In [1]:
```
# importing important libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
import calendar
import plotly.graph_objects as go
%matplotlib inline
```

In [2]:
```
df=pd.read_csv('C:/Users/dell/Downloads/dataset/unemployment/Unemployment_Rate_upto_11_2020.csv')
```

In [3]:
```
df
```

Out[3]:

	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	79.740
1	Andhra Pradesh	29-02-2020	M	5.83	16545652	40.90	South	15.9129	79.740
2	Andhra Pradesh	31-03-2020	M	5.79	15881197	39.18	South	15.9129	79.740
3	Andhra Pradesh	30-04-2020	M	20.51	11336911	33.10	South	15.9129	79.740
4	Andhra Pradesh	31-05-2020	M	17.43	12988845	36.46	South	15.9129	79.740
...	...	...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310	40.39	East	22.9868	87.855
263	West Bengal	31-07-2020	M	6.83	35372506	46.17	East	22.9868	87.855
264	West Bengal	31-08-2020	M	14.87	33298644	47.48	East	22.9868	87.855
265	West Bengal	30-09-2020	M	9.35	35707239	47.73	East	22.9868	87.855
266	West Bengal	31-10-2020	M	9.98	33962549	45.63	East	22.9868	87.855

267 rows × 9 columns

In [4]:
```
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 [5]:
```
df.columns=['state','date','frequency','estimated unemployment rate','estimated employed','estimated labour participation rat
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['state', 'date', 'frequency', 'estimated unemployment rate',
       'estimated employed', 'estimated labour participation rate', 'region',
       'longitude', 'latitude'],
      dtype='object')
```

In [7]:
```
df.describe()
```
Out[7]:

	estimated unemployment rate	estimated employed	estimated labour participation rate	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

In [8]:
```
df.isnull().sum()
```
Out[8]:
```
state                                  0
date                                   0
frequency                              0
estimated unemployment rate            0
estimated employed                     0
estimated labour participation rate    0
region                                 0
longitude                              0
latitude                               0
dtype: int64
```

In [9]:
```
df.state.value_counts()
