Unemployment Anaylsis

In order to comprehend the causes and effects of unemployment, statistics on unemployment rates are gathered, examined, and interpreted through the process of unemployment analysis. The method of automating unemployment analysis may be done with the help of Python, a strong computer language.

The steps for doing a Python unemployment study are as follows:

1. Gather information on unemployment rates - You may get information on unemployment rates from a variety of sources, including the World Bank, the Bureau of Labour Statistics (BLS), and the International Labour Organisation (ILO).
2. Data preparation and cleaning - After gathering the data, you must clean it and get it ready for analysis. This can entail eliminating redundant data, adding missing values, and transforming the data into a Python-compatible format.
3. Review the data - There are several statistical methods available for analysing unemployment rates. Descriptive statistics, regression analysis, and time series analysis are a few popular methods.
4. Make the data visible - The use of visualisation to convey the findings of an unemployment investigation is quite effective. Charts, graphs, and maps are just a few examples of the visualisations you can make with Python.
5. Analyse the outcomes - The results of the unemployment analysis are then interpreted. Drawing conclusions on the causes and effects of unemployment is required for this.

Although unemployment analysis is a challenging undertaking, Python may be an effective tool for making it simpler. Python may aid in your decision-making process by automating the gathering, cleaning, preparation, analysis, visualisation, and interpretation of data. This will allow you to better comprehend the unemployment problem.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import calendar
```

```
import datetime as dt
import plotly.io as pio
pio.templates
```

```
    Templates configuration
    -----------------------
        Default template: 'plotly'
        Available templates:
            ['ggplot2', 'seaborn', 'simple_white', 'plotly',
             'plotly_white', 'plotly_dark', 'presentation', 'xgridoff',
             'ygridoff', 'gridon', 'none']
```

```
import plotly.express as px
import plotly.graph_objects as go
import plotly.figure_factory as ff
from IPython.display import HTML
```

```
df = pd.read_csv('/content/Unemployment in India.csv')
df = pd.read_csv('/content/Unemployment_Rate_upto_11_2020.csv')
```

```
df.head()
```

	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-	M	5.83	16545652	40.90	South	15.9129	79.74

```
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
```

```
df.isnull().sum()
```

```
Region                                    0
 Date                                     0
 Frequency                                0
 Estimated Unemployment Rate (%)          0
 Estimated Employed                       0
 Estimated Labour Participation Rate (%)  0
Region.1                                  0
longitude                                 0
latitude                                  0
dtype: int64
```

```
df.columns =['States','Date','Frequency','Estimated Unemployment Rate','Estimated Employed','Estimated Labour Participation Rate','Region','l
```

```
df['Date'] = pd.to_datetime(df['Date'],dayfirst=True)
```

```
df['Frequency']= df['Frequency'].astype('category')
```

```
df['Month'] =  df['Date'].dt.month
```

```
df['Month_int'] = df['Month'].apply(lambda x : int(x))
```

```
df['Month_name'] =  df['Month_int'].apply(lambda x: calendar.month_abbr[x])
```

```
df['Region'] = df['Region'].astype('category')
```

```
df.drop(columns='Month',inplace=True)
df.head(3)
```

	States	Date	Frequency	Estimated Unemployment Rate	Estimated Employed	Estimated Labour Participation Rate	Region	longitude	latitude	Month
0	Andhra Pradesh	2020-01-31	M	5.48	16635535	41.02	South	15.9129	79.74	
1	Andhra Pradesh	2020-02-29	M	5.83	16545652	40.90	South	15.9129	79.74	

```
 df_stats = df[['Estimated Unemployment Rate',
       'Estimated Employed', 'Estimated Labour Participation Rate']]
```

