Global Temperature Analysis

This notebook contains a basic analysis through some visualizations of Global Temperaute and climate change

The analysis is broken up into 3 sections:

- Data Loading and Preparation.
- Exploration and visualization.
- Conclusion.

1. Data Loading and Preparation

1.1 Loading Modules

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from sklearn.metrics import mean_squared_error
from math import sqrt
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.tools as tls
import time
import nltk
import string
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc,accuracy_score,f1_score,precision_score,
from sklearn.linear_model import ElasticNet, Lasso,  BayesianRidge, LassoLarsIC,Rid
from sklearn.kernel_ridge import KernelRidge
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import RobustScaler
from sklearn.svm import LinearSVC
from sklearn.base import BaseEstimator, TransformerMixin, RegressorMixin, clone
from sklearn.model_selection import KFold, cross_val_score, train_test_split
from sklearn.metrics import precision_score,recall_score,auc
```

1.2 Loading Data

In [2]:

```
gltc = pd.read_csv("data/GlobalLandTemperaturesByCountry.csv")
```

In [3]:

```
global_temp = pd.read_csv("data/GlobalTemperatures.csv" )
```

In [4]:

```
global_temp_country = pd.read_csv("data/GlobalLandTemperaturesByCountry.csv")
```

1.3 Data Preparation

In [5]:

```
df = gltc
```

In [6]:

```
df.head()
```

Out[6]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
0	1743-11-01	4.384	2.294	Åland
1	1743-12-01	NaN	NaN	Åland
2	1744-01-01	NaN	NaN	Åland
3	1744-02-01	NaN	NaN	Åland
4	1744-03-01	NaN	NaN	Åland

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 577462 entries, 0 to 577461
Data columns (total 4 columns):
 #   Column                         Non-Null Count   Dtype  
---  ------                         --------------   -----  
 0   dt                             577462 non-null  object 
 1   AverageTemperature             544811 non-null  float64
 2   AverageTemperatureUncertainty  545550 non-null  float64
 3   Country                        577462 non-null  object 
dtypes: float64(2), object(2)
memory usage: 17.6+ MB
```

In [8]:

```
df.describe(include = 'all')
```

Out[8]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
count	577462	544811.000000	545550.000000	577462
unique	3239	NaN	NaN	243
top	2013-09-01	NaN	NaN	Åland
freq	243	NaN	NaN	3239
mean	NaN	17.193354	1.019057	NaN
std	NaN	10.953966	1.201930	NaN
min	NaN	-37.658000	0.052000	NaN
25%	NaN	10.025000	0.323000	NaN
50%	NaN	20.901000	0.571000	NaN
75%	NaN	25.814000	1.206000	NaN
max	NaN	38.842000	15.003000	NaN

Mapping average temperature in the Countries

In [9]:

```
global_temp_country_clear = global_temp_country[~global_temp_country['Country'].isi
    ['Denmark', 'Antarctica', 'France', 'Europe', 'Netherlands',
     'United Kingdom', 'Africa', 'South America'])]
# remove the duplicate countries and countries for which there is no information ab
```

In [10]:

```
global_temp_country_clear = global_temp_country_clear.replace(
   ['Denmark (Europe)', 'France (Europe)', 'Netherlands (Europe)', 'United Kingdom
   ['Denmark', 'France', 'Netherlands', 'United Kingdom'])
```

In [11]:

```
countries = np.unique(global_temp_country_clear['Country'])
mean_temp = []
for country in countries:
    mean_temp.append(global_temp_country_clear[global_temp_country_clear['Country']
                                                country]['AverageTemperature'].mean(
```

In [12]:

```
data = [ dict(
        type = 'choropleth',
        locations = countries,
        z = mean_temp,
        locationmode = 'country names',
        text = countries,
        marker = dict(
            line = dict(color = 'rgb(0,0,0)', width = 1)),
            colorbar = dict(autotick = True, tickprefix = '',
            title = '# Average\nTemperature,\n°C')
            )
       ]
```

In [13]:

```
#Extract the year from a date
years = np.unique(global_temp_country_clear['dt'].apply(lambda x: x[:4]))
```

```
#Let's create an array and add the values of average temperatures in the countries
mean_temp_year_country = [ [0] * len(countries) for i in range(len(years[::10]))]

j = 0
for country in countries:
    all_temp_country = global_temp_country_clear[global_temp_country_clear['Country
    i = 0
    for year in years[::10]:
        mean_temp_year_country[i][j] = all_temp_country[all_temp_country['dt'].appl
                lambda x: x[:4]) == year]['AverageTemperature'].mean()
        i +=1
    j += 1
```

2. Exploration and Visualization

Through our exploration we are going to visualize and analyse:

- Countries by yearly temperature
- Seasonal Temperature
- Global Average Temperature
- Continents by average yearly temperature
- Average temperature for each country

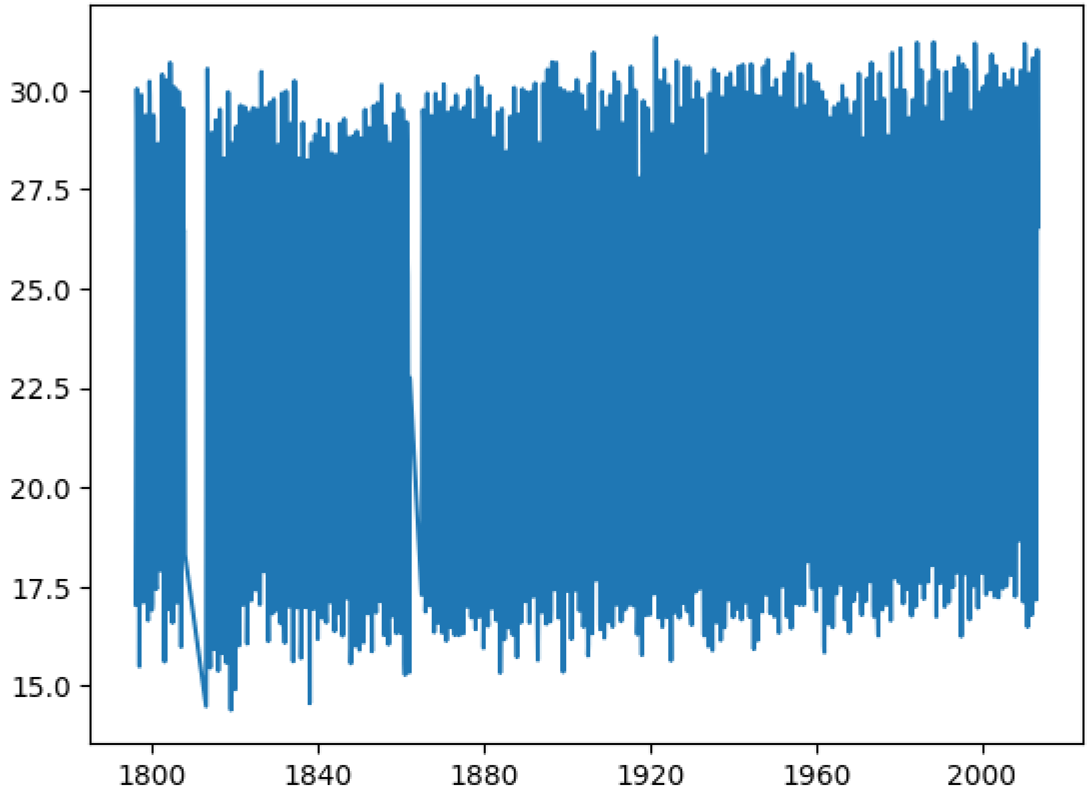
2.1 Countries by yearly temperature

In [14]:

