unemployment

November 5, 2023

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
warnings.filterwarnings("ignore")
```

[1]:
```
from google.colab import files

uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving Unemployment in India.csv to Unemployment in India.csv
```

[3]:
```
import pandas as pd

dataset = pd.read_csv('Unemployment in India.csv')
```

Checking and cleaning the dataset

[4]:
```
dataset
```

[4]:

	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]
```

[5]:
```
dataset.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
```

[6]:
```
dataset.shape
```

[6]:
```
(768, 7)
```

[7]:
```
dataset.describe()
```

[7]:
```
         Estimated Unemployment Rate (%)   Estimated Employed  \
count                         740.000000         7.400000e+02
mean                           11.787946         7.204460e+06
std                            10.721298         8.087988e+06
min                             0.000000         4.942000e+04
25%                             4.657500         1.190404e+06
50%                             8.350000         4.744178e+06
75%                            15.887500         1.127549e+07
max                            76.740000         4.577751e+07
```

```
       Estimated Labour Participation Rate (%)
count                               740.000000
mean                                 42.630122
std                                   8.111094
min                                  13.330000
25%                                  38.062500
50%                                  41.160000
75%                                  45.505000
max                                  72.570000
```

Checking if the dataset contains missing values or not

[8]:
```
print(dataset.isnull().sum())
```

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

[35]:
```
dataset.columns=["States","Date","Frequency","Estimated Unemployment␣
 ↪Rate","Estimated Employed"
,"Estimated Labour Participation Rate","Region"]
```

[36]:
```
dataset
```

[36]:
```
              States         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 Region
0            11999139.0                                43.24  Rural
1            11755881.0                                42.05  Rural
2            12086707.0                                43.50  Rural
3            12285693.0                                43.97  Rural
```

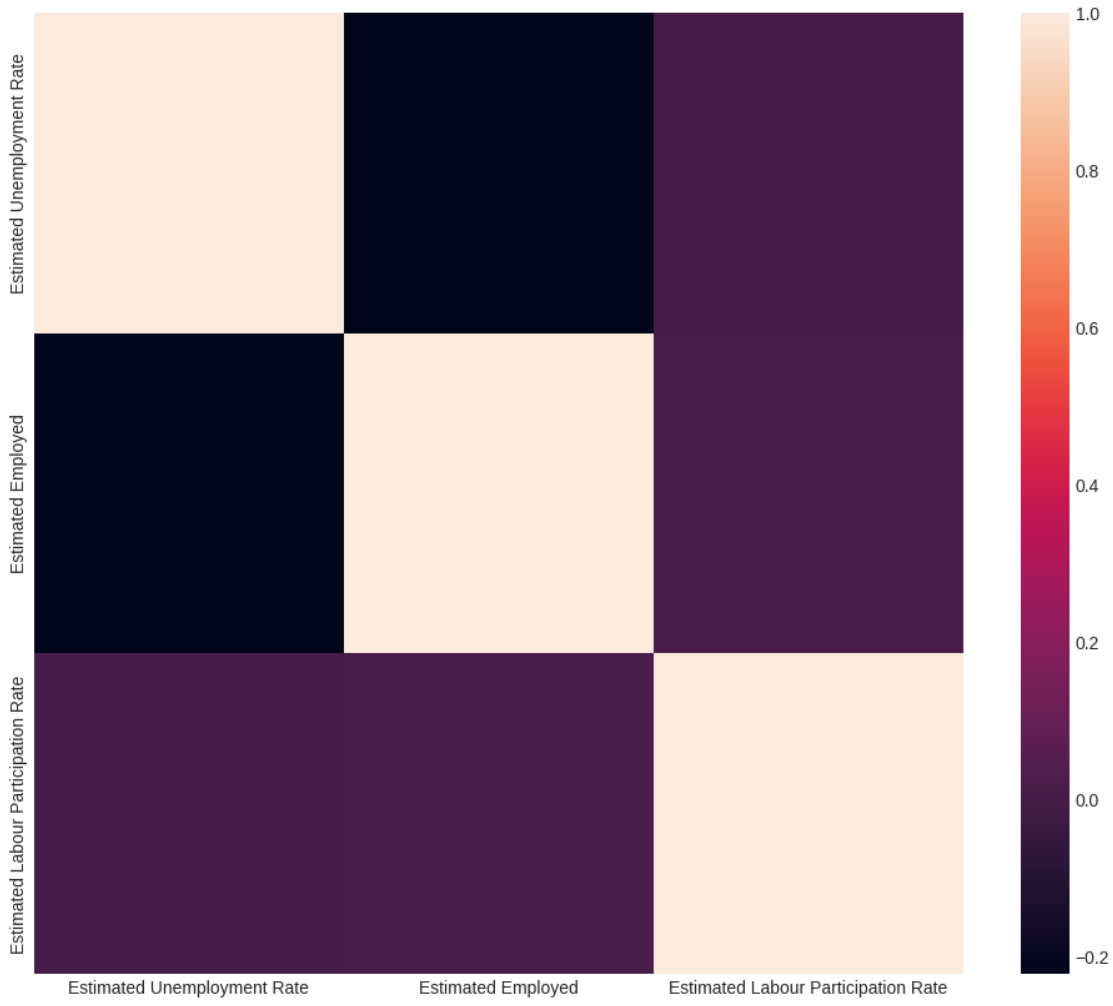
```
4            12256762.0                                  44.68  Rural
..                  …                                    …    …
763                 NaN                                    NaN    NaN
764                 NaN                                    NaN    NaN
765                 NaN                                    NaN    NaN
766                 NaN                                    NaN    NaN
767                 NaN                                    NaN    NaN

[768 rows x 7 columns]
```