```
round(df_stats.describe().T,2)
```

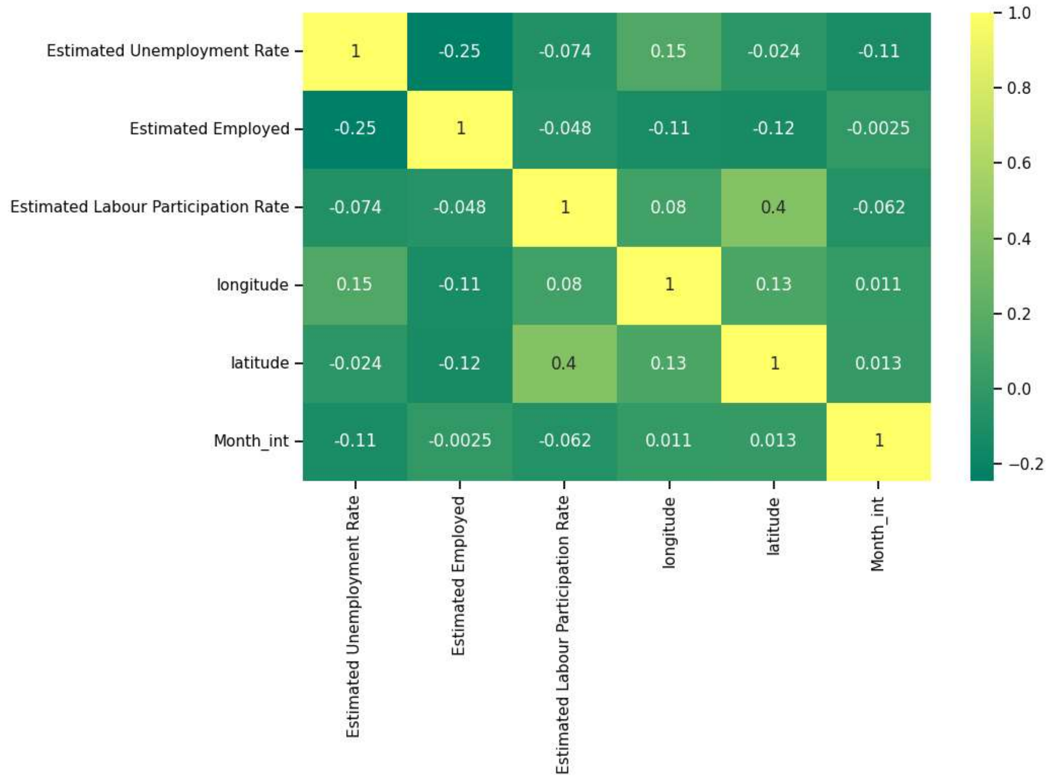
```
                    count       mean        std       min       25%       50%       75%       ma
```

```
region_stats = df.groupby(['Region'])[['Estimated Unemployment Rate','Estimated Employed','Estimated Labour Participation Rate']].mean().rese

