Unemployment in India

Context

The story behind this datasets is how lock-down affects employment opportunities and how the unemployment rate increases during the Covid-19. Content

This dataset contains the unemployment rate of all the states in India **bold text**

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

1.Problem Statement

Unemployment is measured by the unemployment rate which is the number of people who are unemployed as a percentage of the total labour force. We have seen a sharp increase in the unemployment rate during Covid-19. So, the analysis intends to shed light on the socio-economic consequences of the pandemic on India's workforce and labor market.

This dataset aids in comprehending the unemployment dynamics across India's states during the COVID-19 crisis. It offers valuable insights into how the unemployment rate, employment figures, and labor participation rates have been impacted across different regions in the country.

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
df = pd.read_csv('/content/Unemployment in India.csv')
df
```

Out [4]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...	...
763	NaN	NaN	NaN	NaN	NaN	NaN	NaN
764	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
765	NaN	NaN	NaN	NaN	NaN	NaN	NaN
766	NaN	NaN	NaN	NaN	NaN	NaN	NaN
767	NaN	NaN	NaN	NaN	NaN	NaN	NaN

768 rows × 7 columns

In [ ]:
```
df.head()
```

Out [5]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural

In [ ]:
```
df.tail()
```

Out [6]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
763	NaN	NaN	NaN	NaN	NaN	NaN	NaN
764	NaN	NaN	NaN	NaN	NaN	NaN	NaN
765	NaN	NaN	NaN	NaN	NaN	NaN	NaN
766	NaN	NaN	NaN	NaN	NaN	NaN	NaN
767	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [ ]:
```
df.describe
```

Out [7]:
```
<bound method NDFrame.describe of             Region         Date  Frequency   Estimated Unemployment
Rate (%)  \
0    Andhra Pradesh   31-05-2019    Monthly                           3.65
1    Andhra Pradesh   30-06-2019    Monthly                           3.05
2    Andhra Pradesh   31-07-2019    Monthly                           3.75
3    Andhra Pradesh   31-08-2019    Monthly                           3.32
4    Andhra Pradesh   30-09-2019    Monthly                           5.17
..              ...          ...        ...                            ...
763             NaN          NaN        NaN                            NaN
764             NaN          NaN        NaN                            NaN
765             NaN          NaN        NaN                            NaN
766             NaN          NaN        NaN                            NaN
767             NaN          NaN        NaN                            NaN

      Estimated Employed   Estimated Labour Participation Rate (%)   Area
0             11999139.0                                    43.24  Rural
1             11755881.0                                    42.05  Rural
2             12086707.0                                    43.50  Rural
3             12285693.0                                    43.97  Rural
4             12256762.0                                    44.68  Rural
..                   ...                                      ...    ...
763                  NaN                                      NaN    NaN
764                  NaN                                      NaN    NaN
765                  NaN                                      NaN    NaN
```

```
766                  NaN                                    NaN    NaN
767                  NaN                                    NaN    NaN

[768 rows x 7 columns]>
```

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 7 columns):
 #   Column                                   Non-Null Count  Dtype
---  ------                                   --------------  -----
 0   Region                                   740 non-null    object
 1    Date                                    740 non-null    object
 2    Frequency                               740 non-null    object
 3    Estimated Unemployment Rate (%)         740 non-null    float64
 4    Estimated Employed                      740 non-null    float64
 5    Estimated Labour Participation Rate (%) 740 non-null    float64
 6   Area                                     740 non-null    object
dtypes: float64(3), object(4)
memory usage: 42.1+ KB
```

In [ ]:
```
df.shape
```

Out [9]:
```
(768, 7)
```

In [ ]:
```
df.isna().sum()
```

Out [10]:
```
Region                                   28
 Date                                    28
 Frequency                               28
 Estimated Unemployment Rate (%)         28
 Estimated Employed                      28
 Estimated Labour Participation Rate (%) 28
Area                                     28
dtype: int64
```

Exploratory data analysis

In [ ]:
```
df = df.drop_duplicates() #removing duplicates
df.shape
```

Out [11]:
```
(741, 7)
```

In [ ]:
```
df.dtypes
```

Out [12]:
```
Region                                    object
 Date                                     object
 Frequency                                object
 Estimated Unemployment Rate (%)         float64
 Estimated Employed                      float64
 Estimated Labour Participation Rate (%) float64
Area                                      object
dtype: object
```