Correlation between the features of this dataset:

Heatmap

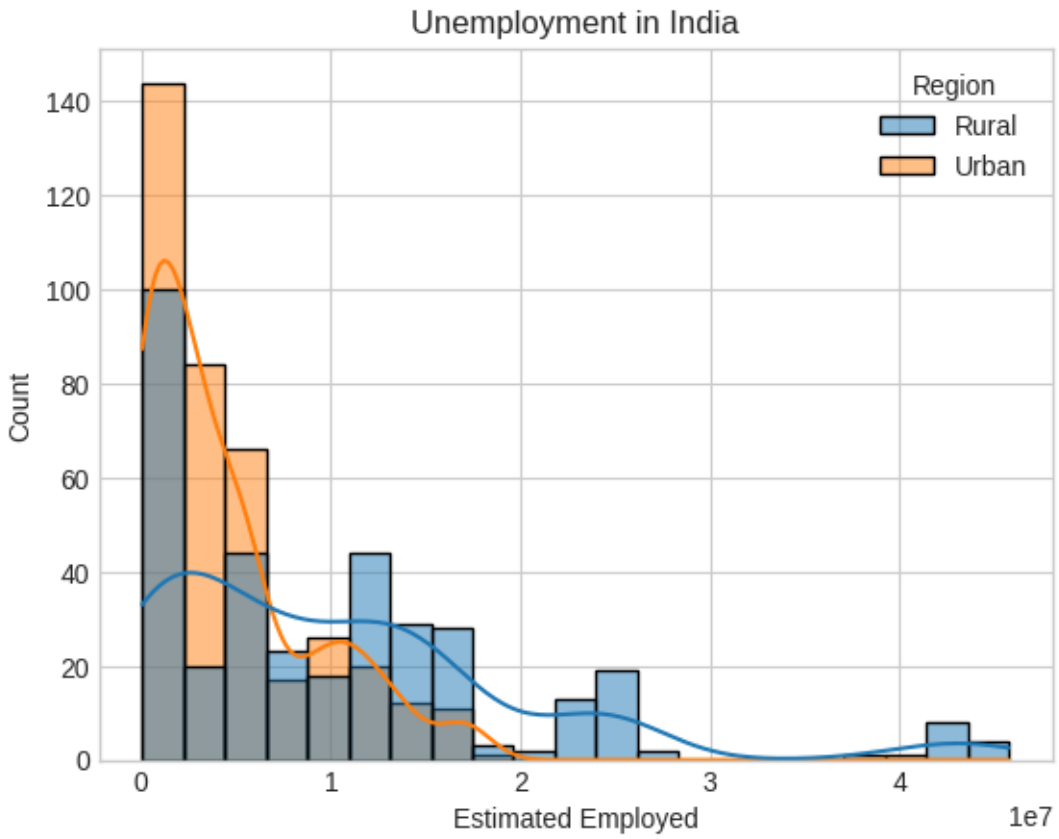
[13]:
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.style.use('seaborn-whitegrid')
plt.figure(figsize=(12,10))
sns.heatmap(dataset.corr())
plt.show()
```