region_stats = round(region_stats,2)
```

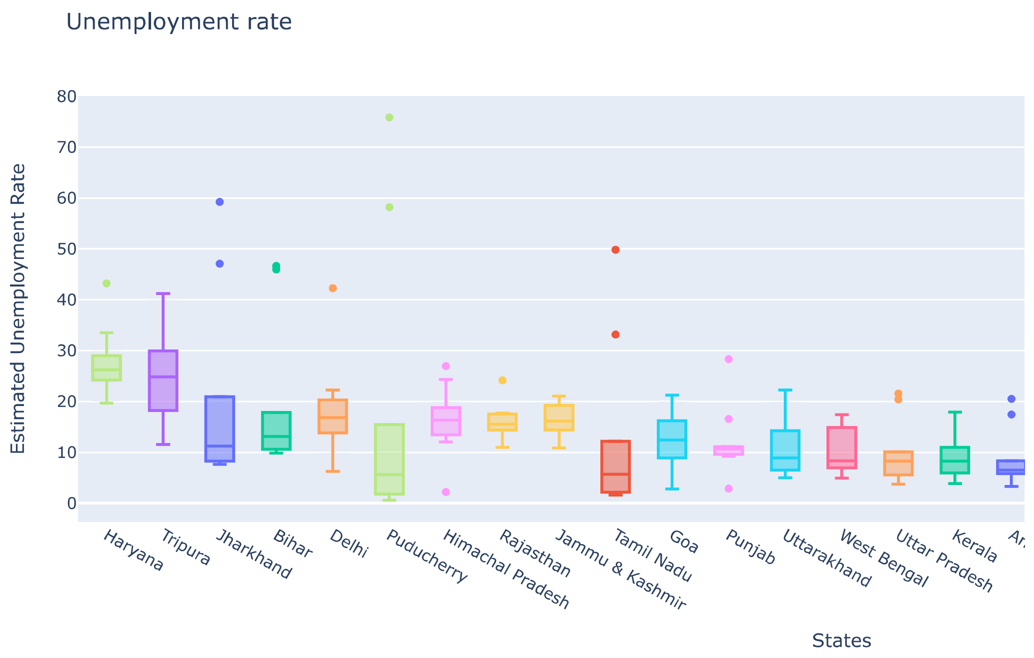
```
region_stats
```

	Region	Estimated Unemployment Rate	Estimated Employed	Estimated Labour Participation Rate
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

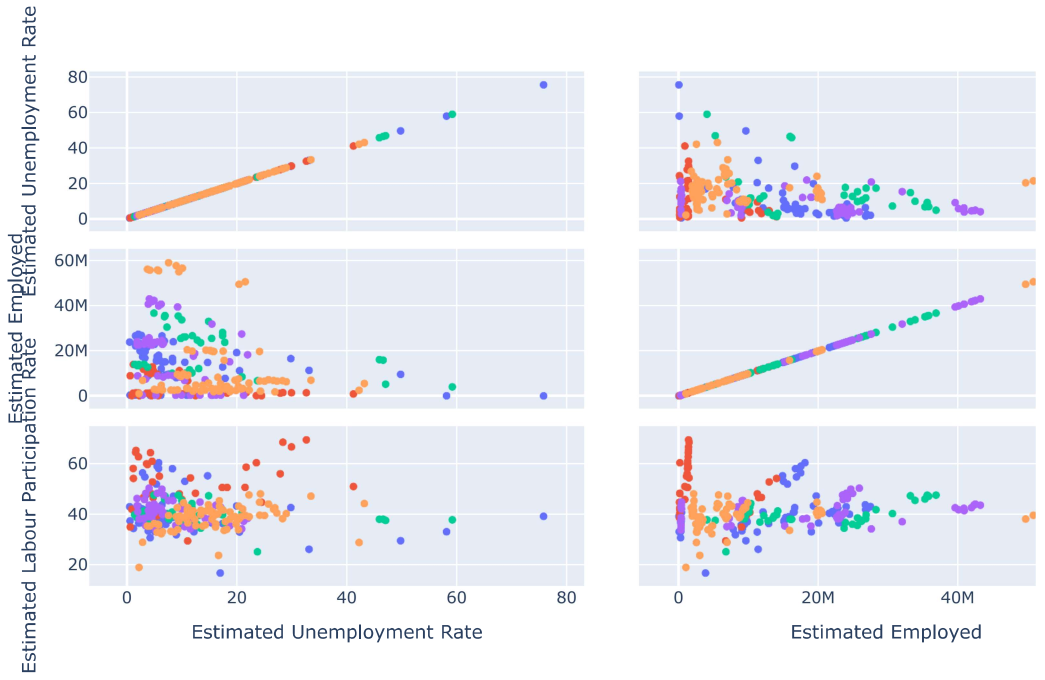
```
heat_maps = df[['Estimated Unemployment Rate',
       'Estimated Employed', 'Estimated Labour Participation Rate',
       'longitude', 'latitude', 'Month_int']]

heat_maps = heat_maps.corr()