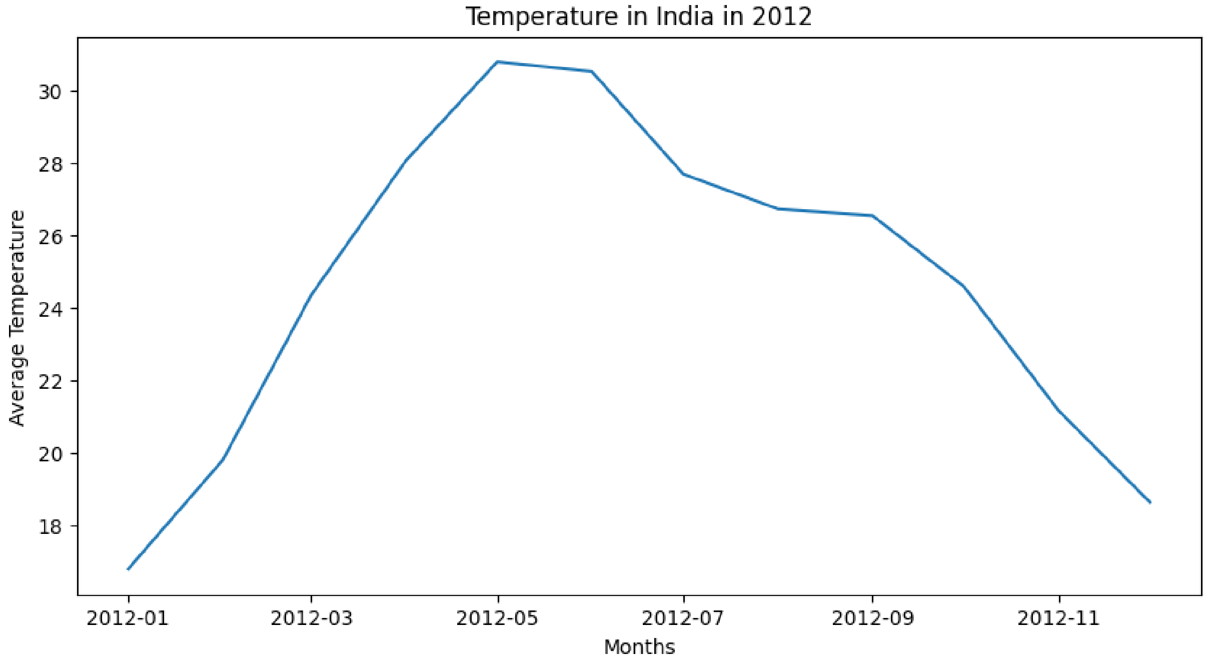
```
df = gltc[gltc['Country']=='India']

#dropping rows with NaN values
df.dropna(inplace=True)

# first lets bifurcate the months and year data for the dt
df.loc[:,'dt'] = pd.to_datetime(df['dt'])

df.loc[:,'month'] = [x.month for x in list(df['dt'])]
df.loc[:,'year'] = [x.year for x in list(df['dt'])]
```

In [15]:

```
plt.plot(df['dt'], df['AverageTemperature'])
plt.show()
```



```
In [16]: fig = plt.figure(figsize=(10,5))
         plt.plot(df.loc[df['year']==2012, 'dt'], df.loc[df['year']==2012,'AverageTemperatur
         plt.title('Temperature in India in 2012')
         plt.xlabel('Months')
         plt.ylabel('Average Temperature')
         plt.show()
```



From the above plot we can understand that the temperature in India reaches it's highest point in the month of May and the lowest on Dec-Feb.

Country with minimum average temperature

In [17]:

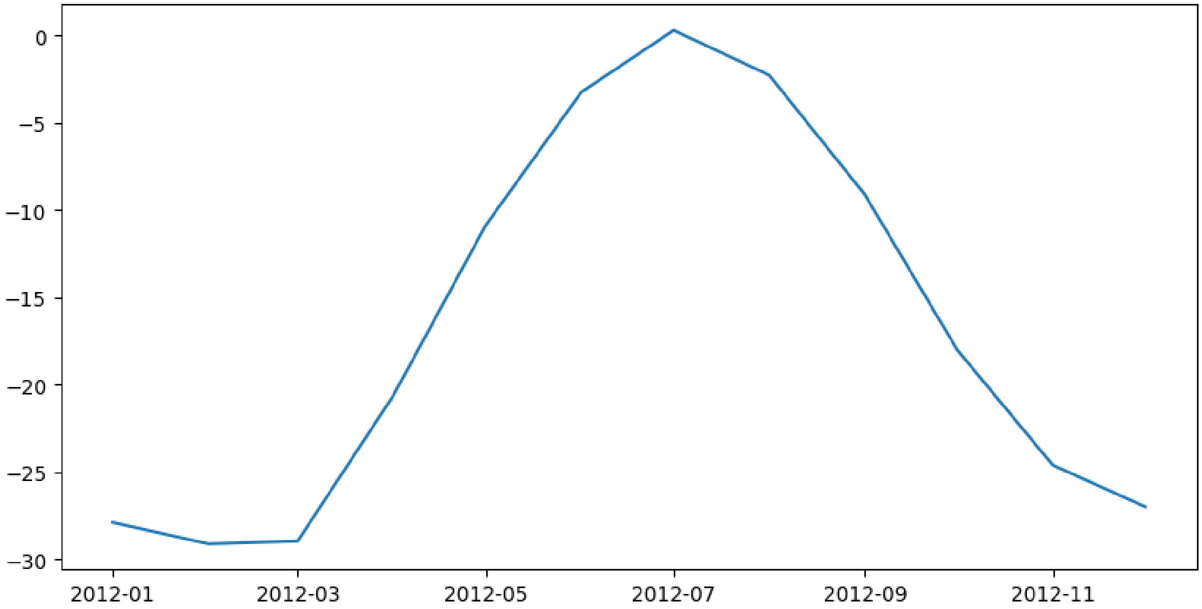
```
gltc[gltc['AverageTemperature']==gltc['AverageTemperature'].min()]
```

Out[17]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
210436	1868-02-01	-37.658	6.111	Greenland

In [18]:

```
df = gltc[gltc['Country']=='Greenland']
df.dropna(inplace=True)
df.loc[:,'dt'] = pd.to_datetime(df['dt'])
df.loc[:,'month'] = [x.month for x in list(df['dt'])]
df.loc[:,'year'] = [x.year for x in list(df['dt'])]
fig = plt.figure(figsize=(10,5))
plt.plot(df.loc[df['year']==2012, 'dt'], df.loc[df['year']==2012,'AverageTemperatur
plt.show()
```



From the above plot we can understand that the lowest temperature is in Greenland reaches it's highest point in the month of July and the lowest on Dec-Feb.