In [ ]:
```
df['Area'].value_counts()
```

Out [13]:
```
Urban    381
Rural    359
Name: Area, dtype: int64
```

In [ ]:
```
df['Region'].value_counts()
```

Out [14]:
```
Andhra Pradesh      28
Kerala              28
West Bengal         28
Uttar Pradesh       28
Tripura             28
Telangana           28
Tamil Nadu          28
Rajasthan           28
Punjab              28
```

```
Odisha              28
Madhya Pradesh      28
Maharashtra         28
Karnataka           28
Jharkhand           28
Himachal Pradesh    28
Haryana             28
Gujarat             28
Delhi               28
Chhattisgarh        28
Bihar               28
Meghalaya           27
Uttarakhand         27
Assam               26
Puducherry          26
Goa                 24
Jammu & Kashmir     21
Sikkim              17
Chandigarh          12
Name: Region, dtype: int64
```

In [ ]:
```
df[' Frequency'].value_counts()
```

Out [15]:
```
Monthly     381
 Monthly    359
Name:  Frequency, dtype: int64
```

In [ ]:
```
df[' Estimated Employed'].value_counts()
```

Out [16]:
```
11999139.0    1
1183770.0     1
241366.0      1
246596.0      1
227804.0      1
             ..
6021921.0     1
6395022.0     1
6164215.0     1
6189471.0     1
9088931.0     1
Name:  Estimated Employed, Length: 740, dtype: int64
```

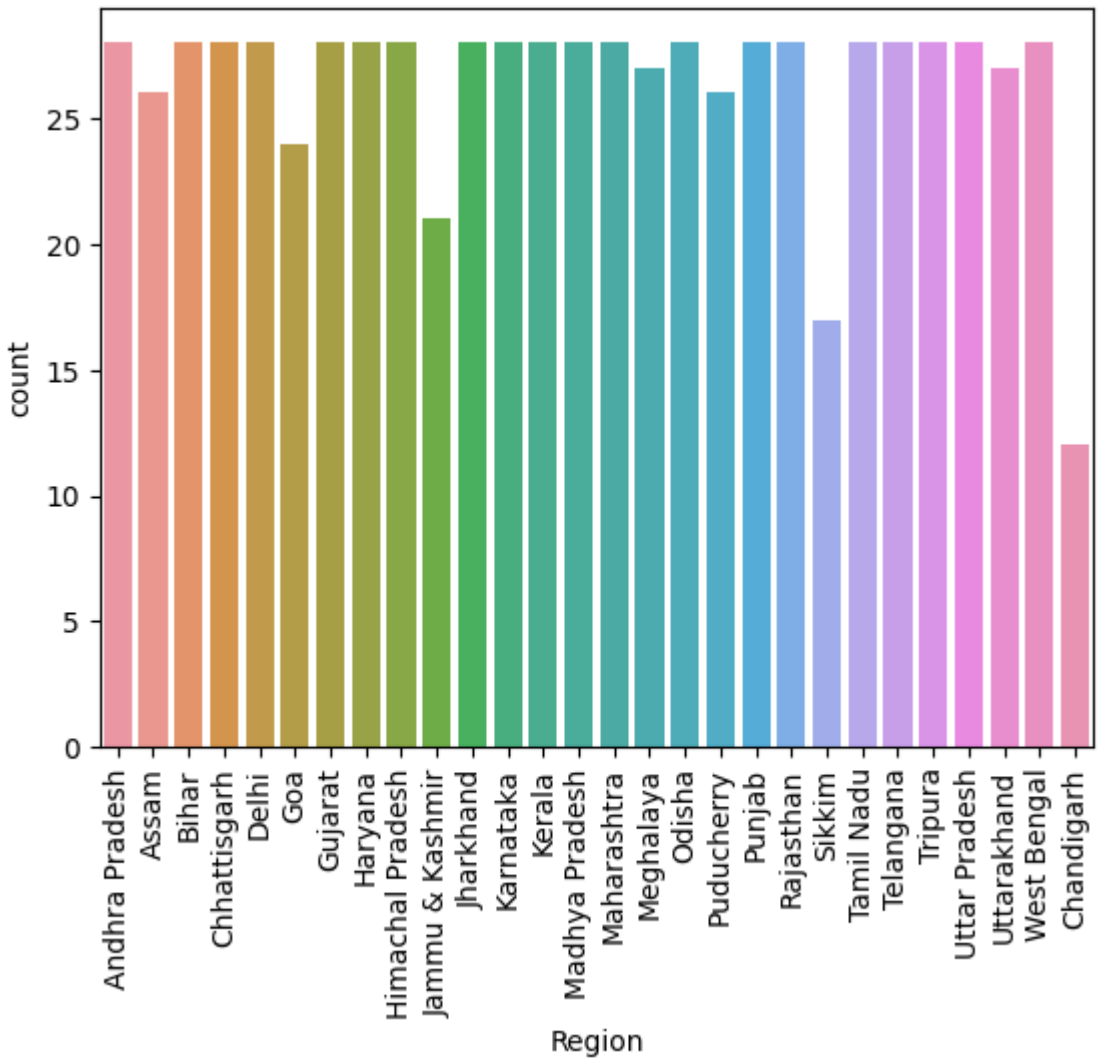
In [ ]:
```
df[' Date'].value_counts
```

