UNEMPLOYMENT ANALYSIS

OASIS INFOBYTE
(DATA SCIENCE INTERNSHIP)

Name: Aris Venitia Fernandes

Problem Statement

The COVID-19 pandemic has resulted in a significant rise in the unemployment rate, making it an intriguing subject for a data science project. This project aims to analyze the unemployment rate by examining key factors and data sources. By investigating the most affected regions, and valuable insights can be gained.

Objective

Evaluate the effects of the COVID-19 pandemic on the job market, analyzing which states have demonstrated resilience and which have experienced a more significant impact. By examining the various aspects of the dataset, a comprehensive understanding of how different states have fared in the face of the pandemic's economic challenges can be gained.

Dataset Description

The dataset provides information on the estimated unemployment rate, employment rate, and labor force participation rate for different states in India. The data is recorded on a monthly basis and covers a specific date range.

Data Dictionary

This dataset contains the unemployment rate of all the states in India

Region = states in India

Date = date which the unemployment rate observed

Frequency = measuring frequency (Monthly)

Estimated Unemployment Rate(%) = percentage of people unemployed in each States of India

Estimated Employed = percentage of people employed

Estimated Labour Participation Rate(%) = labour force participation rate by dividing the number of people actively participating in the labour force by the total number of people eligible to participate in the labor force

Region.1 = Parts of the state for the respective data (Direction)

longitude = longitude of states in the respective region

latitude = latitude of states in the respective region

Import the required libraries

In [7]:
```
# import numpy
import numpy as np

# import pandas
import pandas as pd

# import matplotlib.pyplot
import matplotlib.pyplot as plt

# import seaborn
import seaborn as sns

# to suppress the warnings
import warnings
warnings.filterwarnings('ignore')

# import plotly.express
# to create interactive visualizations and plots
import plotly.express as px
```

Understanding the data

In [8]:
```
# load the csv file
df = pd.read_csv('Unemployment_Rate_upto_11_2020.csv')
```

In [9]:
```
# display the first five observations
df.head()
```

Out[9]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Region.1	longitude	latitude
0	Andhra Pradesh	31-01-2020	M	5.48	16635535	41.02	South	15.9129	79.74
1	Andhra Pradesh	29-02-2020	M	5.83	16545652	40.90	South	15.9129	79.74
2	Andhra Pradesh	31-03-2020	M	5.79	15881197	39.18	South	15.9129	79.74
3	Andhra Pradesh	30-04-2020	M	20.51	11336911	33.10	South	15.9129	79.74
4	Andhra Pradesh	31-05-2020	M	17.43	12988845	36.46	South	15.9129	79.74

In [10]:
```
# check the no. of rows and columns
df.shape
```

Out[10]:
```
(267, 9)
```

In [11]:
```
# Check the data types of the columns for the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 267 entries, 0 to 266
Data columns (total 9 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Region                                    267 non-null    object
 1    Date                                     267 non-null    object
 2    Frequency                                267 non-null    object
 3    Estimated Unemployment Rate (%)          267 non-null    float64
 4    Estimated Employed                       267 non-null    int64
 5    Estimated Labour Participation Rate (%)  267 non-null    float64
 6   Region.1                                  267 non-null    object
 7   longitude                                 267 non-null    float64
 8   latitude                                  267 non-null    float64
dtypes: float64(4), int64(1), object(4)
memory usage: 18.9+ KB
```

In [12]:
```
# renaming columns in appropriate format
df = df.rename(columns={df.columns[0]:'State', df.columns[3]:'EUR', df.columns[4]:'EE', df.columns[5]:'ELPR',
                        df.columns[6]:'Region', df.columns[7]:'Longitude', df.columns[8]:'Latitude'})

# display column names
df.columns
```

Out[12]:
```
Index(['State', ' Date', ' Frequency', 'EUR', 'EE', 'ELPR', 'Region',
       'Longitude', 'Latitude'],
      dtype='object')
```

In [13]:
```
# to retrive an array of unique values from the "State"
df.State.unique()
```

Out[13]:
```
array(['Andhra Pradesh', 'Assam', 'Bihar', 'Chhattisgarh', 'Delhi', 'Goa',
       'Gujarat', 'Haryana', 'Himachal Pradesh', 'Jammu & Kashmir',
       'Jharkhand', 'Karnataka', 'Kerala', 'Madhya Pradesh',
       'Maharashtra', 'Meghalaya', 'Odisha', 'Puducherry', 'Punjab',
       'Rajasthan', 'Sikkim', 'Tamil Nadu', 'Telangana', 'Tripura',
       'Uttar Pradesh', 'Uttarakhand', 'West Bengal'], dtype=object)
```