Country with mamixmum average temperature

In [19]:

```
gltc[gltc['AverageTemperature']==gltc['AverageTemperature'].max()]
```

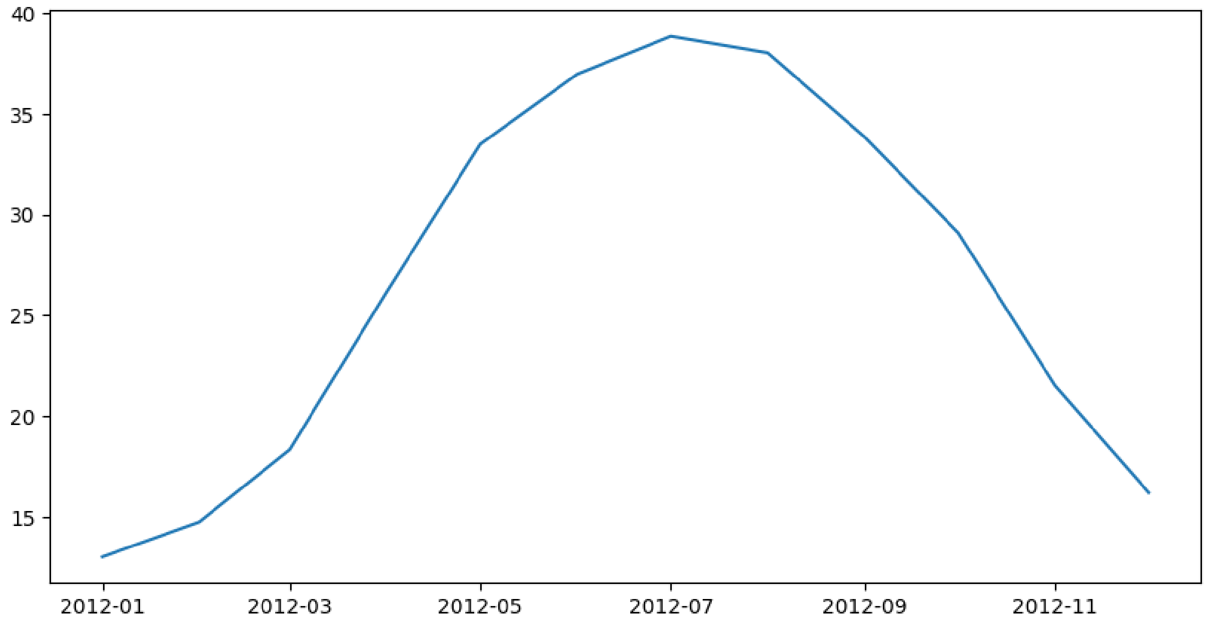
Out[19]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
284851	2012-07-01	38.842	0.464	Kuwait

In [20]:

```
df = gltc[gltc['Country']=='Kuwait']
df.dropna(inplace=True)
df.loc[:,'dt'] = pd.to_datetime(df['dt'])
df.loc[:,'month'] = [x.month for x in list(df['dt'])]
```

```python
df.loc[:,'year'] = [x.year for x in list(df['dt'])]
fig = plt.figure(figsize=(10,5))
plt.plot(df.loc[df['year']==2012, 'dt'], df.loc[df['year']==2012,'AverageTemperatur
plt.show()
```



From the above plot we can understand that the lowest temperature is in Kuwait reaches it's highest point in the month of July and the lowest on Dec-Mar.

2.2 Seasonal Temperature

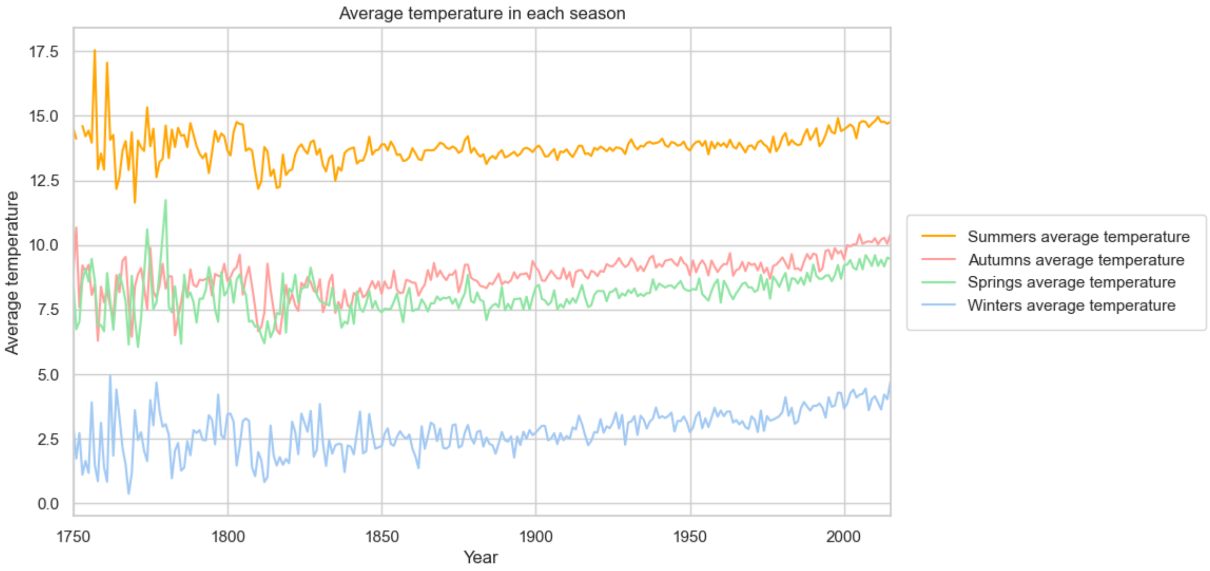
In [21]:

```python
# drop unnecessary columns
global_temp = global_temp[['dt', 'LandAverageTemperature']]

global_temp['dt'] = pd.to_datetime(global_temp['dt'])
global_temp['year'] = global_temp['dt'].map(lambda x: x.year)
global_temp['month'] = global_temp['dt'].map(lambda x: x.month)

def get_season(month):
    if month >= 3 and month <= 5:
        return 'spring'
    elif month >= 6 and month <= 8:
        return 'summer'
    elif month >= 9 and month <= 11:
        return 'autumn'
    else:
        return 'winter'

min_year = global_temp['year'].min()
max_year = global_temp['year'].max()
years = range(min_year, max_year + 1)

global_temp['season'] = global_temp['month'].apply(get_season)

spring_temps = []
summer_temps = []
```

```
autumn_temps = []
winter_temps = []

for year in years:
    curr_years_data = global_temp[global_temp['year'] == year]
    spring_temps.append(curr_years_data[curr_years_data['season'] == 'spring']['Lan
    summer_temps.append(curr_years_data[curr_years_data['season'] == 'summer']['Lan
    autumn_temps.append(curr_years_data[curr_years_data['season'] == 'autumn']['Lan
    winter_temps.append(curr_years_data[curr_years_data['season'] == 'winter']['Lan
```

In [22]:

```
sns.set(style="whitegrid")
sns.set_color_codes("pastel")
f, ax = plt.subplots(figsize=(10, 6))

plt.plot(years, summer_temps, label='Summers average temperature', color='orange')
plt.plot(years, autumn_temps, label='Autumns average temperature', color='r')
plt.plot(years, spring_temps, label='Springs average temperature', color='g')
plt.plot(years, winter_temps, label='Winters average temperature', color='b')

plt.xlim(min_year, max_year)

ax.set_ylabel('Average temperature')
ax.set_xlabel('Year')
ax.set_title('Average temperature in each season')
legend = plt.legend(loc='center left', bbox_to_anchor=(1, 0.5), frameon=True, borde
```