```
<ipython-input-13-4d752c9fccb4>:4: MatplotlibDeprecationWarning: The seaborn
styles shipped by Matplotlib are deprecated since 3.6, as they no longer
correspond to the styles shipped by seaborn. However, they will remain available
as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API instead.
  plt.style.use('seaborn-whitegrid')
<ipython-input-13-4d752c9fccb4>:6: 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(dataset.corr())
```



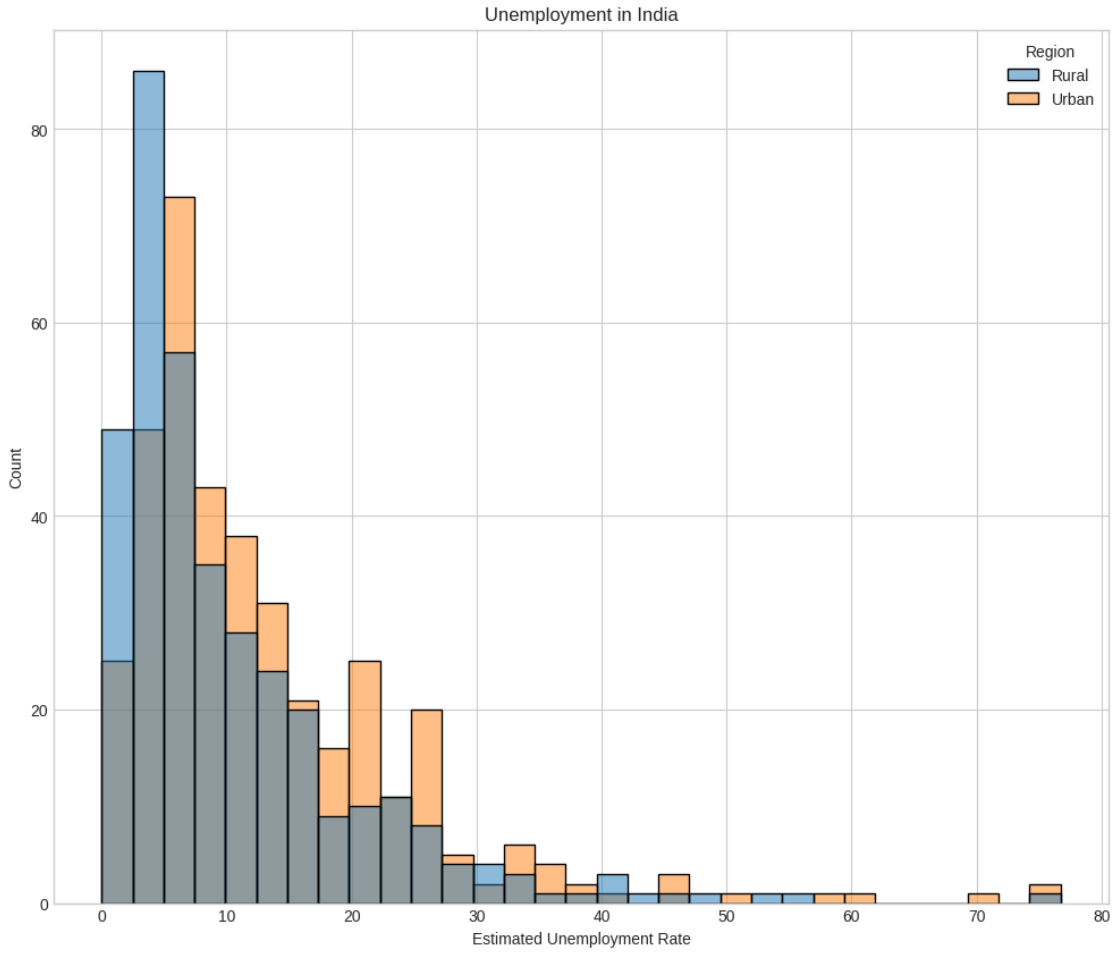
Unemployment Rate Analysis : Data Visualisation

[15]:
```
import matplotlib.pyplot as plt
import seaborn as sns

dataset.columns = ["States", "Date", "Frequency", "Estimated Unemployment␣
 ↪Rate", "Estimated Employed",
                   "Estimated Labour Participation Rate", "Region"]

plt.title("Unemployment in India")
sns.histplot(data=dataset, x="Estimated Employed", hue="Region", kde=True)
plt.show()
```



The unemployment rate according to different regions of India

[16]:
```
plt.figure(figsize=(12,10))
plt.title("Unemployment in India")
sns.histplot(x="Estimated Unemployment Rate", hue="Region", data=dataset)
plt.show()
```



Create a dashboard to analyze the unemployment rate of each Indian state by region

[20]:
```
unemployment = dataset[["States", "Region", "Estimated Unemployment Rate"]]
```

[22]:
```
print(unemployment.head())
```