Out [17]:
```
<bound method IndexOpsMixin.value_counts of 0       31-05-2019
1       30-06-2019
2       31-07-2019
3       31-08-2019
4       30-09-2019
          ...
749     29-02-2020
750     31-03-2020
751     30-04-2020
752     31-05-2020
753     30-06-2020
Name:  Date, Length: 741, dtype: object>
```

In [ ]:
```
sns.countplot(x=df['Region'])
plt.xticks(rotation=90)
```

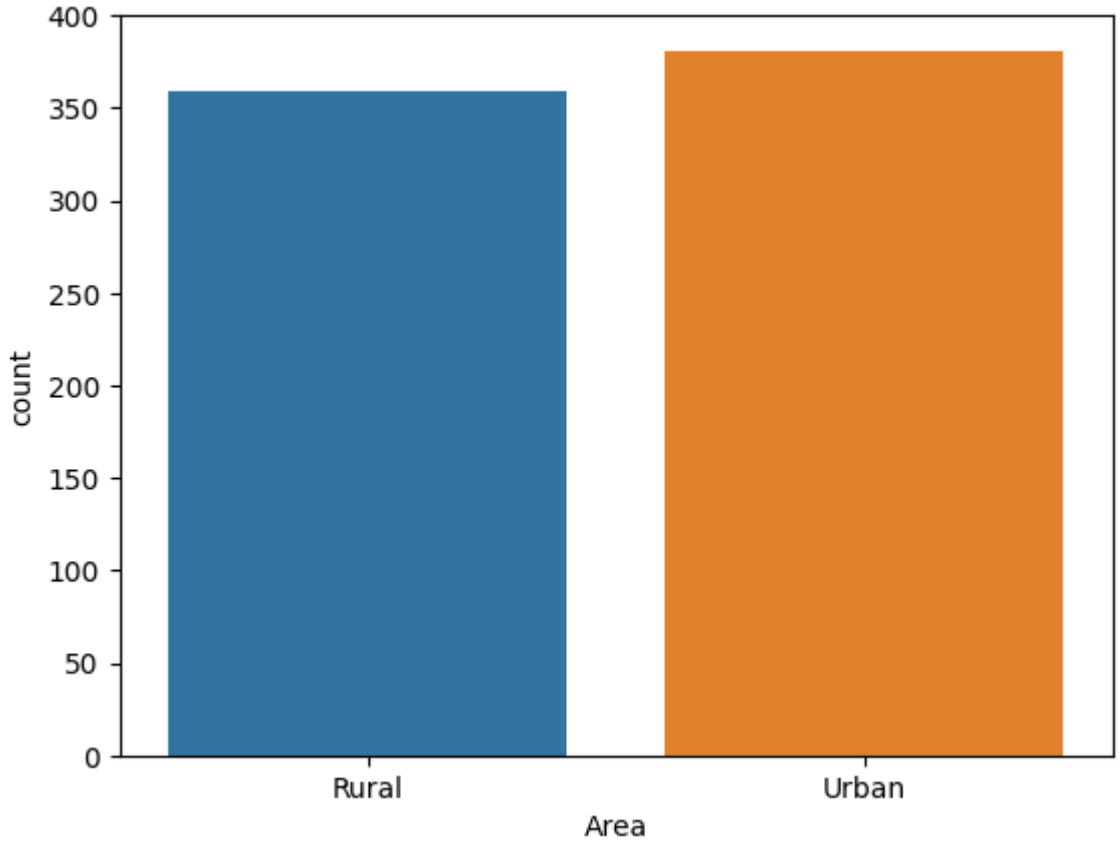
Out [18]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]),
 [Text(0, 0, 'Andhra Pradesh'),
  Text(1, 0, 'Assam'),
  Text(2, 0, 'Bihar'),
  Text(3, 0, 'Chhattisgarh'),
  Text(4, 0, 'Delhi'),
  Text(5, 0, 'Goa'),
  Text(6, 0, 'Gujarat'),
  Text(7, 0, 'Haryana'),
  Text(8, 0, 'Himachal Pradesh'),
  Text(9, 0, 'Jammu & Kashmir'),
  Text(10, 0, 'Jharkhand'),
  Text(11, 0, 'Karnataka'),
  Text(12, 0, 'Kerala'),
  Text(13, 0, 'Madhya Pradesh'),
  Text(14, 0, 'Maharashtra'),
  Text(15, 0, 'Meghalaya'),
  Text(16, 0, 'Odisha'),
  Text(17, 0, 'Puducherry'),
  Text(18, 0, 'Punjab'),
  Text(19, 0, 'Rajasthan'),
  Text(20, 0, 'Sikkim'),
  Text(21, 0, 'Tamil Nadu'),
  Text(22, 0, 'Telangana'),
```

