NAME: NERELLA VENKATA RADHAKRISHNA

ID: 190031187

PRACTICAL-4

In [ ]:

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

In [ ]:

```
data_2015=pd.read_csv('2015.csv')
data_2016=pd.read_csv('2016.csv')
data_2017=pd.read_csv('2017.csv')
```

In [ ]:

```
data_2015.columns= [each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for
data_2016.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for e
data_2017.columns=[each.replace("."," ") for each in data_2017.columns]
data_2017.columns=[each.split()[0] if(len(each.split())>2) else each.replace(" ","_") for e
```

In [ ]:

```
data_2015.head()
```

Out[17]:

	Country	Region	Happiness_Rank	Happiness_Score	Standard_Error	Economy	Family
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261

In [ ]:

```
print("Are There Missing Data? :",data_2015.isnull().any().any())
print(data_2015.isnull().sum())
```

```
Are There Missing Data? : False
Country                          0
Region                           0
Happiness Rank                   0
Happiness Score                  0
Standard Error                   0
Economy (GDP per Capita)         0
Family                           0
Health (Life Expectancy)         0
Freedom                          0
Trust (Government Corruption)    0
Generosity                       0
Dystopia Residual                0
dtype: int64
```

In [ ]:

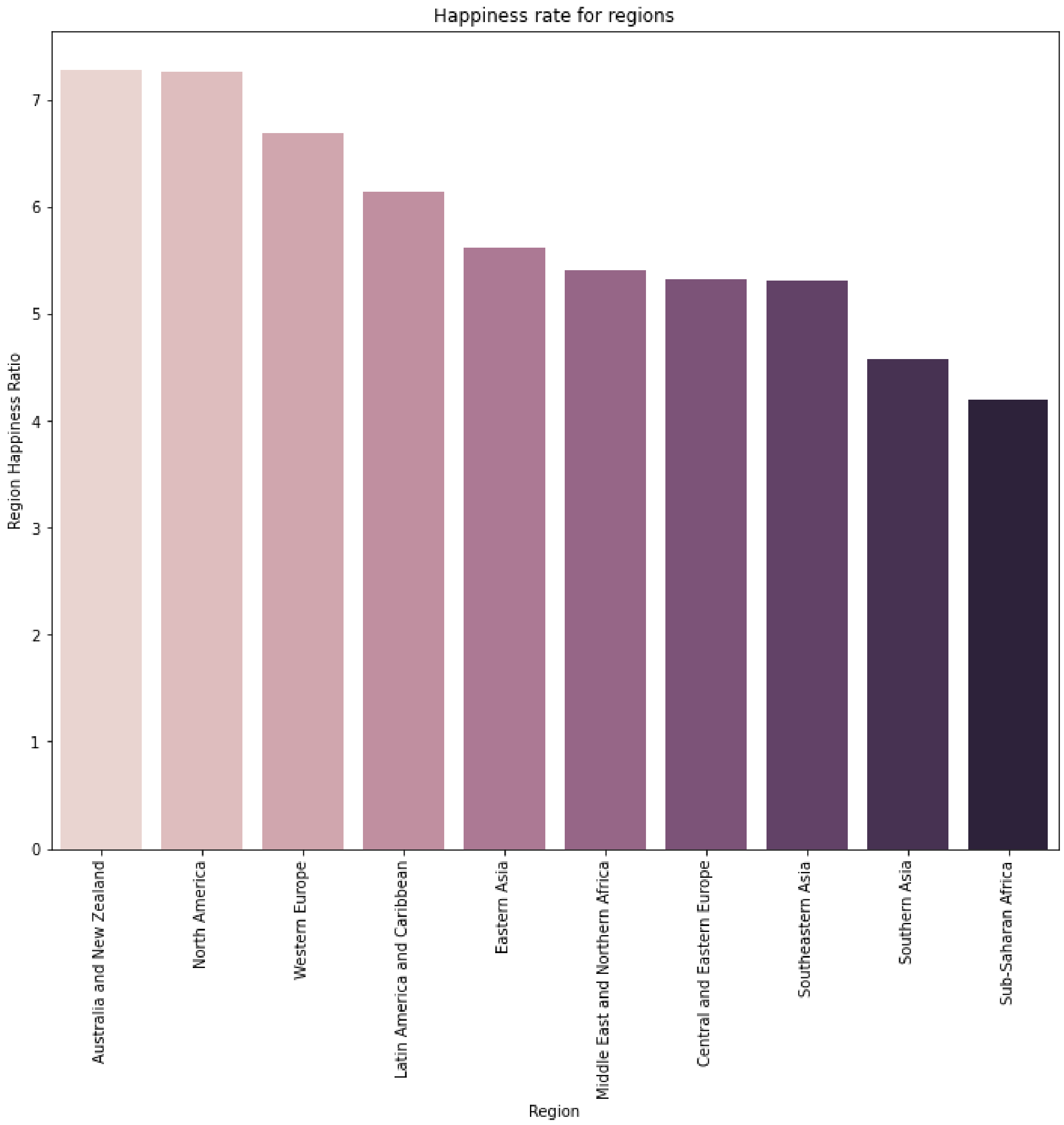
```
region_lists=list(data_2015['Region'].unique())
region_happiness_ratio=[]
for each in region_lists:
    region=data_2015[data_2015['Region']==each]
    region_happiness_rate=sum(region.Happiness_Score)/len(region)
    region_happiness_ratio.append(region_happiness_rate)

data=pd.DataFrame({'region':region_lists,'region_happiness_ratio':region_happiness_ratio})
new_index=(data['region_happiness_ratio'].sort_values(ascending=False)).index.values
sorted_data = data.reindex(new_index)

sorted_data
```