In [14]:
```
# to retrive an array of unique values from the "Region"
df.Region.unique()
```

Out[14]:
```
array(['South', 'Northeast', 'East', 'West', 'North'], dtype=object)
```

In [15]:
```
# five point summary of continuous variables
df.describe()
```

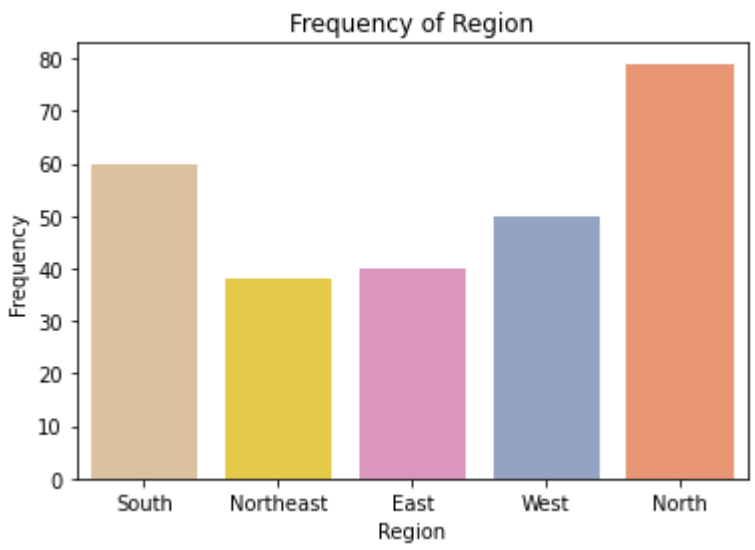
Out[15]:

	EUR	EE	ELPR	Longitude	Latitude
count	267.000000	2.670000e+02	267.000000	267.000000	267.000000
mean	12.236929	1.396211e+07	41.681573	22.826048	80.532425
std	10.803283	1.336632e+07	7.845419	6.270731	5.831738
min	0.500000	1.175420e+05	16.770000	10.850500	71.192400
25%	4.845000	2.838930e+06	37.265000	18.112400	76.085600
50%	9.650000	9.732417e+06	40.390000	23.610200	79.019300
75%	16.755000	2.187869e+07	44.055000	27.278400	85.279900
max	75.850000	5.943376e+07	69.690000	33.778200	92.937600

Data Visualisation

Frequency of Region

In [16]:
```
# creating a barplot of the Frequency of Region
sns.countplot(data = df, x = 'Region', palette = 'Set2_r')
plt.title('Frequency of Region')
plt.ylabel('Frequency')
plt.show()
df.groupby('Region').size()
```



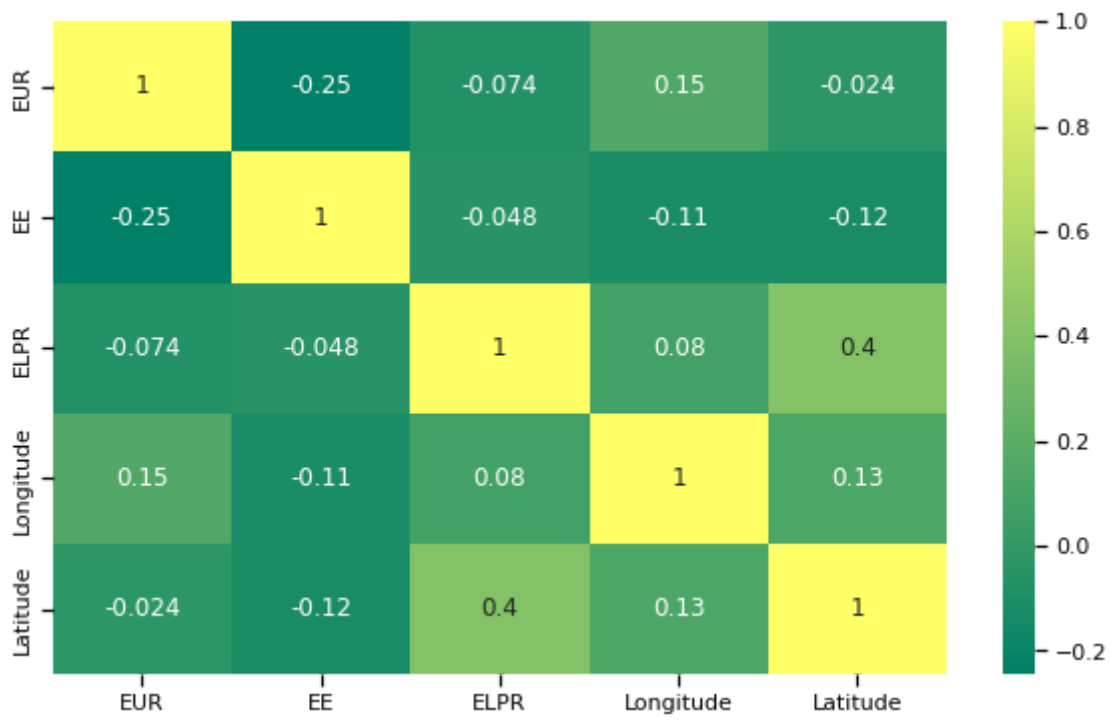
Out[16]:
```
Region
East         40
North        79
Northeast    38
South        60
West         50
dtype: int64
```

