Unemployment Analysis using Python

Author: Krishna Sayanti Deb

Data Description

There are two datasets that contain the unemployment rate of all the states in India. Features of Dataset1: Region = states in India Date = date which the unemployment rate observed Frequency = measuring frequency (Monthly) Estimated Unemployment Rate(%) = percentage of people unemployed in each State of India Estimated Employed = percentage of people employed Estimated Labour Participation Rate(%) = the number of people actively participating in the labour force/the total number of people eligible to participate in the labor force *100 Features of Dataset2: Region = states in India Date = date which the unemployment rate observed Frequency = measuring frequency (Monthly) Estimated Unemployment Rate(%) = percentage of people unemployed in each State of India Estimated Employed = percentage of people employed Estimated Labour Participation Rate(%) = the number of people actively participating in the labour force/the total number of people eligible to participate in the labor force *100 Region.1 = Region Latitude= Latitunial extension Longitude = Longitudinal extension

Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Importing Datasets

In [2]:
```
#importing dataset 1
data1 = pd.read_excel(r'C:\Users\LENOVO\Desktop\Internship\CipherByte Technoligies\Task-2\Unemployment in Ir

#import the daataset 2
data2 = pd.read_excel(r'C:\Users\LENOVO\Desktop\Internship\CipherByte Technoligies\Task-2\Unemployment_Rate_
```

In [3]:
```
#view dataset 1
data1
```

Out[3]:

	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
...	...	...	...	...	...	...	...
749	West Bengal	29-02-2020	Monthly	7.55	10871168.0	44.09	Urban
750	West Bengal	31-03-2020	Monthly	6.67	10806105.0	43.34	Urban
751	West Bengal	30-04-2020	Monthly	15.63	9299466.0	41.20	Urban
752	West Bengal	31-05-2020	Monthly	15.22	9240903.0	40.67	Urban
753	West Bengal	30-06-2020	Monthly	9.86	9088931.0	37.57	Urban

754 rows × 7 columns

In [4]:
```
#view dataset 2
```

```
data2
```

Out[4]:

	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	16635535.0	41.02	South	15.9129	79.740
1	Andhra Pradesh	29-02-2020	M	5.83	16545652.0	40.90	South	15.9129	79.740
2	Andhra Pradesh	31-03-2020	M	5.79	15881197.0	39.18	South	15.9129	79.740
3	Andhra Pradesh	30-04-2020	M	20.51	11336911.0	33.10	South	15.9129	79.740
4	Andhra Pradesh	31-05-2020	M	17.43	12988845.0	36.46	South	15.9129	79.740
...	...	...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	East	22.9868	87.855
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	East	22.9868	87.855
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	East	22.9868	87.855
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	East	22.9868	87.855
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	East	22.9868	87.855

267 rows × 9 columns

Summary and Data Cleaning

Dataset:1

In [5]:
```
#view shape of dataset1
data1.shape
```

Out[5]:
```
(754, 7)
```

In [6]:
```
#checking whether there is any missing values in dataset1
data1.isnull().sum()
```

Out[6]:
```
Region                                   14
Date                                     14
Frequency                                14
Estimated Unemployment Rate (%)          14
Estimated Employed                       14
Estimated Labour Participation Rate (%)  14
Area                                     14
dtype: int64
```

In [7]:
```
#dropping null values from dataset1
data1 = data1.dropna()
```

In [8]:
```
#shape
data1.shape
```

Out[8]:
```
(740, 7)
```

In [9]:
```
#checking if there is any duplicate values in dataset 1
data1.duplicated().sum()
```

Out[9]:
```
0
```

In [10]:
```
#cleaned dataset1
data1
```

Out[10]:

	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
...	...	...	...	...	...	...	...
749	West Bengal	29-02-2020	Monthly	7.55	10871168.0	44.09	Urban
750	West Bengal	31-03-2020	Monthly	6.67	10806105.0	43.34	Urban
751	West Bengal	30-04-2020	Monthly	15.63	9299466.0	41.20	Urban
752	West Bengal	31-05-2020	Monthly	15.22	9240903.0	40.67	Urban
753	West Bengal	30-06-2020	Monthly	9.86	9088931.0	37.57	Urban

740 rows × 7 columns

Dataset2

In [11]:
```
#view shape of dataset2
data2.shape
```

Out[11]:
```
(267, 9)
```

In [12]:
```
#checking if there is any null values in dataset2
data2.isna().sum()
```

Out[12]:
```
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
```

In [13]:
```
#checking if there is any duplicate values in dataset2
data2.duplicated().sum()
```

Out[13]:
```
0
```

In [14]:
```
#cleaned dataset 2
data2
```

Out[14]:

	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	16635535.0	41.02	South	15.9129	79.740
1	Andhra Pradesh	29-02-2020	M	5.83	16545652.0	40.90	South	15.9129	79.740
2	Andhra Pradesh	31-03-2020	M	5.79	15881197.0	39.18	South	15.9129	79.740
3	Andhra Pradesh	30-04-2020	M	20.51	11336911.0	33.10	South	15.9129	79.740
4	Andhra Pradesh	31-05-2020	M	17.43	12988845.0	36.46	South	15.9129	79.740
...	...	...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	East	22.9868	87.855
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	East	22.9868	87.855
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	East	22.9868	87.855
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	East	22.9868	87.855
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	East	22.9868	87.855

267 rows × 9 columns

Merge Datasets

The task at hand involves merging two datasets that both contain data organized by dates. In order to create the final dataset, it's necessary to combine these two datasets.

In [15]:
```
#make a list of two datasets
merge_data = [data1,data2]
```

In [16]:
```
#Concate two dataset
final_data = pd.concat(merge_data)
```

In [17]:
```
# final Dataset
final_data
```

Out[17]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area	Region.1	Longitude	Latitude
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural	NaN	NaN	NaN
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural	NaN	NaN	NaN
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural	NaN	NaN	NaN
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural	NaN	NaN	NaN
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	NaN	East	22.9868	87.855
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	NaN	East	22.9868	87.855
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	NaN	East	22.9868	87.855
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	NaN	East	22.9868	87.855
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	NaN	East	22.9868	87.855

1007 rows × 10 columns

Data Mining

To create the dataset, it's required to extract information from the "Region" column and then carry out feature engineering on the "Region.1" column.

In [18]:
```
#print unique region
final_data['Region.1'].unique()
```

Out[18]:
```
array([nan, 'South', 'Northeast', 'East', 'West', 'North'], dtype=object)
```

In [19]:
```
#find unique region number in dataset2
data2.Region.nunique()
```

Out[19]:
```
27
```

In [20]:
```
#find unique region name in dataset2
data2.Region.unique()
```

Out[20]:
```
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 [21]:
```
#find unique region number in dataset1
data1.Region.nunique()
```

Out[21]:
```
28
```

In [22]:
```
#find unique region name in dataset1
data1.Region.unique()
```

Out[22]:
```
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', 'Chandigarh'],
      dtype=object)
```

In [23]:
```
# Perform groupby on the 'Region' column
grouped = data2.groupby(['Region','Region.1'])
```

In [24]:
```
grouped
```

Out[24]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000002713EE07850>
```

In [25]:
```
#making list of state and region
list_of_state = []

#for state in grouped:
for i,j in grouped:
    list = list_of_state.append(i)
```

In [26]:
```
#list of the state, region
list_of_state
```

Out[26]:
```
[('Andhra Pradesh', 'South'),
 ('Assam', 'Northeast'),
 ('Bihar', 'East'),
 ('Chhattisgarh', 'West'),
 ('Delhi', 'North'),
 ('Goa', 'West'),
 ('Gujarat', 'West'),
 ('Haryana', 'North'),
 ('Himachal Pradesh', 'North'),
 ('Jammu & Kashmir', 'North'),
 ('Jharkhand', 'East'),
 ('Karnataka', 'South'),
 ('Kerala', 'South'),
 ('Madhya Pradesh', 'West'),
 ('Maharashtra', 'West'),
 ('Meghalaya', 'Northeast'),
 ('Odisha', 'East'),
 ('Puducherry', 'South'),
 ('Punjab', 'North'),
 ('Rajasthan', 'North'),
 ('Sikkim', 'Northeast'),
 ('Tamil Nadu', 'South'),
 ('Telangana', 'South'),
 ('Tripura', 'Northeast'),
 ('Uttar Pradesh', 'North'),
 ('Uttarakhand', 'North'),
 ('West Bengal', 'East')]
```

In [27]:
```
#make a dictionary on  state, region
new_region_map = dict(list_of_state)
```

In [28]:
```
#print dictionary
new_region_map
```

Out[28]:
```
{'Andhra Pradesh': 'South',
 'Assam': 'Northeast',
 'Bihar': 'East',
 'Chhattisgarh': 'West',
 'Delhi': 'North',
 'Goa': 'West',
 'Gujarat': 'West',
 'Haryana': 'North',
 'Himachal Pradesh': 'North',
 'Jammu & Kashmir': 'North',
 'Jharkhand': 'East',
 'Karnataka': 'South',
 'Kerala': 'South',
 'Madhya Pradesh': 'West',
 'Maharashtra': 'West',
 'Meghalaya': 'Northeast',
 'Odisha': 'East',
 'Puducherry': 'South',
 'Punjab': 'North',
 'Rajasthan': 'North',
 'Sikkim': 'Northeast',
 'Tamil Nadu': 'South',
 'Telangana': 'South',
 'Tripura': 'Northeast',
 'Uttar Pradesh': 'North',
 'Uttarakhand': 'North',
 'West Bengal': 'East'}
```

In [29]:
```
#mapped the region column
final_data['New Region'] = final_data['Region'].map(new_region_map)
```

In [30]:
```
#final_data
final_data
```

Out[30]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area	Region.1	Longitude	Latitude	New Region
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural	NaN	NaN	NaN	South
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural	NaN	NaN	NaN	South
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural	NaN	NaN	NaN	South
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural	NaN	NaN	NaN	South
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural	NaN	NaN	NaN	South
...	...	...	...	...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	NaN	East	22.9868	87.855	East
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	NaN	East	22.9868	87.855	East
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	NaN	East	22.9868	87.855	East
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	NaN	East	22.9868	87.855	East
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	NaN	East	22.9868	87.855	East

1007 rows × 11 columns

In [31]:
```
#Checking the null values
final_data.isna().sum()
```

Out[31]:
```
Region                                     0
Date                                       0
Frequency                                  0
Estimated Unemployment Rate (%)            0
Estimated Employed                         0
Estimated Labour Participation Rate (%)    0
Area                                     267
Region.1                                 740
Longitude                                740
Latitude                                 740
New Region                                12
dtype: int64
```

In [32]:
```
final_data.Area.unique()
```

Out[32]:
```
array(['Rural', 'Urban', nan], dtype=object)
```

Inferences

1. As the latitude and longitude columns have no impact on Unemployment. So these two columns are not required.
2. Using Region.1 column, mapped a New Region column. Hence, dropping the Region.1 column.
3. In case of Area Column we observe missing values. Since we can not identified whether it is a Rural or Urban data, this column cannot be mapped. Hence, Area column will be dropped.