Out[17]:

	region	region_happiness_ratio
2	Australia and New Zealand	7.285000
1	North America	7.273000
0	Western Europe	6.689619
4	Latin America and Caribbean	6.144682
7	Eastern Asia	5.626167
3	Middle East and Northern Africa	5.406900
6	Central and Eastern Europe	5.332931
5	Southeastern Asia	5.317444
9	Southern Asia	4.580857
8	Sub-Saharan Africa	4.202800

In [ ]:

```
#BarPlot
plt.figure(figsize=(12,10))
sns.barplot(x=sorted_data['region'], y=sorted_data['region_happiness_ratio'])

plt.xticks(rotation= 90)
plt.xlabel('Region')
plt.ylabel('Region Happiness Ratio')
plt.title('Happiness rate for regions')
plt.show()
```



In [ ]:

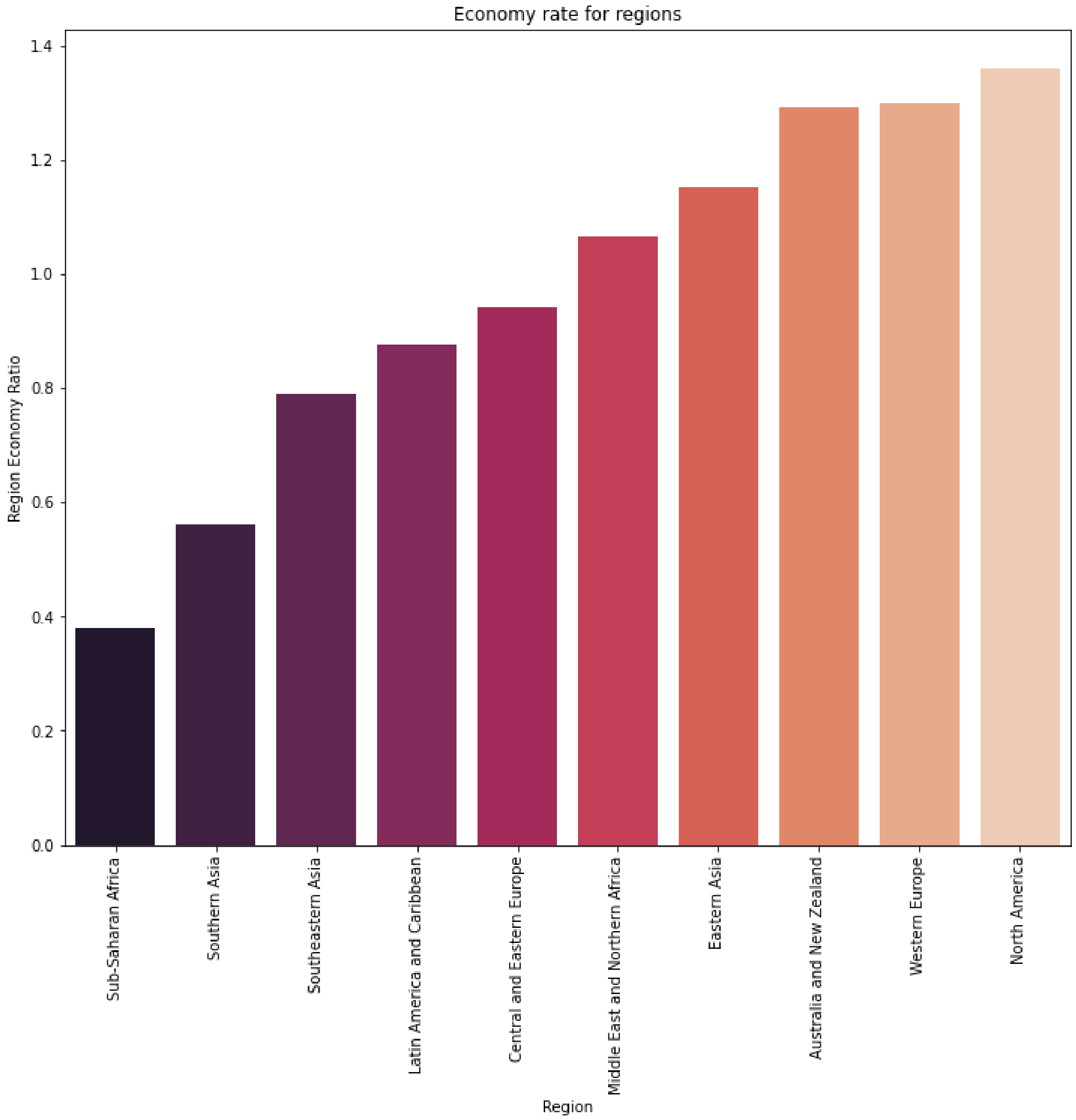
```
region_lists=list(data_2015['Region'].unique())
region_economy_ratio=[]
for each in region_lists:
    region=data_2015[data_2015['Region']==each]
    region_economy_rate=sum(region.Economy)/len(region)
    region_economy_ratio.append(region_economy_rate)

data_economy=pd.DataFrame({'region':region_lists,'region_economy_ratio':region_economy_rati
new_index_economy=(data_economy['region_economy_ratio'].sort_values(ascending=True)).index.
sorted_data_economy = data_economy.reindex(new_index_economy)
sorted_data_economy
```

Out[16]:

	region	region_economy_ratio
8	Sub-Saharan Africa	0.380473
9	Southern Asia	0.560486
5	Southeastern Asia	0.789054
4	Latin America and Caribbean	0.876815
6	Central and Eastern Europe	0.942438
3	Middle East and Northern Africa	1.066973
7	Eastern Asia	1.151780
2	Australia and New Zealand	1.291880
0	Western Europe	1.298596
1	North America	1.360400