In [17]:
```
# checking avaerage of 'EUR','EE','ELPR' across 'Region'
region_stats = df.groupby(['Region'])[['EUR','EE','ELPR']].mean().reset_index()
region_stats = round(region_stats,2)
region_stats
```

Out[17]:

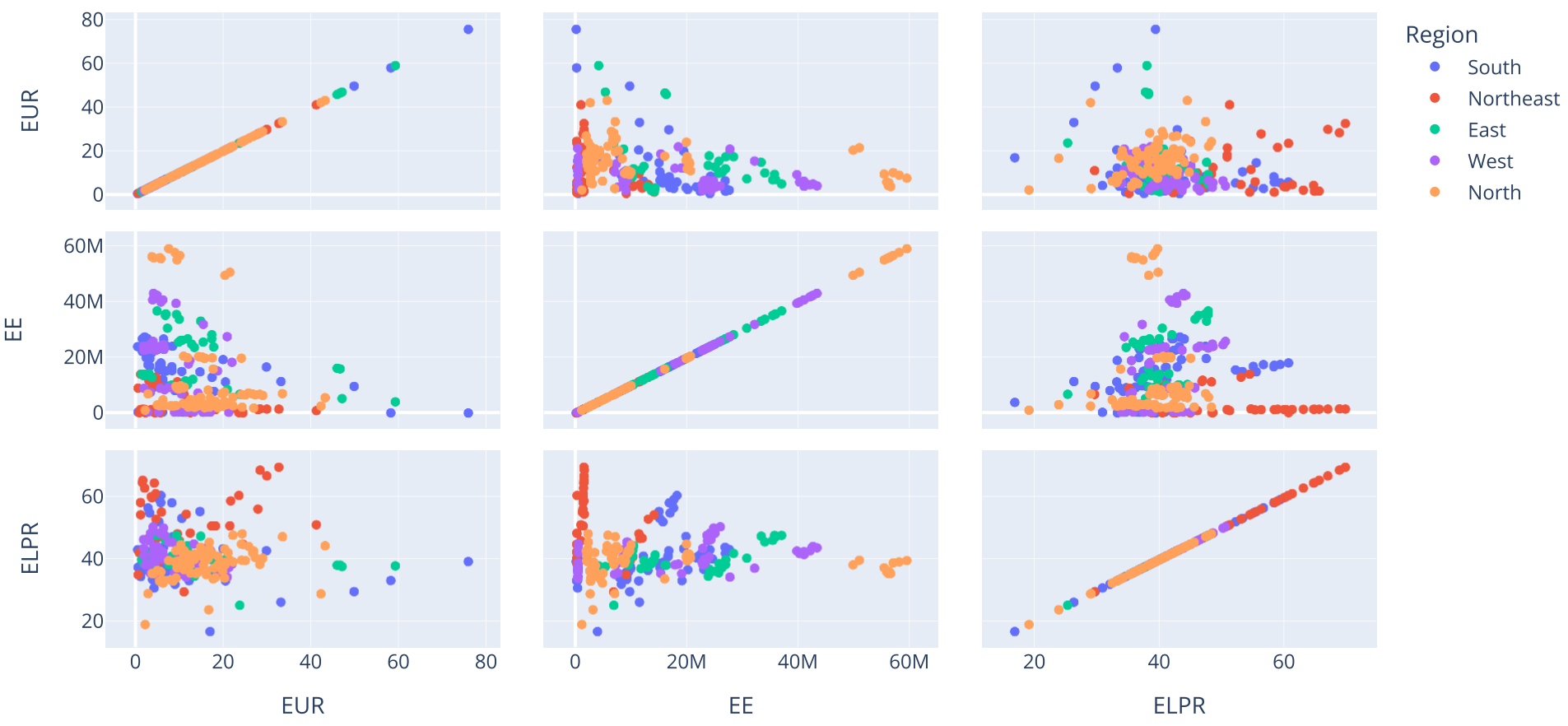
	Region	EUR	EE	ELPR
0	East	13.92	19602366.90	40.11
1	North	15.89	13072487.92	38.70
2	Northeast	10.95	3617105.53	52.06
3	South	10.45	14040589.33	40.44
4	West	8.24	18623512.72	41.26