```
Out[9]:
```
Jharkhand           10
Tripura             10
Uttar Pradesh       10
Himachal Pradesh    10
Karnataka           10
Haryana             10
Assam               10
Telangana           10
Punjab              10
West Bengal         10
Rajasthan           10
Meghalaya           10
Uttarakhand         10
Goa                 10
Delhi               10
Puducherry          10
Madhya Pradesh      10
Maharashtra         10
Odisha              10
Chhattisgarh        10
Tamil Nadu          10
Kerala              10
Bihar               10
Gujarat             10
Andhra Pradesh      10
Jammu & Kashmir      9
Sikkim               8
Name: state, dtype: int64
```

In [10]:
```
# changing the datatype of 'data' from object to datetime
df['date']=pd.to_datetime(df['date'],dayfirst=True)
df.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 267 entries, 0 to 266
Data columns (total 9 columns):
 #   Column                               Non-Null Count  Dtype         
---  ------                               --------------  -----         
 0   state                                267 non-null    object        
 1   date                                 267 non-null    datetime64[ns]
 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                               267 non-null    object        
 7   longitude                            267 non-null    float64       
 8   latitude                             267 non-null    float64       
dtypes: datetime64[ns](1), float64(4), int64(1), object(3)
memory usage: 18.9+ KB
```

In [11]:
```
# Extracting month from date attribute
df['month_int']=df['date'].dt.month
df
```

Out[11]:

	state	date	frequency	estimated unemployment rate	estimated employed	estimated labour participation rate	region	longitude	latitude	month_int
0	Andhra Pradesh	2020-01-31	M	5.48	16635535	41.02	South	15.9129	79.740	1
1	Andhra Pradesh	2020-02-29	M	5.83	16545652	40.90	South	15.9129	79.740	2
2	Andhra Pradesh	2020-03-31	M	5.79	15881197	39.18	South	15.9129	79.740	3
3	Andhra Pradesh	2020-04-30	M	20.51	11336911	33.10	South	15.9129	79.740	4
4	Andhra Pradesh	2020-05-31	M	17.43	12988845	36.46	South	15.9129	79.740	5