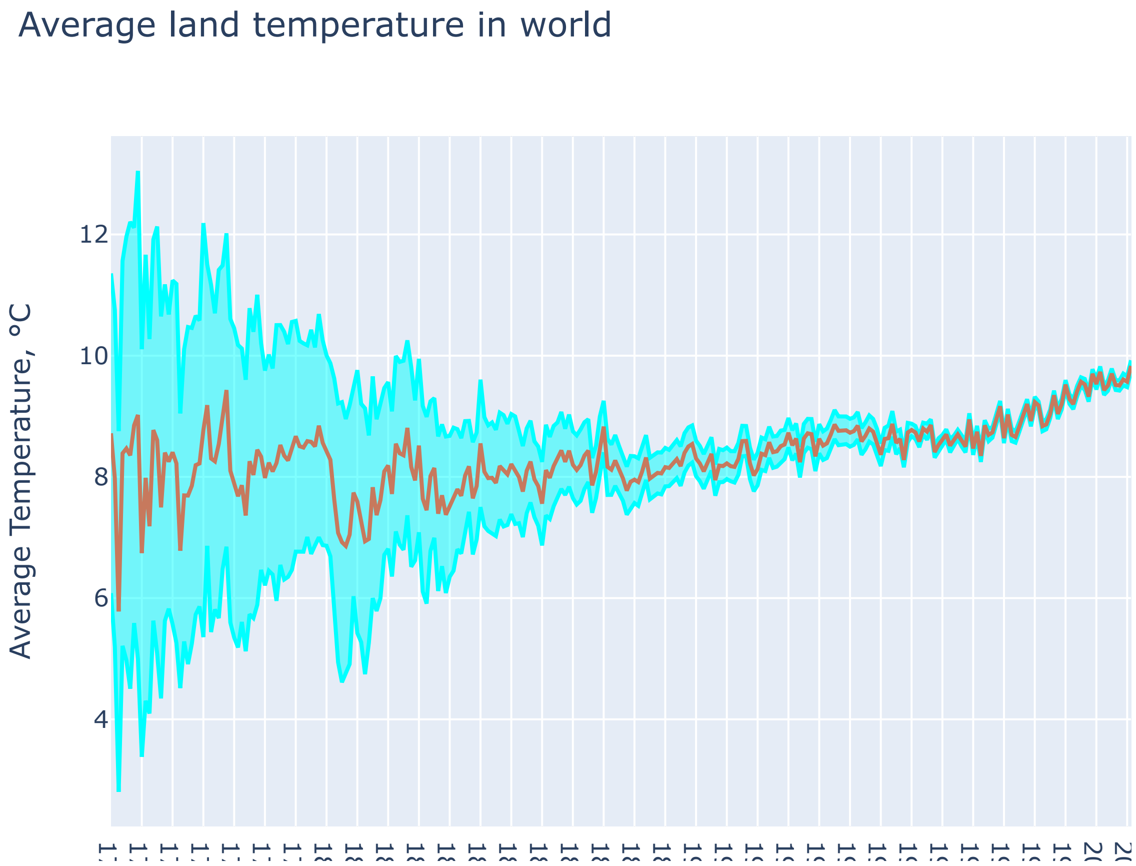
2.3 Global Average Temperature

In [23]:

```
global_temp = pd.read_csv("data/GlobalTemperatures.csv" )
```

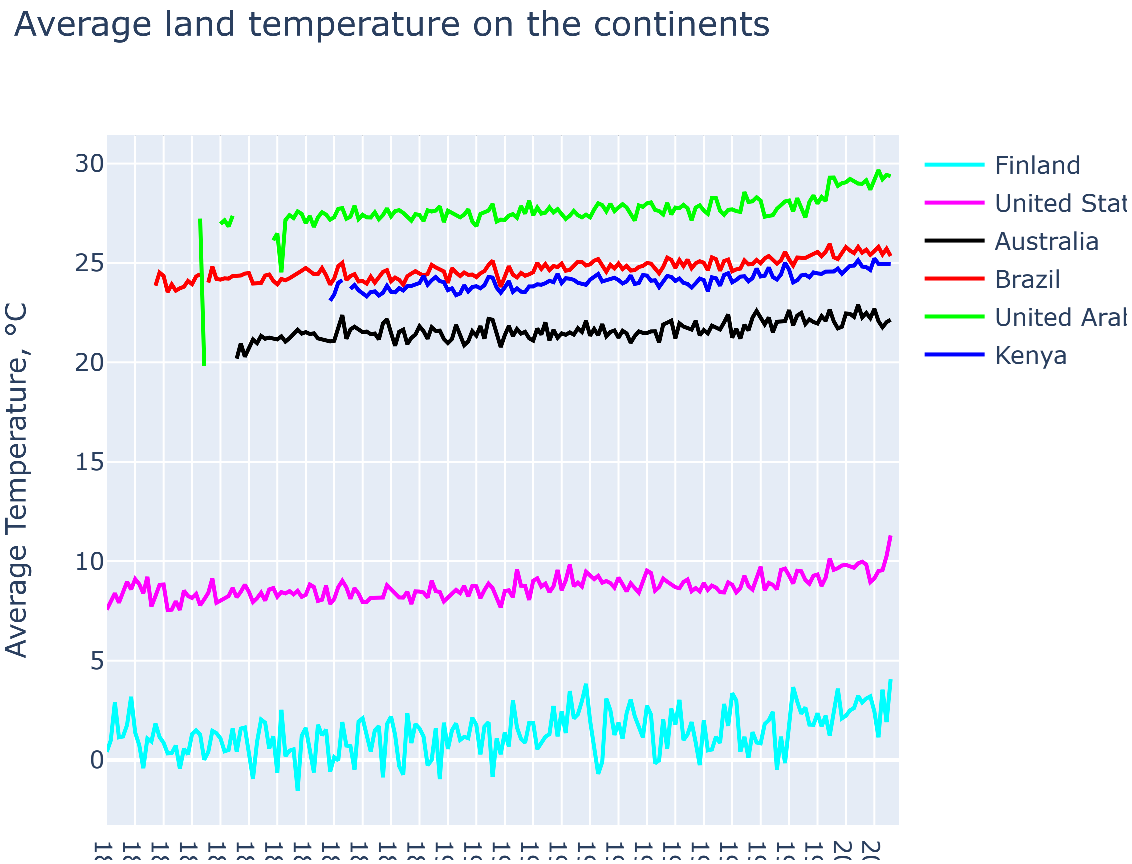
In [24]:

```
#Extract the year from a date
years = np.unique(global_temp['dt'].apply(lambda x: x[:4]))
mean_temp_world = []
mean_temp_world_uncertainty = []

for year in years:
    mean_temp_world.append(global_temp[global_temp['dt'].apply(
        lambda x: x[:4]) == year]['LandAverageTemperature'].mean())
    mean_temp_world_uncertainty.append(global_temp[global_temp['dt'].apply(
                lambda x: x[:4]) == year]['LandAverageTemperatureUncertainty'].mean

trace0 = go.Scatter(
    x = years,
    y = np.array(mean_temp_world) + np.array(mean_temp_world_uncertainty),
    fill= None,
    mode='lines',
    name='Uncertainty top',
    line=dict(
        color='rgb(0, 255, 255)',
    )
)
trace1 = go.Scatter(
    x = years,
    y = np.array(mean_temp_world) - np.array(mean_temp_world_uncertainty),
    fill='tonexty',
    mode='lines',
    name='Uncertainty bot',
    line=dict(
        color='rgb(0, 255, 255)',
    )
)

trace2 = go.Scatter(
    x = years,
    y = mean_temp_world,
    name='Average Temperature',
    line=dict(
        color='rgb(199, 121, 093)',
    )
)
data = [trace0, trace1, trace2]

layout = go.Layout(
    xaxis=dict(title='year'),
    yaxis=dict(title='Average Temperature, °C'),
    title='Average land temperature in world',
    showlegend = False)

fig = go.Figure(data=data, layout=layout)
py.iplot(fig)
```