In [33]:
```
#Removing unnecessary column
final_data = final_data.drop(['Region.1','Longitude','Latitude', 'Area'], axis=1)
```

In [34]:
```
#rename column
final_data = final_data.rename(columns={'Region':'State',
                                        'New Region':'Region'})
```

In [35]:
```
#handle the null value in Region Column
final_data.Region = final_data.Region.fillna("North")
```

In [36]:
```
#Checking for any remaining missing values in the dataset
final_data.isna().sum()
```

Out[36]:
```
State                                    0
Date                                     0
Frequency                                0
Estimated Unemployment Rate (%)          0
Estimated Employed                       0
Estimated Labour Participation Rate (%)  0
Region                                   0
dtype: int64
```

In [37]:
```
#cleanned data
final_data
```

Out[37]:

	State	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Region
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	South
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	South
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	South
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	South
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	South
...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	East
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	East
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	East
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	East
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	East

1007 rows × 7 columns

In [38]:
```
#Shape
final_data.shape
```

Out[38]:
```
(1007, 7)
```

In [39]:
```
#print all the columns
final_data.columns
```

Out[39]:
```
Index(['State', 'Date', 'Frequency', 'Estimated Unemployment Rate (%)',
       'Estimated Employed', 'Estimated Labour Participation Rate (%)',
       'Region'],
      dtype='object')
```

In [40]:
```
# Left-trim column names using lambda function
final_data = final_data.rename(columns=lambda x: x.lstrip())
```

In [41]:
```
final_data
```

Out[41]:

	State	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Region
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	South
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	South
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	South
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	South
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	South
...	...	...	...	...	...	...	...
262	West Bengal	30-06-2020	M	7.29	30726310.0	40.39	East
263	West Bengal	31-07-2020	M	6.83	35372506.0	46.17	East
264	West Bengal	31-08-2020	M	14.87	33298644.0	47.48	East
265	West Bengal	30-09-2020	M	9.35	35707239.0	47.73	East
266	West Bengal	31-10-2020	M	9.98	33962549.0	45.63	East

1007 rows × 7 columns

In [42]:
```
final_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1007 entries, 0 to 266
Data columns (total 7 columns):
 #   Column                                   Non-Null Count  Dtype  
---  ------                                   --------------  -----  
 0   State                                    1007 non-null   object 
 1   Date                                     1007 non-null   object 
 2   Frequency                                1007 non-null   object 
 3   Estimated Unemployment Rate (%)          1007 non-null   float64
 4   Estimated Employed                       1007 non-null   float64
 5   Estimated Labour Participation Rate (%)  1007 non-null   float64
 6   Region                                   1007 non-null   object 
dtypes: float64(3), object(4)
memory usage: 62.9+ KB
```

In [43]:
```
final_data.Frequency.unique()
```

Out[43]:
```
array(['Monthly', 'M'], dtype=object)
```

As the frequency column contains only single unique data that is monthly,the Frequency column can be dropped.

In [44]:
```
#Drop the Frequency Column
final_data = final_data.drop(['Frequency'],axis=1)
```

In [45]:
```
#View data
final_data.head()
```

Out[45]:

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

The Date Column is of 'object' datatype. Converting the column to Date datatype and setting it as the Index Column.

In [46]:
```
#Convert datatype
final_data.Date = pd.to_datetime(final_data.Date)
```

In [47]:
```
#Check the datetype
final_data.Date.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 1007 entries, 0 to 266
Series name: Date
Non-Null Count  Dtype
--------------  -----
1007 non-null   datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 15.7 KB
```

In [48]:
```
#Convert datatype
final_data.Date = pd.to_datetime(final_data.Date)#Check the datetype
final_data.Date.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 1007 entries, 0 to 266
Series name: Date
Non-Null Count  Dtype
--------------  -----
1007 non-null   datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 15.7 KB
```

In [49]:
```
#Check the datetype
final_data.Date.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 1007 entries, 0 to 266
Series name: Date
Non-Null Count  Dtype
--------------  -----
1007 non-null   datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 15.7 KB
```

In [50]:
```
#Set index
final_data = final_data.set_index('Date')
```

In [51]:
```
#Reorder the column
new_order = ['State', 'Region','Estimated Unemployment Rate (%)','Estimated Employed','Estimated Labour Part
final_data = final_data[new_order]
```

In [52]:
```
#View the Final data
final_data
```

Out[52]:

Date	State	Region	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)
2019-05-31	Andhra Pradesh	South	3.65	11999139.0	43.24
2019-06-30	Andhra Pradesh	South	3.05	11755881.0	42.05
2019-07-31	Andhra Pradesh	South	3.75	12086707.0	43.50
2019-08-31	Andhra Pradesh	South	3.32	12285693.0	43.97
2019-09-30	Andhra Pradesh	South	5.17	12256762.0	44.68
...	...	...	...	...	...
2020-06-30	West Bengal	East	7.29	30726310.0	40.39
2020-07-31	West Bengal	East	6.83	35372506.0	46.17
2020-08-31	West Bengal	East	14.87	33298644.0	47.48
2020-09-30	West Bengal	East	9.35	35707239.0	47.73
2020-10-31	West Bengal	East	9.98	33962549.0	45.63

1007 rows × 5 columns

Statistical Measure

In [53]:
```
#data describe
final_data.describe().T
```

Out[53]:

	count	mean	std	min	25%	50%	75%	max
Estimated Unemployment Rate (%)	1007.0	1.190699e+01	1.073955e+01	0.00	4.685	8.89	1.612500e+01	76.74
Estimated Employed	1007.0	8.996209e+06	1.020784e+07	49420.00	1639125.000	5543380.00	1.287115e+07	59433759.00
Estimated Labour Participation Rate (%)	1007.0	4.237862e+01	8.048542e+00	13.33	37.835	40.88	4.531500e+01	72.57

Summary of Attributes Explanation:

Estimated Unemployment Rate (%):

1. The mean and standard deviation of the unemployment rate are approximately 11.91% and 10.74%, respectively.
2. The lowest recorded unemployment rate is 0%.
3. The first quartile (25th percentile) of the unemployment rate is around 4.685%.
4. The median (50th percentile) of the unemployment rate is roughly 8.89%.
5. The third quartile (75th percentile) of the unemployment rate is about 16.125%.
6. The highest recorded unemployment rate is 76.74%.
7. Notably, there is a significant difference between the mean and median values.
8. The wide gap between the third quartile and the maximum value indicates a right-skewed distribution.

Estimated Employed:

1. The mean and standard deviation of the estimated number of employed individuals are approximately 8,996,209 and 10,207,840, respectively.
2. The lowest recorded number of employed individuals is 49,420.
3. The first quartile (25th percentile) of the estimated number of employed individuals is roughly 1,639,125.
4. The median (50th percentile) of the estimated number of employed individuals is around 5,543,380.
5. The third quartile (75th percentile) of the estimated number of employed individuals is approximately 12,871,150.

6. The highest recorded number of employed individuals is 59,433,759.
7. The significant difference between the third quartile and the maximum value indicates a right-skewed distribution.

Estimated Labour Participation Rate (%):

1. The mean labour participation rate is approximately 42.38%.
2. The standard deviation of the labour participation rate is roughly 8.05%.
3. The lowest recorded labour participation rate is 13.33%.
4. The first quartile (25th percentile) of the labour participation rate is about 37.835%.
5. The median (50th percentile) of the labour participation rate is approximately 40.88%.
6. The third quartile (75th percentile) of the labour participation rate is roughly 45.315%.
7. The highest recorded labour participation rate is 72.57%.

In [54]:
```
#data description of the Object column
final_data.describe(include='O').T
```

Out[54]:

	count	unique	top	freq
State	1007	28	Andhra Pradesh	38
Region	1007	5	North	307

Summary of Objective Features Explanation:

State:

1. The dataset covers data from 28 unique Indian states.
2. The state with the highest frequency is Andhra Pradesh, appearing 38 times.

Region:

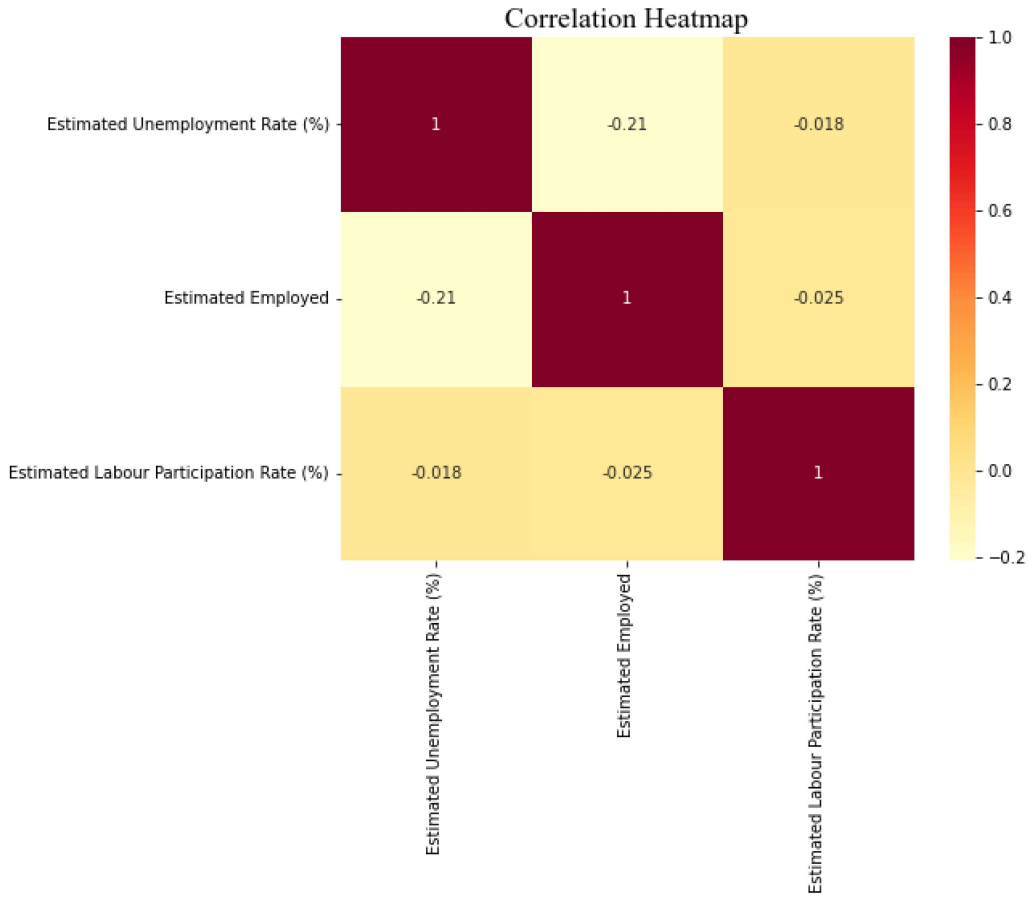
1. The dataset categorizes data into 5 distinct regions.
2. Among these five regions, "North" has the highest frequency with 307 occurrences.

In [55]:
```
# compute correletaion
final_data.corr()
```

Out[55]:

	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)
Estimated Unemployment Rate (%)	1.000000	-0.209495	-0.018137
Estimated Employed	-0.209495	1.000000	-0.024770
Estimated Labour Participation Rate (%)	-0.018137	-0.024770	1.000000

In [56]:
```
#plot correlation
plt.figure(figsize=(8,6))
sns.heatmap(final_data.corr(), cmap= 'YlOrRd', annot=True)
plt.title('Correlation Heatmap', fontsize= (18), fontname='Times New Roman')
plt.show()
```

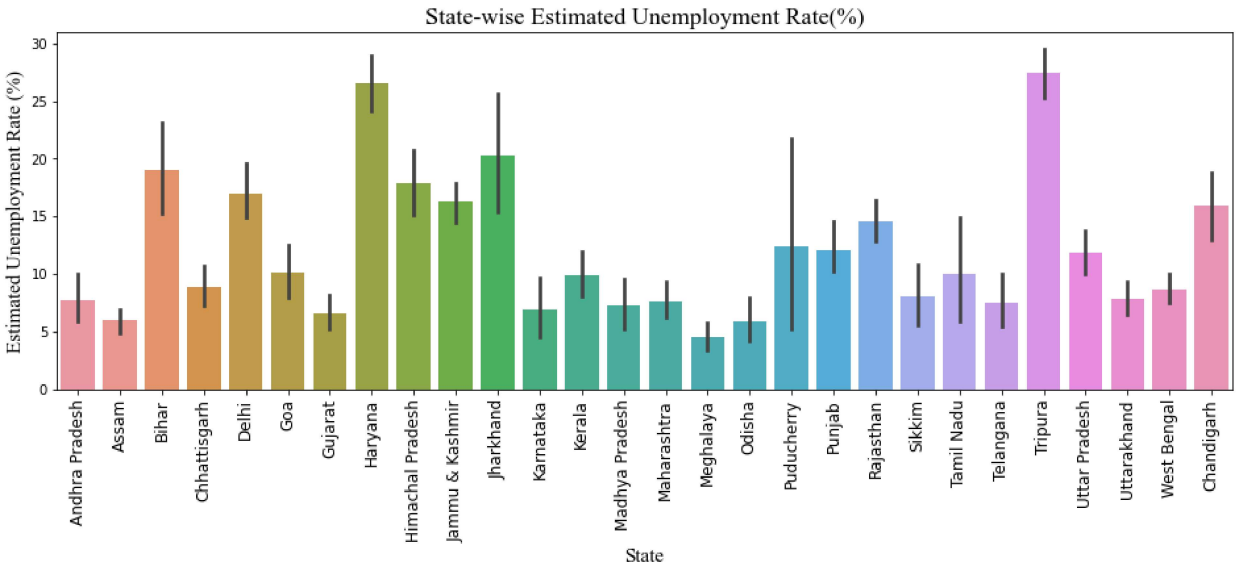


Inferences:

The graphical representation of the heatmap makes it clear that there is no strong correlation within the dataset.