...	...	...	...	...	...	...	...	...	...	...
262	West Bengal	2020-06-30	M	7.29	30726310	40.39	East	22.9868	87.855	6
263	West Bengal	2020-07-31	M	6.83	35372506	46.17	East	22.9868	87.855	7
264	West Bengal	2020-08-31	M	14.87	33298644	47.48	East	22.9868	87.855	8
265	West Bengal	2020-09-30	M	9.35	35707239	47.73	East	22.9868	87.855	9
266	West Bengal	2020-10-31	M	9.98	33962549	45.63	East	22.9868	87.855	10

267 rows × 10 columns

In [12]:
```
# The months are in integer datetype. We need to convert the months into words for better analysis
df['month']=df['month_int'].apply(lambda x: calendar.month_abbr[x])
df
```

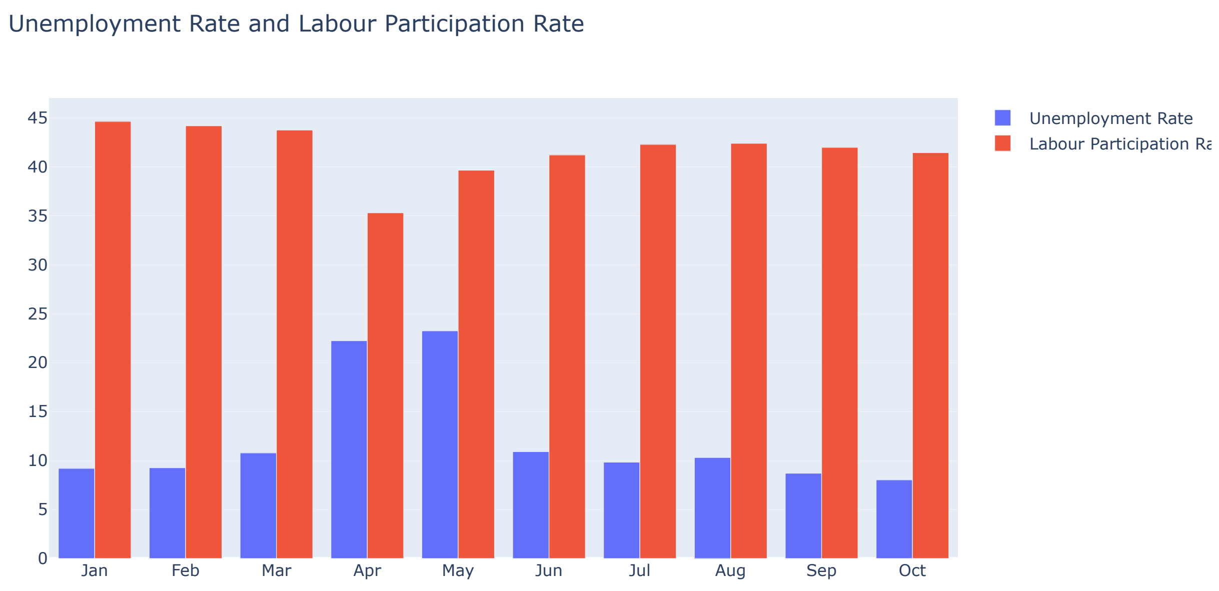
Out[12]:

	state	date	frequency	estimated unemployment rate	estimated employed	estimated labour participation rate	region	longitude	latitude	month_int	month
0	Andhra Pradesh	2020-01-31	M	5.48	16635535	41.02	South	15.9129	79.740	1	Jan
1	Andhra Pradesh	2020-02-29	M	5.83	16545652	40.90	South	15.9129	79.740	2	Feb
2	Andhra Pradesh	2020-03-31	M	5.79	15881197	39.18	South	15.9129	79.740	3	Mar
3	Andhra Pradesh	2020-04-30	M	20.51	11336911	33.10	South	15.9129	79.740	4	Apr
4	Andhra Pradesh	2020-05-31	M	17.43	12988845	36.46	South	15.9129	79.740	5	May
...	...	...	...	...	...	...	...	...	...	...	...
262	West Bengal	2020-06-30	M	7.29	30726310	40.39	East	22.9868	87.855	6	Jun
263	West Bengal	2020-07-31	M	6.83	35372506	46.17	East	22.9868	87.855	7	Jul
264	West Bengal	2020-08-31	M	14.87	33298644	47.48	East	22.9868	87.855	8	Aug
265	West Bengal	2020-09-30	M	9.35	35707239	47.73	East	22.9868	87.855	9	Sep
266	West Bengal	2020-10-31	M	9.98	33962549	45.63	East	22.9868	87.855	10	Oct

267 rows × 11 columns

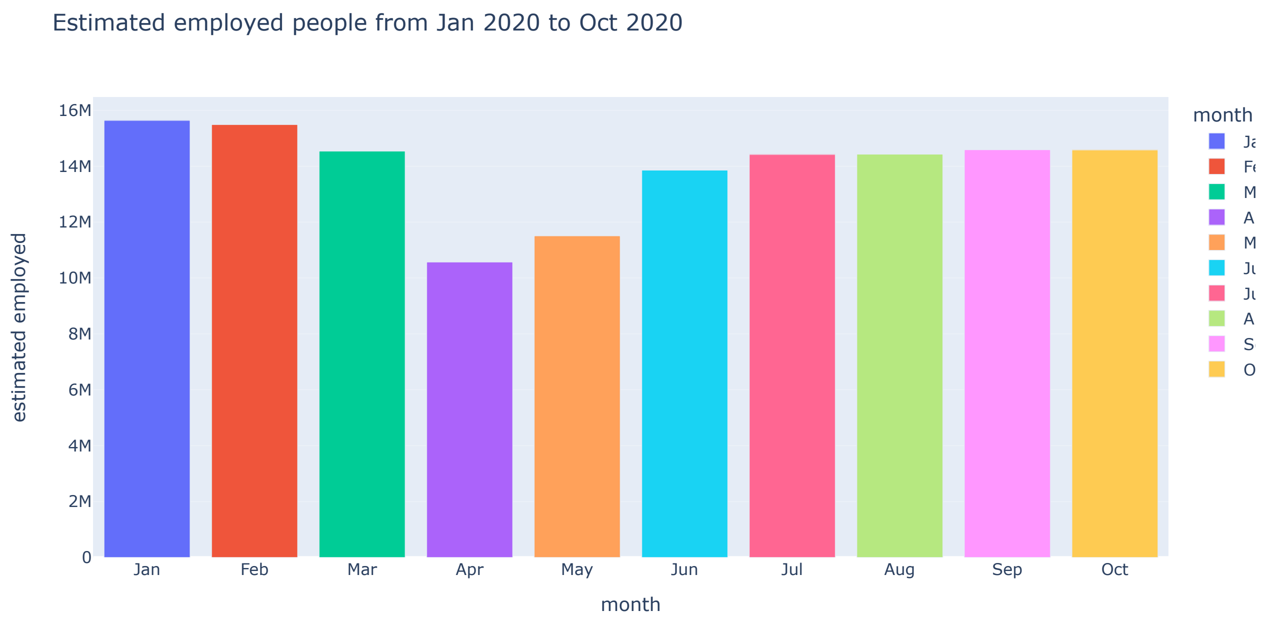
In [13]:
```
# Numeric data grouped by months
data=df.groupby(['month'])[['estimated unemployment rate','estimated employed','estimated labour participation rate']].mean(
data=pd.DataFrame(data).reset_index()
```

In [14]:
```python
# Bar plot of umemployment rate and labour participation rate
month=data.month
unemployment_rate=data['estimated unemployment rate']
labour_participation_rate=data['estimated labour participation rate']