2.4 Continents by average yearly temperature

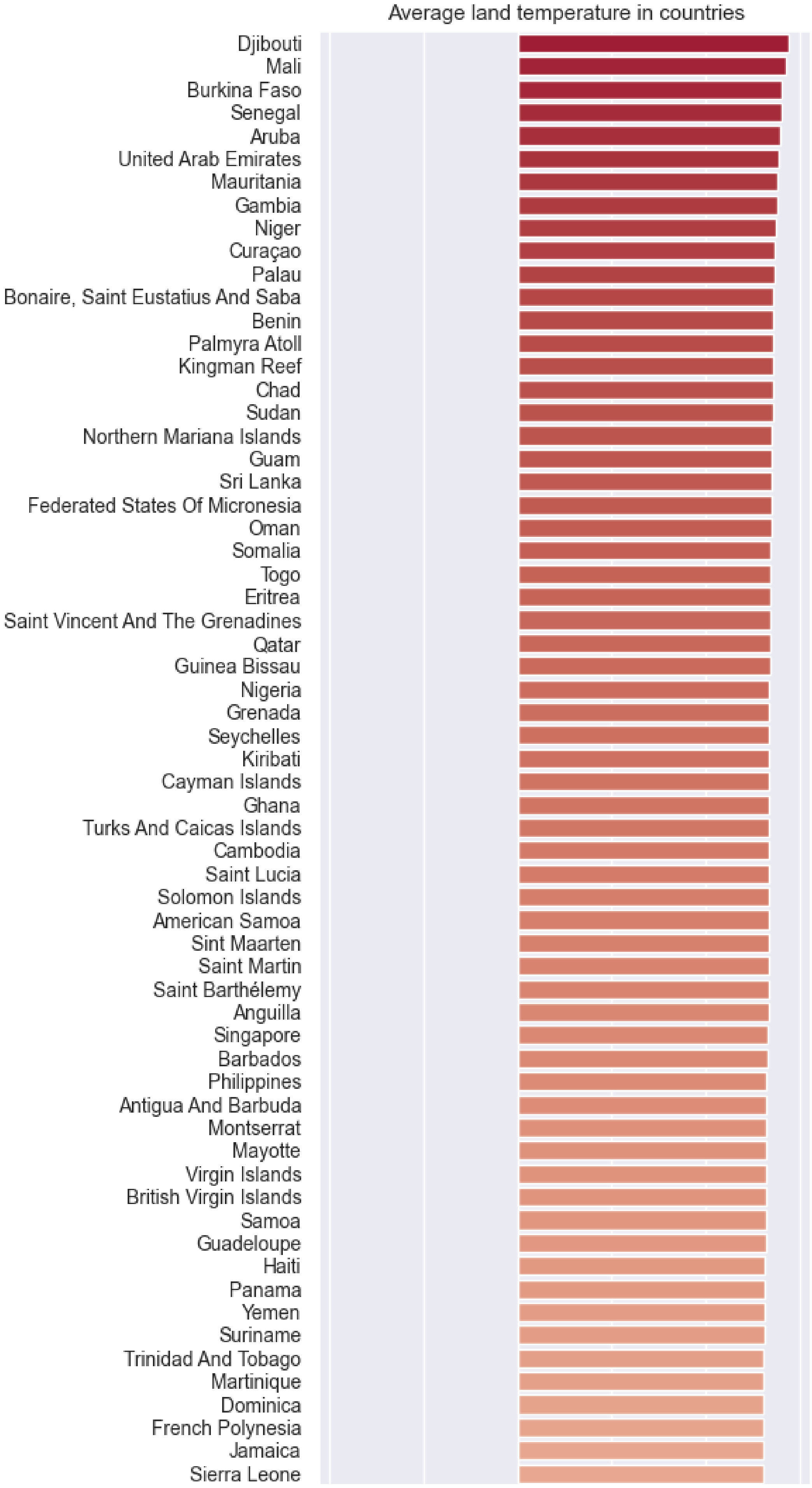
In [25]:
```
continent = ['Finland', 'United States', 'Australia', 'Brazil', 'United Arab Emirat
mean_temp_year_country = [ [0] * len(years[70:]) for i in range(len(continent))]
j = 0
for country in continent:
    all_temp_country = global_temp_country_clear[global_temp_country_clear['Country
    i = 0
    for year in years[70:]:
        mean_temp_year_country[j][i] = all_temp_country[all_temp_country['dt'].appl
                lambda x: x[:4]) == year]['AverageTemperature'].mean()
        i +=1
    j += 1

traces = []
colors = ['rgb(0, 255, 255)', 'rgb(255, 0, 255)', 'rgb(0, 0, 0)',
          'rgb(255, 0, 0)', 'rgb(0, 255, 0)', 'rgb(0, 0, 255)']
for i in range(len(continent)):
    traces.append(go.Scatter(
        x=years[70:],
        y=mean_temp_year_country[i],
        mode='lines',
        name=continent[i],
        line=dict(color=colors[i]),
    ))

layout = go.Layout(
    xaxis=dict(title='year'),
    yaxis=dict(title='Average Temperature, °C'),
    title='Average land temperature on the continents',)

fig = go.Figure(data=traces, layout=layout)
py.iplot(fig)
```