plt.figure(figsize=(10,6))
sns.set_context('notebook',font_scale=1)
sns.heatmap(heat_maps, annot=True,cmap='summer');
```



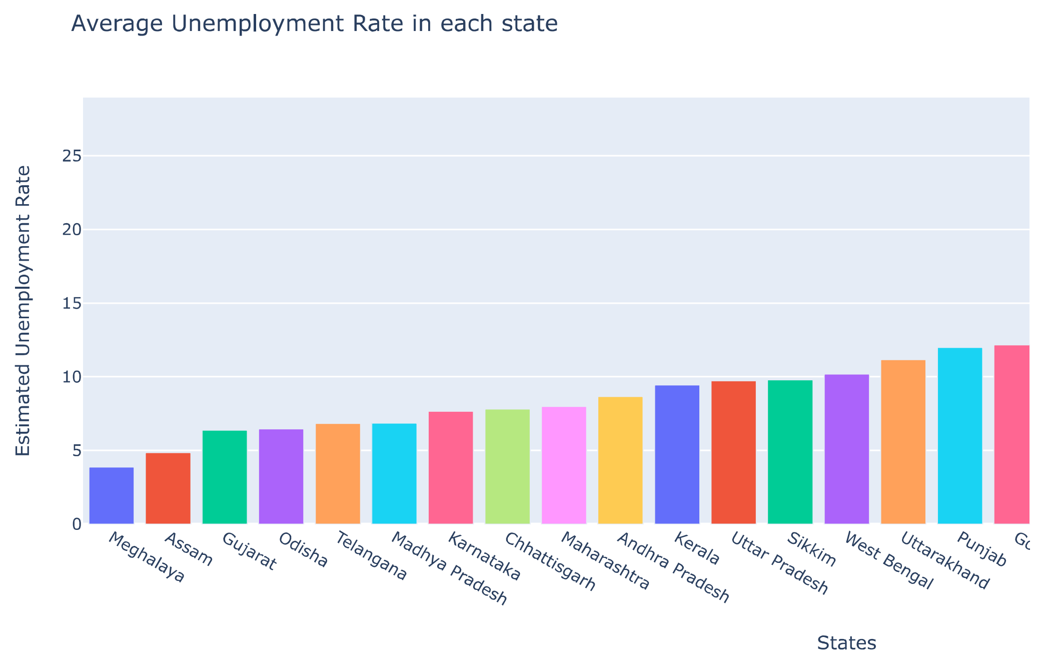
```
fig = px.box(df,x='States',y='Estimated Unemployment Rate',color='States',title='Unemployment rate',template='plotly')
fig.update_layout(xaxis={'categoryorder':'total descending'})
fig.show()
```



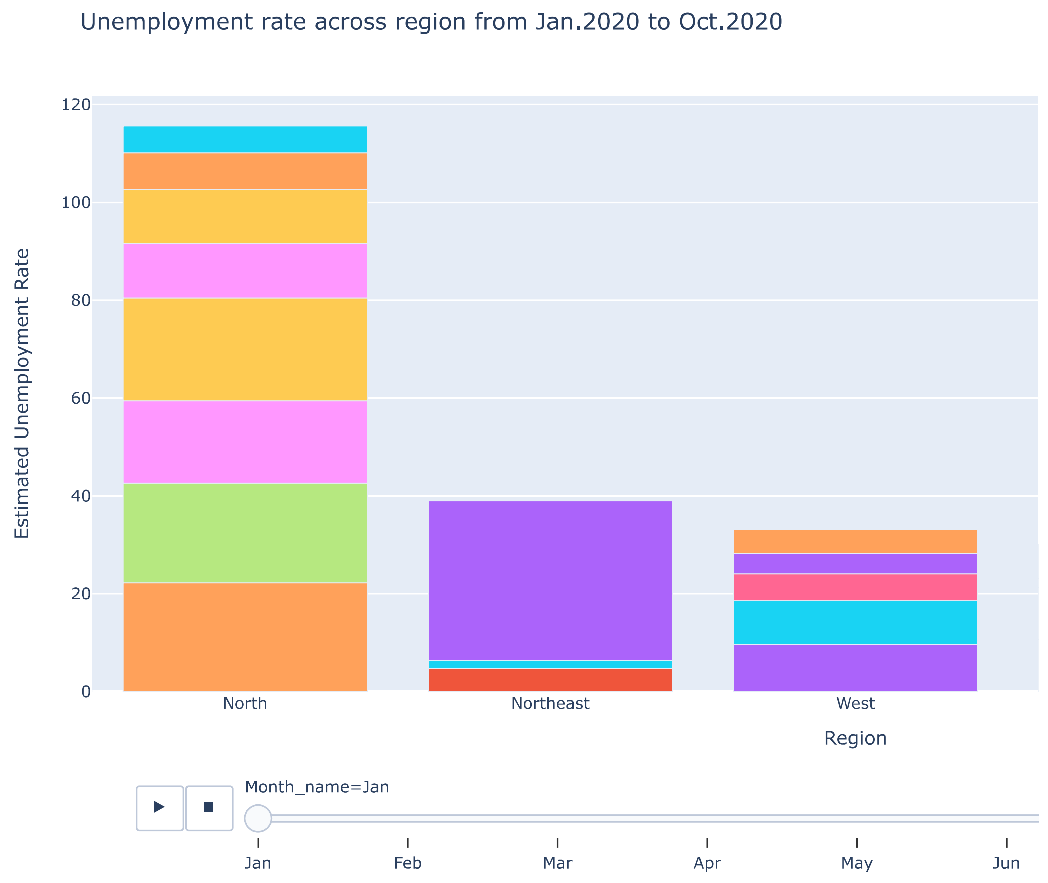
```
fig = px.scatter_matrix(df,template='plotly',
    dimensions=['Estimated Unemployment Rate','Estimated Employed',
                'Estimated Labour Participation Rate'],
    color='Region')
fig.show()
```



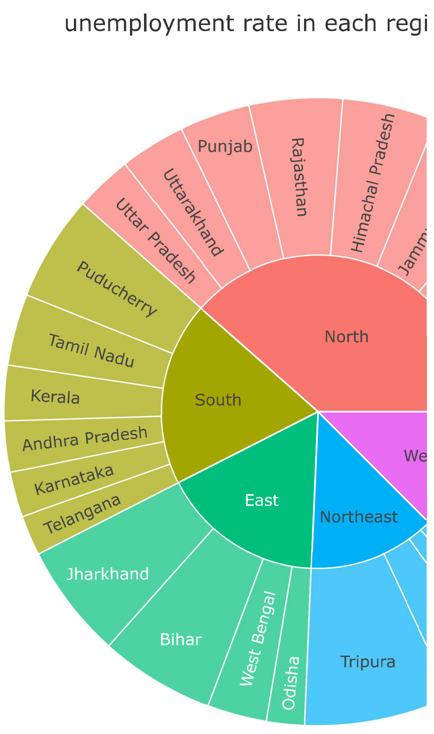
```
plot_ump = df[['Estimated Unemployment Rate','States']]