fig=go.Figure()

fig.add_trace(go.Bar(x=month,y=unemployment_rate,name='Unemployment Rate'))
fig.add_trace(go.Bar(x=month,y=labour_participation_rate,name='Labour Participation Rate'))

fig.update_layout(title='Unemployment Rate and Labour Participation Rate',xaxis={'categoryorder':'array','categoryarray':['Ja
fig.show()
```



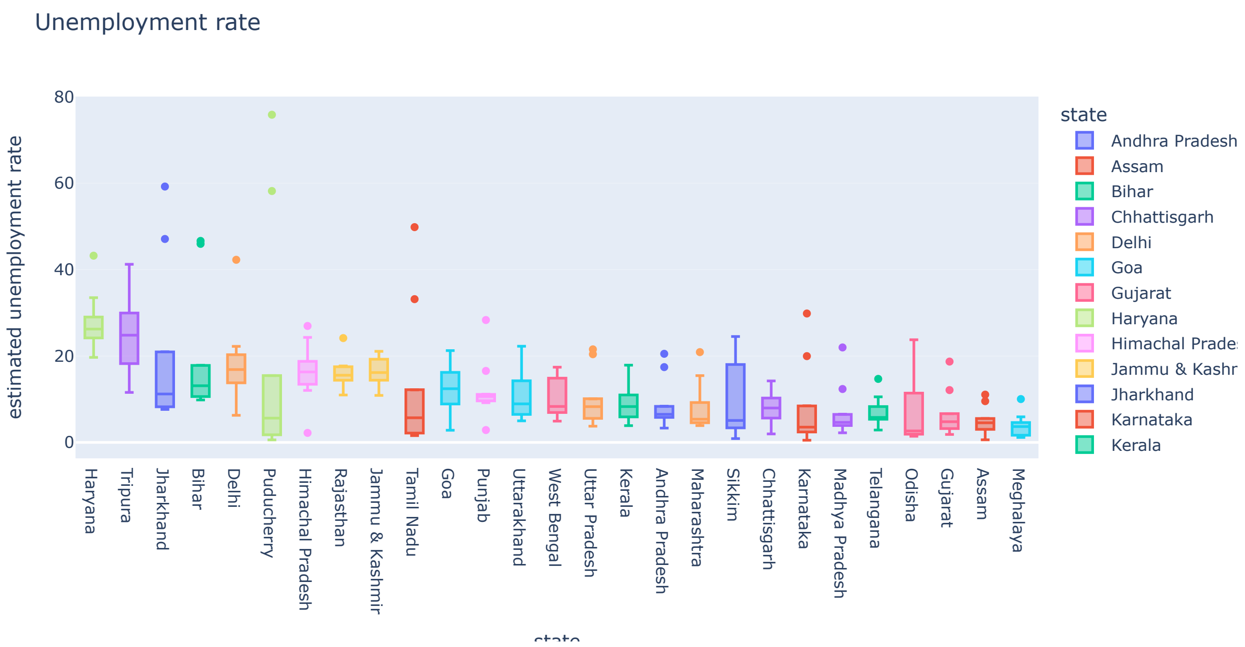
In [15]:
```python
import plotly.express as px
```

In [16]:
```python
fig=px.bar(data,x='month',y='estimated employed',color='month',category_orders={'month':['Jan','Feb','Mar','Apr','May','Jun',
fig.show()
```

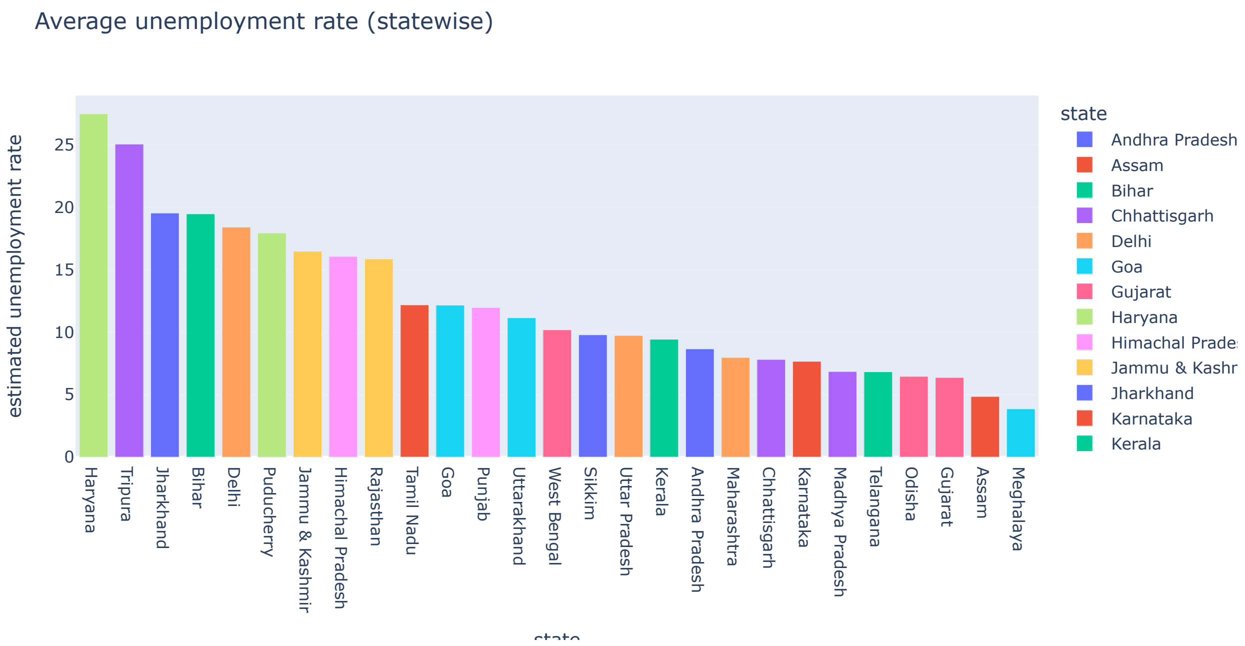