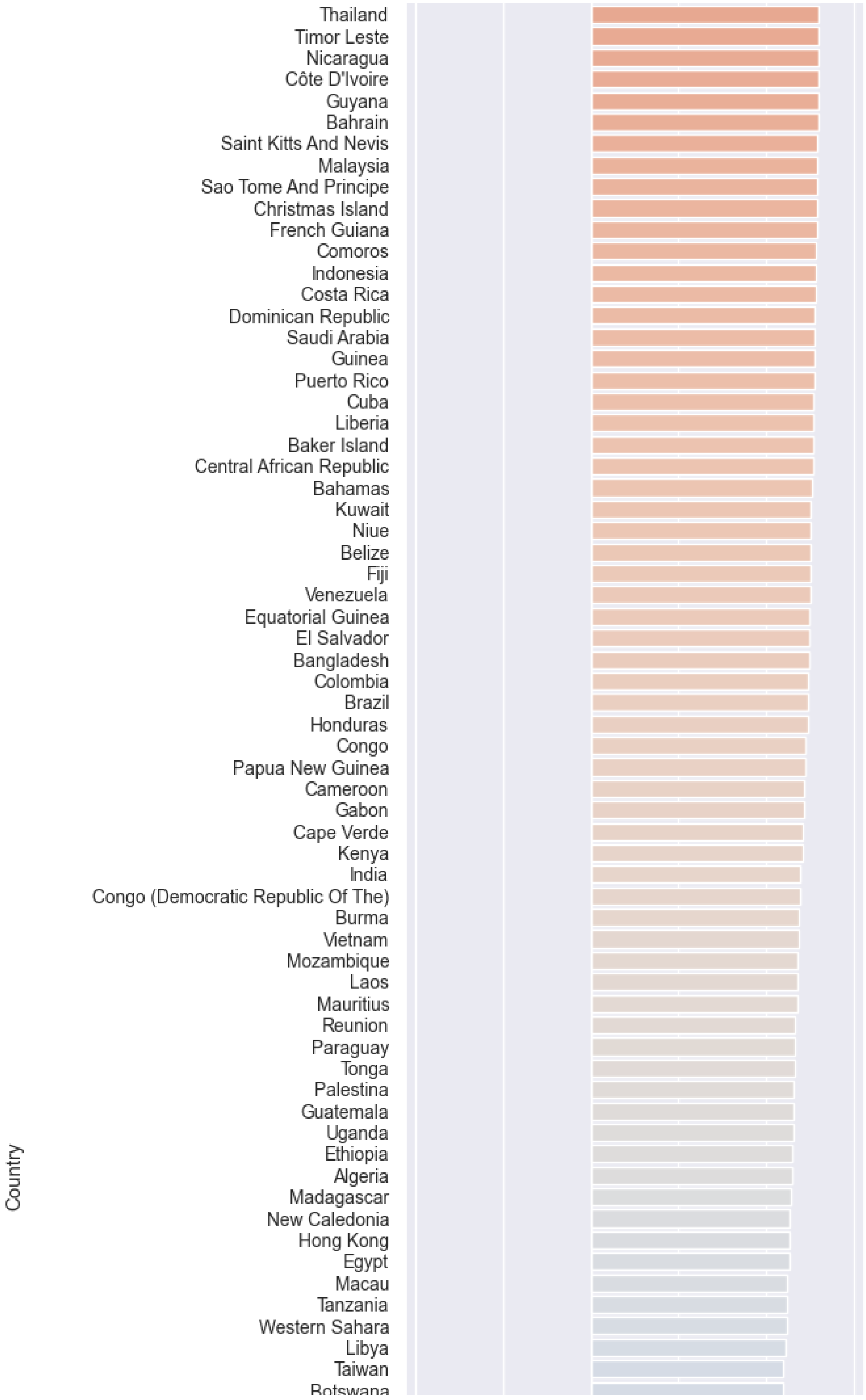


2.5 Average temperature for each country

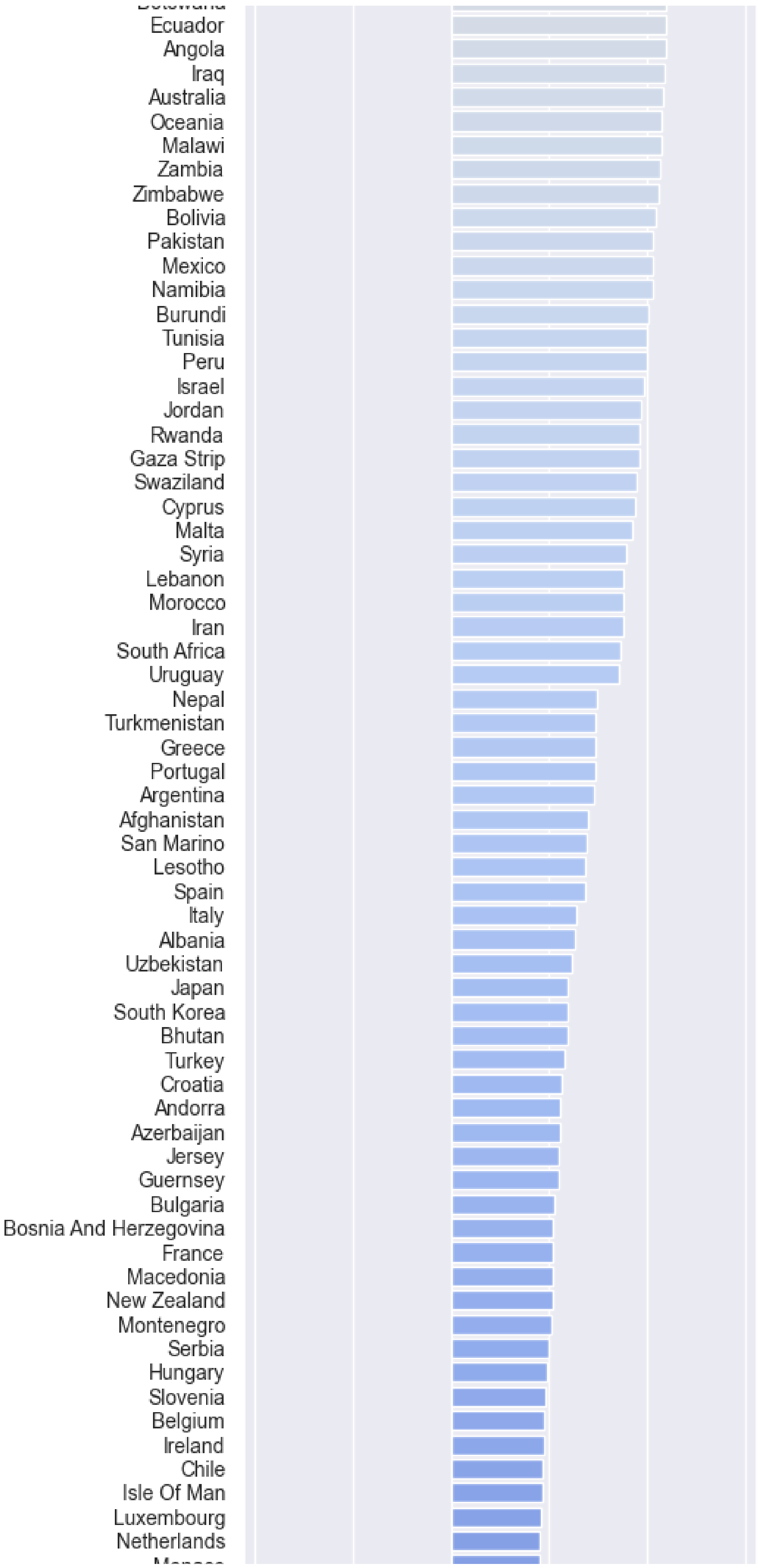
In [26]:

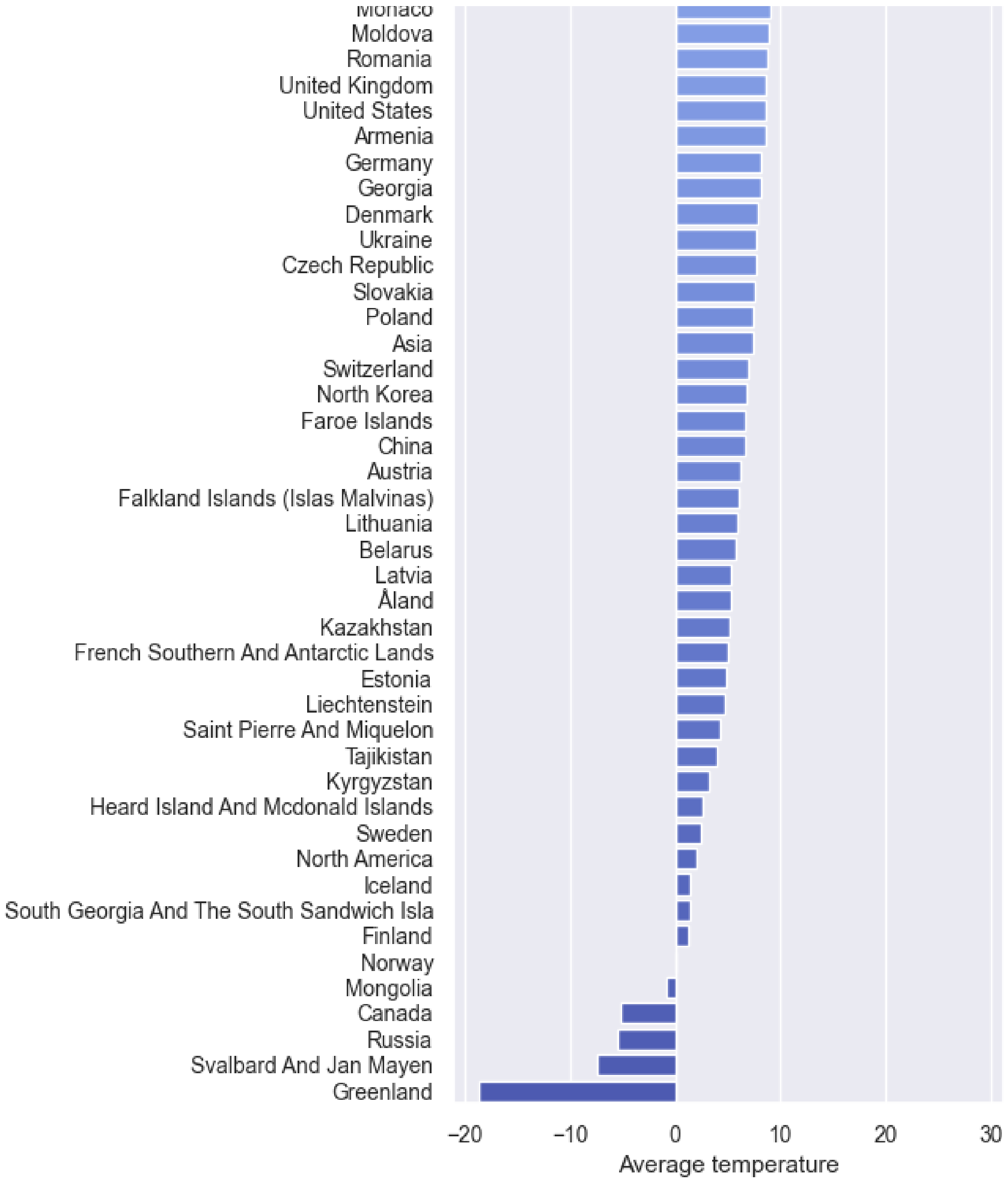
```
mean_temp_bar, countries_bar = (list(x) for x in zip(*sorted(zip(mean_temp, countri
data = pd.DataFrame({'Average temperature': mean_temp_bar, 'Country': countries_bar
sns.set(font_scale=0.9)
f, ax = plt.subplots(figsize=(4.5, 50))
colors_cw = sns.color_palette('coolwarm', len(countries))
sns.barplot(data=data, x='Average temperature', y='Country', palette=colors_cw[::-1
Text = ax.set(xlabel='Average temperature', title='Average land temperature in coun
```


Average land temperature in countries
Djibouti
Mali
Burkina Faso
Senegal
Aruba
United Arab Emirates
Mauritania
Gambia
Niger
Curaçao
Palau
Bonaire, Saint Eustatius And Saba
Benin
Palmyra Atoll
Kingman Reef
Chad
Sudan
Northern Mariana Islands
Guam
Sri Lanka
Federated States Of Micronesia
Oman
Somalia
Togo
Eritrea
Saint Vincent And The Grenadines
Qatar
Guinea Bissau
Nigeria
Grenada
Seychelles
Kiribati
Cayman Islands
Ghana
Turks And Caicas Islands
Cambodia
Saint Lucia
Solomon Islands
American Samoa
Sint Maarten
Saint Martin
Saint Barthélemy
Anguilla
Singapore
Barbados
Philippines
Antigua And Barbuda
Montserrat
Mayotte
Virgin Islands
British Virgin Islands
Samoa
Guadeloupe
Haiti
Panama
Yemen
Suriname
Trinidad And Tobago
Martinique
Dominica
French Polynesia
Jamaica
Sierra Leone



Country
Thailand
Timor Leste
Nicaragua
Côte D'Ivoire
Guyana
Bahrain
Saint Kitts And Nevis
Malaysia
Sao Tome And Principe
Christmas Island
French Guiana
Comoros
Indonesia
Costa Rica
Dominican Republic
Saudi Arabia
Guinea
Puerto Rico
Cuba
Liberia
Baker Island
Central African Republic
Bahamas
Kuwait
Niue
Belize
Fiji
Venezuela
Equatorial Guinea
El Salvador
Bangladesh
Colombia
Brazil
Honduras
Congo
Papua New Guinea
Cameroon
Gabon
Cape Verde
Kenya
India
Congo (Democratic Republic Of The)
Burma
Vietnam
Mozambique
Laos
Mauritius
Reunion
Paraguay
Tonga
Palestina
Guatemala
Uganda
Ethiopia
Algeria
Madagascar
New Caledonia
Hong Kong
Egypt
Macau
Tanzania
Western Sahara
Libya
Taiwan
Botswana