```
Text(23, 0, 'Tripura'),
Text(24, 0, 'Uttar Pradesh'),
Text(25, 0, 'Uttarakhand'),
Text(26, 0, 'West Bengal'),
Text(27, 0, 'Chandigarh')])
```



In [ ]:
```
sns.countplot(x=df['Area'])
```

Out [19]: <Axes: xlabel='Area', ylabel='count'>



In [ ]:
```
df = df.drop([' Frequency'],axis = 1)
```

In [ ]:
```
df
```

Out [21]:

	Region	Date	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...
749	West Bengal	29-02-2020	7.55	10871168.0	44.09	Urban
750	West Bengal	31-03-2020	6.67	10806105.0	43.34	Urban
751	West Bengal	30-04-2020	15.63	9299466.0	41.20	Urban
752	West Bengal	31-05-2020	15.22	9240903.0	40.67	Urban
753	West Bengal	30-06-2020	9.86	9088931.0	37.57	Urban

741 rows × 6 columns

In [ ]:
```
# Renaming columns for easier access
df1= df.rename(columns={ ' Estimated Unemployment Rate (%)' : 'est_unemp_perc', ' E
                         ' Estimated Labour Participation Rate (%)' : 'est_labour_
```

In [ ]:
```
df1
```

Out [23]:

	Region	Date	est_unemp_perc	est_mil_emp	est_labour_perc	Area
0	Andhra Pradesh	31-05-2019	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...
736	West Bengal	29-02-2020	7.55	10871168.0	44.09	Urban
737	West Bengal	31-03-2020	6.67	10806105.0	43.34	Urban
738	West Bengal	30-04-2020	15.63	9299466.0	41.20	Urban
739	West Bengal	31-05-2020	15.22	9240903.0	40.67	Urban
740	West Bengal	30-06-2020	9.86	9088931.0	37.57	Urban

741 rows × 6 columns

In [ ]:
```
df1.isna().sum()
```

Out [24]:
```
Region            1
 Date             1
est_unemp_perc    1
est_mil_emp       1
est_labour_perc   1
Area              1
dtype: int64
```

In [ ]:
```
df1[' Date'] = pd.to_datetime(df1[' Date'])
```

In [ ]:
```
import datetime as dt  #commonly used for working with dates and times.
```

In [ ]:
```
# df1['year']=df1[' Date'].dt.isocalendar().year    # extracts the year from the 'D
# df1['month']=df1[' Date'].dt.month                #extracts the month from the 'D
```

In [ ]:
```
# df1[' Date'] = pd.to_numeric(df1[' Date'], errors='coerce')
```

In [ ]:
```
df1[' Date']=df1[' Date'].fillna((df1[' Date']).mean())
df1['Region']=df1['Region'].fillna((df1['Region']).mode()[0])
df1['est_unemp_perc']=df1['est_unemp_perc'].fillna((df1['est_unemp_perc']).mean())
df1['est_mil_emp']=df1['est_mil_emp'].fillna((df1['est_mil_emp']).mean())
df1['est_labour_perc']=df1['est_labour_perc'].fillna((df1['est_labour_perc']).mean(
df1['Area']=df1['Area'].fillna((df1['Area']).mode()[0])
```

In [ ]:
```
df1.corr()
```