```
In [17]: # now comes state wise analysis
         state=df.groupby(['state'])[['estimated unemployment rate','estimated employed','estimated labour participation rate']].mean(
         state=pd.DataFrame(state).reset_index()
```

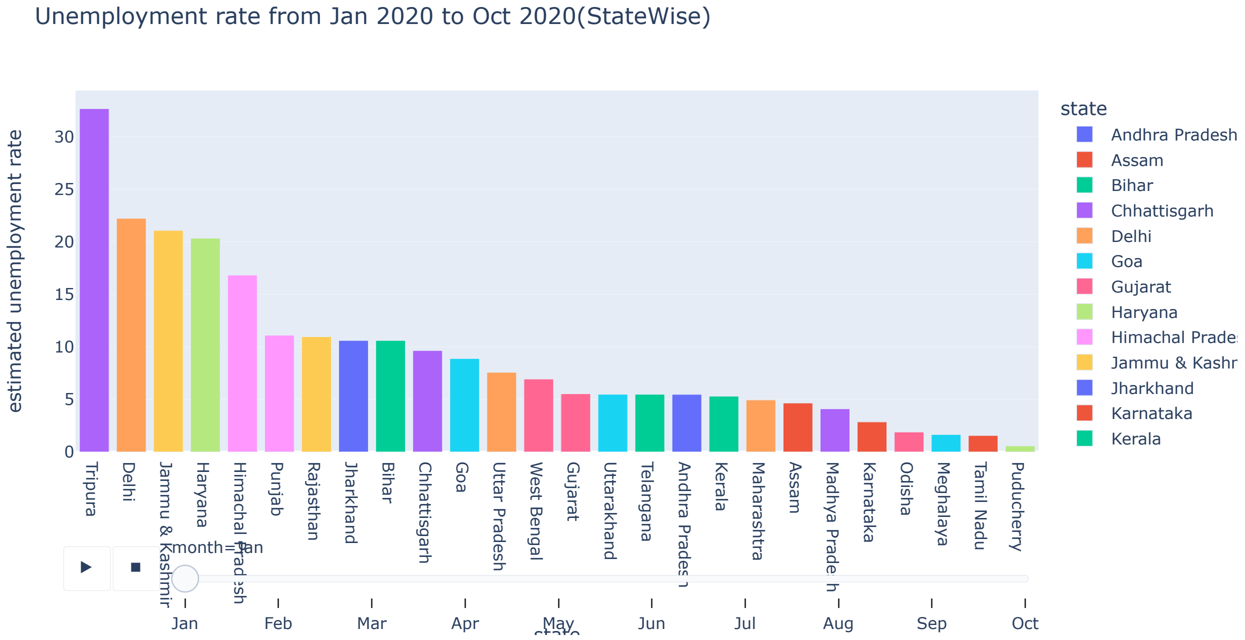
```
In [18]: # box plot
         fig=px.box(data_frame=df,x='state',y='estimated unemployment rate',color='state',title='Unemployment rate')
         fig.update_layout(xaxis={'categoryorder':'total descending'})
         fig.show()
```



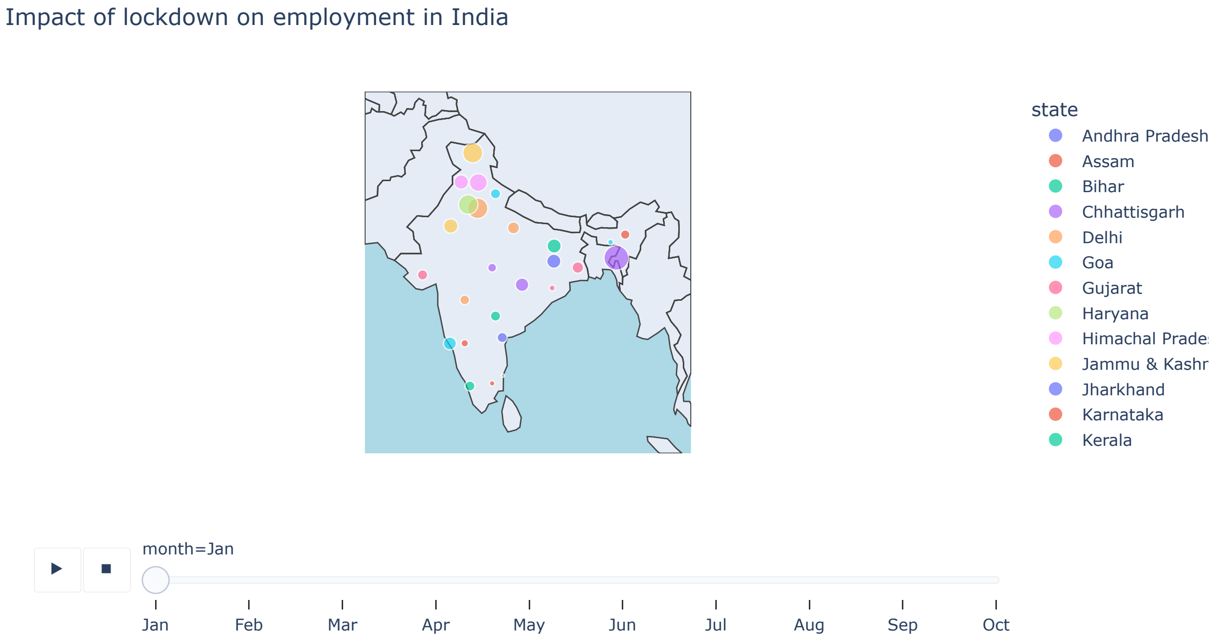
```
In [19]: # average unemployment rate bar plot
         fig=px.bar(state,x='state',y='estimated unemployment rate',color='state',title='Average unemployment rate (statewise)')
         fig.update_layout(xaxis={'categoryorder':'total descending'})
         fig.show()
```