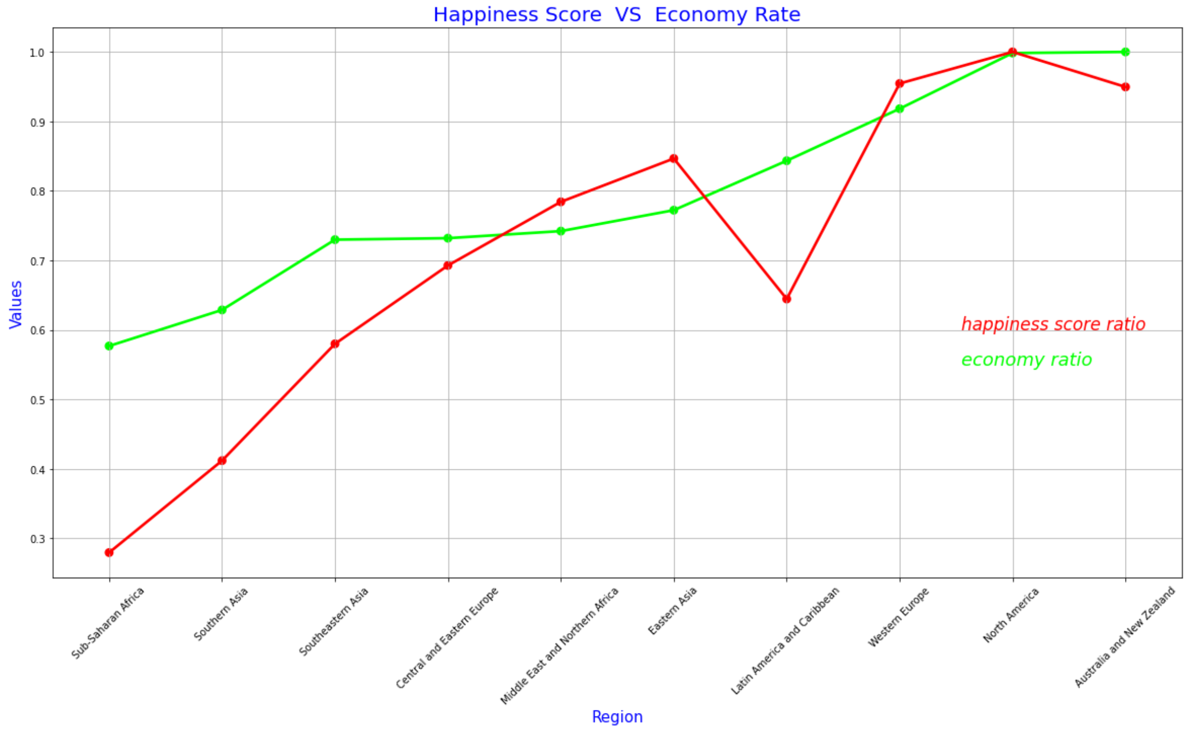
In [ ]:

```
#BarPlot
f,ax1 = plt.subplots(figsize =(12,10))
sns.barplot(x=sorted_data_economy['region'], y=sorted_data_economy['region_economy_ratio'],

plt.xticks(rotation= 90)
plt.xlabel('Region')
plt.ylabel('Region Economy Ratio')
plt.title('Economy rate for regions')
plt.show()
```



In [ ]:

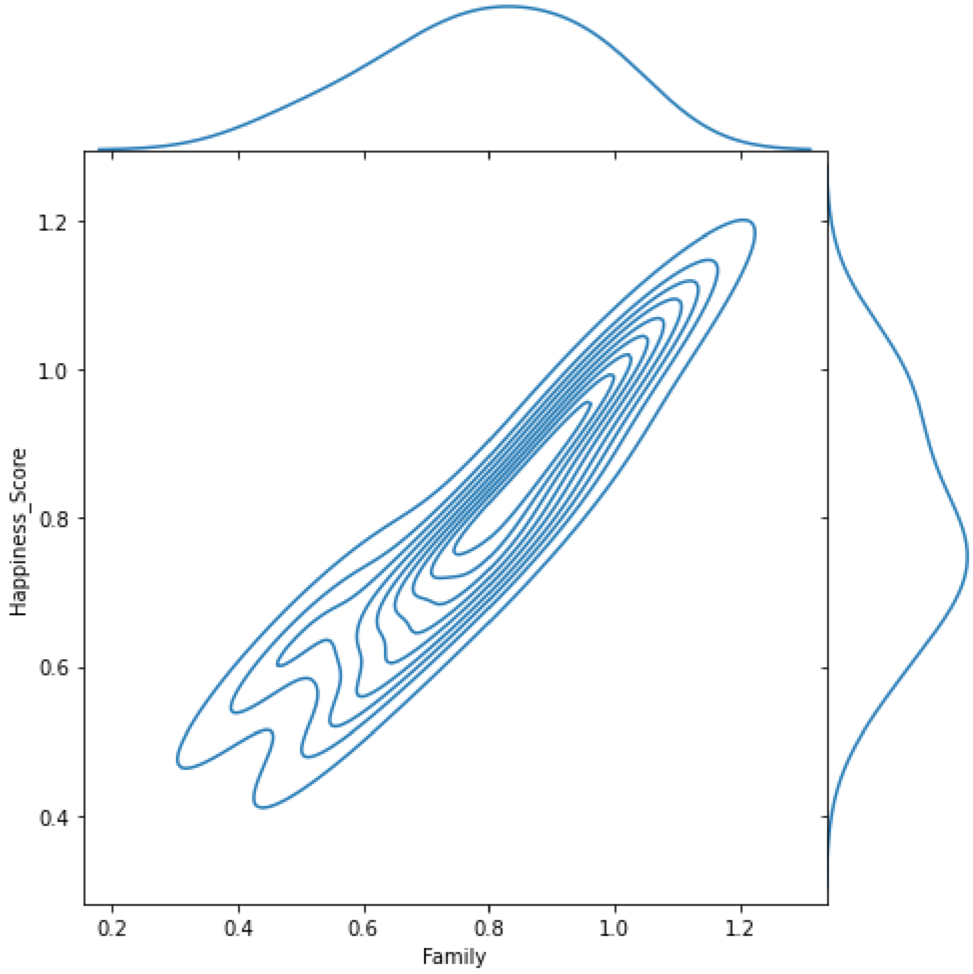
```
#PointPlot
sorted_data['region_happiness_ratio']=sorted_data['region_happiness_ratio']/max(sorted_data
sorted_data_economy['region_economy_ratio']=sorted_data_economy['region_economy_ratio']/max

data=pd.concat([sorted_data,sorted_data_economy['region_economy_ratio']],axis=1)
data.sort_values('region_happiness_ratio',inplace=True)


f,ax1 = plt.subplots(figsize =(20,10))
sns.pointplot(x='region',y='region_happiness_ratio',data=data,color='lime',alpha=0.8)
sns.pointplot(x='region',y='region_economy_ratio',data=data,color='red',alpha=0.8)
plt.text(7.55,0.6,'happiness score ratio',color='red',fontsize = 17,style = 'italic')
plt.text(7.55,0.55,'economy ratio',color='lime',fontsize = 18,style = 'italic')
plt.xticks(rotation=45)
plt.xlabel('Region',fontsize = 15,color='blue')
plt.ylabel('Values',fontsize = 15,color='blue')
plt.title('Happiness Score  VS  Economy Rate',fontsize = 20,color='blue')
plt.grid()
plt.show()
```



In [ ]:

```
#JointPlot
dataframe=pd.pivot_table(data_2015, index = 'Region', values=["Happiness_Score","Family"])

dataframe["Happiness_Score"]=dataframe["Happiness_Score"]/max(dataframe["Happiness_Score"])
dataframe["Family"]=dataframe["Family"]/max(dataframe["Family"])
sns.jointplot(dataframe.Family,dataframe.Happiness_Score,kind="kde",height=7,space=0)
plt.savefig('graph.png')
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/_decorators.py:43: FutureWarn
ing: Pass the following variables as keyword args: x, y. From version 0.12,
the only valid positional argument will be `data`, and passing other argumen
ts without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



In [ ]:

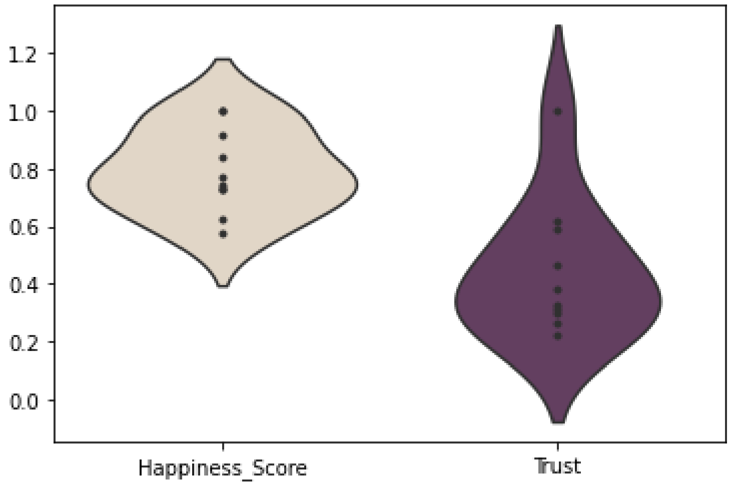
```
data_2015['Year']=2015
data_2016['Year']=2016
data_2017['Year']=2017

data_concat=pd.concat([data_2015,data_2016,data_2017],axis=0,sort = False)

df=pd.pivot_table(data_concat, index = 'Year', values="Happiness_Score")
df
```

Out[5]:

	Happiness_Score
Year	
2015	5.375734
2016	5.382185
2017	5.354019

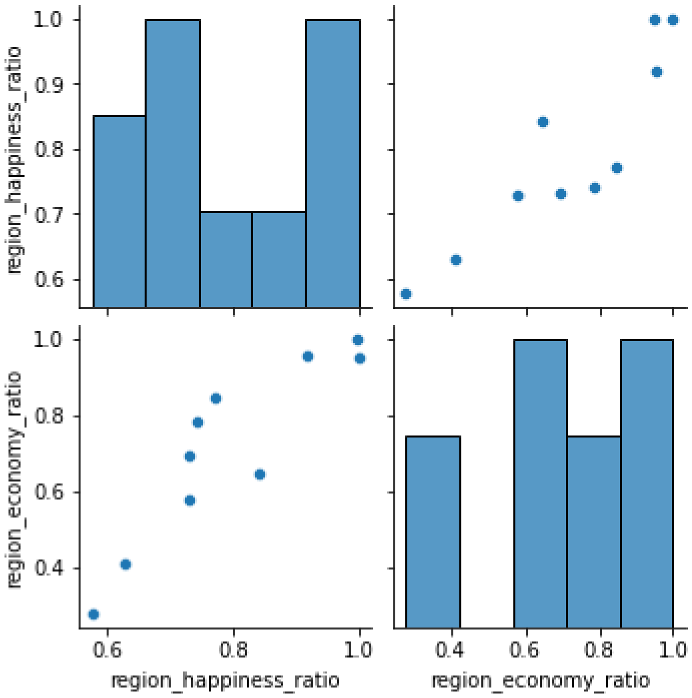
In [ ]:

```
#ViolinPlot
dataframe2=pd.pivot_table(data_2015, index = 'Region', values=["Happiness_Score","Trust"])
#to normalize
dataframe2["Happiness_Score"]=dataframe2["Happiness_Score"]/max(dataframe2["Happiness_Score
dataframe2["Trust"]=dataframe2["Trust"]/max(dataframe2["Trust"])
pal=sns.cubehelix_palette(2,rot=.5,dark=.3)
sns.violinplot(data=dataframe2, palette=pal, inner="points")
plt.show()
```



In [ ]:

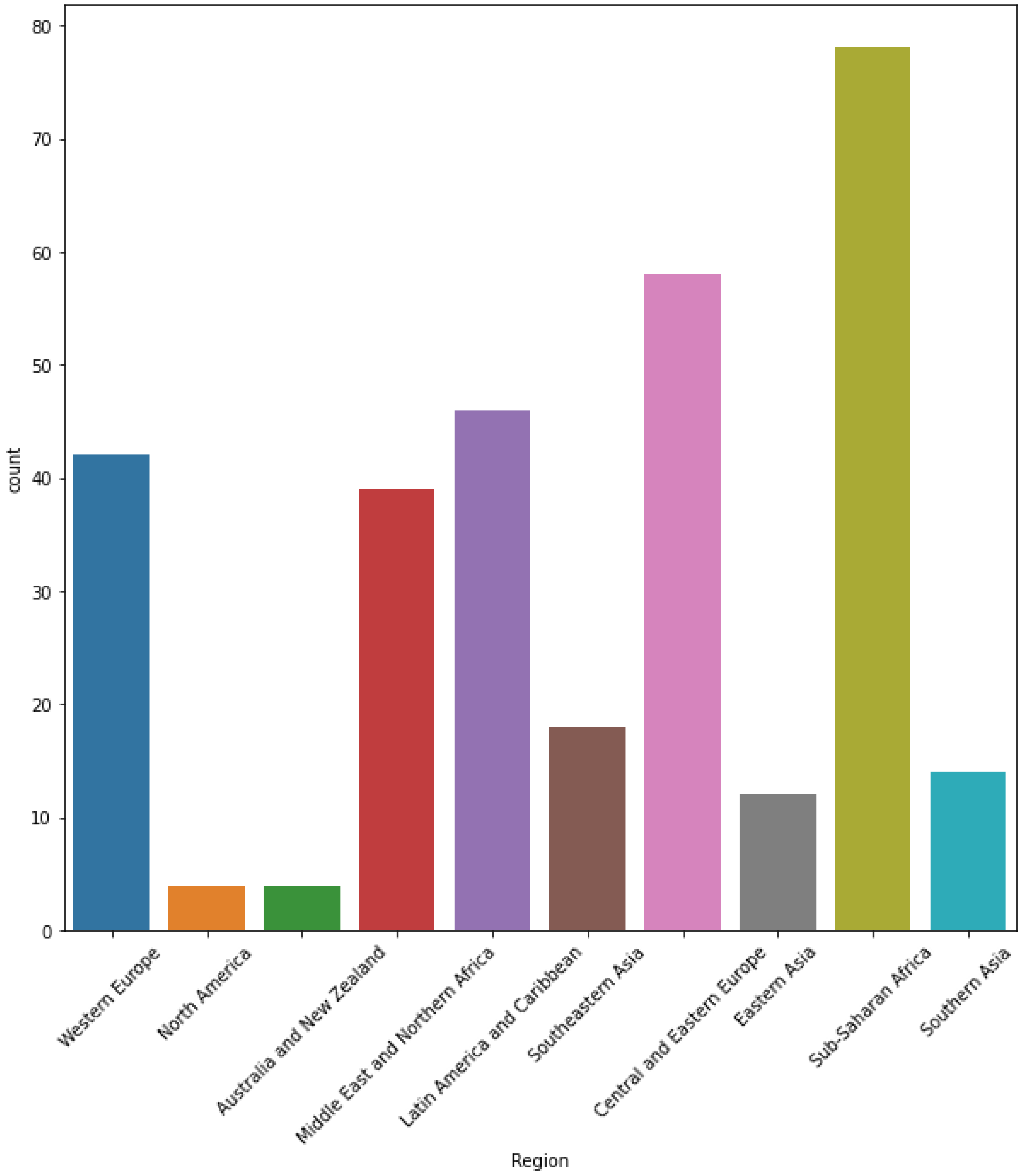
```
#Pair Plot
sns.pairplot(data)
plt.show()
```



In [ ]:

```
#CountPlot
f,ax = plt.subplots(figsize =(10,10))
sns.countplot(data_concat.Region,ax=ax)
plt.xticks(rotation= 45)
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/_decorators.py:43: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the
only valid positional argument will be `data`, and passing other arguments w
ithout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```


count
Western Europe
North America
Australia and New Zealand
Middle East and Northern Africa
Latin America and Caribbean
Southeastern Asia
Central and Eastern Europe
Eastern Asia
Sub-Saharan Africa
Southern Asia
Region