|  |  |  |
| --- | --- | --- |
| **Phase** | **Task** | **Language and Tools Used** |
| 1 | Data Cleaning | Python, Pandas, NumPy |
| 2 | EDA (Exploratory Data Analysis) | Seaborn, Matplotlib |
| 3 | Geographical Mapping | Plotly |
| 4 | VBAC Analysis | Pandas |
| 5 | Output Visualization | Dashboards, Plots |
| 6 | Report Generation | Jupyter Notebook / PDF |

**Design:**

****

**Implementation:**

Write your analysis steps:

1. Load and clean the dataset
2. Create a normalized rate of C-sections and normal deliveries per region
3. Map regions using Plotly (choropleth map)
4. Filter data based on parity = 1 or more, and analyze VBAC trends
5. Calculate success rates of VBAC per region/parity group
6. Yearly trends: C-section and VBAC rates
7. Generate visuals to summarize key findings

**Code and Explanation:**

import pandas as pd

import plotly.express as px

import json

class IndiaDeliveryStatsVisualizer:

    # 3. Map regions using Plotly (choropleth map)

    def \_\_init\_\_(self, csv\_path, geojson\_path):

        self.csv\_path = csv\_path

        self.geojson\_path = geojson\_path # GeoJSON file is a format for encoding geographical data structures using JSON

        self.df = None

        self.geojson = None

    def load\_data(self):

        self.df = pd.read\_csv(self.csv\_path)

        self.df.columns = self.df.columns.str.strip()

        self.\_normalize\_state\_names()

    def load\_geojson(self):

        with open(self.geojson\_path) as f:

            self.geojson = json.load(f)

    def \_normalize\_state\_names(self) # Map dataset state names to GeoJSON-compatible names

        state\_map = {

            'Andaman & Nicobar Islands': 'Andaman and Nicobar',

            'Delhi': 'Delhi',

            'Odisha': 'Orissa',

            'Uttarakhand': 'Uttaranchal',

            'Pondicherry': 'Puducherry',

        }

        self.df['State'] = self.df['State'].replace(state\_map)

    def calculate\_metrics(self): # Add C-section rate column

        self.df['CSection\_Rate'] = (self.df['Cesarean\_Births'] / self.df['Total\_Births']) \* 100

    def plot\_choropleth(self, metric='CSection\_Rate', title='C-section Rate per State in India'):

        if self.df is None or self.geojson is None:

            raise ValueError("Data or GeoJSON not loaded.")

        fig = px.choropleth(

            self.df,

            geojson=self.geojson,

            featureidkey="properties.NAME\_1", # NAME\_1 is the featureidkey that we found out

            locations="State",

            color=metric,

            color\_continuous\_scale="Reds",

            title=title

        )

        fig.update\_geos(fitbounds="locations", visible=False)

        fig.show()

    # 4. Filter data based on parity = 1 or more, and analyze VBAC trends

    def filter\_multiparous(self): #Filter dataset to only include women with parity ≥ 1 (multiparous - 2nd birth or more).

        self.df = self.df[self.df['Multiparous\_Births'] > 0]

    def analyze\_vbac(self):

        self.df['VBAC\_Attempt\_Rate'] = (self.df['VBAC\_Attempts'] / self.df['Prev\_CSection\_Mothers'].replace(0, pd.NA)) \* 100

        self.df['VBAC\_Success\_Rate'] = (self.df['VBAC\_Successes'] / self.df['VBAC\_Attempts'].replace(0, pd.NA)) \* 100

    def plot\_vbac\_success\_rate(self):

        #Plot VBAC Success Rate by state (only for parity ≥ 1 data)

        fig = px.choropleth(

            self.df,

            geojson=self.geojson,

            featureidkey="properties.NAME\_1", # NAME\_1 is the featureidkey that we found out

            locations="State",

            color="VBAC\_Success\_Rate",

            color\_continuous\_scale="Greens",

            title="VBAC Success Rate by State (Parity ≥ 1)"

        )

        fig.update\_geos(fitbounds="locations", visible=False)

        fig.show()

    # 5. Calculate success rates of VBAC per region/parity group

    def calculate\_vbac\_success\_by\_parity\_group(self):

        """

        Group data by State and parity group (Primiparous / Multiparous), and calculate VBAC success rates.

        Returns a new DataFrame.