```
In [20]: fig = px.bar(df,x='state',y='estimated unemployment rate',animation_frame='month',color='state',
                  title='Unemployment rate from Jan 2020 to Oct 2020(StateWise)')

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



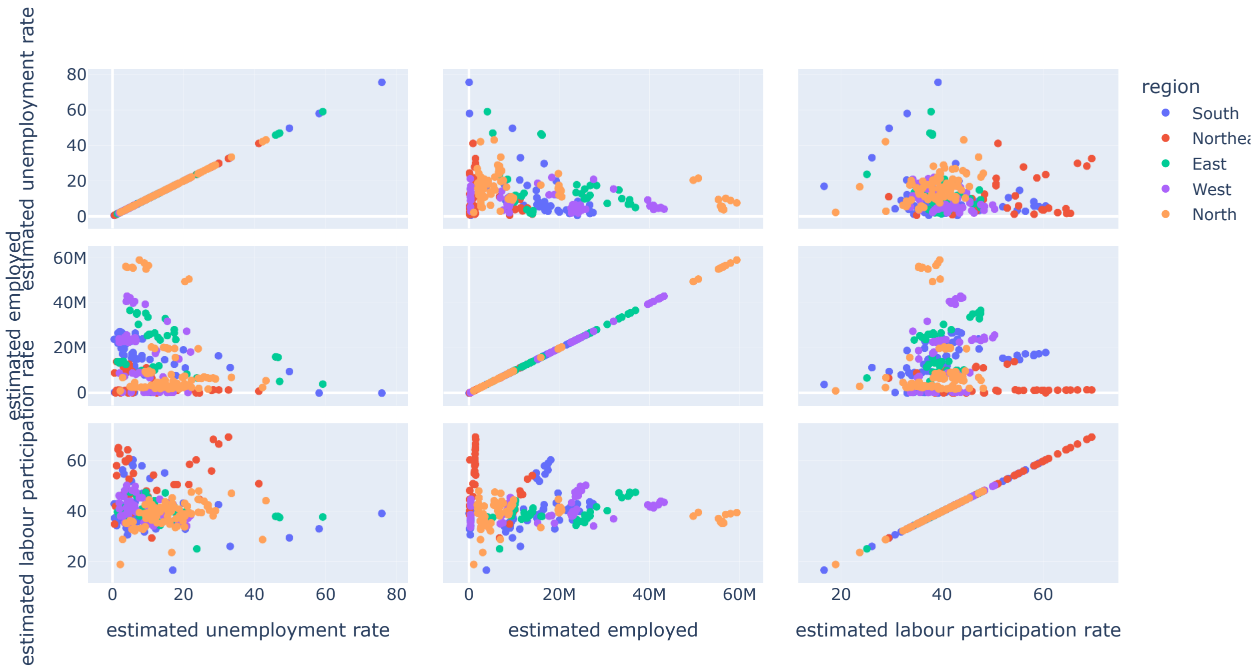
```
In [21]: fig=px.scatter_geo(df,'longitude','latitude',color='state',
                            hover_name='state',size='estimated unemployment rate',
                            animation_frame='month',scope='asia',title='Impact of lockdown on employment in India')

         fig.layout.updatemenus[0].buttons[0].args[1]['frame']['duration'] =2000
         fig.update_geos(lataxis_range=[5,40],lonaxis_range=[65,100],oceancolor='lightblue',
                         showocean=True)

         fig.show()
```



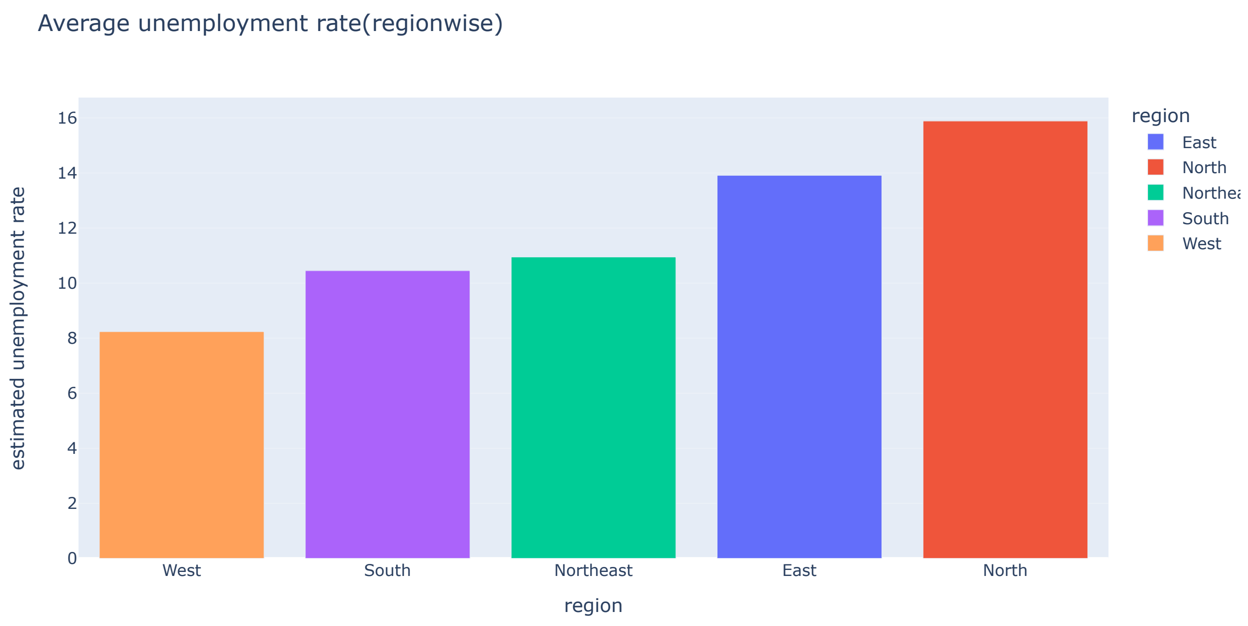
```
In [22]: df.region.unique()

Out[22]: array(['South', 'Northeast', 'East', 'West', 'North'], dtype=object)
```

In [23]:
```
# numeric data grouped by region

region = df.groupby(['region'])[['estimated unemployment rate','estimated employed','estimated labour participation rate']].r
region = pd.DataFrame(region).reset_index()
```