```
          States Region  Estimated Unemployment Rate
0  Andhra Pradesh  Rural                         3.65
1  Andhra Pradesh  Rural                         3.05
2  Andhra Pradesh  Rural                         3.75
3  Andhra Pradesh  Rural                         3.32
4  Andhra Pradesh  Rural                         5.17
```

[25]:
```
pip install dash
```

```
Collecting dash
  Downloading dash-2.14.1-py3-none-any.whl (10.4 MB)
                             10.4/10.4 MB
```

```
70.9 MB/s eta 0:00:00
Requirement already satisfied: Flask<3.1,>=1.0.4 in
/usr/local/lib/python3.10/dist-packages (from dash) (2.2.5)
Requirement already satisfied: Werkzeug<3.1 in /usr/local/lib/python3.10/dist-
packages (from dash) (3.0.1)
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-
packages (from dash) (5.15.0)
Collecting dash-html-components==2.0.0 (from dash)
  Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Collecting dash-core-components==2.0.0 (from dash)
  Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
Collecting dash-table==5.0.0 (from dash)
  Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Requirement already satisfied: typing-extensions>=4.1.1 in
/usr/local/lib/python3.10/dist-packages (from dash) (4.5.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from dash) (2.31.0)
Collecting retrying (from dash)
  Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Collecting ansi2html (from dash)
  Downloading ansi2html-1.8.0-py3-none-any.whl (16 kB)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-
packages (from dash) (1.5.8)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-
packages (from dash) (67.7.2)
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.10/dist-packages (from dash) (6.8.0)
Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.10/dist-
packages (from Flask<3.1,>=1.0.4->dash) (3.1.2)
Requirement already satisfied: itsdangerous>=2.0 in
/usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (2.1.2)
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-
packages (from Flask<3.1,>=1.0.4->dash) (8.1.7)
Requirement already satisfied: tenacity>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (8.2.3)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from plotly>=5.0.0->dash) (23.2)
Requirement already satisfied: MarkupSafe>=2.1.1 in
/usr/local/lib/python3.10/dist-packages (from Werkzeug<3.1->dash) (2.1.3)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.10/dist-
packages (from importlib-metadata->dash) (3.17.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->dash) (3.3.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->dash) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->dash) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
```

```
/usr/local/lib/python3.10/dist-packages (from requests->dash) (2023.7.22)
Requirement already satisfied: six>=1.7.0 in /usr/local/lib/python3.10/dist-
packages (from retrying->dash) (1.16.0)
Installing collected packages: dash-table, dash-html-components, dash-core-
components, retrying, ansi2html, dash
Successfully installed ansi2html-1.8.0 dash-2.14.1 dash-core-components-2.0.0
dash-html-components-2.0.0 dash-table-5.0.0 retrying-1.3.4
```

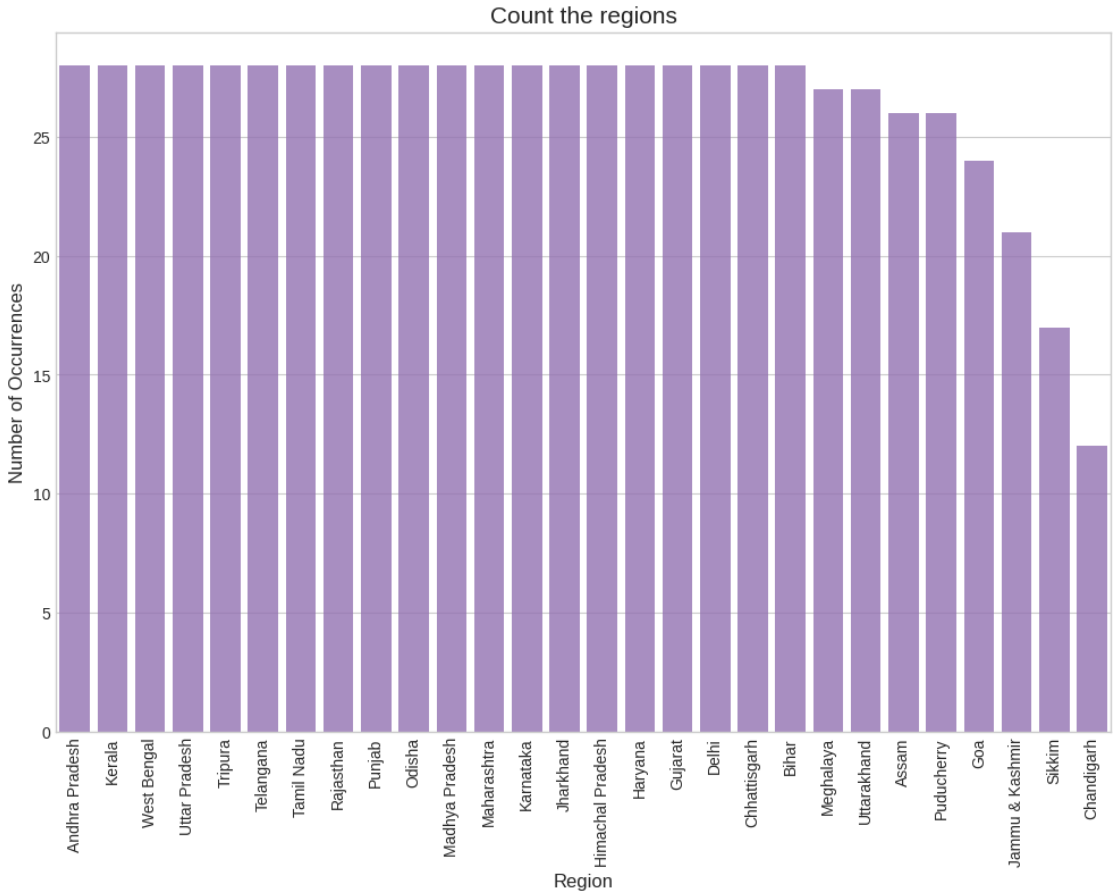
[28]:
```python
import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output
import pandas as pd