In [18]:
```
# creating a dataframe of all continuous variables
conti_var = df[['EUR','EE', 'ELPR','Longitude', 'Latitude']]

# checking correlation
conti_corr = conti_var.corr()

# creating a heatmap for all continuous variables
plt.figure(figsize = (10,6))
sns.set_context('notebook', font_scale = 1)
sns.heatmap(conti_corr, annot = True, cmap = 'summer')
plt.show()
```



The above graph shows negative correlation between EUR, ELPR and EE.

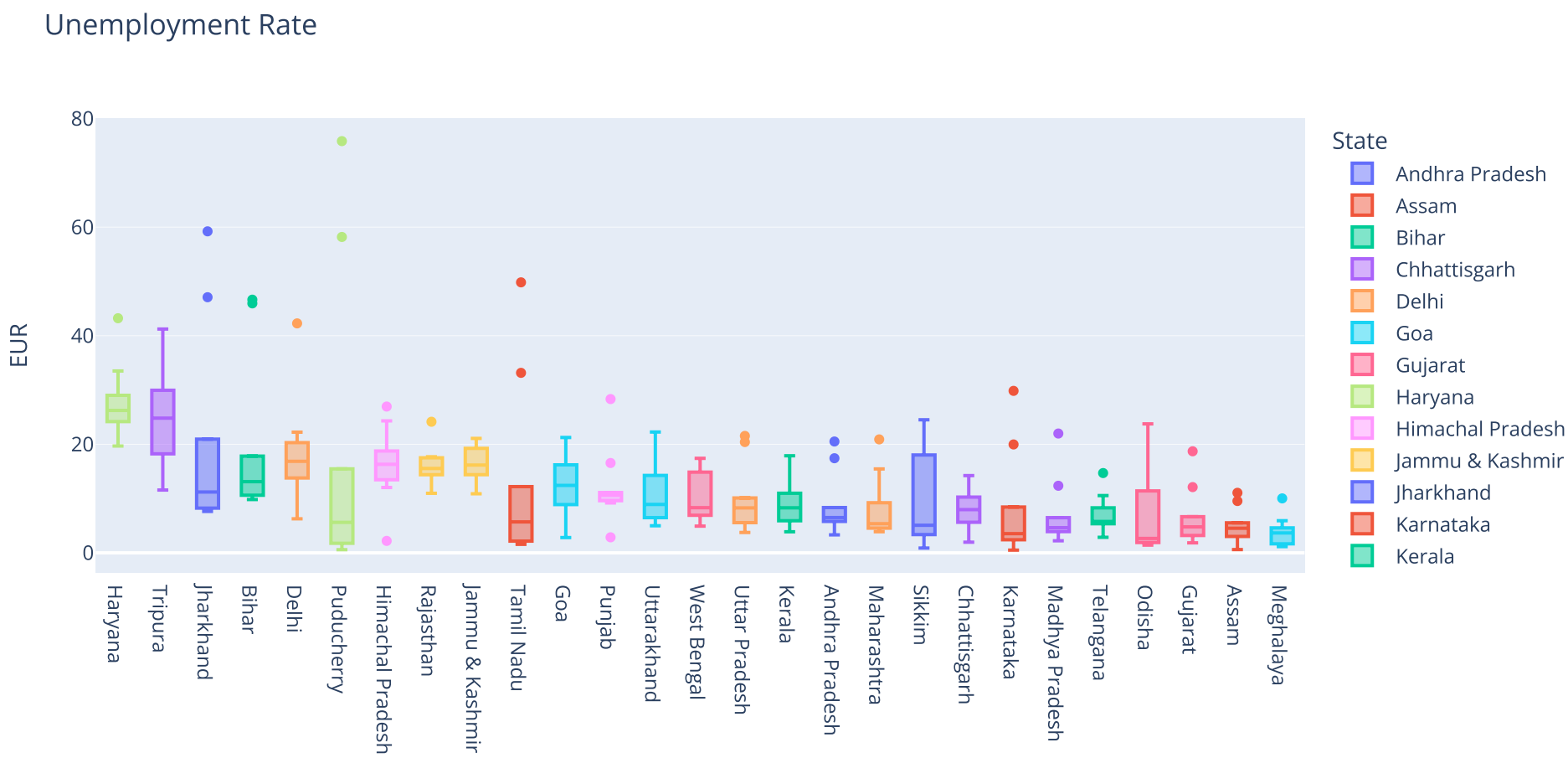
In [19]:
```
# creating scatter matrix to understand correlation across 'Region'
fig = px.scatter_matrix(df,template='plotly', dimensions=['EUR','EE','ELPR'], color='Region')
fig.show()
```