        """

        data = []

        for \_, row in self.df.iterrows():

            state = row['State']

            if row['Multiparous\_Births'] > 0:

                data.append({

                    'State': state,

                    'Parity\_Group': 'Multiparous',

                    'VBAC\_Attempts': row['VBAC\_Attempts'],

                    'VBAC\_Successes': row['VBAC\_Successes']

                })

        parity\_df = pd.DataFrame(data)

        result = (

            parity\_df.groupby(['State', 'Parity\_Group'])

            .sum()

            .reset\_index()

        )

        result['VBAC\_Success\_Rate'] = (result['VBAC\_Successes'] / result['VBAC\_Attempts'].replace(0, pd.NA)) \* 100

        return result

    def plot\_vbac\_by\_parity\_group(self, vbac\_summary\_df): # Bar plot of VBAC Success Rate by State and Parity Group

        fig = px.bar(

            vbac\_summary\_df,

            x='State',

            y='VBAC\_Success\_Rate',

            color='Parity\_Group',

            barmode='group',

            title='VBAC Success Rate by State and Parity Group',

            text\_auto='.2f'

        )

        fig.update\_layout(xaxis\_tickangle=-45)

        fig.show()

if \_\_name\_\_ == "\_\_main\_\_":

    visualizer = IndiaDeliveryStatsVisualizer(

        csv\_path="delivery.csv",

        geojson\_path="india\_state.geojson"

    )

    visualizer.load\_data()

    visualizer.load\_geojson()

    visualizer.calculate\_metrics()

    visualizer.plot\_choropleth()

if \_\_name\_\_ == "\_\_main\_\_":

    visualizer = IndiaDeliveryStatsVisualizer(

        csv\_path="delivery.csv",

        geojson\_path="india\_state.geojson"

    )

    visualizer.load\_data()

    visualizer.load\_geojson()

    visualizer.filter\_multiparous()

    visualizer.analyze\_vbac()

    visualizer.plot\_vbac\_success\_rate()

if \_\_name\_\_ == "\_\_main\_\_":

    visualizer = IndiaDeliveryStatsVisualizer(

        csv\_path="delivery.csv",

        geojson\_path="india\_state.geojson"

    )

    visualizer.load\_data()

    visualizer.load\_geojson()

    visualizer.analyze\_vbac()

    vbac\_summary = visualizer.calculate\_vbac\_success\_by\_parity\_group()

    visualizer.plot\_vbac\_by\_parity\_group(vbac\_summary)

import pandas as pd

import plotly.express as px

class DeliveryAnalysis:

    def \_\_init\_\_(self, file\_path):

        self.df = pd.read\_csv(file\_path)

        self.clean\_data()

    def clean\_data(self):

        self.df.dropna(inplace=True)

        self.df['Normal\_Births'] = self.df['Total\_Births'] - self.df['Cesarean\_Births']

        self.df['CSection\_Rate'] = (self.df['Cesarean\_Births'] / self.df['Total\_Births']) \* 100

        self.df['Normal\_Rate'] = (self.df['Normal\_Births'] / self.df['Total\_Births']) \* 100

        self.df['VBAC\_Attempt\_Rate'] = (self.df['VBAC\_Attempts'] / self.df['Prev\_CSection\_Mothers']) \* 100

        self.df['VBAC\_Success\_Rate'] = (self.df['VBAC\_Successes'] / self.df['VBAC\_Attempts']) \* 100

    def plot\_yearly\_trends(self):

        # Group by year and get average rates

        yearly = self.df.groupby('Year')[['CSection\_Rate', 'VBAC\_Success\_Rate']].mean().reset\_index()

        # Plot using Plotly

        fig = px.line(

            yearly,

            x='Year',

            y=['CSection\_Rate', 'VBAC\_Success\_Rate'],

            labels={'value': 'Rate (%)', 'variable': 'Type'},

            title='Yearly Trends: C-Section Rate vs VBAC Success Rate'

        )

        fig.update\_traces(mode='lines+markers')