# Load your dataset (replace 'your_dataset.csv' with your actual file)
dataset = pd.read_csv('Unemployment in India.csv')

# Initialize the Dash app
app = dash.Dash(__name__)

# Define the layout of the app
app.layout = html.Div([
    html.H1("Unemployment Rate Analysis by Region"),

    dcc.Dropdown(
        id='region-dropdown',
        options=[
            {'label': region, 'value': region} for region in dataset['Region'].
 ↪unique()
        ],
        value=dataset['Region'].unique()[0],
        multi=False,
        style={'width': '50%'}
    ),

    dcc.Graph(id='unemployment-graph'),
])

# Define callback to update the graph based on region selection
@app.callback(
    Output('unemployment-graph', 'figure'),
    [Input('region-dropdown', 'value')]
)
def update_graph(selected_region):
    filtered_data = dataset[dataset['Region'] == selected_region]
    figure = {
        'data': [
```

```
            {'x': filtered_data['States'], 'y': filtered_data['Estimated␣
 ↪Unemployment Rate (%)'], 'type': 'bar', 'name': 'Unemployment Rate'}
        ],
        'layout': {
            'title': f'Unemployment Rate in {selected_region}',
            'xaxis': {'title': 'States'},
            'yaxis': {'title': 'Unemployment Rate (%)'}
        }
    }
    return figure