df_unemp = plot_ump.groupby('States').mean().reset_index()

df_unemp = df_unemp.sort_values('Estimated Unemployment Rate')

fig = px.bar(df_unemp, x='States',y='Estimated Unemployment Rate',color='States',
            title='Average Unemployment Rate in each state',template='plotly')

fig.show()
```



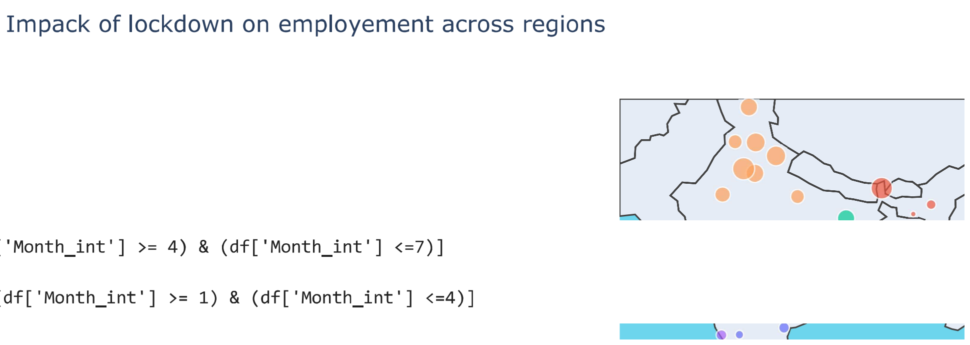
```
fig = px.bar(df, x='Region',y='Estimated Unemployment Rate',animation_frame = 'Month_name',color='States',
            title='Unemployment rate across region from Jan.2020 to Oct.2020', height=700,template='plotly')