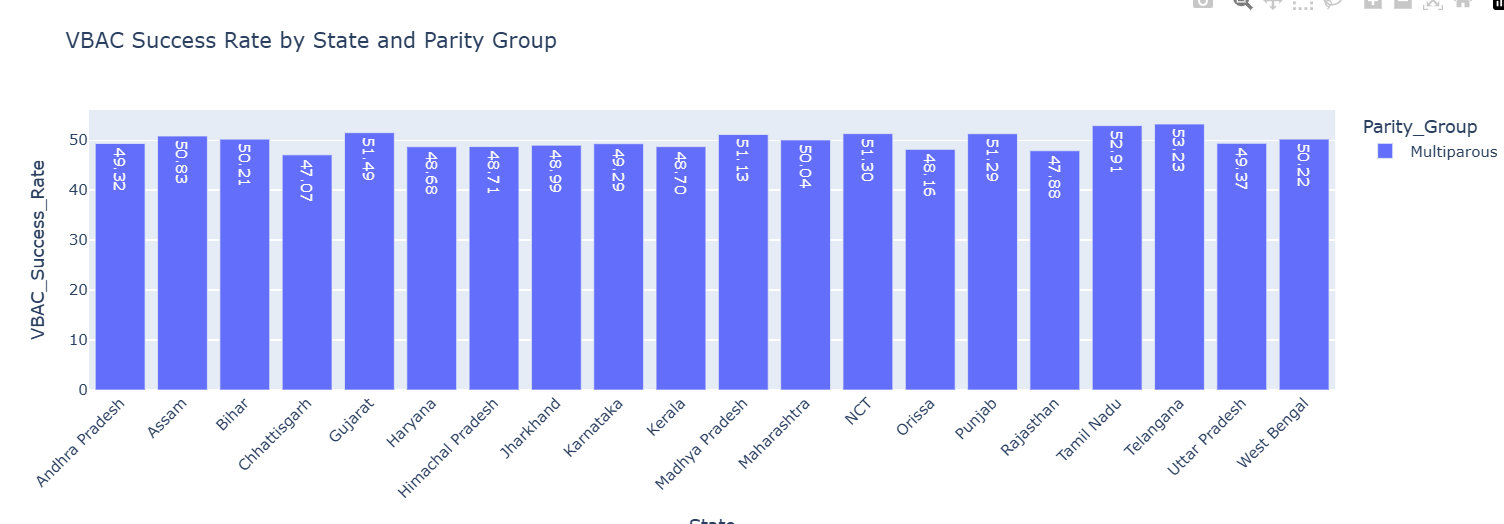
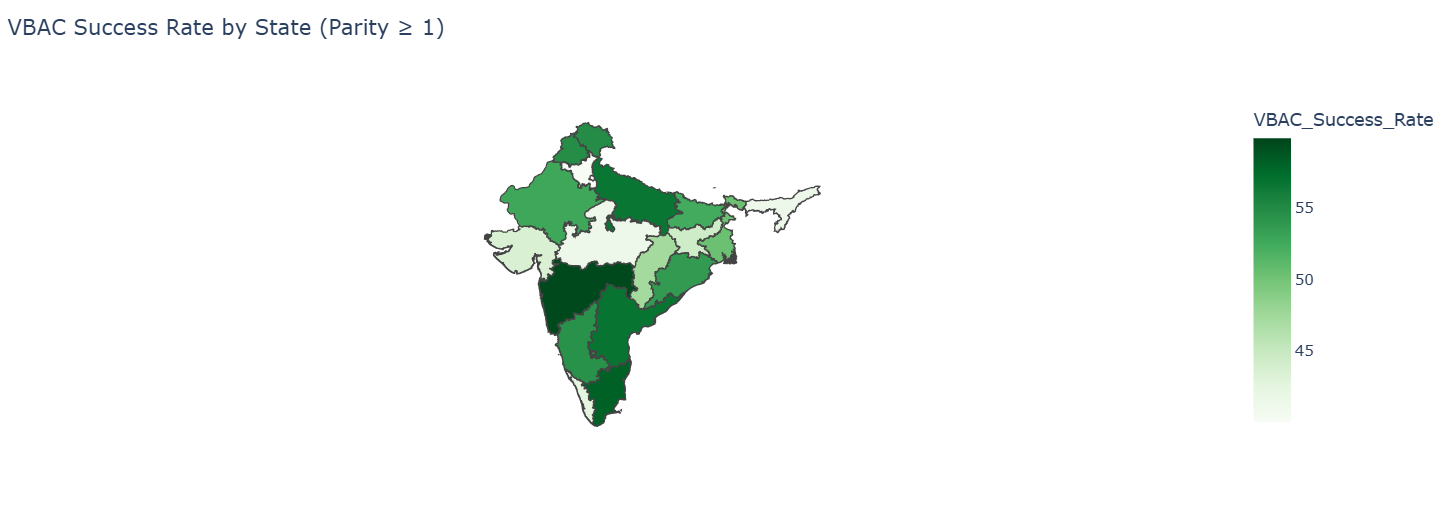
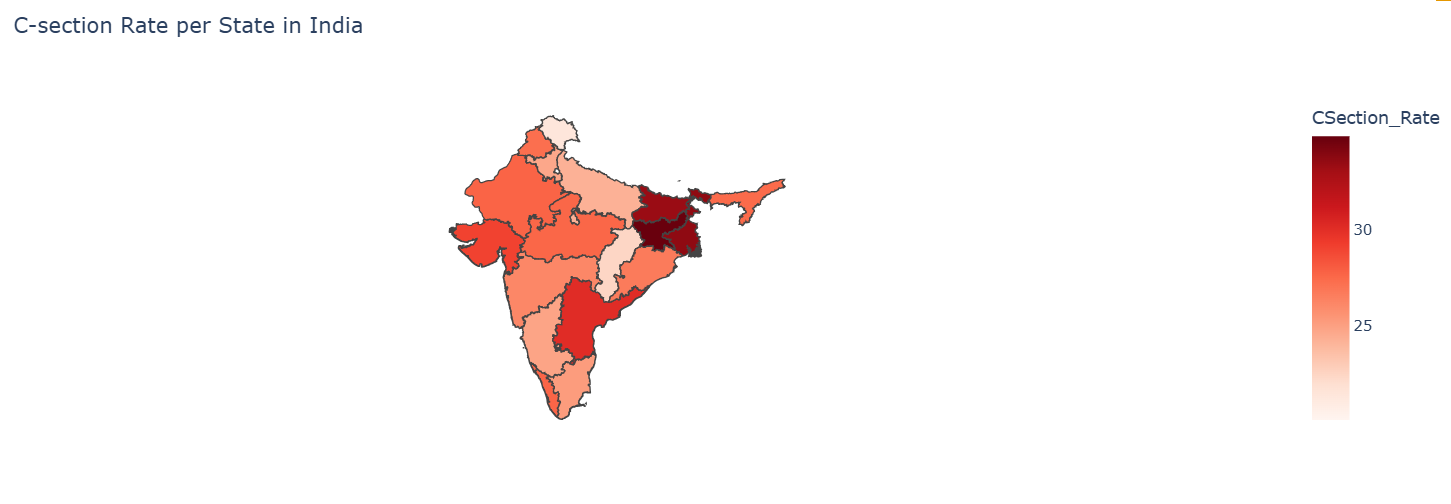
        fig.show()