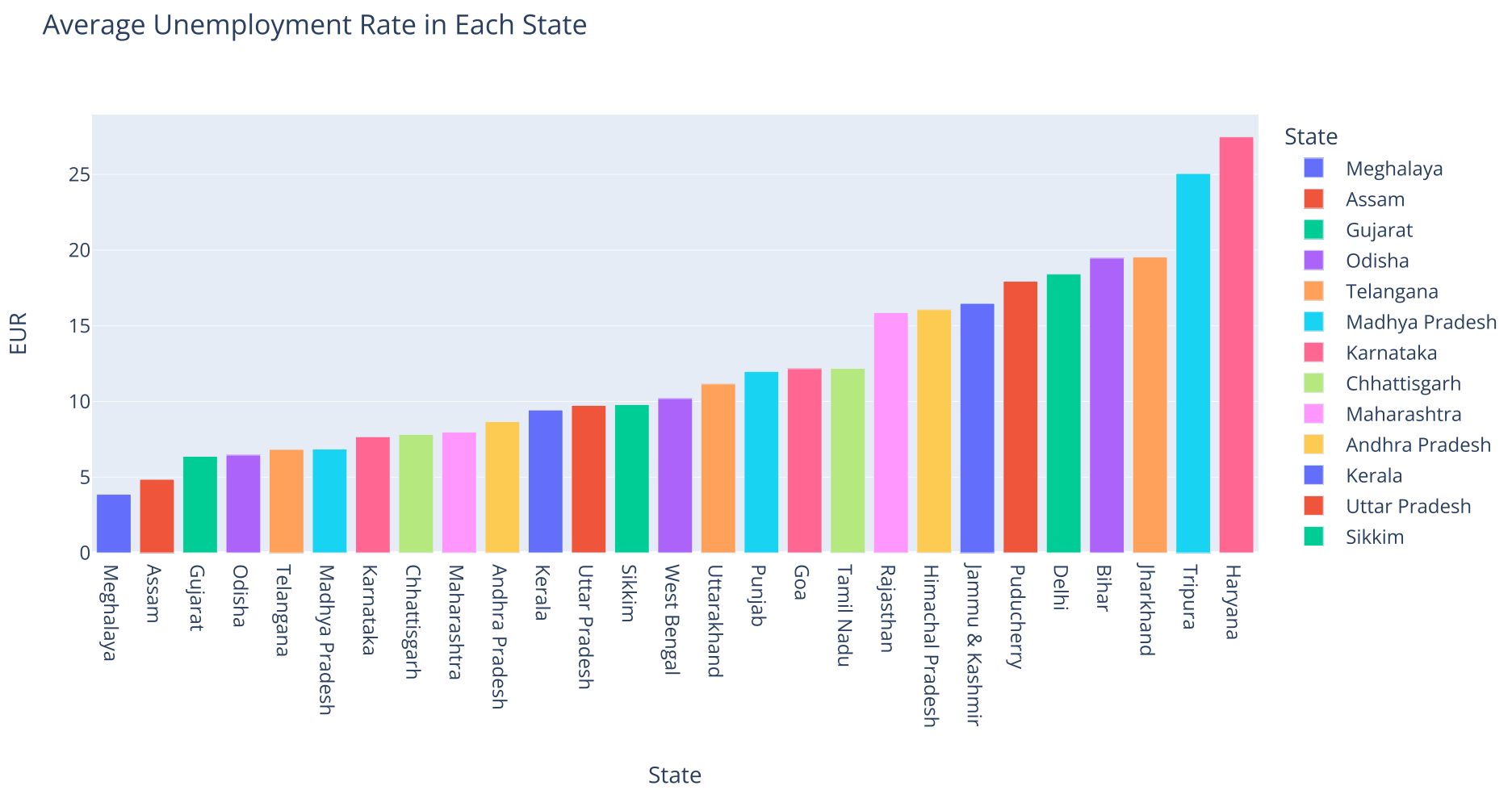
From the above scatter plots we can say that, there is a negative correlation between EUR, ELPR and EE.

Unemployment Rate by State

In [20]:
```
# The below box shows unemployment rate in each state in India
fig = px.box(df, x ='State', y ='EUR', color ='State', title = 'Unemployment Rate', template = 'plotly')
fig.update_layout(xaxis = {'categoryorder': 'total descending'})
fig.show()
```

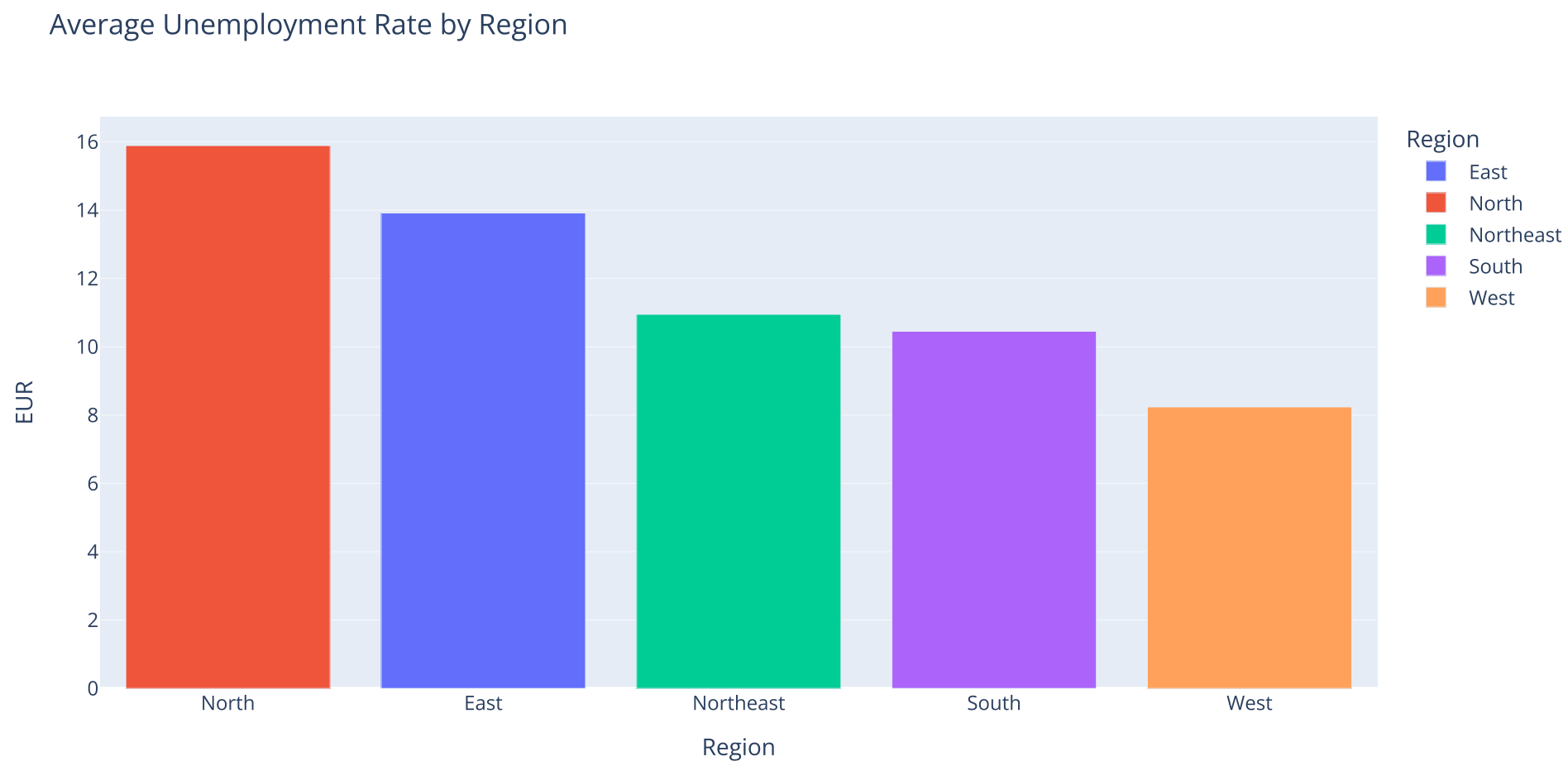


In [21]:
```
# creating a dataframe of 'EUR' and 'State'
plot_df = df[['EUR','State']]

# grouping EUR wrt State and finding its average
unemp = plot_df.groupby('State').mean().reset_index()
unemp = unemp.sort_values('EUR')

# The below box shows average unemployment rate in each state in India
px.bar(unemp, x = 'State', y = 'EUR', color = 'State', title = 'Average Unemployment Rate in Each State', template = 'plotly')
```



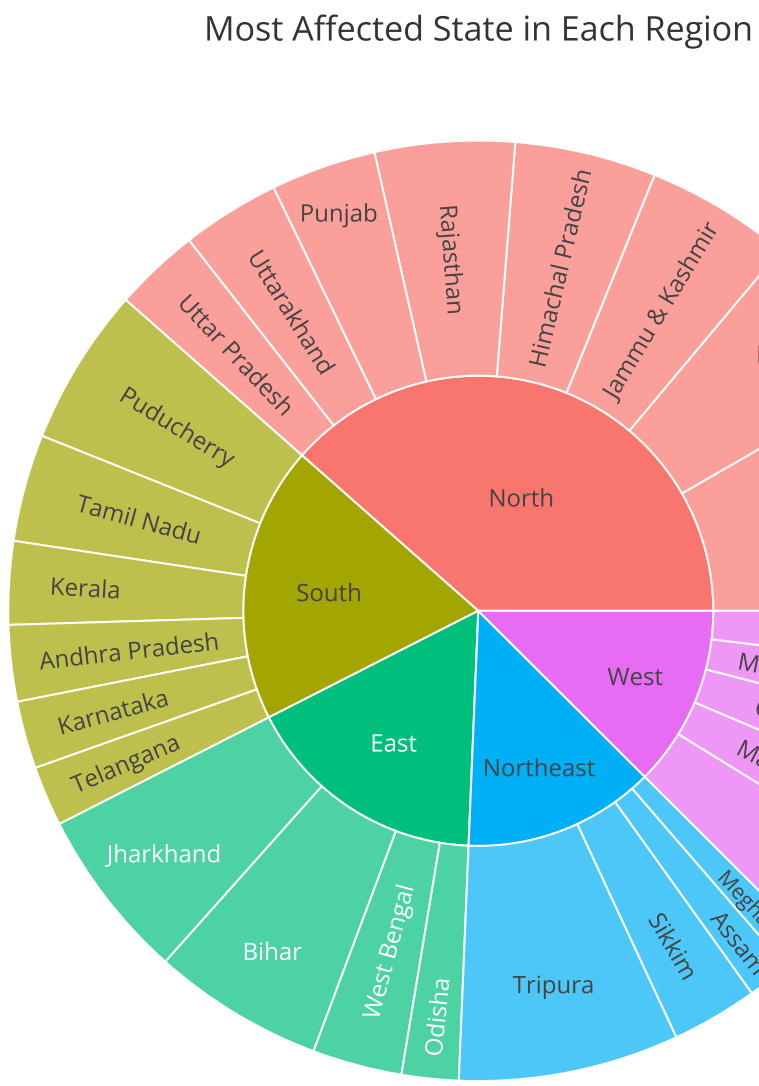
Unemployment Rate by Region

In [22]:
```
# creating a dataframe of 'EUR' and 'State'
region = df.groupby(["Region"])[['EUR', 'EE', 'ELPR']].mean()
region = pd.DataFrame(region).reset_index()

# The below box shows unemployment rate in each region
fig = px.bar(region, x="Region", y="EUR", color="Region", title="Average Unemployment Rate by Region")
fig.update_layout(xaxis={'categoryorder':'total descending'})
fig.show()
```



Northern Region of the various States has the highest number of Average Unemployment Rate.

In [23]:
```
# creating a dataframe
unemp = df[['State','Region','EUR','EE','ELPR']]
unemp = unemp.groupby(['Region','State'])['EUR'].mean().reset_index()

# create a graph to check the most affected states in every region
fig = px.sunburst(unemp, path = ['Region','State'], values = 'EUR', color_continuous_scale = 'Plasma',
                  title = 'Most Affected State in Each Region', height = 650, template = 'ggplot2')
fig.show()
```