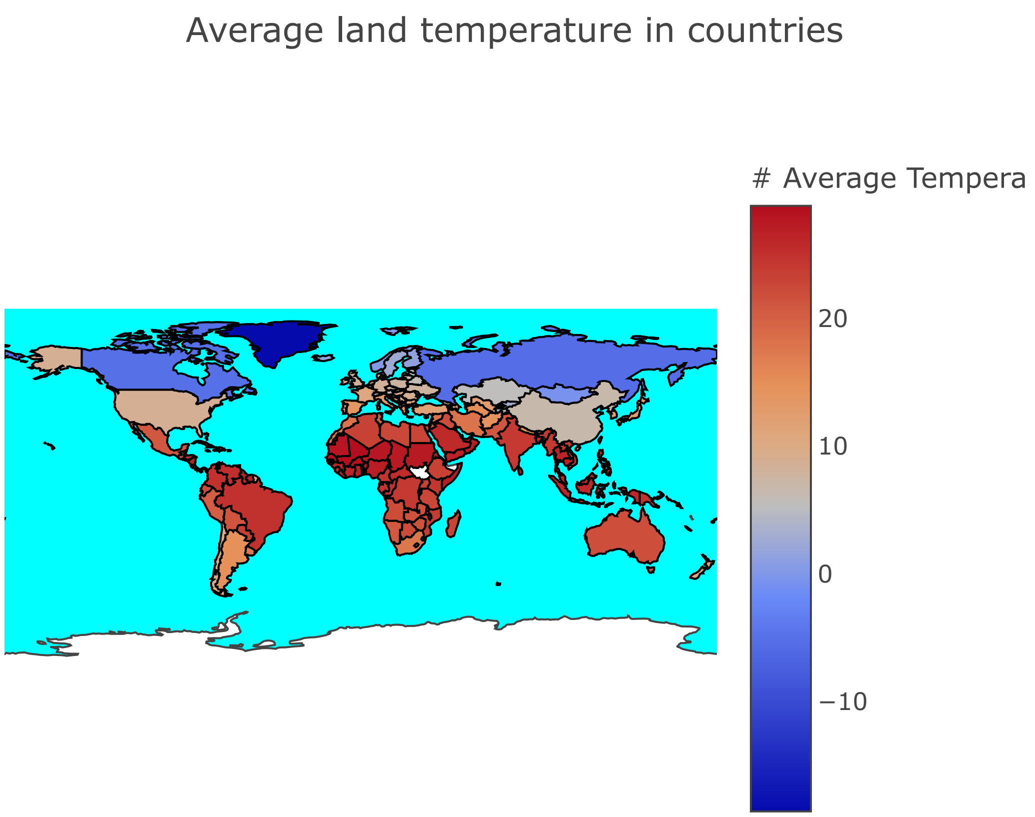

Ecuador
Angola
Iraq
Australia
Oceania
Malawi
Zambia
Zimbabwe
Bolivia
Pakistan
Mexico
Namibia
Burundi
Tunisia
Peru
Israel
Jordan
Rwanda
Gaza Strip
Swaziland
Cyprus
Malta
Syria
Lebanon
Morocco
Iran
South Africa
Uruguay
Nepal
Turkmenistan
Greece
Portugal
Argentina
Afghanistan
San Marino
Lesotho
Spain
Italy
Albania
Uzbekistan
Japan
South Korea
Bhutan
Turkey
Croatia
Andorra
Azerbaijan
Jersey
Guernsey
Bulgaria
Bosnia And Herzegovina
France
Macedonia
New Zealand
Montenegro
Serbia
Hungary
Slovenia
Belgium
Ireland
Chile
Isle Of Man
Luxembourg
Netherlands



Dynamic map

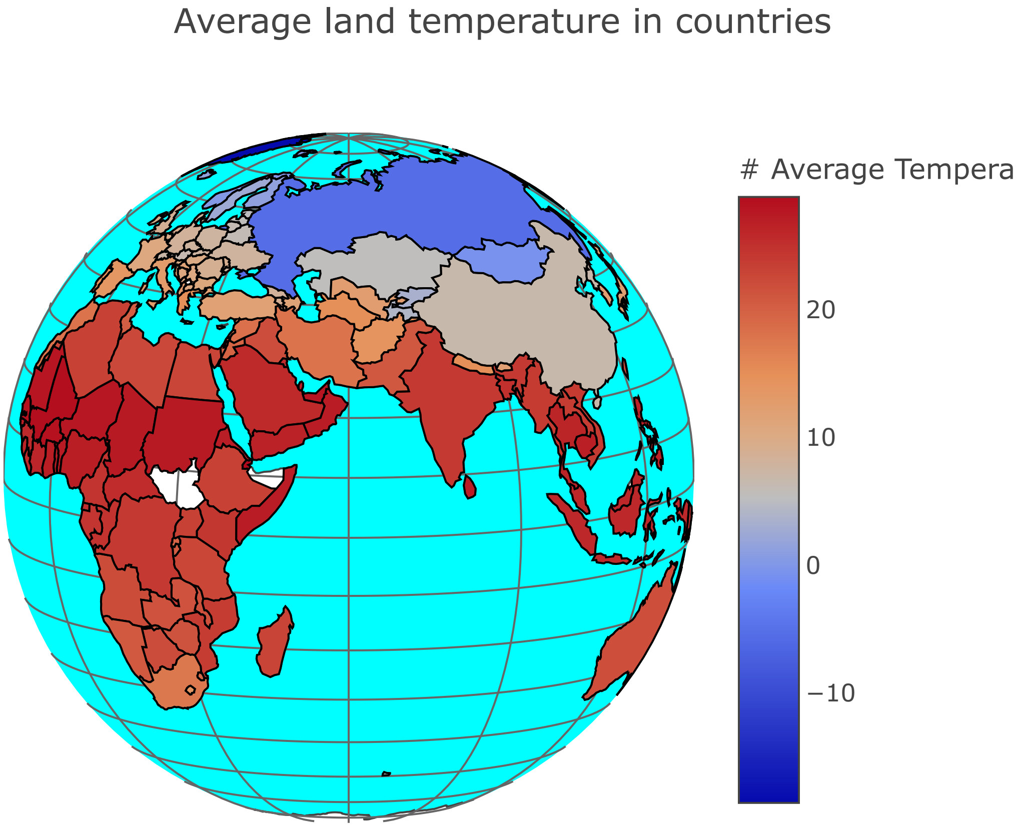
In [27]:

```
#Let's create a Streaming in Plotly (here, alas, does not work, so commented out)
#stream_tokens = tls.get_credentials_file()['stream_ids']
#token =  stream_tokens[-1]
#stream_id = dict(token=token, maxpoints=60)

data = [ dict(
        type = 'choropleth',
        locations = countries,
        z = mean_temp,
        locationmode = 'country names',
        text = countries,
        marker = dict(
            line = dict(color = 'rgb(0,0,0)', width = 1)),
            colorbar = dict(autotick = True, tickprefix = '',
            title = '# Average\nTemperature,\n°C'),
        #The following line is also needed to create Stream
        #stream = stream_id
            )
       ]

layout = dict(
    title = 'Average land temperature in countries',
    geo = dict(
        showframe = False,
        showocean = True,
        oceancolor = 'rgb(0,255,255)',
        type = 'equirectangular'
    ),
)

fig = dict(data=data, layout=layout)
py.iplot(fig, validate=False, filename='world_temp_map')
```



In [28]:

```
layout = dict(
    title = 'Average land temperature in countries',
    geo = dict(
        showframe = False,
        showocean = True,
        oceancolor = 'rgb(0,255,255)',
        projection = dict(
        type = 'orthographic',
            rotation = dict(
                    lon = 60,
                    lat = 10),
        ),
        lonaxis =  dict(
                showgrid = True,
                gridcolor = 'rgb(102, 102, 102)'
            ),
        lataxis = dict(
                showgrid = True,
                gridcolor = 'rgb(102, 102, 102)'
                )
            ),
        )

fig = dict(data=data, layout=layout)
py.iplot(fig, validate=False, filename='worldmap')
```



Result

Russia has one of the lowest average temperature same as Canada. The lowest temperature in Greenland (it is distinctly visible on the map). The hottest country in Africa, on the equator.

3. Conclusion

Through our analysis of global temperature data, we have discovered the following information:

- The average global temperature has been steadily increasing over the past century, with the warmest years on record occurring in the last decade.
- We have also noticed that there are regional variations in the impact of global warming, with some areas being disproportionately affected, such as developing countries and vulnerable communities.
- We have identified seasonal and temporal patterns in global temperature trends, such as the increase in temperature during the summer months.
- Countries located near the equator experience high temperatures throughout the year.
- There is a need for further investigation into the causes of global temperature variations, including natural factors such as solar activity and volcanic eruptions, as well as human factors.
- Finally, we have identified the importance of addressing climate change not only as a challenge but also as an opportunity to create a more sustainable and equitable future for all.