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

Index:

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

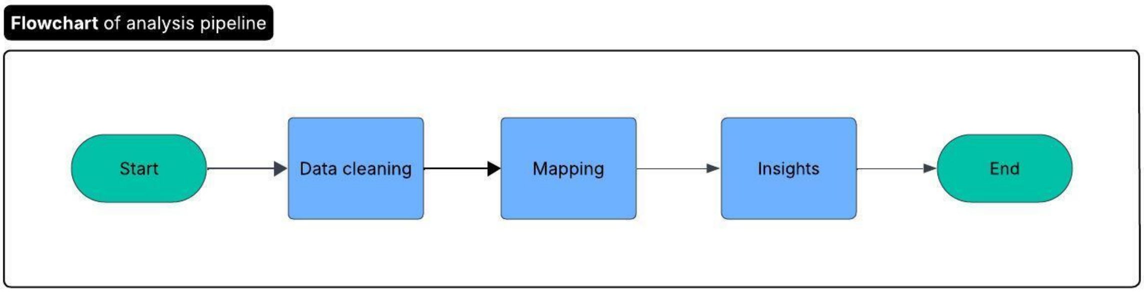
Benefits:

- 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:

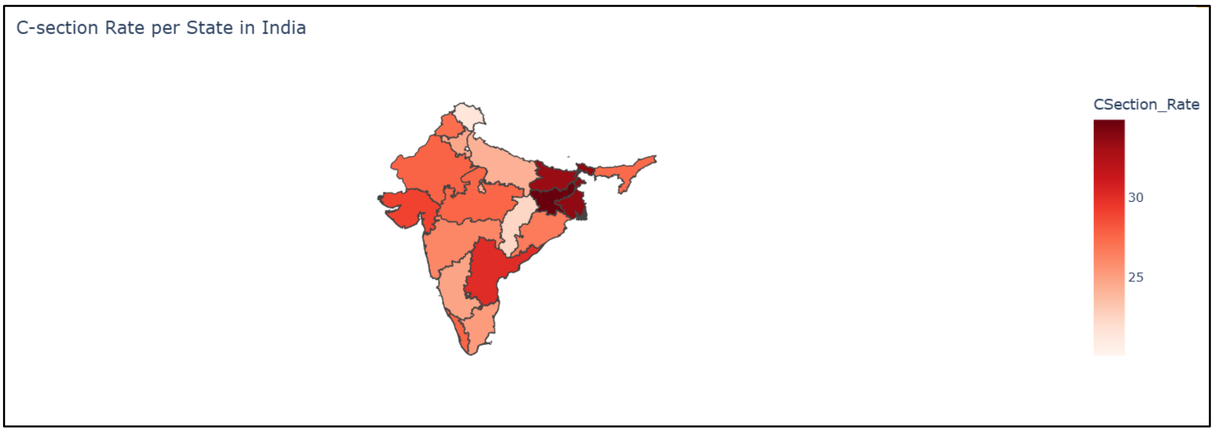
1. Load and clean the dataset
2. Create a normalized rate of C-sections and normal deliveries per region
3. Map regions using Plotly (choropleth map)
4. Filter data based on parity = 1 or more, and analyze VBAC trends
5. Calculate success rates of VBAC per region/parity group
6. Yearly trends: C-section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

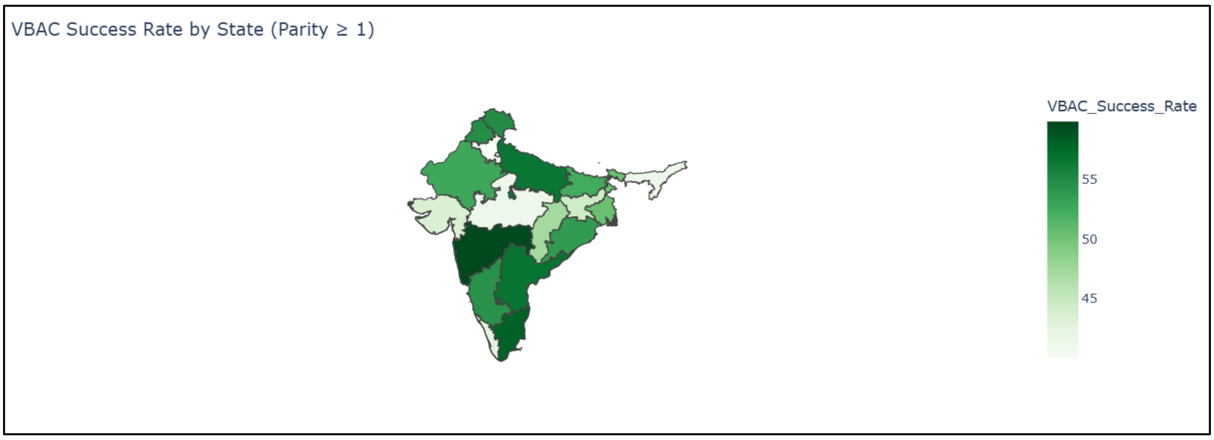
Code and Explanation:

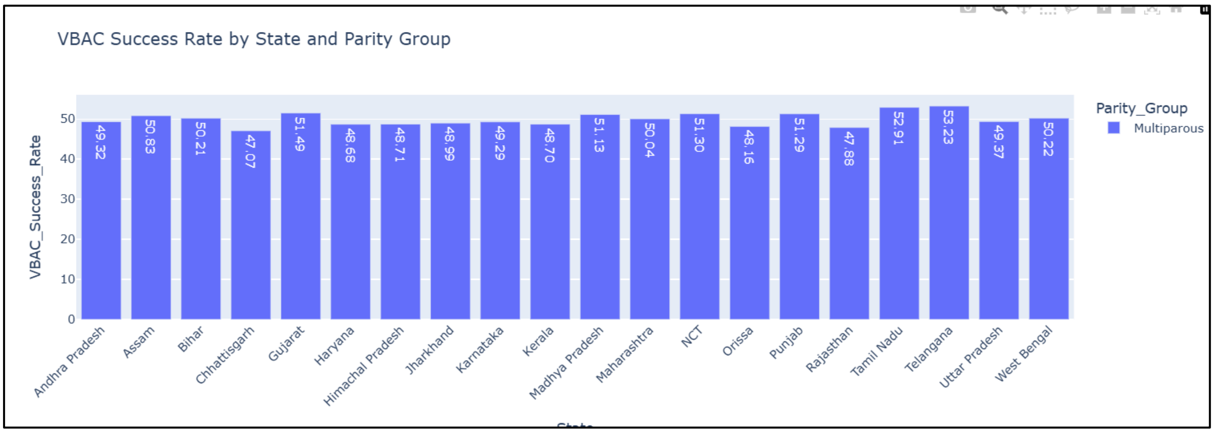
```
import pandas as pd
import plotly.express as px
import json


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
```

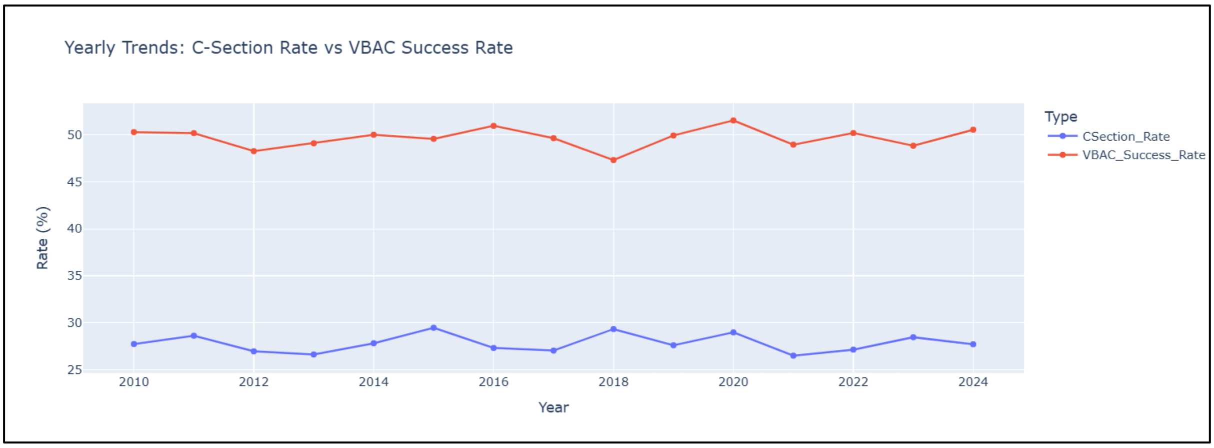
```
def load_data(self):
    self.df = pd.read_csv(self.csv_path)
    self.df.columns = self.df.columns.str.strip()
    self._normalize_state_names()

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()
```