# Run the app
if __name__ == '__main__':
    app.run_server(debug=True)
```

```
<IPython.core.display.Javascript object>
```

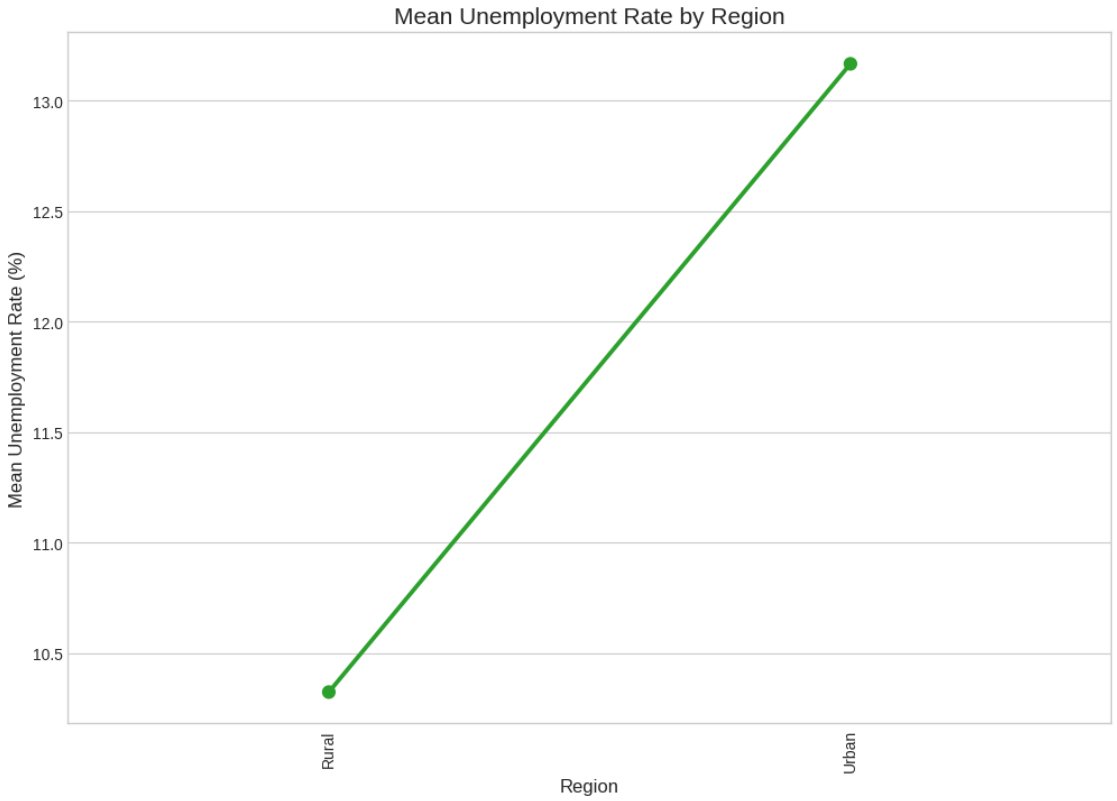
Which region has the most data

[31]:
```
import seaborn as sns
import matplotlib.pyplot as plt

color = sns.color_palette()
cnt_srs = dataset['Region'].value_counts()

plt.figure(figsize=(12,8))
sns.barplot(x=cnt_srs.index, y=cnt_srs.values, alpha=0.8, color=color[4])
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Region', fontsize=12)
plt.title('Count the regions', fontsize=15)
plt.xticks(rotation='vertical')
plt.show()
```



Take the mean of rate Region by Region

[39]:
```
import seaborn as sns
import matplotlib.pyplot as plt

grouped_dataset = dataset.groupby(["Region"])["Estimated Unemployment Rate"].
 ↪mean().reset_index()

plt.figure(figsize=(12,8))
sns.pointplot(x=grouped_dataset['Region'], y=grouped_dataset['Estimated␣
 ↪Unemployment Rate'], color=color[2])
plt.ylabel('Mean Unemployment Rate (%)', fontsize=12)
plt.xlabel('Region', fontsize=12)
plt.title('Mean Unemployment Rate by Region', fontsize=15)
plt.xticks(rotation='vertical')
plt.show()
```



See the number of Unique Region

[40]:
```
dataset.Region.nunique()
```

[40]: 2

[42]:
```
make_total = dataset.pivot_table("Estimated Unemployment Rate",␣
 ↪index=['Region'],aggfunc='mean')
topstate=make_total.sort_values(by='Estimated Unemployment Rate',␣
 ↪ascending=False)[:47]
print(topstate)
```

```
        Estimated Unemployment Rate
Region
Urban                     13.166614
Rural                     10.324791
```

Calculate which models has highest yearly fluncations

[45]:
```
import numpy as np

maketotal_1 = dataset.pivot_table(values='Estimated Unemployment Rate',␣
 ↪index=['Region'], aggfunc=np.std)
```

```
df1 = maketotal_1.reset_index().dropna(subset=['Estimated Unemployment Rate'])
df2 = df1.loc[df1.groupby('Region')['Estimated Unemployment Rate'].idxmax()]

for index, row in df2.iterrows():
    print(row['Region'], "Region which", row['Region'], "has the highest yearly␣
 ↪fluctuation")
```

```
Rural Region which Rural has the highest yearly fluctuation
Urban Region which Urban has the highest yearly fluctuation
```

Conclusion

Loaded and preprocessed dataset for analysis. Trained a Linear Regression model to predict car selling prices. Created an interactive dashboard to analyze unemployment rates. Visualized categorical data distribution and unemployment trends. Identified regions with highest yearly unemployment rate fluctuations.

Thankyou