```
<ipython-input-30-49b3fcfeb4d1>:1: FutureWarning: The default value of numeric_only in DataFrame.corr
is deprecated. In a future version, it will default to False. Select only valid columns or specify
the value of numeric_only to silence this warning.
  df1.corr()
```

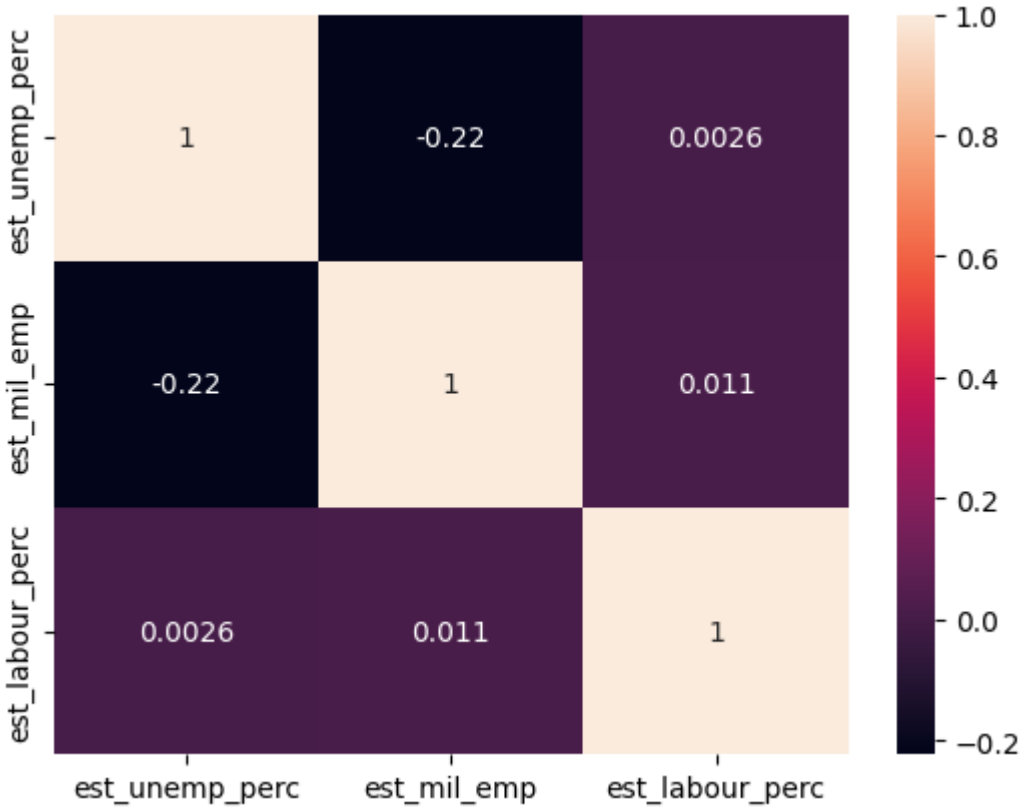
Out [30]:

	est_unemp_perc	est_mil_emp	est_labour_perc
est_unemp_perc	1.000000	-0.222876	0.002558
est_mil_emp	-0.222876	1.000000	0.011300
est_labour_perc	0.002558	0.011300	1.000000

In [ ]:
```
sns.heatmap(df1.corr(),annot= True)
```

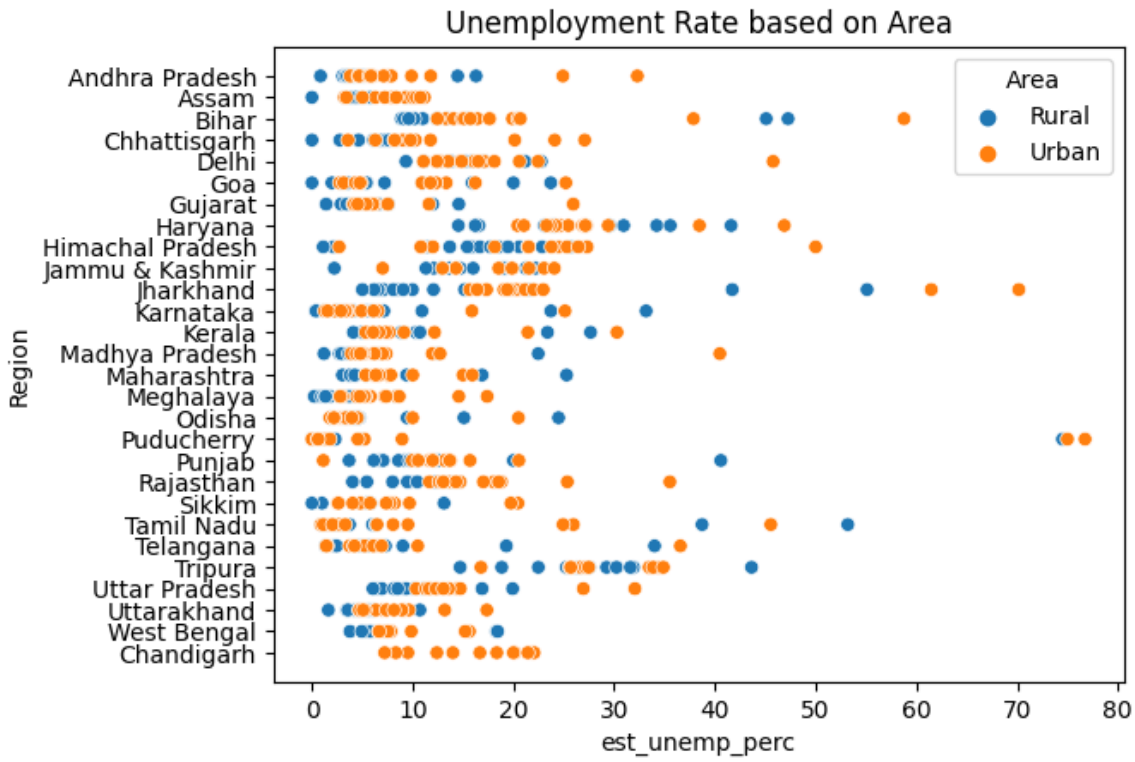
```
<ipython-input-38-729c09d24423>:1: FutureWarning: The default value of numeric_only in DataFrame.corr