fig.update_layout(xaxis={'categoryorder':'total descending'})

fig.layout.updatemenus[0].buttons[0].args[1]["frame"]["duration"] = 2000

fig.show()
```



```
unemplo_df = df[['States','Region','Estimated Unemployment Rate','Estimated Employed','Estimated Labour Participation Rate']]

unemplo = unemplo_df.groupby(['Region','States'])['Estimated Unemployment Rate'].mean().reset_index()


fig = px.sunburst(unemplo, path=['Region','States'], values='Estimated Unemployment Rate',
                  color_continuous_scale='Plasma',title= 'unemployment rate in each region and state',
                  height=650,template='ggplot2')


fig.show()
```



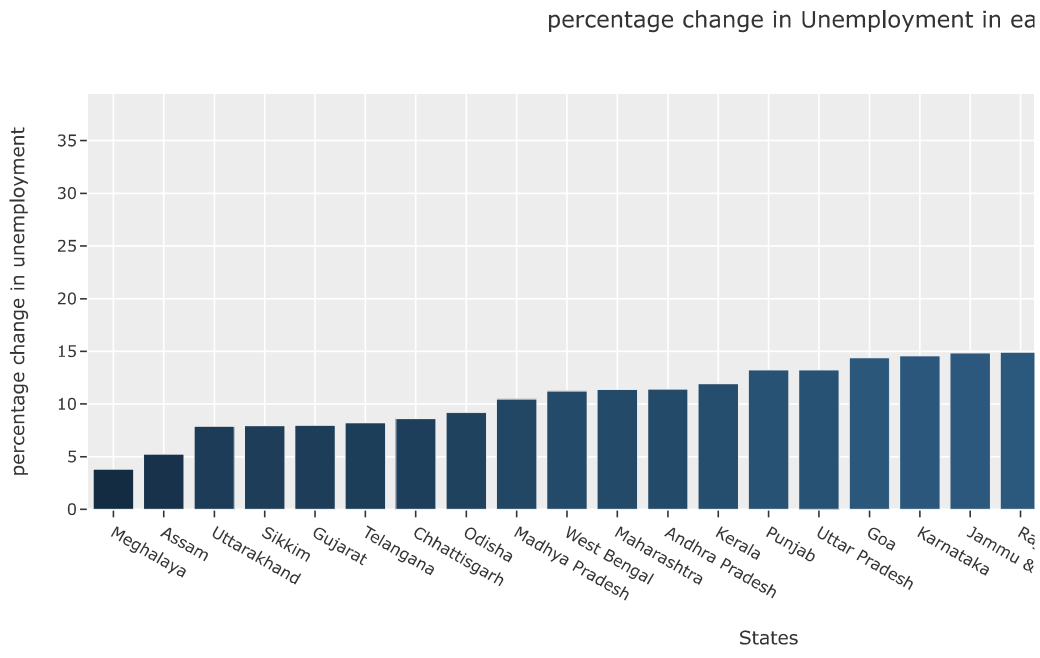
```
fig = px.scatter_geo(df,'longitude', 'latitude', color="Region",
                     hover_name="States", size="Estimated Unemployment Rate",
                     animation_frame="Month_name",scope='asia',template='plotly',title='Impack of lockdown on employement across regions')

fig.layout.updatemenus[0].buttons[0].args[1]["frame"]["duration"] = 2000

fig.update_geos(lataxis_range=[5,35], lonaxis_range=[65, 100],oceancolor="#6dd5ed",
    showocean=True)

fig.show()
```



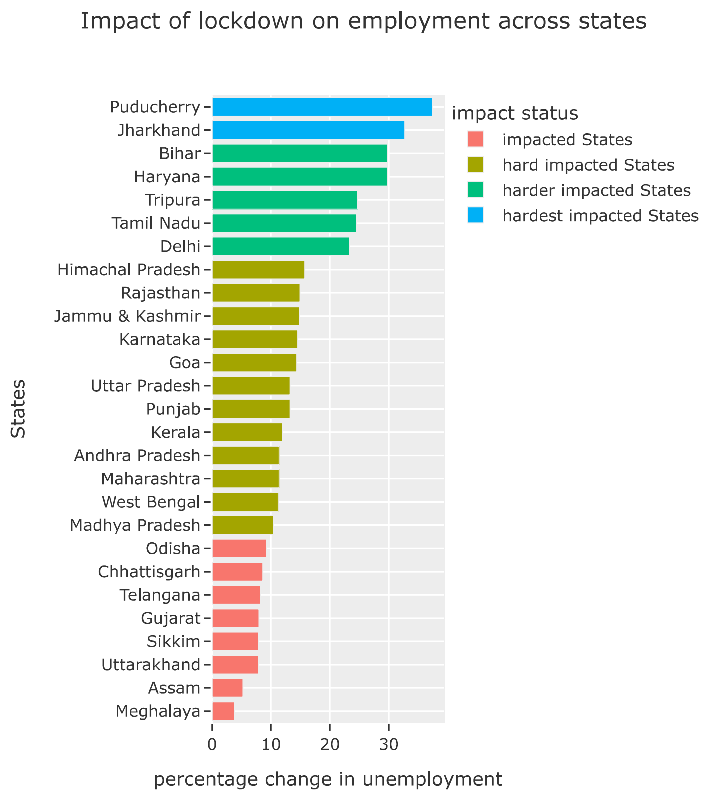
```
lock = df[(df['Month_int'] >= 4) & (df['Month_int'] <=7)]

bf_lock = df[(df['Month_int'] >= 1) & (df['Month_int'] <=4)]

g_lock = lock.groupby('States')['Estimated Unemployment Rate'].mean().reset_index()

g_bf_lock = bf_lock.groupby('States')['Estimated Unemployment Rate'].mean().reset_index()


g_lock['Unemployment Rate before lockdown'] = g_bf_lock['Estimated Unemployment Rate']

g_lock.columns = ['States','Unemployment Rate after lockdown','Unemployment Rate before lockdown']

g_lock.head(2)
```

	States	Unemployment Rate after lockdown	Unemployment Rate before lockdown
0	Andhra Pradesh	12.3975	9.4025
1	Assam	6.2450	6.2250

```
g_lock['percentage change in unemployment'] = round(g_lock['Unemployment Rate after lockdown'] - g_lock['Unemployment Rate before lockdown']/

plot_per = g_lock.sort_values('percentage change in unemployment')


fig = px.bar(plot_per, x='States',y='percentage change in unemployment',color='percentage change in unemployment',
            title='percentage change in Unemployment in each state after lockdown',template='ggplot2')

fig.show()
```



```
def sort_impact(x):
    if x <= 10:
        return 'impacted States'
    elif x <= 20:
        return 'hard impacted States'
    elif x <= 30:
        return 'harder impacted States'
    elif x <= 40:
        return 'hardest impacted States'
    return x


plot_per['impact status'] = plot_per['percentage change in unemployment'].apply(lambda x:sort_impact(x))


fig = px.bar(plot_per, y='States',x='percentage change in unemployment',color='impact status',
            title='Impact of lockdown on employment across states',template='ggplot2',height=650)


fig.show()
```



✓ 0s completed at 11:38 PM