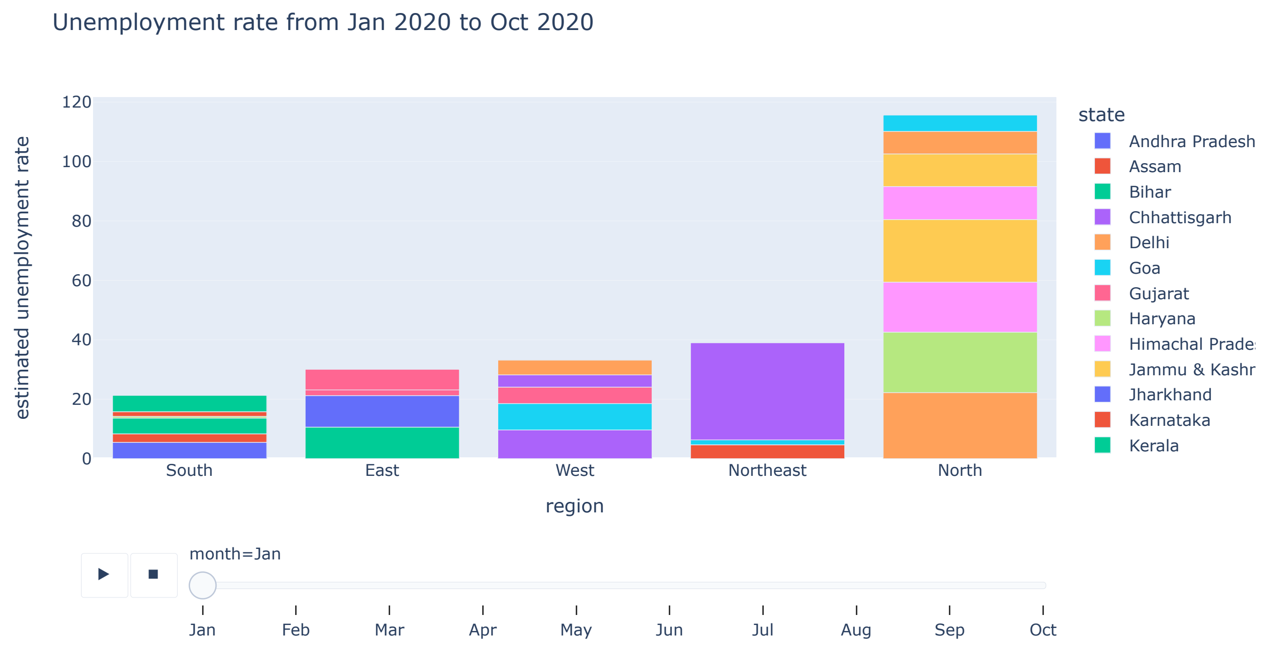
In [24]:
```
#Scatter plot

fig= px.scatter_matrix(df,dimensions=['estimated unemployment rate','estimated employed','estimated labour participation rate
fig.show()
```



In [25]:
```
# Average Unemployment Rate

fig = px.bar(region,x='region',y='estimated unemployment rate',color='region',title='Average unemployment rate(regionwise)')
fig.update_layout(xaxis={'categoryorder':'total ascending'})
fig.show()
```



```
In [26]: fig = px.bar(df,x='region',y='estimated unemployment rate',animation_frame='month',color='state',
                    title='Unemployment rate from Jan 2020 to Oct 2020')

         fig.update_layout(xaxis={'categoryorder':'total ascending'})
         fig.layout.updatemenus[0].buttons[0].args[1]['frame']['duration'] =2000

         fig.show()
```

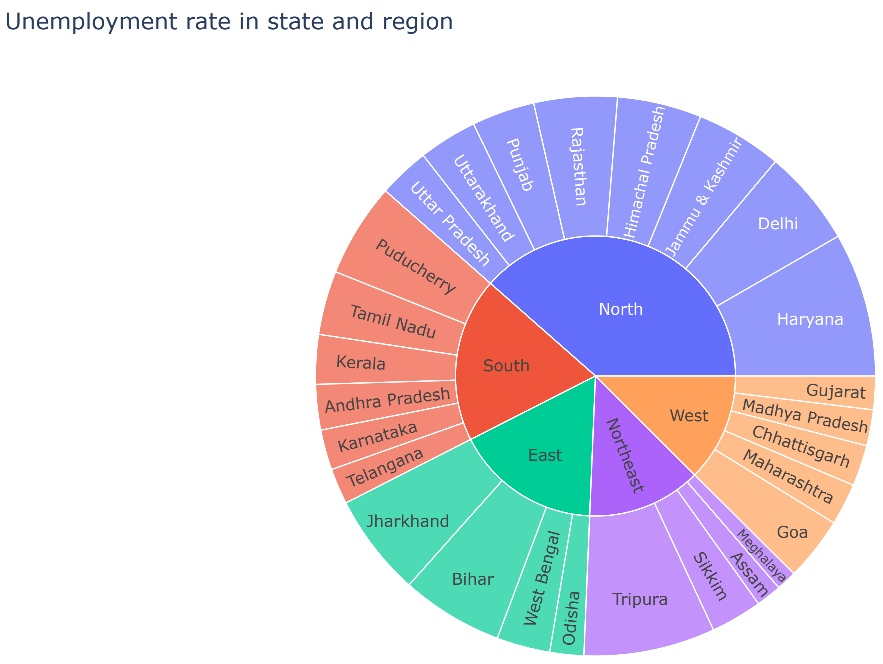


```
In [27]: unemployment =df.groupby(['region','state'])['estimated unemployment rate'].mean().reset_index()
         unemployment.head()
```