is deprecated. In a future version, it will default to False. Select only valid columns or specify
the value of numeric_only to silence this warning.
  sns.heatmap(df1.corr(),annot= True)
```

Out [38]: <Axes: >



In [ ]:
```
plt.title('Unemployment Rate based on Area')
sns.scatterplot(y=df1['Region'],x=df1['est_unemp_perc'],hue=df1['Area'])
```

Out [32]: <Axes: title={'center': 'Unemployment Rate based on Area'}, xlabel='est_unemp_perc', ylabel='Region'>

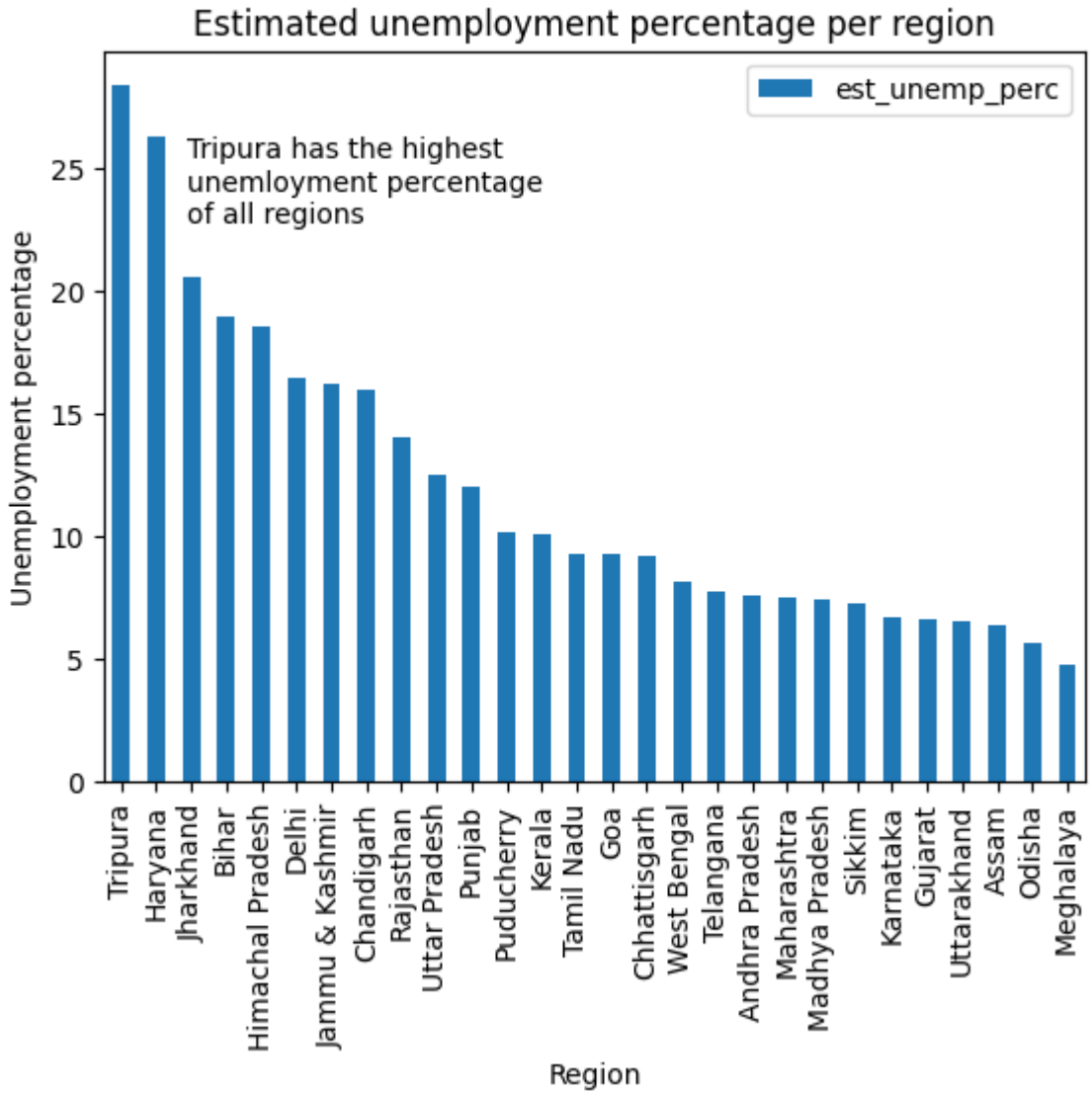


Data visualization

In [ ]:
```
df1[' Date'] = pd.to_datetime(df1[' Date'])
#converted the 'Date' column to a datetime
```

italicized text Discovering the differences in unemployment percentages among regions should tell us the most affected region