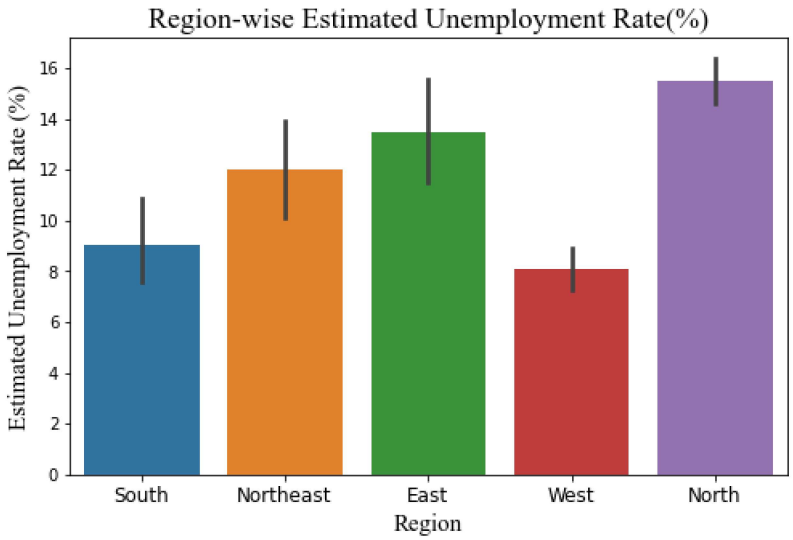
EDA

In [57]:
```
# Plot barchart to show State-wise Estimated Unemployment Rate(%)

plt.figure(figsize=(16,5))
sns.barplot(x='State', y= 'Estimated Unemployment Rate (%)', data= final_data)
plt.xlabel('State', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Unemployment Rate (%)',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(rotation = 90, fontsize =(12))
plt.title("State-wise Estimated Unemployment Rate(%)", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```

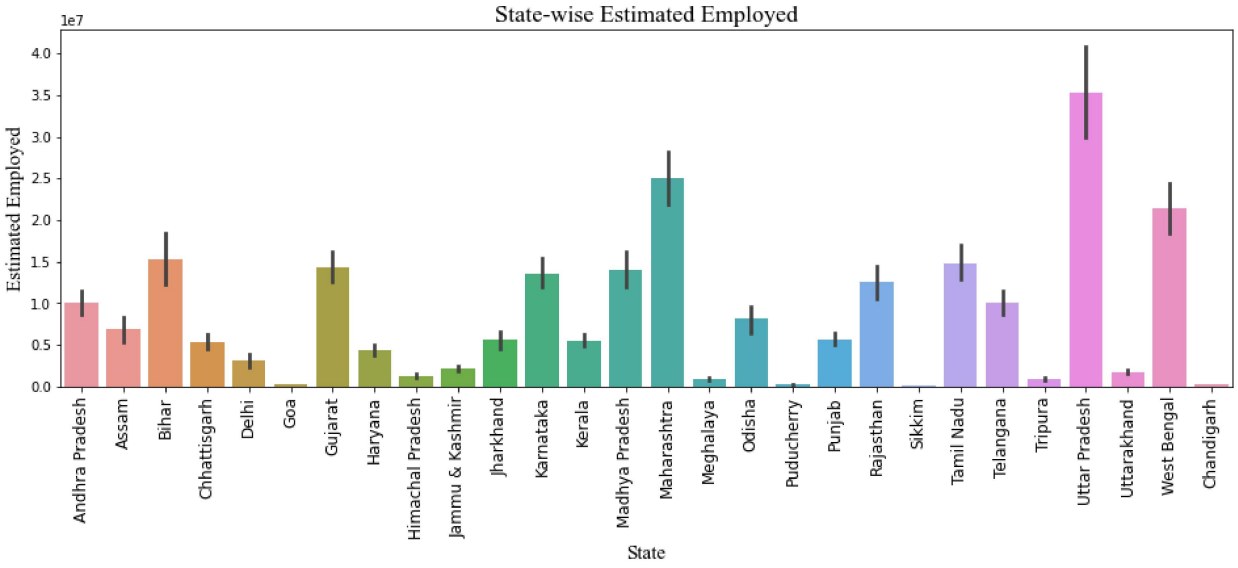


In [58]:
```
# Plot barchart to show Region-wise Estimated Unemployment Rate(%)
```

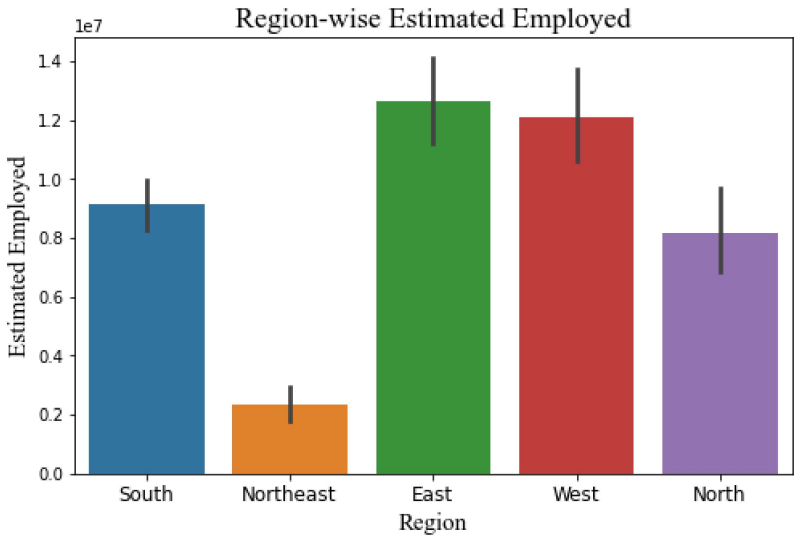
```
plt.figure(figsize=(8,5))
sns.barplot(x='Region', y= 'Estimated Unemployment Rate (%)', data= final_data)
plt.xlabel('Region', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Unemployment Rate (%)',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(fontsize =(12))
plt.title("Region-wise Estimated Unemployment Rate(%)", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```



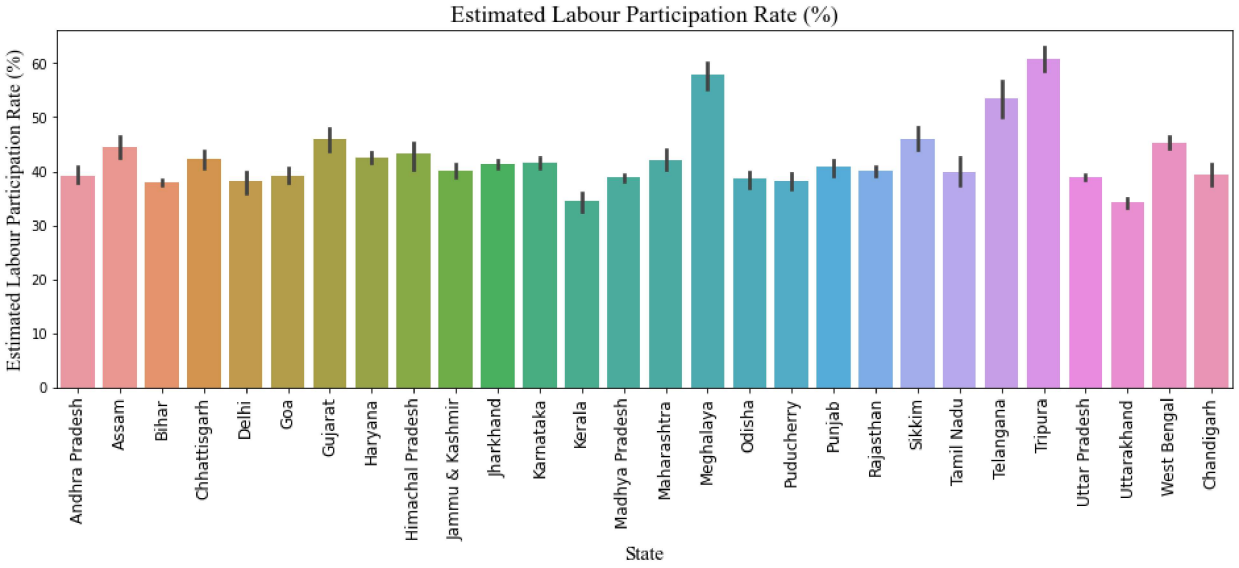
In [59]:
```
# Plot barchart to show State-wise Estimated Employed

plt.figure(figsize=(16,5))
sns.barplot(x='State', y= 'Estimated Employed', data= final_data)
plt.xlabel('State', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Employed',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(rotation = 90, fontsize =(12))
plt.title("State-wise Estimated Employed", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```



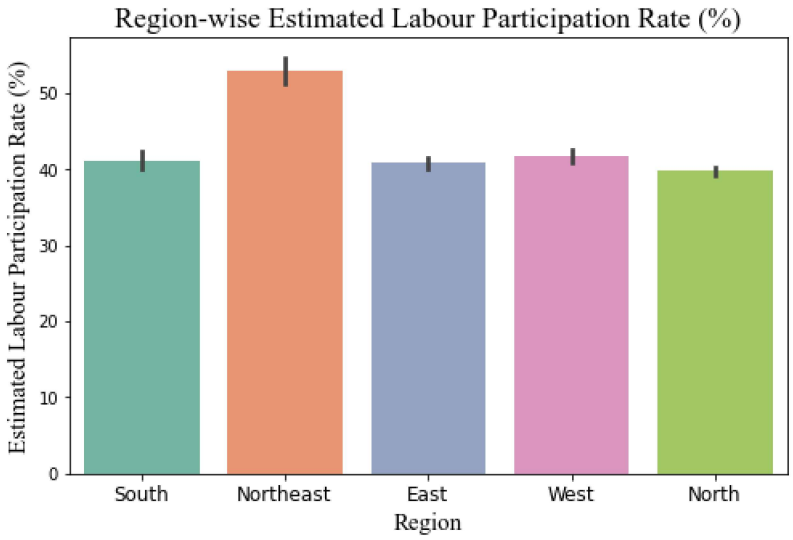
In [60]:
```
# Plot barchart to show Region-wise Estimated Employed

plt.figure(figsize=(8,5))
custom_palette= sns.color_palette("Set3_r")
sns.barplot(x='Region', y= 'Estimated Employed', data= final_data)
plt.xlabel('Region', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Employed',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(fontsize =(12))
plt.title("Region-wise Estimated Employed", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```



In [61]:
```
# Plot barchart to show State-wise Estimated Labour Participation Rate (%)

plt.figure(figsize=(16,5))
sns.barplot(x='State', y= 'Estimated Labour Participation Rate (%)', data= final_data)
plt.xlabel('State', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Labour Participation Rate (%)',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(rotation = 90, fontsize =(12))
plt.title("Estimated Labour Participation Rate (%)", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```



In [62]:
```
# Plot barchart to show Region-wise Estimated Labour Participation Rate (%)

plt.figure(figsize=(8,5))
#custom_palette = ['crimson', 'orange', 'yellow', 'green', 'blue']
custom_palette= sns.color_palette("Set2")
sns.barplot(x='Region', y= 'Estimated Labour Participation Rate (%)', data= final_data,palette=custom_palett
plt.xlabel('Region', fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Estimated Labour Participation Rate (%)',fontsize=(15),fontname= 'Times New Roman')
plt.xticks(fontsize =(12))
plt.title("Region-wise Estimated Labour Participation Rate (%)", fontsize=(18) ,fontname='Times New Roman')
plt.show()
```



Inferences:

1. The highest Estimated Unemployment Rate (%) is registered in Tripura, while the lowest is in Meghalaya.
2. Unemployment rates are generally higher in the North region and lower in the West region.
3. Uttar Pradesh has a high number of Estimated Employed individuals, while Goa, Puducherry, Sikkim, and Chandigarh have very low numbers.
4. Estimated Employed numbers are relatively higher in the East and West regions, while Northeast states have very low numbers.
5. Estimated Labour Participation Rate (%) shows slight variations across states, with peaks in Meghalaya, Tripura, and Telangana.
6. Estimated Labour Participation Rate (%) is relatively consistent across regions, with the highest value observed in the Northeast region.

In [63]:
```
#make statelist
statelist = []
for i in final_data.State.unique():
    #print(i)
    statelist.append(i)
```

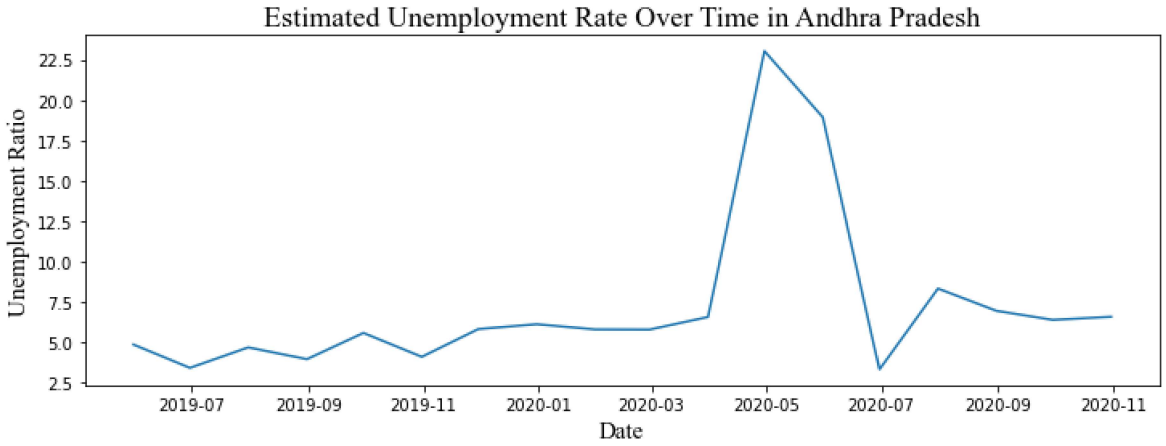
In [64]:
```
#print statelist
statelist
```

Out[64]:
```
['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',
 'Chandigarh']
```

Satetwise Unemployment Rate

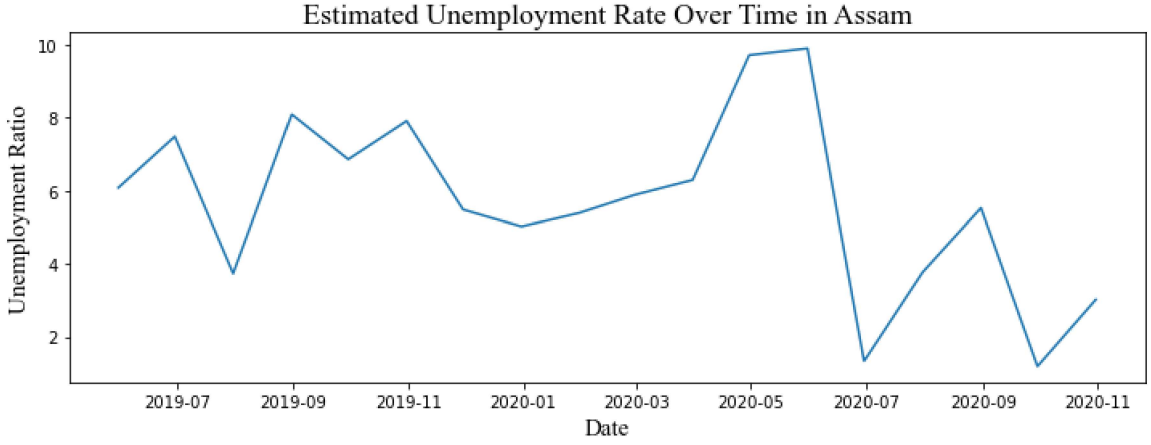
Andhra Pradesh

In [65]:
```
#define data
Andhra_Pradesh_df = final_data[final_data['State'] == 'Andhra Pradesh']

# Aggregate duplicate values by taking the mean
Andhra_Pradesh_df = Andhra_Pradesh_df.groupby(Andhra_Pradesh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Andhra_Pradesh_df, x = Andhra_Pradesh_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Andhra Pradesh',fontsize=(18),fontname= 'Times New Roman
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



Assam

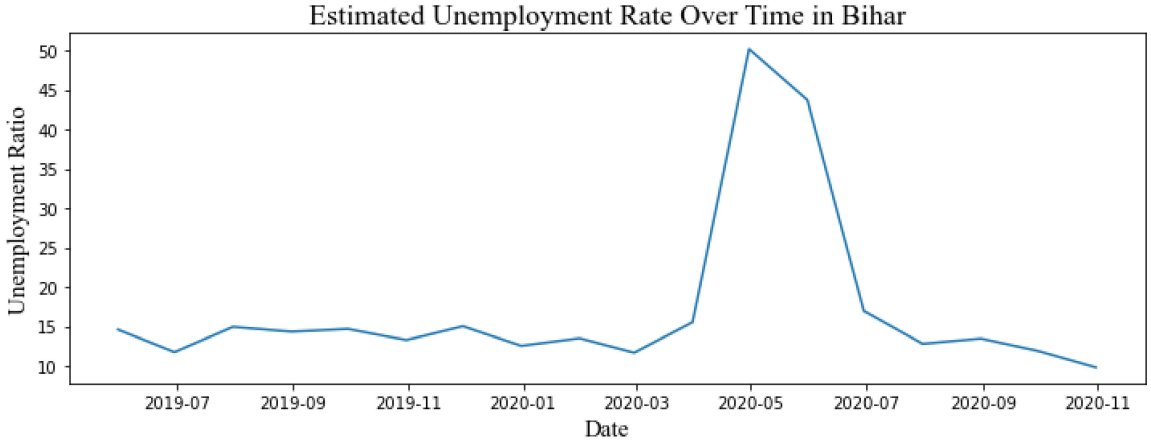
In [66]:
```
#define data
Assam_df = final_data[final_data['State'] == 'Assam']

# Aggregate duplicate values by taking the mean
Assam_df = Assam_df.groupby(Assam_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Assam_df, x = Assam_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Assam',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



Bihar

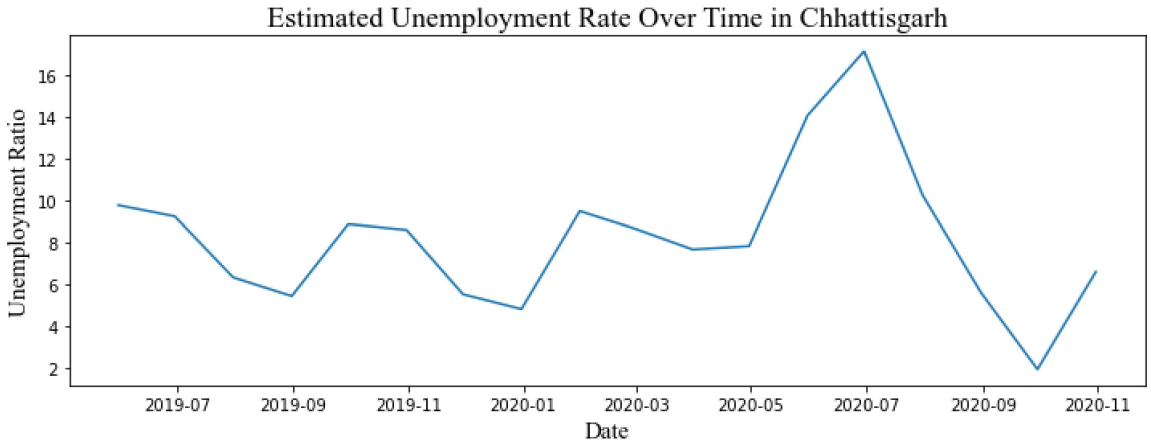
In [67]:
```
#define data
Bihar_df = final_data[final_data['State'] == 'Bihar']
```

```
# Aggregate duplicate values by taking the mean
Bihar_df = Bihar_df.groupby(Bihar_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Bihar_df, x = Bihar_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Bihar',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



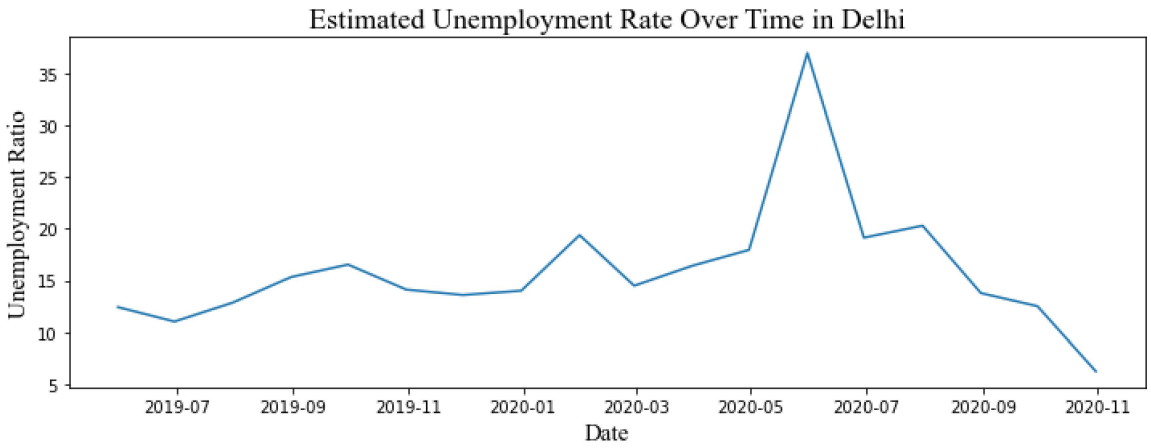
Chhattisgarh

In [68]:
```
#define data
Chhattisgarh_df = final_data[final_data['State'] == 'Chhattisgarh']

# Aggregate duplicate values by taking the mean
Chhattisgarh_df = Chhattisgarh_df.groupby(Chhattisgarh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Chhattisgarh_df, x = Chhattisgarh_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Chhattisgarh',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



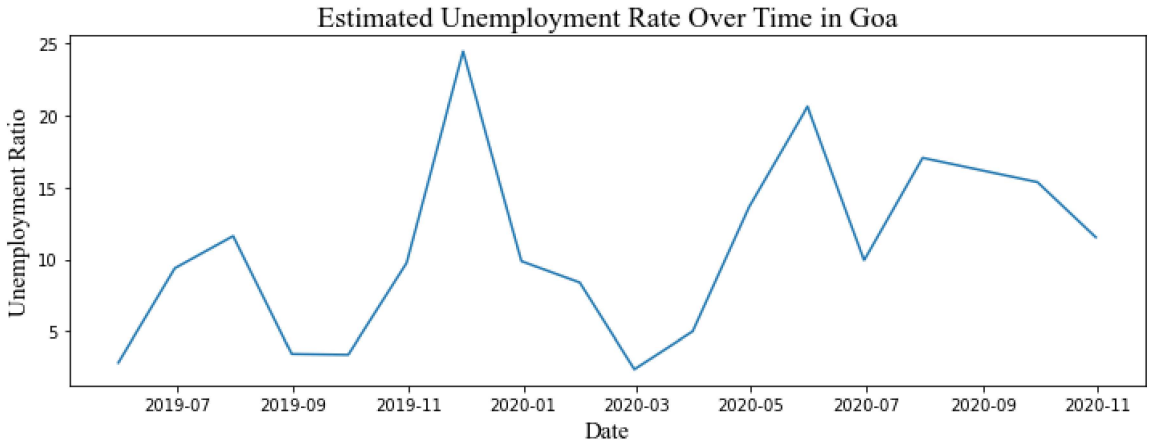
Delhi

In [69]:
```
#define data
Delhi_df = final_data[final_data['State'] == 'Delhi']

# Aggregate duplicate values by taking the mean
Delhi_df = Delhi_df.groupby(Delhi_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Delhi_df, x = Delhi_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Delhi',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



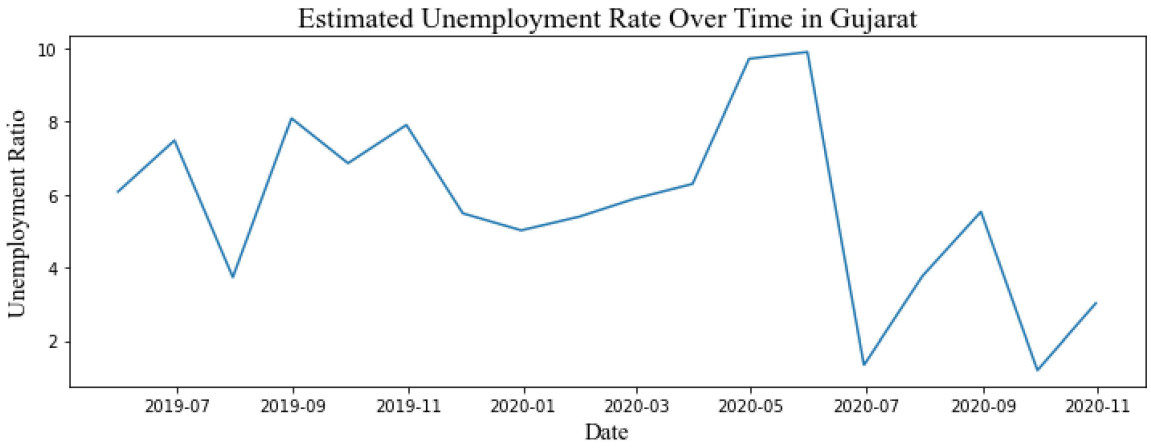
Goa

In [70]:
```
#define data
Goa_df = final_data[final_data['State'] == 'Goa']

# Aggregate duplicate values by taking the mean
Goa_df = Goa_df.groupby(Goa_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Goa_df, x = Goa_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Goa',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



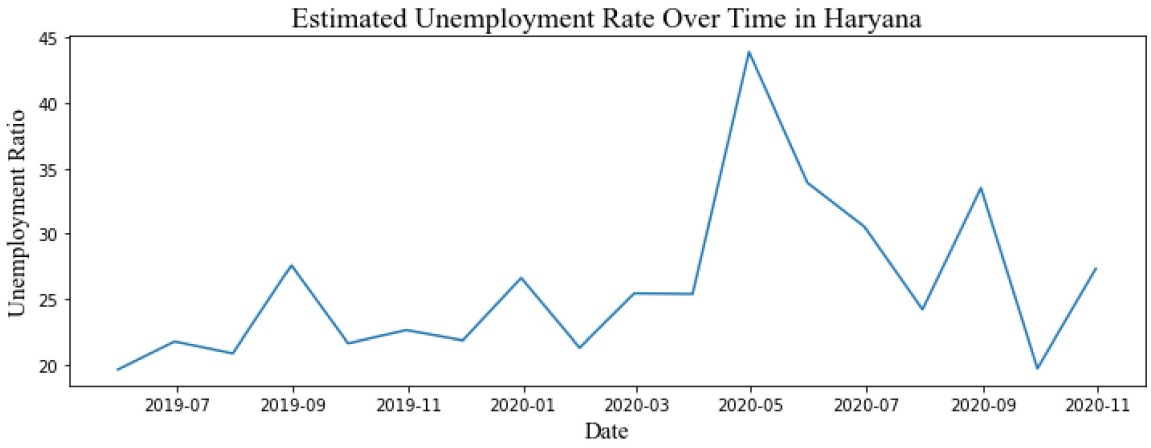
Gujarat

In [71]:
```
#define data
Gujarat_df = final_data[final_data['State'] == 'Assam']

# Aggregate duplicate values by taking the mean
Gujarat_df = Gujarat_df.groupby(Gujarat_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Gujarat_df, x = Gujarat_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Gujarat',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



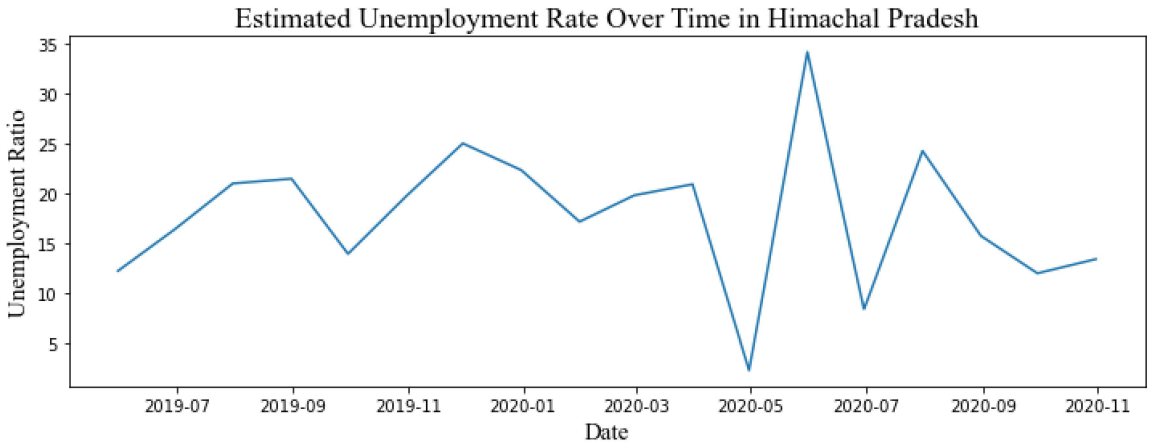
Haryana

In [72]:
```
#define data
Haryana_df = final_data[final_data['State'] == 'Haryana']

# Aggregate duplicate values by taking the mean
Haryana_df = Haryana_df.groupby(Haryana_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Haryana_df, x = Haryana_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Haryana',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



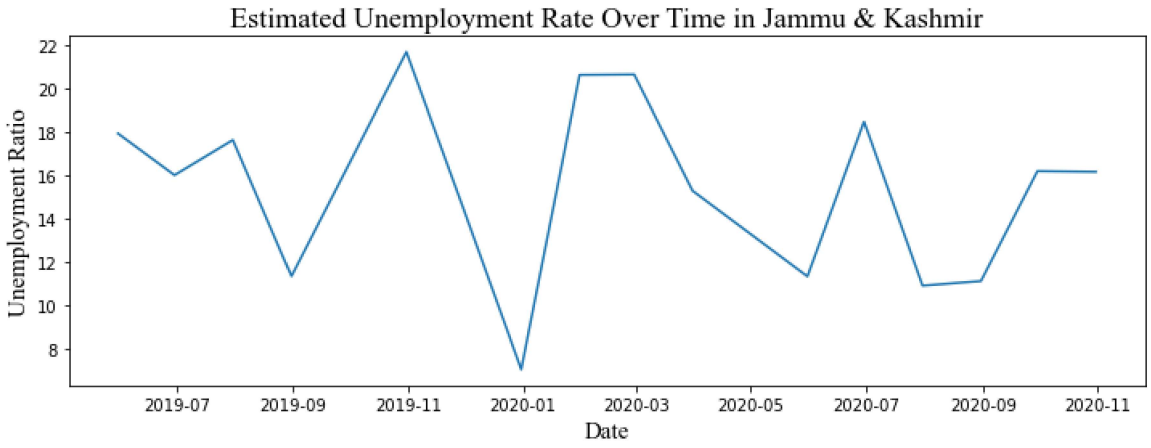
Himachal Pradesh

In [73]:
```
#define data
Himachal_Pradesh_df = final_data[final_data['State'] == 'Himachal Pradesh']

# Aggregate duplicate values by taking the mean
Himachal_Pradesh_df = Himachal_Pradesh_df.groupby(Himachal_Pradesh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Himachal_Pradesh_df, x = Himachal_Pradesh_df.index, y= 'Estimated Unemployment Rate (%)
plt.title('Estimated Unemployment Rate Over Time in Himachal Pradesh',fontsize=(18),fontname= 'Times New Ro
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



Jammu & Kashmir

In [74]:
```
#define data
Jammu_Kashmir_df = final_data[final_data['State'] == 'Jammu & Kashmir']

# Aggregate duplicate values by taking the mean
Jammu_Kashmir_df = Jammu_Kashmir_df.groupby(Jammu_Kashmir_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Jammu_Kashmir_df, x = Jammu_Kashmir_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Jammu & Kashmir',fontsize=(18),fontname= 'Times New Roma
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```

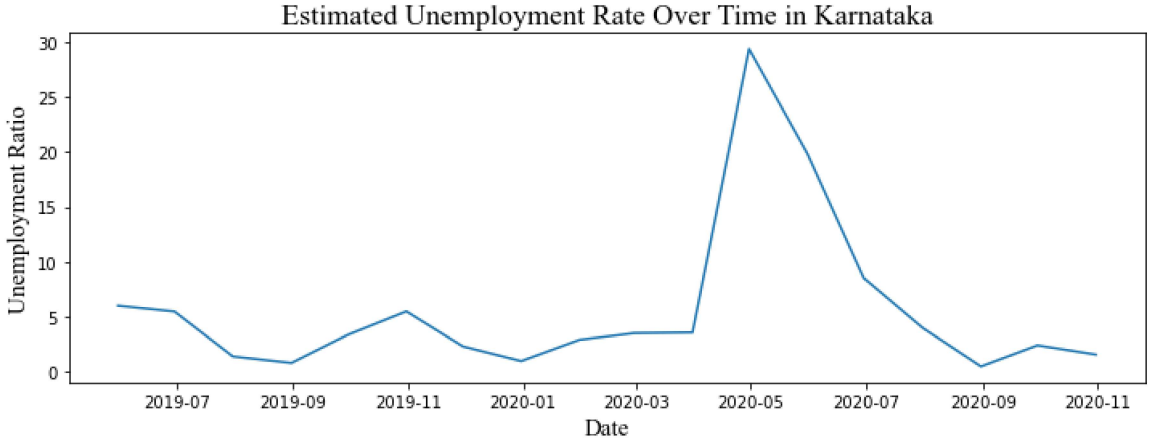


Jharkhand

In [75]:
```
#define data
Jharkhand_df = final_data[final_data['State'] == 'Jharkhand']

# Aggregate duplicate values by taking the mean
Jharkhand_df = Jharkhand_df.groupby(Jharkhand_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Jharkhand_df, x = Jharkhand_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Jharkhand',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```

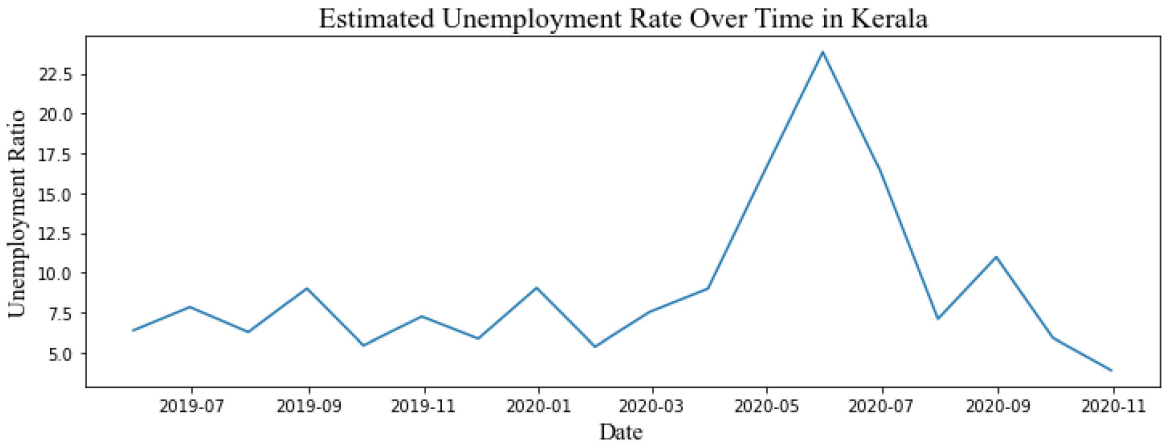
Karnataka

In [76]:
```
#define data
Karnataka_df = final_data[final_data['State'] == 'Karnataka']

# Aggregate duplicate values by taking the mean
Karnataka_df = Karnataka_df.groupby(Karnataka_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Karnataka_df, x = Karnataka_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Karnataka',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



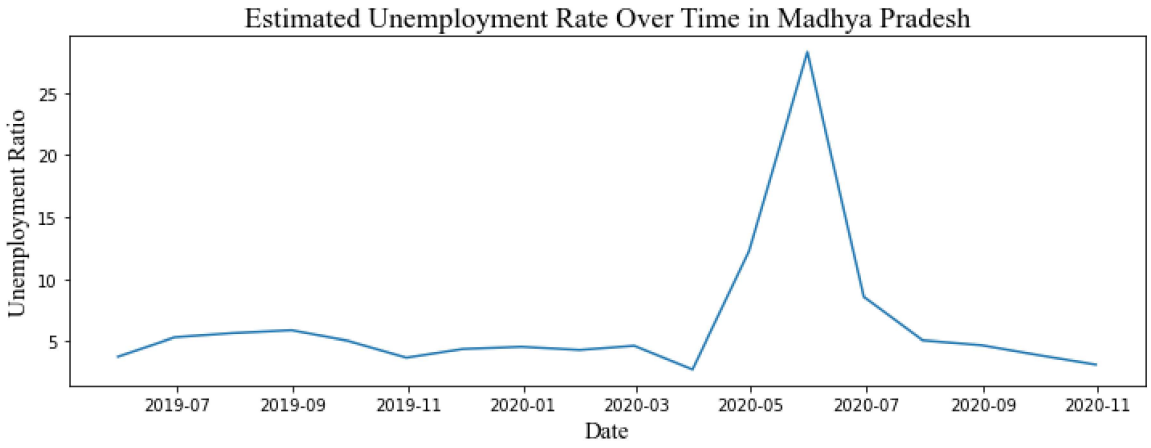
Kerala

In [77]:
```
#define data
Kerala_df = final_data[final_data['State'] == 'Kerala']

# Aggregate duplicate values by taking the mean
Kerala_df = Kerala_df.groupby(Kerala_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Kerala_df, x = Kerala_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Kerala',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



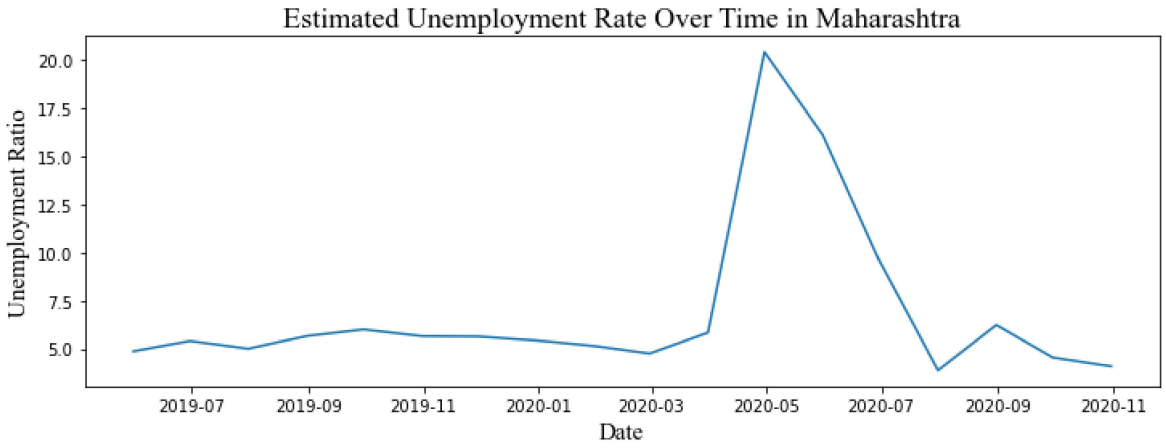
Madhya Pradesh

In [78]:
```
#define data
Madhya_Pradesh_df = final_data[final_data['State'] == 'Madhya Pradesh']

# Aggregate duplicate values by taking the mean
Madhya_Pradesh_df = Madhya_Pradesh_df.groupby(Madhya_Pradesh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Madhya_Pradesh_df, x = Madhya_Pradesh_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Madhya Pradesh',fontsize=(18),fontname= 'Times New Roman
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



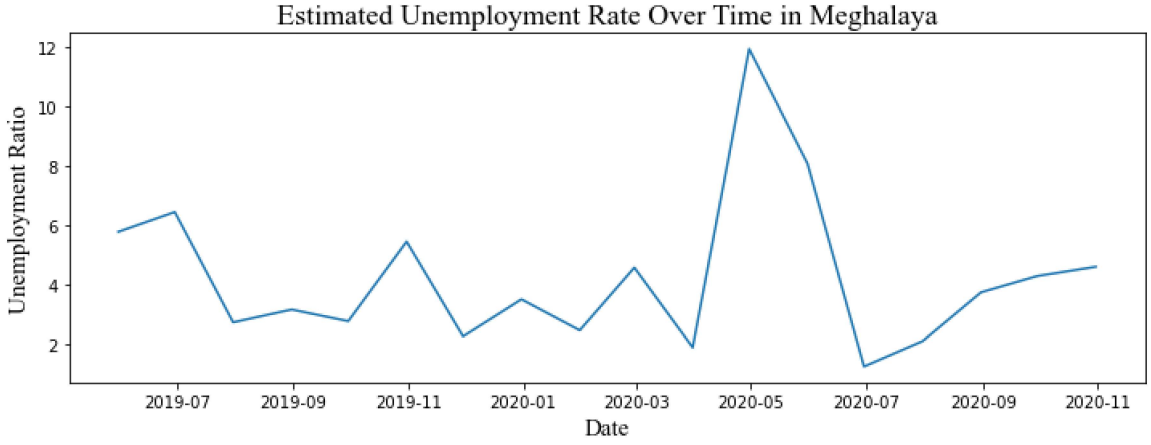
Maharashtra

In [79]:
```
#define data
Maharashtra_df = final_data[final_data['State'] == 'Maharashtra']

# Aggregate duplicate values by taking the mean
Maharashtra_df = Maharashtra_df.groupby(Maharashtra_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Maharashtra_df, x = Maharashtra_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Maharashtra',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



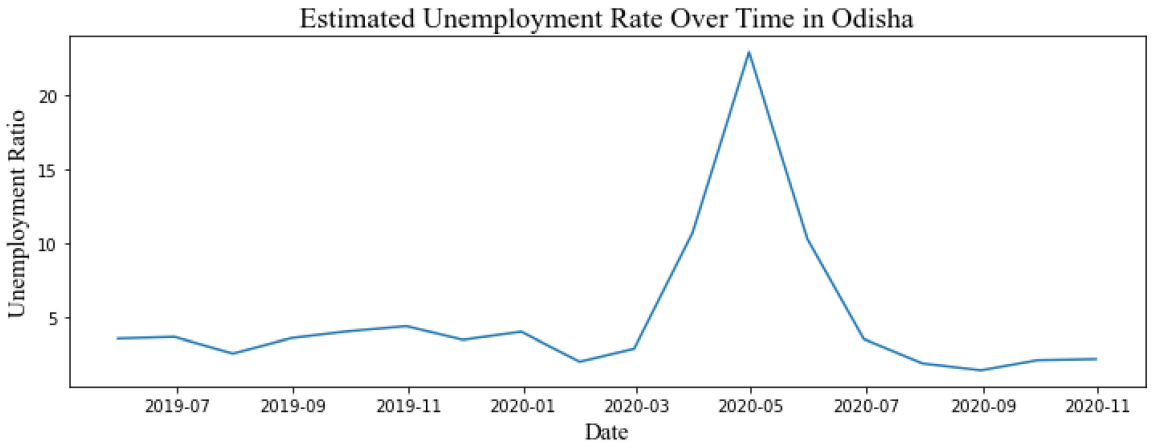
Meghalaya

In [80]:
```
#define data
Meghalaya_df = final_data[final_data['State'] == 'Meghalaya']

# Aggregate duplicate values by taking the mean
Meghalaya_df = Meghalaya_df.groupby(Meghalaya_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Meghalaya_df, x = Meghalaya_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Meghalaya',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



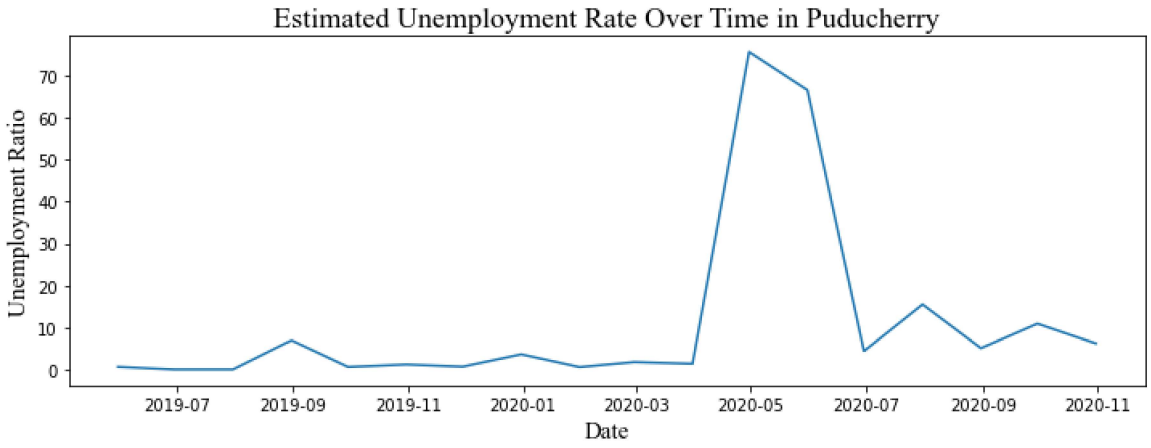
Odisha

In [81]:
```
#define data
Odisha_df = final_data[final_data['State'] == 'Odisha']

# Aggregate duplicate values by taking the mean
Odisha_df = Odisha_df.groupby(Odisha_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Odisha_df, x = Odisha_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Odisha',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



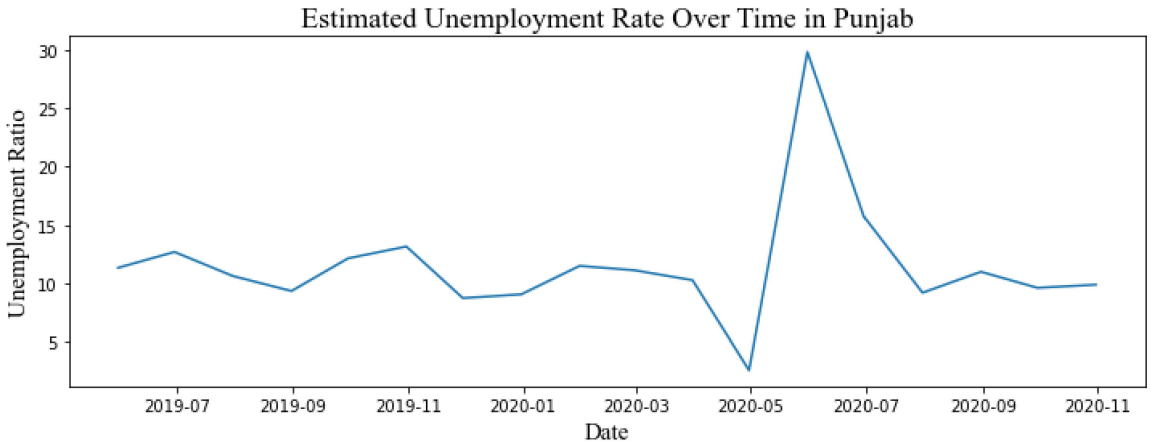
Puducherry

In [82]:
```
#define data
Puducherry_df = final_data[final_data['State'] == 'Puducherry']

# Aggregate duplicate values by taking the mean
Puducherry_df = Puducherry_df.groupby(Puducherry_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Puducherry_df, x = Puducherry_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Puducherry',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



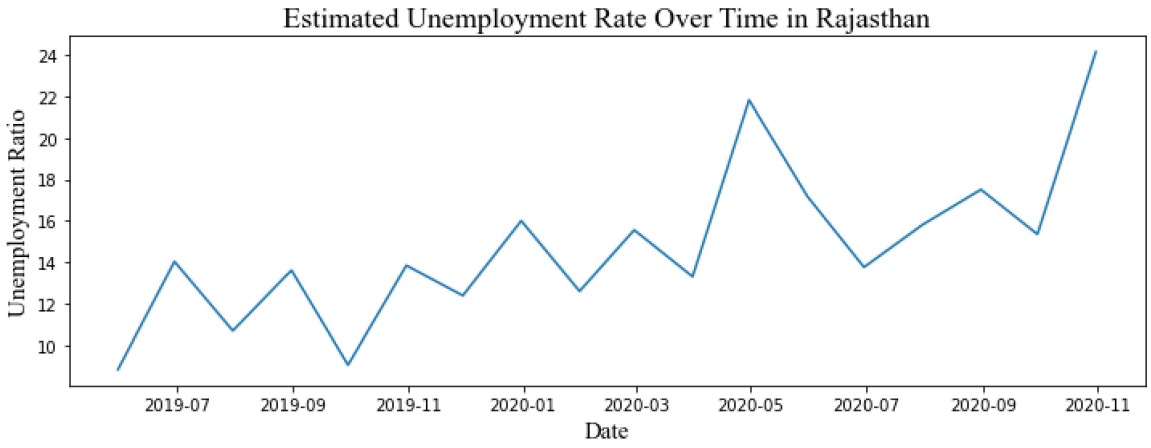
Punjab

In [83]:
```
#define data
Punjab_df = final_data[final_data['State'] == 'Punjab']

# Aggregate duplicate values by taking the mean
Punjab_df = Punjab_df.groupby(Punjab_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Punjab_df, x = Punjab_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Punjab',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



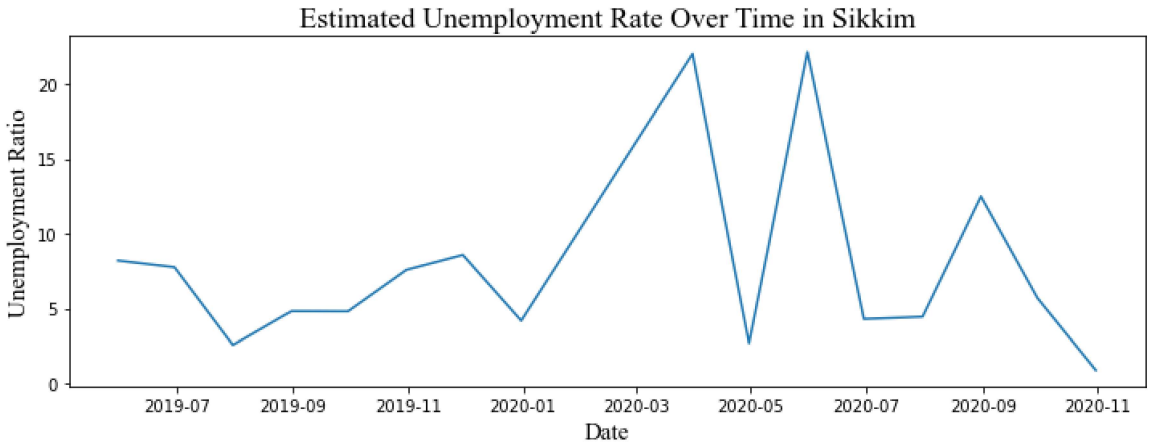
Rajasthan

In [84]:
```
#define data
Rajasthan_df = final_data[final_data['State'] == 'Rajasthan']

# Aggregate duplicate values by taking the mean
Rajasthan_df = Rajasthan_df.groupby(Rajasthan_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Rajasthan_df, x = Rajasthan_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Rajasthan',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



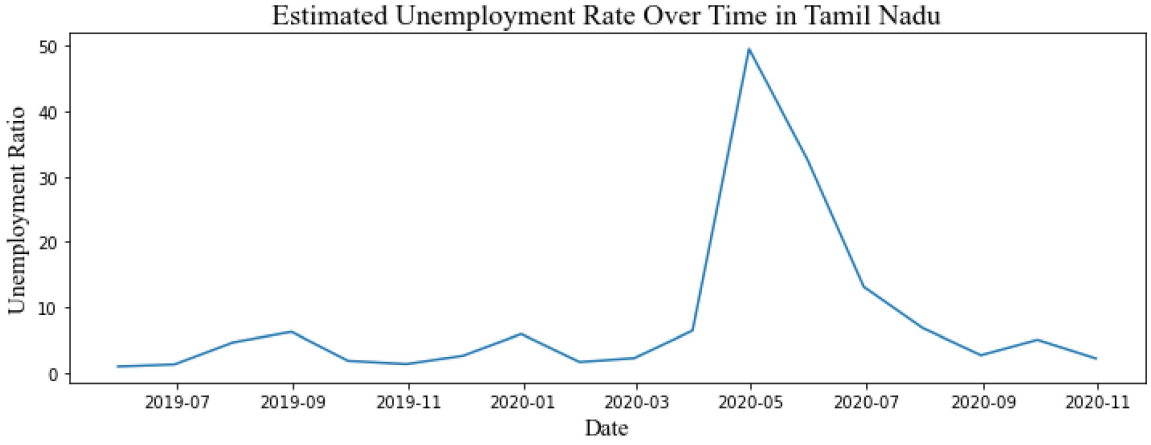
Sikkim

In [85]:
```
#define data
Sikkim_df = final_data[final_data['State'] == 'Sikkim']

# Aggregate duplicate values by taking the mean
Sikkim_df = Sikkim_df.groupby(Sikkim_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Sikkim_df, x = Sikkim_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Sikkim',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



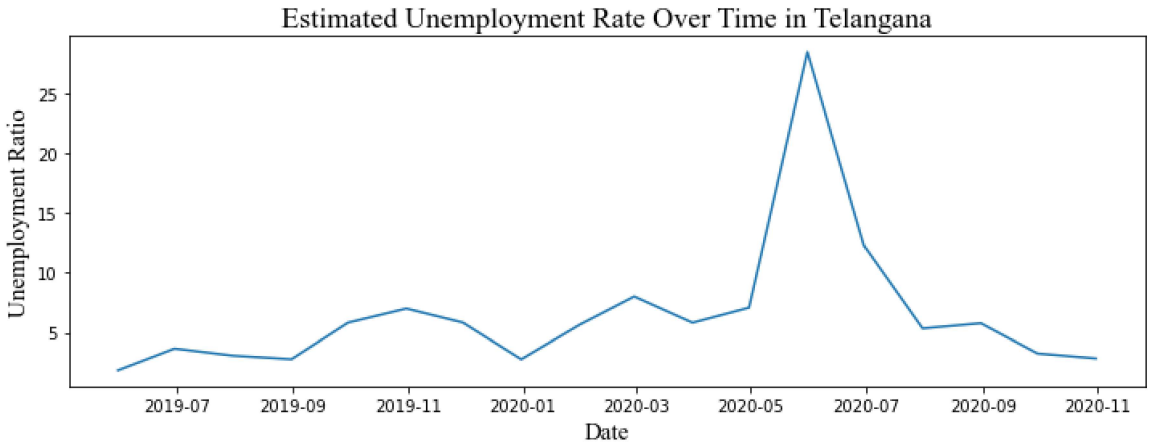
Tamil Nadu

In [86]:
```
#define data
Tamil_Nadu_df = final_data[final_data['State'] == 'Tamil Nadu']

# Aggregate duplicate values by taking the mean
Tamil_Nadu_df = Tamil_Nadu_df.groupby(Tamil_Nadu_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Tamil_Nadu_df, x = Tamil_Nadu_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Tamil Nadu',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



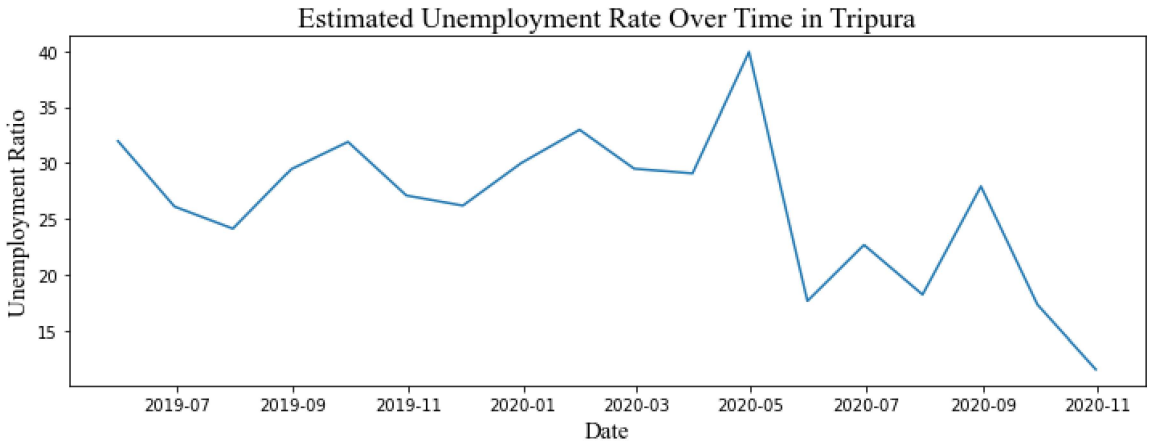
Telangana

In [87]:
```
#define data
Telangana_df = final_data[final_data['State'] == 'Telangana']

# Aggregate duplicate values by taking the mean
Telangana_df = Telangana_df.groupby(Telangana_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Telangana_df, x = Telangana_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Telangana',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



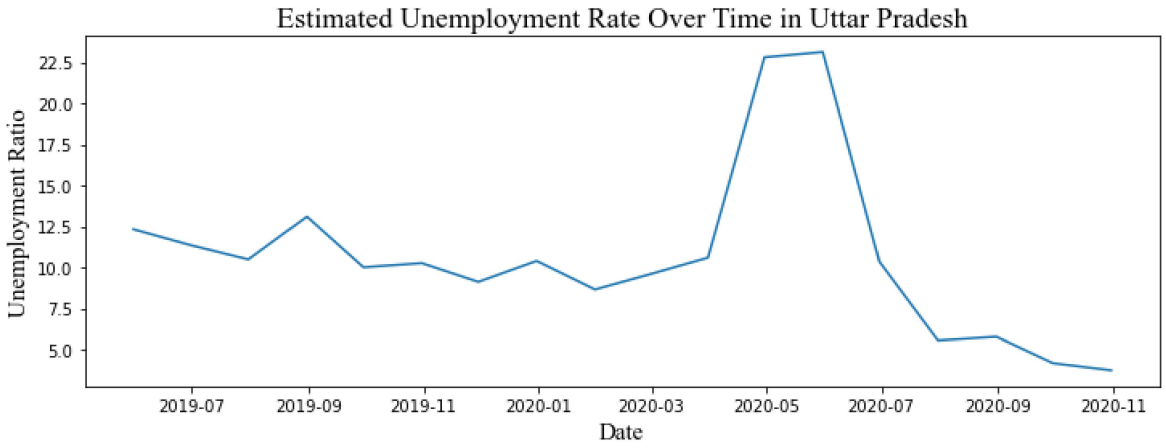
Tripura

In [88]:
```
#define data
Tripura_df = final_data[final_data['State'] == 'Tripura']

# Aggregate duplicate values by taking the mean
Tripura_df = Tripura_df.groupby(Tripura_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Tripura_df, x = Tripura_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Tripura',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



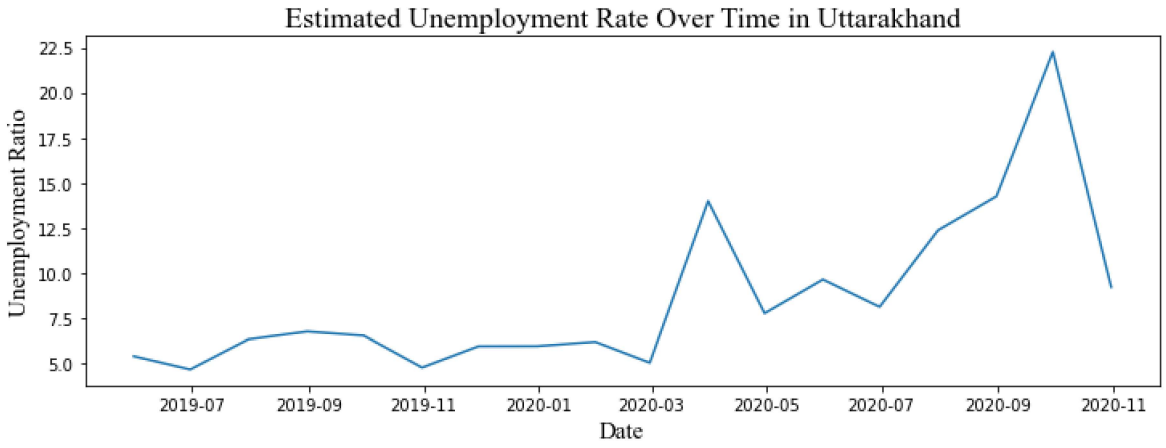
Uttar Pradesh

In [89]:
```
#define data
Uttar_Pradesh_df = final_data[final_data['State'] == 'Uttar Pradesh']

# Aggregate duplicate values by taking the mean
Uttar_Pradesh_df = Uttar_Pradesh_df.groupby(Uttar_Pradesh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Uttar_Pradesh_df, x = Uttar_Pradesh_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Uttar Pradesh',fontsize=(18),fontname= 'Times New Roman'
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



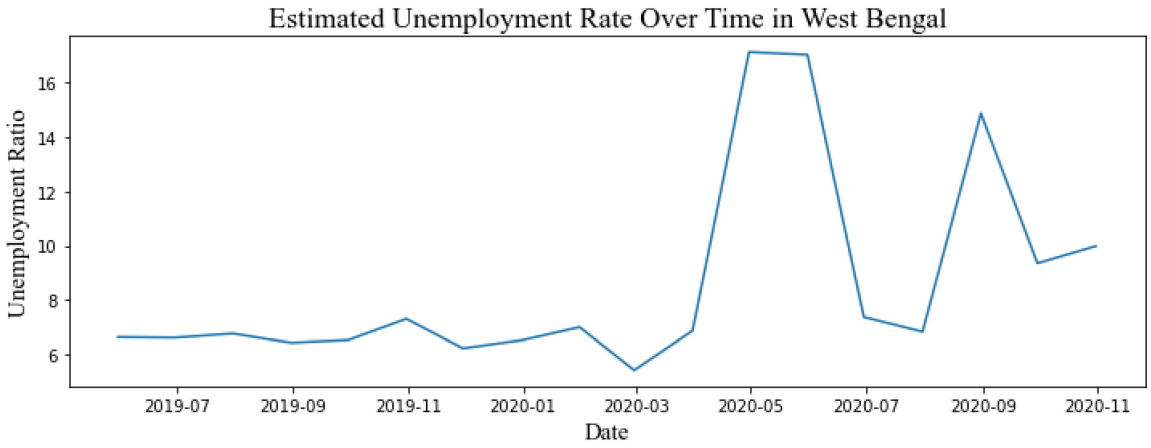
Uttarakhand

In [90]:
```
#define data
Uttarakhand_df = final_data[final_data['State'] == 'Uttarakhand']

# Aggregate duplicate values by taking the mean
Uttarakhand_df = Uttarakhand_df.groupby(Uttarakhand_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Uttarakhand_df, x = Uttarakhand_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Uttarakhand',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



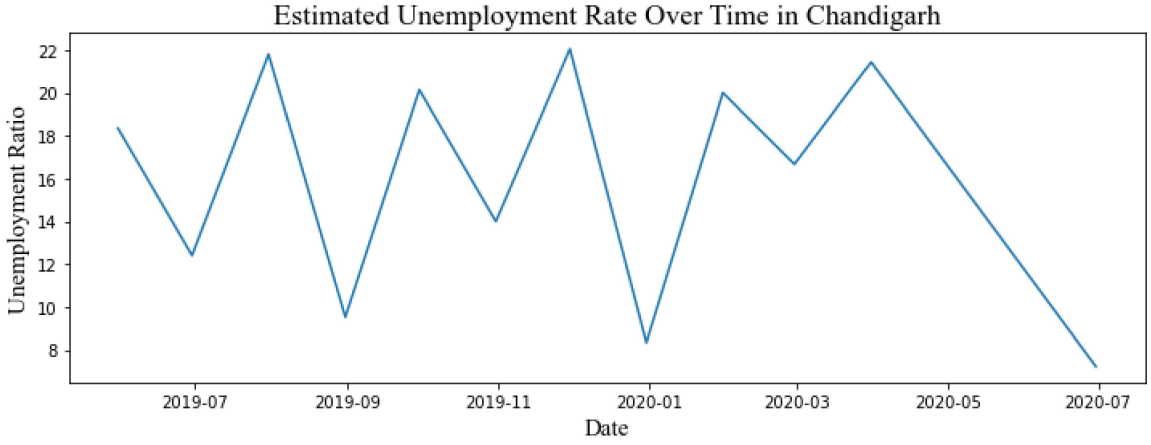
West Bengal

In [91]:
```
#define data
West_Bengal_df = final_data[final_data['State'] == 'West Bengal']

# Aggregate duplicate values by taking the mean
West_Bengal_df = West_Bengal_df.groupby(West_Bengal_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = West_Bengal_df, x = West_Bengal_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in West Bengal',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



Chandigarh

In [92]:
```
#define data
Chandigarh_df = final_data[final_data['State'] == 'Chandigarh']

# Aggregate duplicate values by taking the mean
Chandigarh_df = Chandigarh_df.groupby(Chandigarh_df.index).mean()

#plot data
plt.figure(figsize=(12,4))
sns.lineplot(data = Chandigarh_df, x = Chandigarh_df.index, y= 'Estimated Unemployment Rate (%)')
plt.title('Estimated Unemployment Rate Over Time in Chandigarh',fontsize=(18),fontname= 'Times New Roman')
plt.xlabel('Date',fontsize=(15),fontname= 'Times New Roman')
plt.ylabel('Unemployment Ratio',fontsize=(15),fontname= 'Times New Roman')
plt.show()
```



Inferences:

1.Unemplyment rate sharply increases from March 2020 and decreases by June 2020. Most of the states are in similar situation.
2.In case of Goa, Uttarakhand it continues untill October,2020.It clearly implies that tourism-based states had faced rapid decrease in employment.
3.In case of Rajasthan unemployment rate increases over time.
4.There is a volatile Unemployment Rate in Chandigarh.

In [93]:
```
final_data.Region.unique()
```

Out[93]:
```
array(['South', 'Northeast', 'East', 'West', 'North'], dtype=object)
```

In [94]:
```
South_df = final_data[final_data['Region'] == 'South']
South_df = South_df.groupby(South_df.index)['State','Estimated Unemployment Rate (%)'].sum()
```

In [95]:
```
South_df.head(2)
```

Out[95]:

Date	Estimated Unemployment Rate (%)
2019-05-31	41.36
2019-06-30	43.33

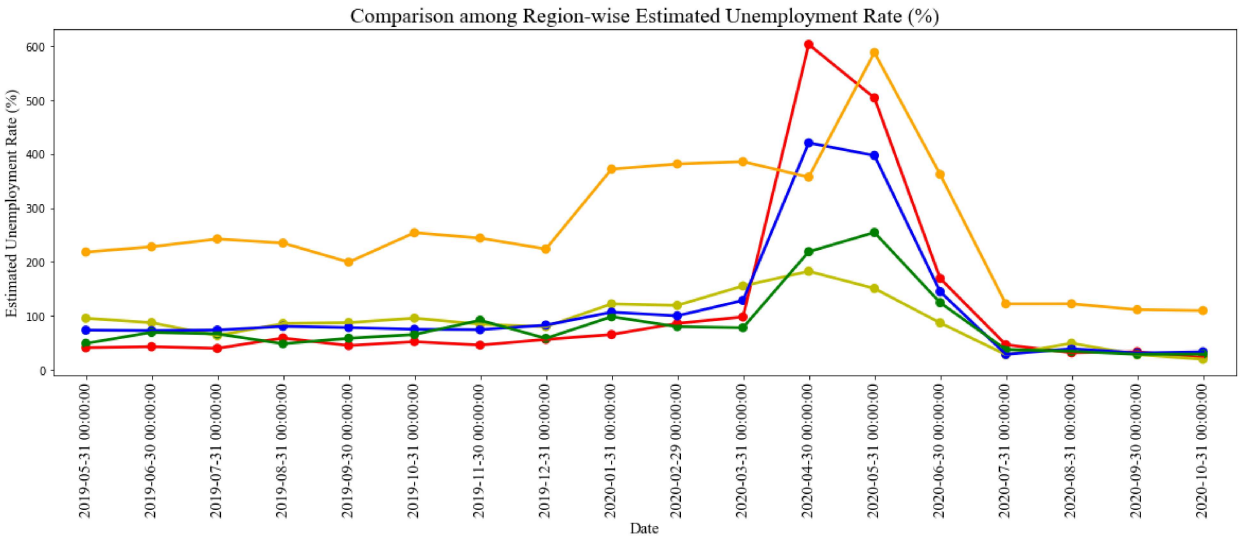
In [96]:
```
Northeast_df = final_data[final_data['Region'] == 'Northeast']
Northeast_df = Northeast_df.groupby(Northeast_df.index)['State','Estimated Unemployment Rate (%)'].sum()
```

In [97]:
```
East_df = final_data[final_data['Region'] == 'East']
East_df = East_df.groupby(East_df.index)['State','Estimated Unemployment Rate (%)'].sum()
```

In [98]:
```
West_df = final_data[final_data['Region'] == 'West']
West_df = West_df.groupby(West_df.index)['State','Estimated Unemployment Rate (%)'].sum()
```

In [99]:
```
North_df = final_data[final_data['Region'] == 'North']
North_df = North_df.groupby(North_df.index)['State', 'Estimated Unemployment Rate (%)'].sum()
```

In [100…
```
plt.figure(figsize=(20,6))
sns.pointplot(x = South_df.index, y= South_df['Estimated Unemployment Rate (%)'], color='r')
sns.pointplot(x = Northeast_df.index, y= Northeast_df['Estimated Unemployment Rate (%)'], color='y')
sns.pointplot(x = East_df.index, y= East_df['Estimated Unemployment Rate (%)'], color='b')
sns.pointplot(x = West_df.index, y= West_df['Estimated Unemployment Rate (%)'], color='g')
sns.pointplot(x = North_df.index, y= North_df['Estimated Unemployment Rate (%)'], color='orange')
plt.title('Comparison among Region-wise Estimated Unemployment Rate (%)',fontsize=(20),fontname= 'Times New
plt.xticks(rotation=90,fontsize=(15),fontname= 'Times New Roman')
plt.xlabel('Date', fontsize= (15), fontname = "Times New Roman")
plt.ylabel('Estimated Unemployment Rate (%)',fontsize=(15),fontname= 'Times New Roman')
#plt.legend(labels = ['South', 'Northeast', 'East', 'West', 'North'])
plt.show()
```



Inferences:

1.As the Lockdown started the Unemployment rate has increased across India.
2.The unemployment rate starts to rise from March,2020 and continues till July,2020.
3.Across all the regions,the highest unemployment rate is found in the North and the lowest in the Northeast region.

----------End----------