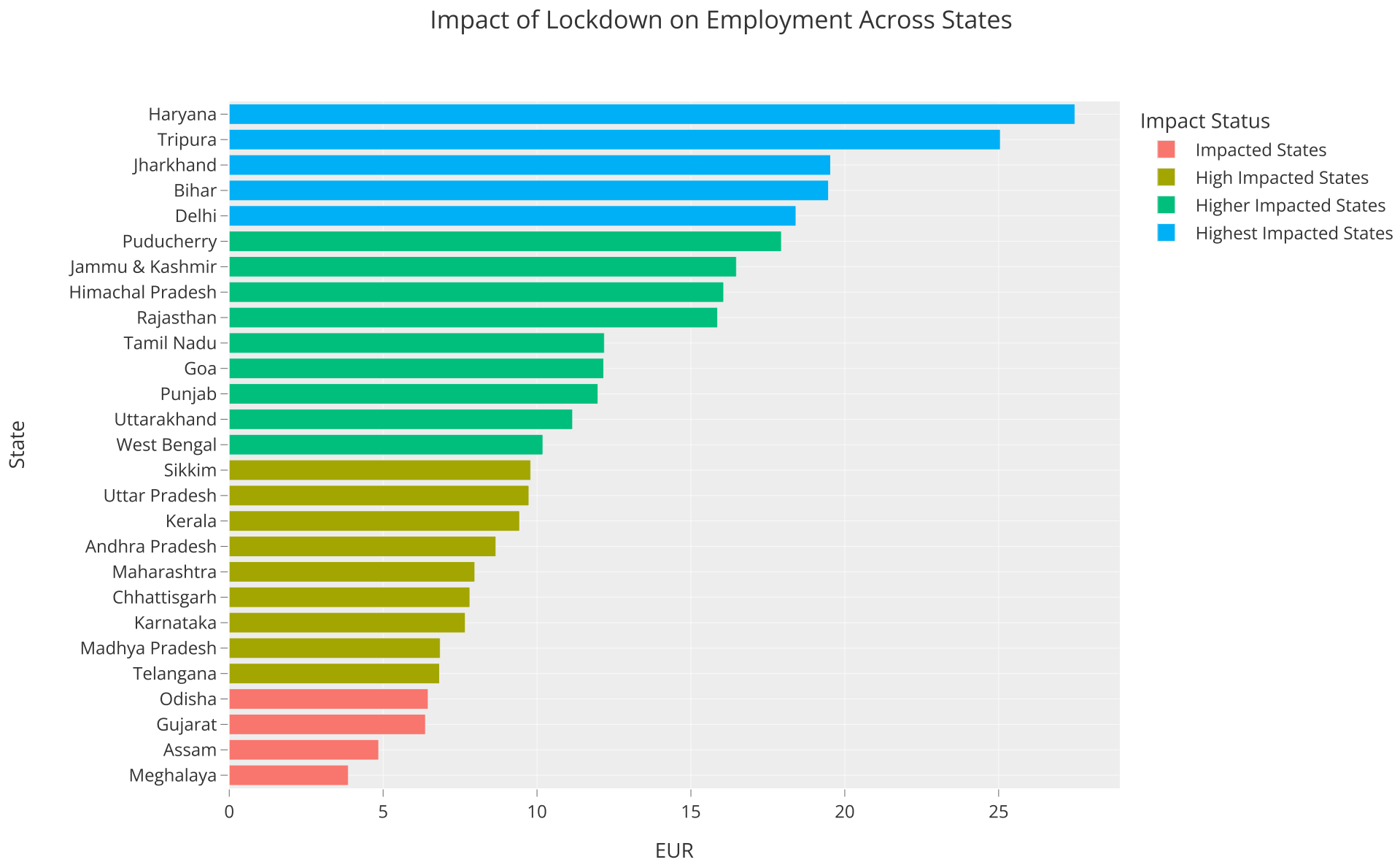


Most Affected State in Each Region

1. North Region: Haryana
2. East Region: Jharkhand
3. West Region: Goa
4. South Region: Puducherry
5. Northeast Region: Tripura

In [26]:
```
# function to sort value based on impact
def impact_type(x):
    if x <= 6.5:
        return 'Impacted States'
    elif x <= 10:
        return 'High Impacted States'
    elif x <= 18:
        return 'Higher Impacted States'
    elif x > 18:
        return 'Highest Impacted States'
    return x
```

In [29]:
```
unemp = df.groupby(['Region','State'])['EUR'].mean().reset_index().sort_values(['EUR'])
unemp['Impact Status'] = unemp['EUR'].apply(lambda x:impact_type(x))

# creating a bar garph
fig = px.bar(unemp, y ='State', x = 'EUR', color = 'Impact Status', template = 'ggplot2', height = 650,
             title ='Impact of Lockdown on Employment Across States')
fig.show()
```



From the above graph, we can see the highest impacted states which suffered unemployment across India.

Conclusion

- The nationwide lockdown imposed by the Indian government in March 2020, in response to the COVID-19 pandemic, had a profound impact on the job market. The restrictions resulted in a significant wave of unemployment across the nation. Based on the data available, states such as Haryana, Tripura, Jharkhand, Bihar, and Delhi emerged as the top five states experiencing the highest rates of unemployment during this period. These findings underscore the widespread economic consequences of the lockdown measures on employment in these regions.
- While the nationwide lockdown in response to the COVID-19 pandemic led to widespread unemployment in India, some states experienced relatively lower levels of impact. Meghalaya, Assam, Gujarat, and Odisha emerged as the states with comparatively lower rates of unemployment during this challenging period.