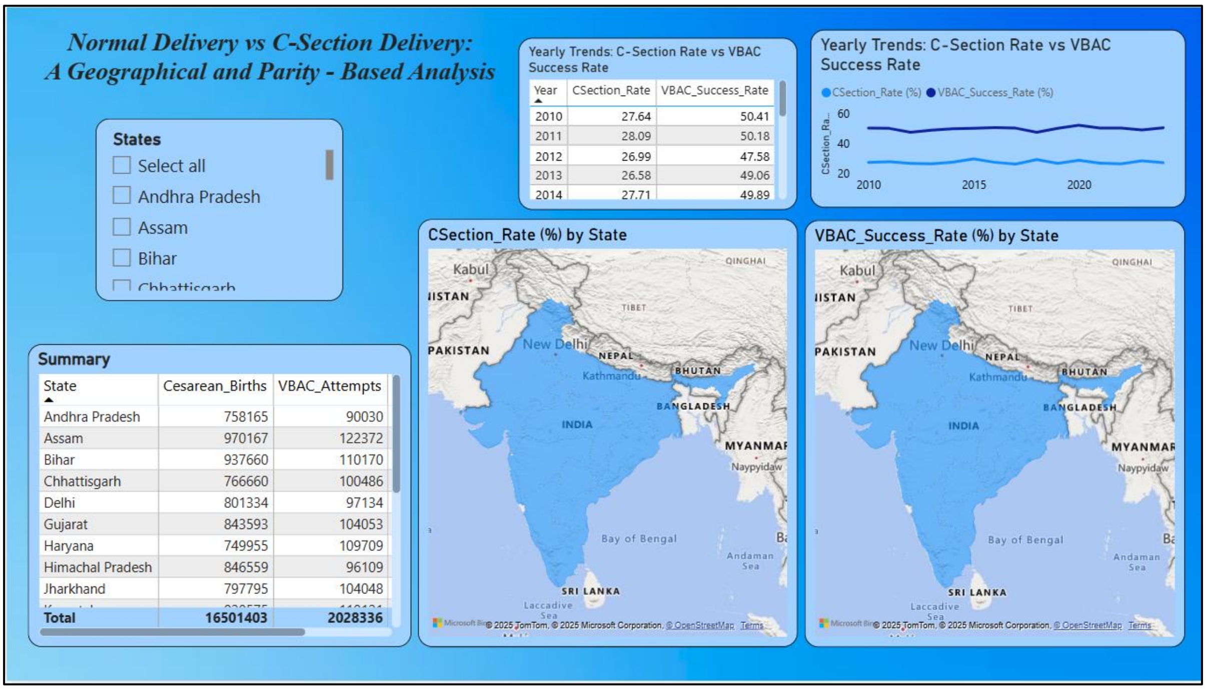
Outputs:


C-section Rate per State in India


VBAC Success Rate by State (Parity ≥ 1)


VBAC Success Rate by State and Parity Group





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

States

□ Select all
□ Andhra Pradesh
□ Assam
□ Bihar
□ Chhatticgarh

Yearly Trends: C-Section Rate vs VBAC Success Rate

Year	CSection_Rate	VBAC_Success_Rate
2010	27.64	50.41
2011	28.09	50.18
2012	26.99	47.58
2013	26.58	49.06
2014	27.71	49.89

Summary

State	Cesarean_Births	VBAC_Attempts
Andhra Pradesh	758165	90030
Assam	970167	122372
Bihar	937660	110170
Chhattisgarh	766660	100486
Delhi	801334	97134
Gujarat	843593	104053
Haryana	749955	109709
Himachal Pradesh	846559	96109
Jharkhand	797795	104048
[illegible]	[illegible]	[illegible]
Total	**16501403**	**2028336**

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