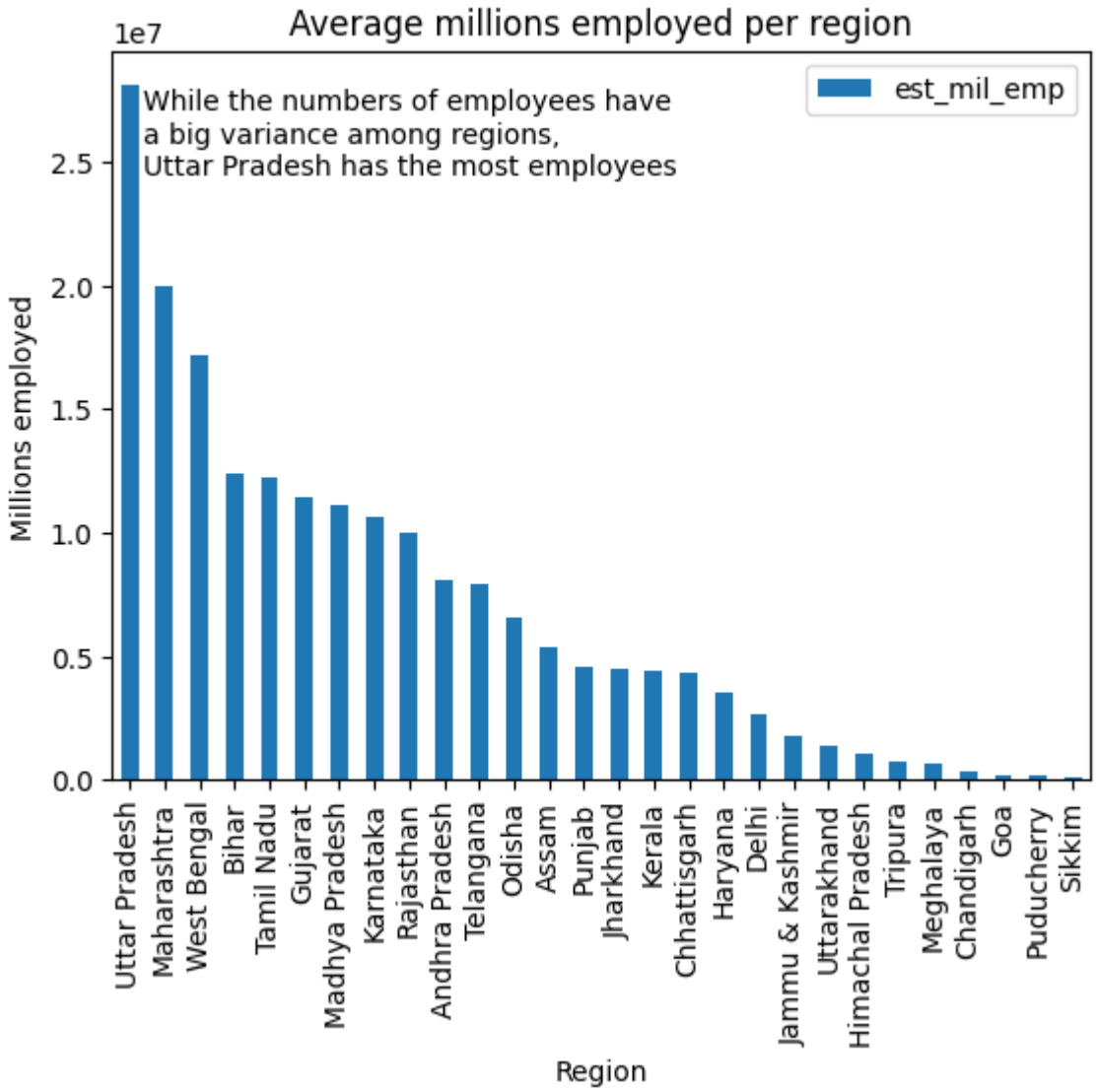
In [ ]:
```
df2 = df1.groupby('Region')[['est_unemp_perc']].mean().sort_values(by='est_unemp_pe
df2.plot(kind='bar')             #used to create a bar plot of the data in a DataFr
plt.title('Estimated unemployment percentage per region')
plt.ylabel('Unemployment percentage')
plt.xticks(rotation= 90)
plt.figtext(x= 0.19, y= 0.7, s= 'Tripura has the highest\nunemloyment percentage\nc
plt.show()
```



The average employed per region will tell us if there a significant difference in employees among states as well as the regions with the highest and lowest number of employees

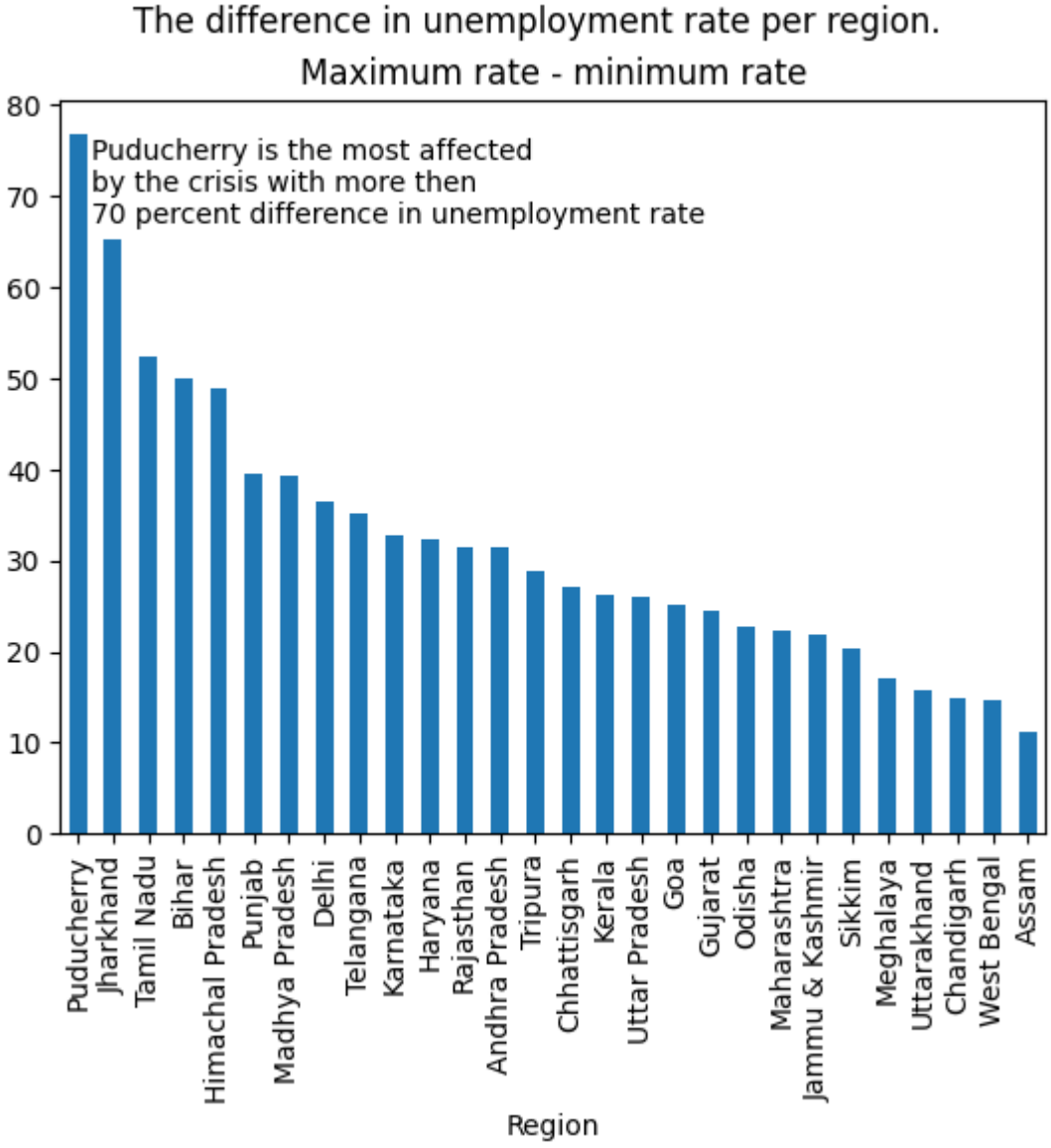
In [ ]:

```
df2 = df1.groupby('Region')[['est_mil_emp']].mean().sort_values(by='est_mil_emp',as
df2.plot(kind='bar')
plt.title('Average millions employed per region')
plt.ylabel('Millions employed')
plt.figtext(x=0.15, y=0.75, s='While the numbers of employees have\na big variance
plt.show()
```



In [ ]:

```
df2= df1.groupby('Region')['est_unemp_perc'].agg(lambda x: max(x) - min(x)).sort_va
plt.suptitle('The difference in unemployment rate per region.')
plt.title('Maximum rate - minimum rate')
plt.figtext(x= 0.15, y= 0.75, s='Puducherry is the most affected\nby the crisis wit
plt.show()
```


The difference in unemployment rate per region.
Maximum rate - minimum rate
Puducherry is the most affected
by the crisis with more then
70 percent difference in unemployment rate
80
70
60
50
40
30
20
10
0
Puducherry
Jharkhand
Tamil Nadu
Bihar
Himachal Pradesh
Punjab
Madhya Pradesh
Delhi
Telangana
Karnataka
Haryana
Rajasthan
Andhra Pradesh
Tripura
Chhattisgarh
Kerala
Uttar Pradesh
Goa
Gujarat
Odisha
Maharashtra
Jammu & Kashmir
Sikkim
Meghalaya
Uttarakhand
Chandigarh
West Bengal
Assam
Region