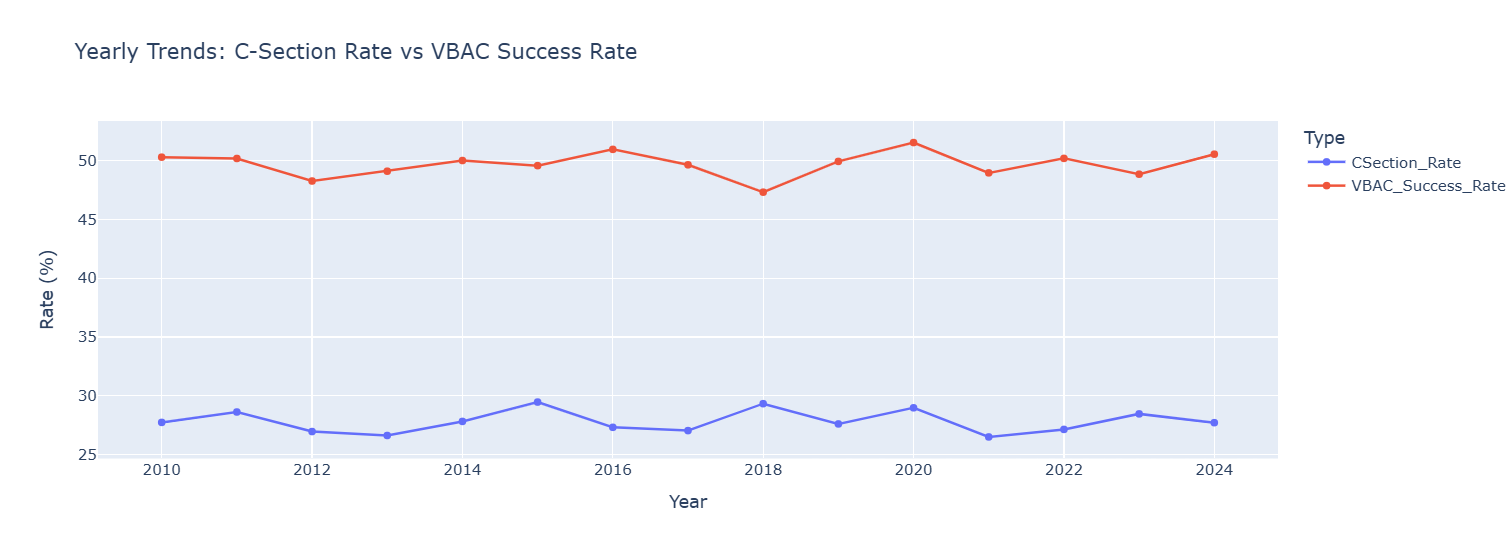
if \_\_name\_\_ == "\_\_main\_\_":

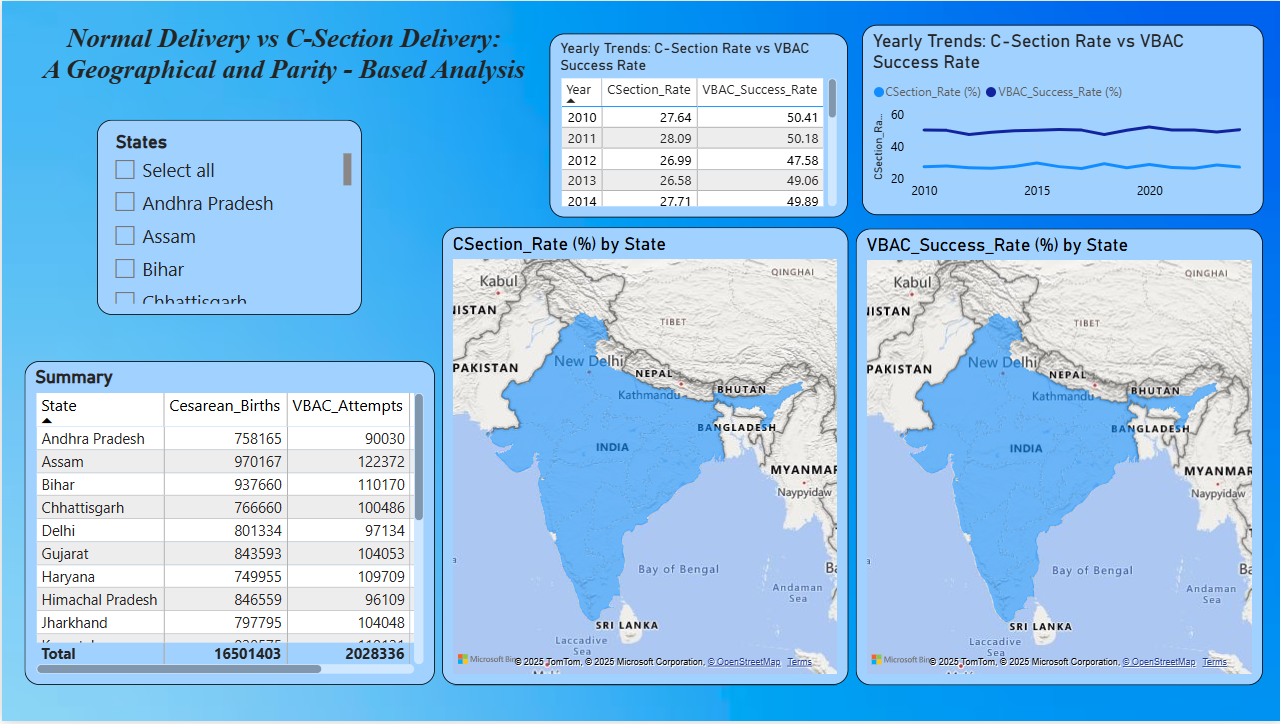
    analysis = DeliveryAnalysis("delivery.csv")

    analysis.plot\_yearly\_trends()

**Outputs:**







**Conclusion:**

This analysis offers a comprehensive state-wise overview of childbirth patterns across India, with a particular focus on **Caesarean (C-section) deliveries** and **VBAC (Vaginal Birth After Caesarean) trends**.

Key insights include:

* **C-section Rates** vary significantly by region, with some states consistently reporting higher-than-recommended rates (above WHO’s suggested 10–15%), indicating potential overuse of surgical delivery.
* **VBAC Attempts** are relatively limited in most states, suggesting a cautious or underutilized approach to offering vaginal delivery options post-C-section.
* **VBAC Success Rates** are generally moderate to high in states where attempts are made, signalling potential for scaling up safe VBAC practices.
* States with a **high rate of multiparous births** and **low VBAC attempts** represent a critical gap — women eligible for VBAC may not be given the opportunity.
* The **dashboard in Power BI** enables interactive filtering by state and parity, making it easier for stakeholders to pinpoint problem areas and monitor progress over time.

**Bibliography:**

1. Dataset on Maternal Health Outcomes in India (2010–2024)
2. National Family Health Survey (NFHS) – Government of India
3. World Health Organization (WHO) – Maternal health and VBAC guidelines