Out[27]:

	region	state	estimated unemployment rate
0	East	Bihar	19.471
1	East	Jharkhand	19.539
2	East	Odisha	6.462
3	East	West Bengal	10.192
4	North	Delhi	18.414

In [28]:
```
fig = px.sunburst(unemployment,path=['region','state'],values='estimated unemployment rate',
                  title ='Unemployment rate in state and region',height=600)
fig.show()
```



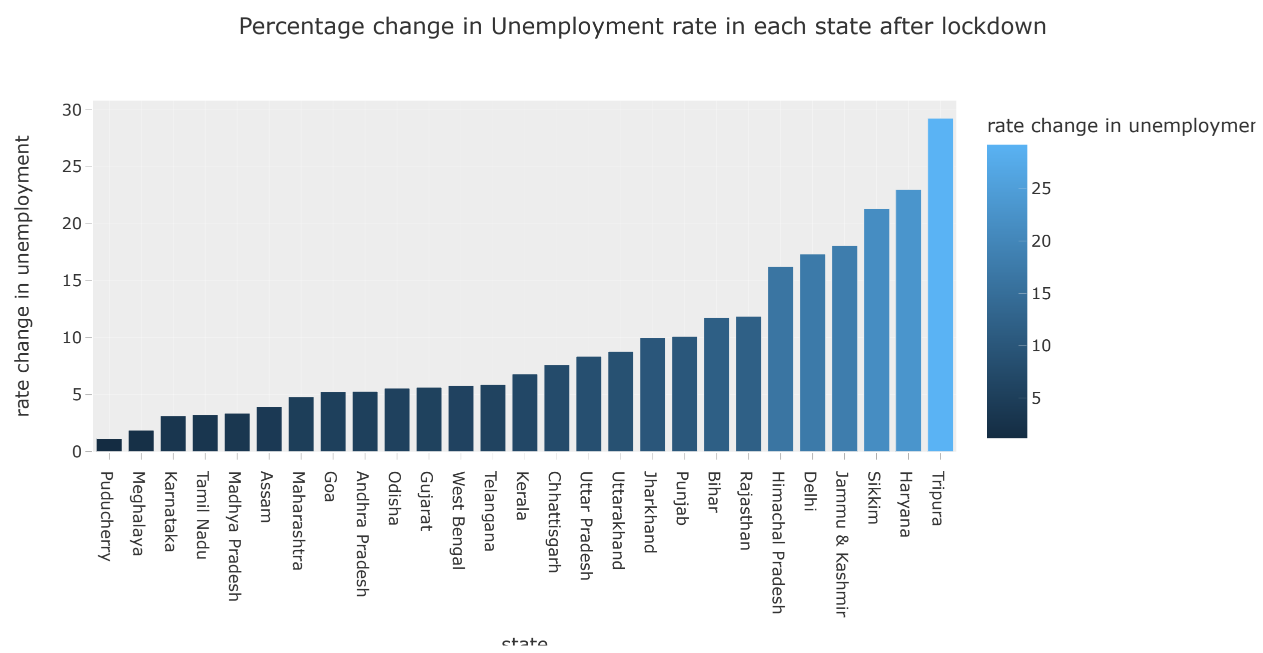
In [29]:
```
# data representation before and after lockdown

before_lockdown = df[(df['month_int']>=1) &(df['month_int'] <4)]
after_lockdown = df[(df['month_int'] >=4) & (df['month_int'] <=6)]
```

In [30]:
```
af_lockdown = after_lockdown.groupby('state')['estimated unemployment rate'].mean().reset_index()

lockdown = before_lockdown.groupby('state')['estimated unemployment rate'].mean().reset_index()
lockdown['unemployment rate before lockdown'] = af_lockdown['estimated unemployment rate']

lockdown.columns = ['state','unemployment rate before lockdown','unemployment rate after lockdown']
lockdown.head()
```

Out[30]:

	state	unemployment rate before lockdown	unemployment rate after lockdown
0	Andhra Pradesh	5.700000	13.750000
1	Assam	4.613333	7.070000
2	Bihar	12.110000	36.806667
3	Chhattisgarh	8.523333	9.380000
4	Delhi	18.036667	25.713333

In [31]:
```
# unenplopyment rate change after lockdown

lockdown['rate change in unemployment'] =round(lockdown['unemployment rate before lockdown']-lockdown['unemployment rate befo
                                              /lockdown['unemployment rate after lockdown'],2)
fig = px.bar(lockdown,x='state',y='rate change in unemployment',color='rate change in unemployment',
            title='Percentage change in Unemployment rate in each state after lockdown',template='ggplot2')
fig.update_layout(xaxis={'categoryorder':'total ascending'})
fig.show()
```



In [ ]: