7.1-Data_Visualization_using_Matplotlib

KeytoDataScience.com

What is Matplotlib?

- Matplotlib is a low level graph plotting library in python that serves as a visualization utility.
- Matplotlib was created by John D. Hunter.
- Matplotlib is open source and we can use it freely.
- Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility.

Table of Contents

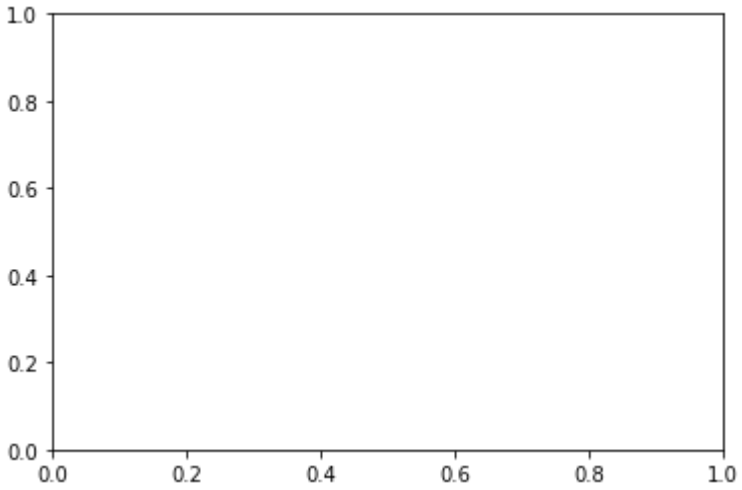
- 21 Matplotlb CheatSheet
- 22 Learn more

In [1]:
```
# uncomment to install the libraries.

# pip intsall matplotlib

# or
# !pip intsall matplotlib
```

In [2]:
```
import matplotlib.pyplot as plt
import os
import pandas as pd
import numpy as np
%matplotlib inline
```

1 Creates a figure object and axes object

In [3]:
```
fig, ax = plt.subplots()
plt.show()
```



In [ ]:
```
# Get the current directory
import os
os.getcwd()
```

In [5]:
```
# csv files are stored at Input/Matplotlib folder
input_files=os.getcwd()+"/Input/Matplotlib/"
```

In [6]:
```
seattle_weather = pd.read_csv(input_files+"seattleWeather_2016-2017.csv")
```

In [7]:
```
print(seattle_weather.head(5))
```

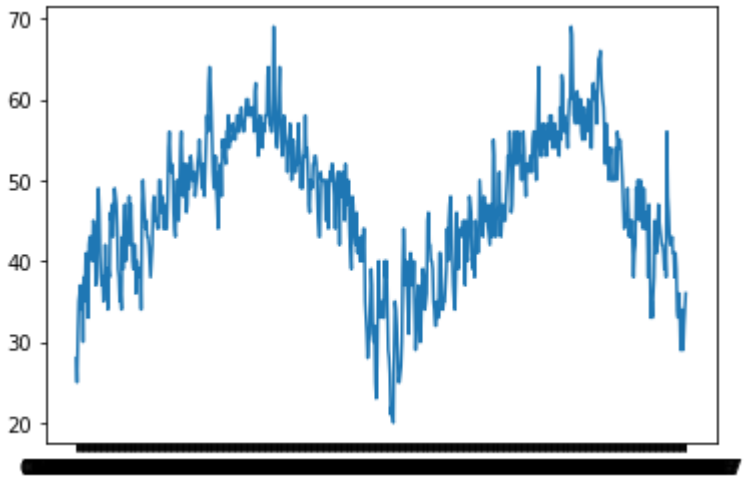
```
        DATE   PRCP  TMAX  TMIN   RAIN
```

```
0  01-01-2016    -     46.0  28.0  False
1  02-01-2016    -     42.0  25.0  False
2  03-01-2016   0.02   40.0  31.0   True
3  04-01-2016   0.15   38.0  35.0   True
4  05-01-2016   0.11   46.0  36.0   True
```

↥ back to top

2 Adding data to the plot

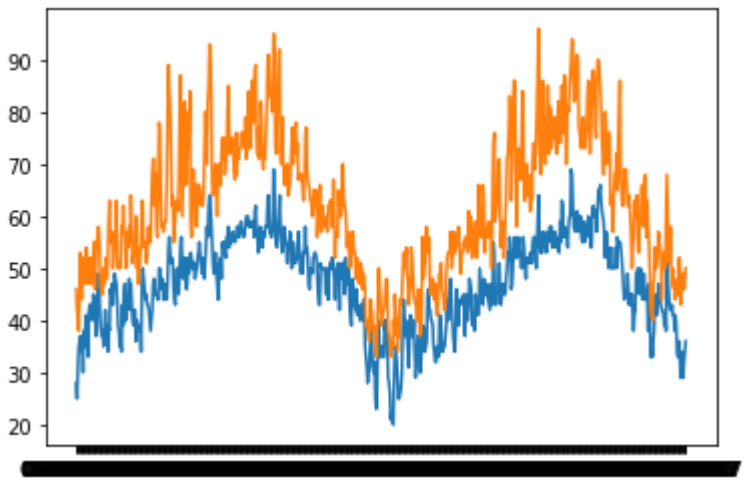
In [8]:
```
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"],seattle_weather["TMIN"])
plt.show()
```



↥ back to top

3 Adding another series to the plot

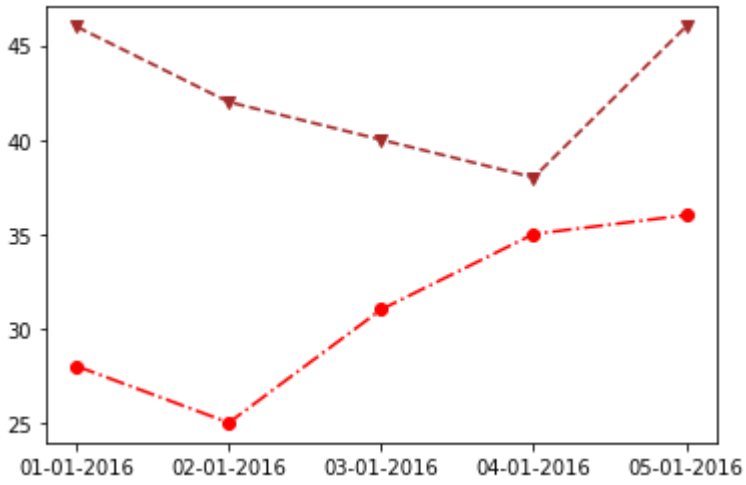
In [9]:
```
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"],seattle_weather["TMIN"])
ax.plot(seattle_weather["DATE"],seattle_weather["TMAX"])
plt.show()
```



↥ back to top

4 Adding markers,color and linestyles to the plot

In [10]:
```
seattle_weather = seattle_weather[:5]

fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"],seattle_weather["TMIN"],marker="o",linestyle='-.',color
ax.plot(seattle_weather["DATE"],seattle_weather["TMAX"],marker="v",linestyle='--',color
plt.show()

# Even works with color = 'blue' or 'red'  etc
```



Link for all markers

- https://matplotlib.org/api/markers_api.html

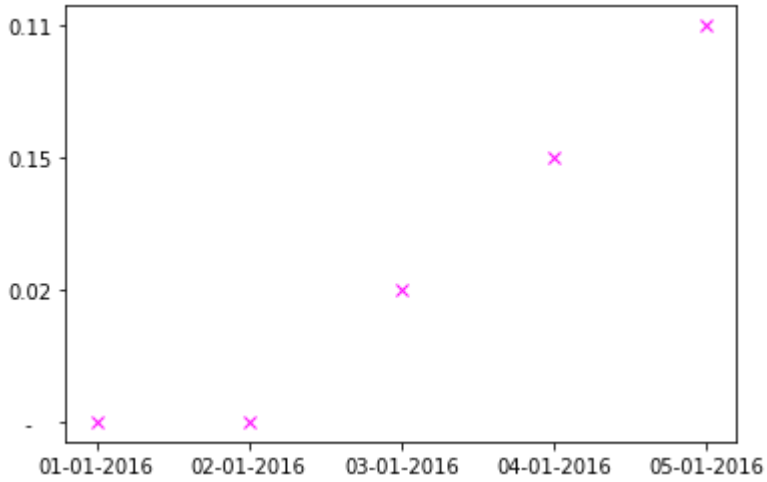
Link for all linestyles

- https://matplotlib.org/gallery/lines_bars_and_markers/line_styles_reference.html

For color codes hex

- https://color.adobe.com/create/
- https://htmlcolorcodes.com/

In [11]:
```
# For plots without Lines joining the points, marker as X and Color as 'Magenta'
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"],seattle_weather["PRCP"],marker="x",linestyle='None',col
plt.show()
```

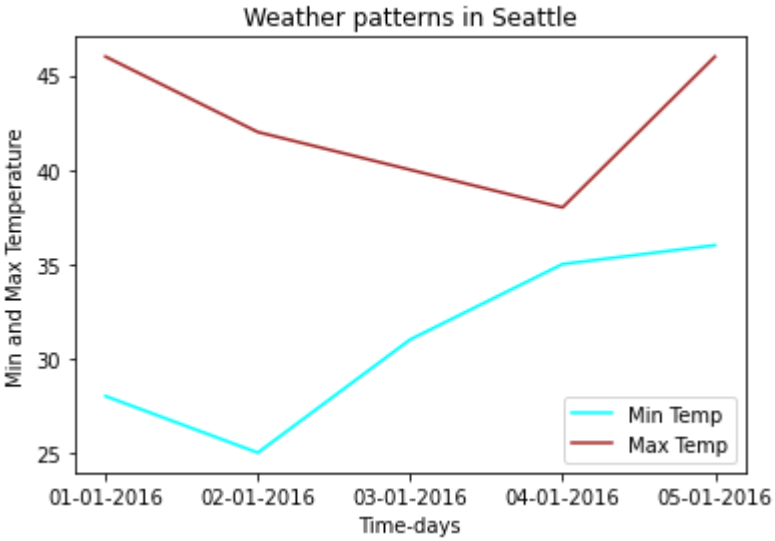


↥ back to top

5 For labeling the axes of the plots and Title of graph, Labels

In [12]:

```
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"],seattle_weather["TMIN"],color='#00FFFF',label='Min Temp
ax.plot(seattle_weather["DATE"],seattle_weather["TMAX"],color='#A52A2A',label='Max Temp

ax.set_xlabel("Time-days")

ax.set_ylabel("Min and Max Temperature")

ax.set_title("Weather patterns in Seattle")

plt.legend()
plt.show()
```

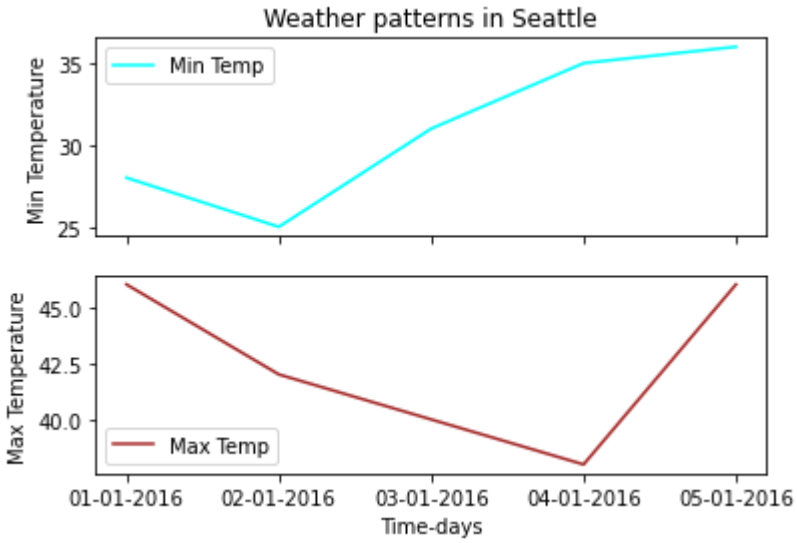


↥ back to top

6 For plotting across multiple subplots with common x axis

In [13]:

```
#fig, ax = plt.subplots(2,1)
fig, ax = plt.subplots(2,1,sharex=True)

ax[0].plot(seattle_weather["DATE"],seattle_weather["TMIN"],color='#00FFFF',label='Min T
ax[0].set_ylabel("Min Temperature")
ax[0].legend()

ax[1].plot(seattle_weather["DATE"],seattle_weather["TMAX"],color='#A52A2A',label='Max T
ax[1].set_ylabel("Max Temperature")
ax[1].legend()

# Common Title and X axis
ax[0].set_title("Weather patterns in Seattle")
ax[1].set_xlabel("Time-days")

plt.show()
```

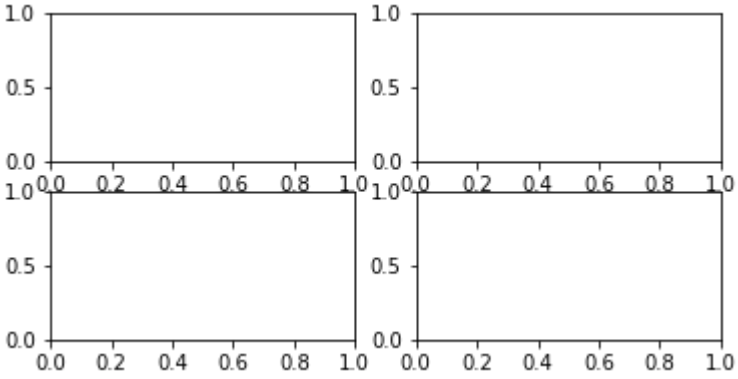


↥ back to top

7 Options to change the canvas area for plots

In [14]:

```
# We need to increase size of the figure using figsize
fig, ax = plt.subplots(2,2,figsize=(6,3))

#or using the below one can change the size permanently
#plt.rcParams["figure.figsize"] = [12,6]
#fig, ax = plt.subplots(2,2)
```

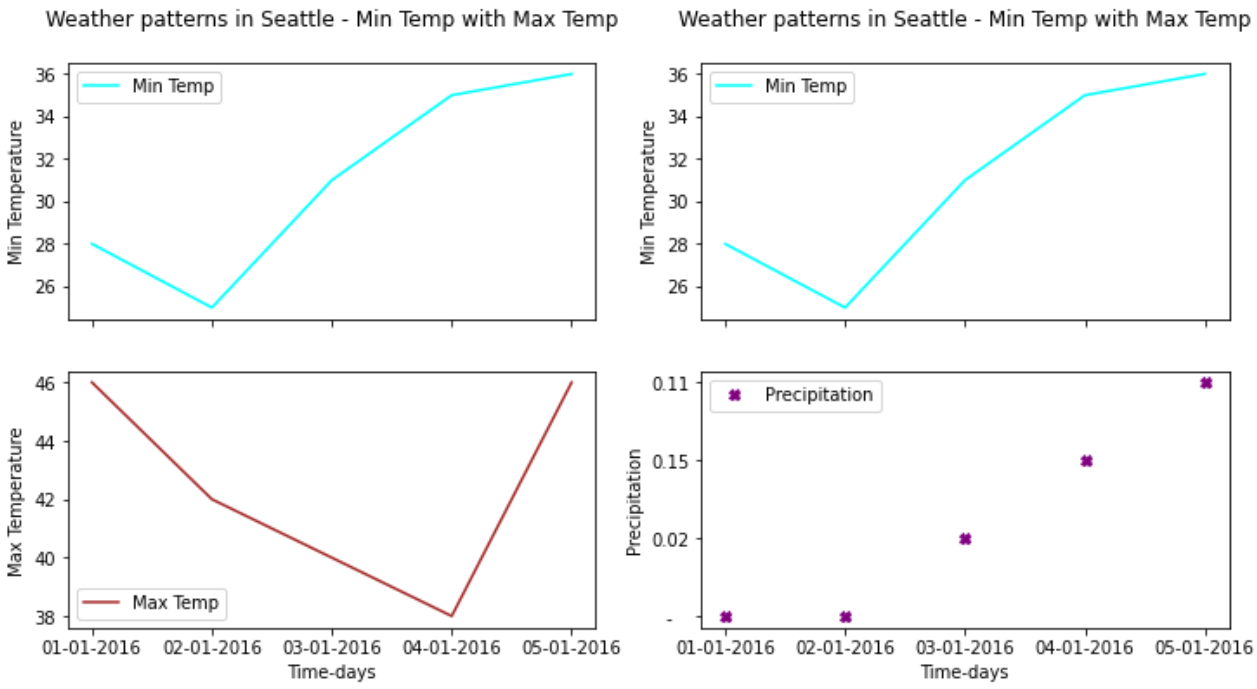


↥ back to top

8 For plotting across multiple subplots 2 X 2

In [15]:

```
fig, ax = plt.subplots(2,2,figsize=(12,6),sharex=True)

# first Column plots
ax[0,0].plot(seattle_weather["DATE"],seattle_weather["TMIN"],color='#00FFFF',label='Min
ax[0,0].set_ylabel("Min Temperature")
ax[0,0].legend()

ax[1,0].plot(seattle_weather["DATE"],seattle_weather["TMAX"],color='#A52A2A',label='Max
ax[1,0].set_ylabel("Max Temperature")
ax[1,0].legend()

ax[1,0].set_xlabel("Time-days")
ax[0,0].set_title("Weather patterns in Seattle - Min Temp with Max Temp\n")

# second Column plots
ax[0,1].plot(seattle_weather["DATE"],seattle_weather["TMIN"],color='#00FFFF',label='Min
ax[0,1].set_ylabel("Min Temperature")
ax[0,1].legend()

ax[1,1].plot(seattle_weather["DATE"],seattle_weather["PRCP"],color='Purple',marker='X',
ax[1,1].set_ylabel("Precipitation")
ax[1,1].legend()

ax[1,1].set_xlabel("Time-days")
ax[0,1].set_title("Weather patterns in Seattle - Min Temp with Max Temp\n")

plt.show()
```



↥ back to top

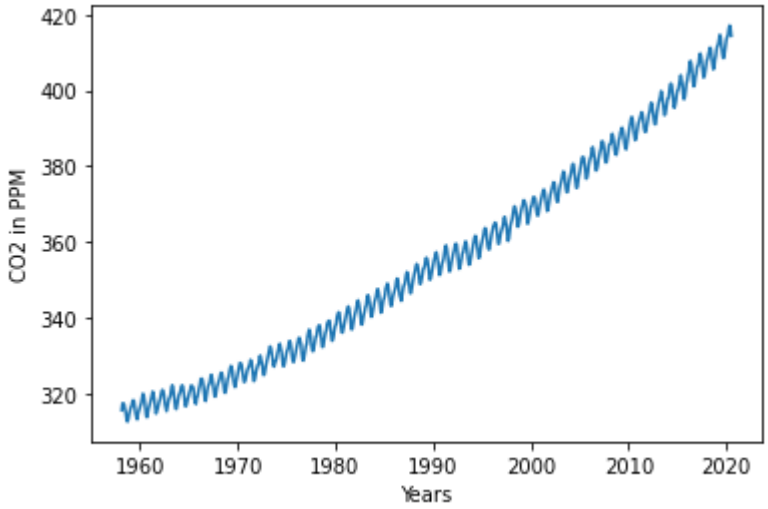
9 Plotting Time Series Data

In [16]:
```
global_co2 =  pd.read_csv(input_files+"monthwise_co2_ppm_data.csv",index_col='Date_Meas
```

In [17]:
```
global_co2.dtypes
```

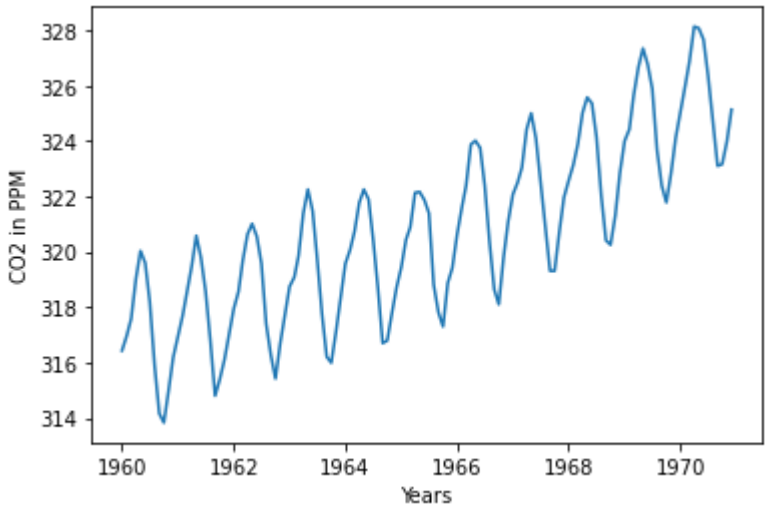
Out[17]:
```
CO2_PPM    float64
dtype: object
```

In [18]:
```
# Plot timeseries data
fig, ax = plt.subplots()
ax.plot(global_co2.index,global_co2["CO2_PPM"])
ax.set_xlabel("Years")
ax.set_ylabel("CO2 in PPM")
plt.show()
```



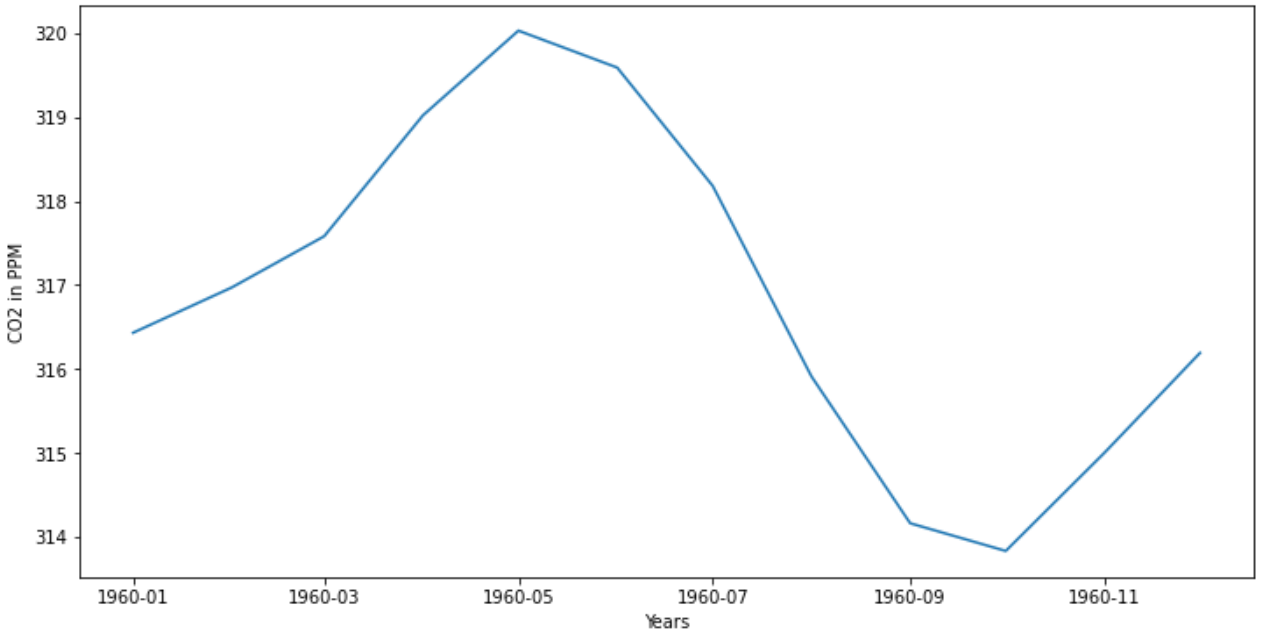
In [19]:

```
# Plot timeseries data for 10 years
global_co2_1960_1970 = global_co2["1960-01-01":"1970-12-01"]
fig, ax = plt.subplots()
ax.plot(global_co2_1960_1970.index,global_co2_1960_1970["CO2_PPM"])
ax.set_xlabel("Years")
ax.set_ylabel("CO2 in PPM")
plt.show()
```



In [20]:

```
# Plot timeseries data for 12 months in 1960
global_co2_1960 = global_co2["1960-01-01":"1960-12-01"]
fig, ax = plt.subplots(figsize=(12,6))
ax.plot(global_co2_1960.index,global_co2_1960["CO2_PPM"])
ax.set_xlabel("Years")
ax.set_ylabel("CO2 in PPM")
plt.show()
```



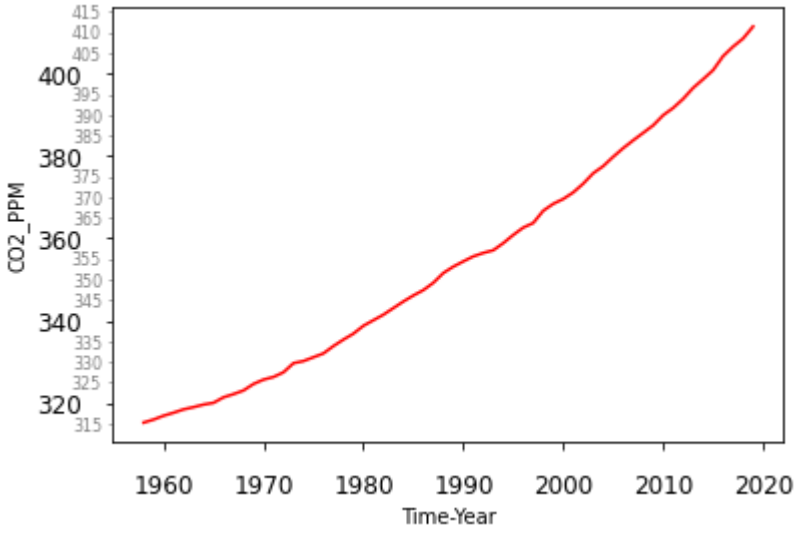
In [21]:
```
global_co2_temp_diff = pd.read_csv(input_files+"GLOBAL_CO2_PPM_TEMP_ANOMALY.csv",index_
global_co2_temp_diff.dtypes
```

Out[21]:
```
Temp_Anomaly    float64
CO2_PPM         float64
dtype: object
```

↥ back to top

10 Tick Parameters and Grid Parameters

In [24]:
```
from matplotlib.ticker import MultipleLocator, ScalarFormatter

fig,ax = plt.subplots()
ax.plot(global_co2_temp_diff.index,global_co2_temp_diff["CO2_PPM"],color='red')
ax.set_xlabel("Time-Year")
ax.set_ylabel("CO2_PPM")
ax.yaxis.set_minor_locator(MultipleLocator(5))
ax.yaxis.set_minor_formatter(ScalarFormatter())

# Tick Parameters
ax.tick_params(axis ='both', which ='major',
               labelsize = 12, pad = 12,
               colors ='black')
ax.tick_params(axis ='y', which ='minor',
               labelsize = 8, colors ='grey')

plt.show()
```



More Tick Parameters

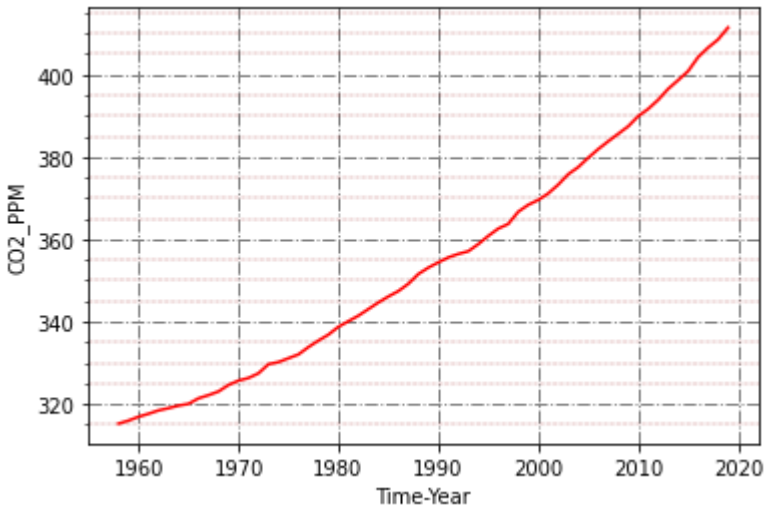
- https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.axes.Axes.tick_params.html

Tick locators

- https://matplotlib.org/3.1.1/gallery/ticks_and_spines/tick-locators.html

In [25]:

```
fig,ax = plt.subplots()
ax.yaxis.set_minor_locator(MultipleLocator(5))
ax.plot(global_co2_temp_diff.index,global_co2_temp_diff["CO2_PPM"],color='red')
ax.set_xlabel("Time-Year")
ax.set_ylabel("CO2_PPM")

# Grid Parameters
ax.grid(b=None, which='major', axis='both',color='grey', linestyle='-.', linewidth=1)
ax.grid(b=None, which='minor', axis='y',color='brown', linestyle='-.', linewidth=0.3)

plt.show()
```

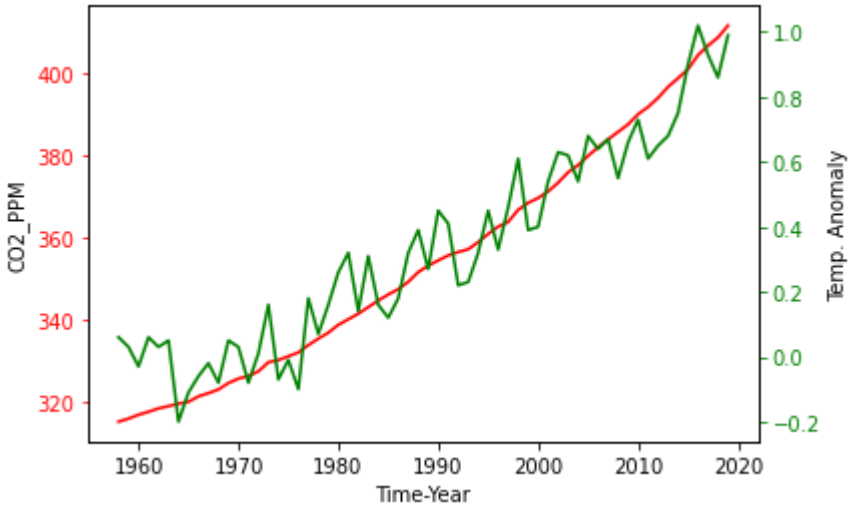


↥ back to top

11 Plot 2 time series data in same plot using Twin Axes

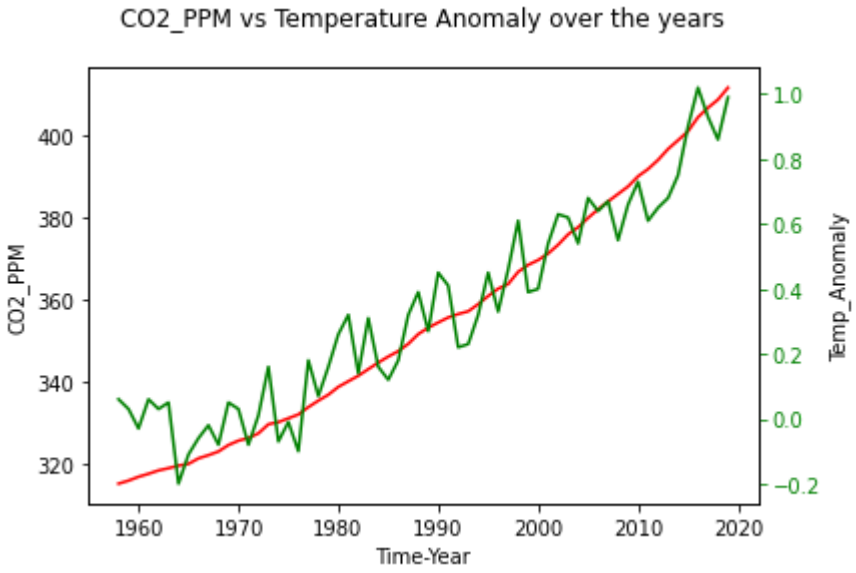
In [26]:

```
fig,ax = plt.subplots()
ax.plot(global_co2_temp_diff.index,global_co2_temp_diff["CO2_PPM"],color='red')
ax.set_xlabel("Time-Year")
ax.set_ylabel("CO2_PPM")
ax.tick_params('y',colors='red')
ax2=ax.twinx()

ax2.plot(global_co2_temp_diff.index,global_co2_temp_diff["Temp_Anomaly"],color='green')
ax2.set_xlabel("Time-Year")
ax2.set_ylabel("Temp. Anomaly")
ax2.tick_params('y',colors='green')
plt.show()
```



In [27]:

```
# Simplify the code using a function

def time_series_plots(axes,x,y,color,xlabel,ylabel):
    axes.plot(x,y,color=color)
    axes.set_xlabel(xlabel)
    axes.set_ylabel(ylabel)
    ax2.tick_params('y',colors=color)


fig,ax = plt.subplots()
time_series_plots(ax,global_co2_temp_diff.index,global_co2_temp_diff["CO2_PPM"],'red',"
ax.set_title("CO2_PPM vs Temperature Anomaly over the years\n")

ax2=ax.twinx()
time_series_plots(ax2,global_co2_temp_diff.index,global_co2_temp_diff["Temp_Anomaly"],'

plt.show()
```

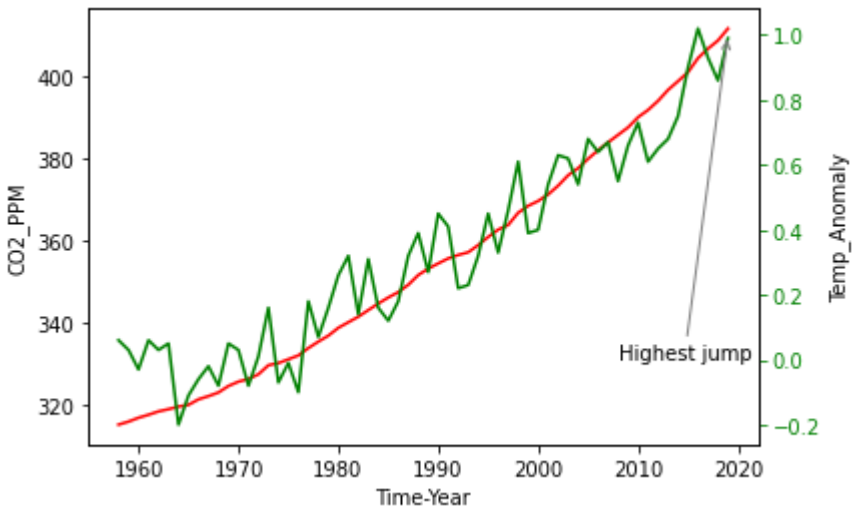


↥ back to top

12 Annotations

In [28]:

```
fig,ax = plt.subplots()
time_series_plots(ax,global_co2_temp_diff.index,global_co2_temp_diff["CO2_PPM"],'red',"

ax2=ax.twinx()
time_series_plots(ax2,global_co2_temp_diff.index,global_co2_temp_diff["Temp_Anomaly"],'

#Annotations
ax2.annotate("Highest jump",
             xy=(pd.Timestamp("2019-01-01"),1),
             xytext=(pd.Timestamp("2008-01-01"),0),
             arrowprops={"arrowstyle":"->","color":"gray"})

plt.show()
```



Annotations Options:

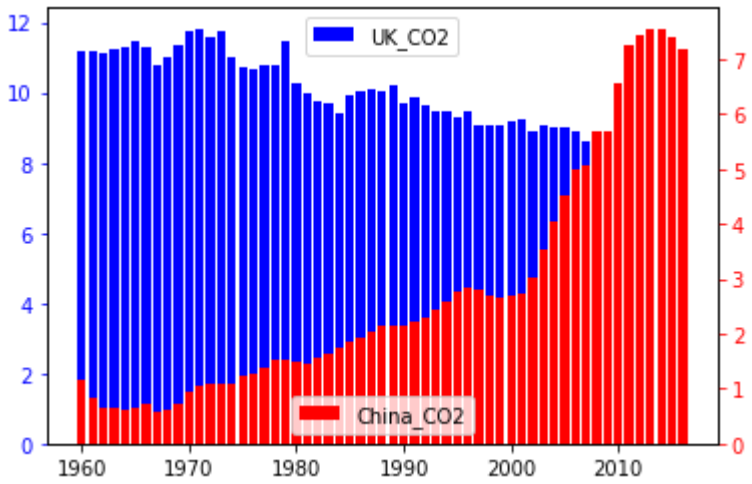
- http://matplotlib.org/users/annotations.html

↥ back to top

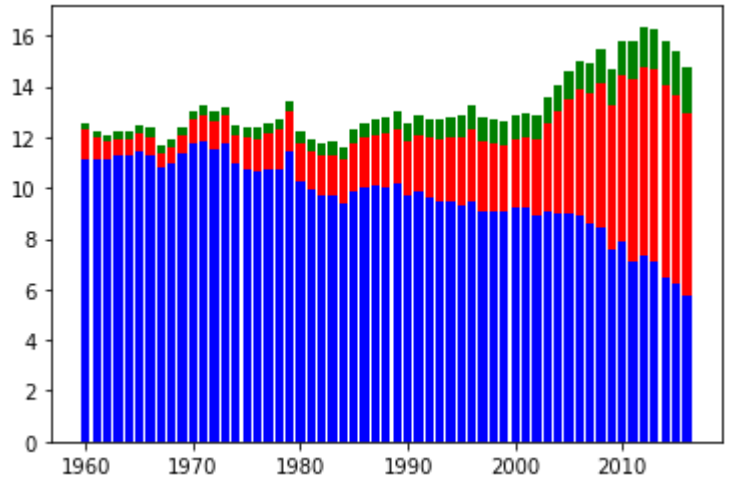
13 Bar Plots

In [29]:
```
CO2_PPM_COUNTRIES =  pd.read_csv(input_files+"CO2_PPM_COUNTRIES.csv")
```

In [30]:
```
# Bar plots coinciding on same axis

data = CO2_PPM_COUNTRIES[["YEAR","United Kingdom","China"]]
fig,ax = plt.subplots()
ax.bar(data["YEAR"],data["United Kingdom"],color='b', label='UK_CO2')
ax.tick_params('y',colors='b')
plt.legend(loc='upper center')

ax2 = ax.twinx()
ax2.bar(data["YEAR"],data["China"],color='r', label='China_CO2')
ax2.tick_params('y',colors='r')
plt.legend(loc='lower center')

plt.show()
```



In [31]:
```
# Stacked bar charts
data = CO2_PPM_COUNTRIES[["YEAR","United Kingdom","China","India"]]
fig,ax = plt.subplots()
ax.bar(data["YEAR"],data["United Kingdom"],color='b')
ax.bar(data["YEAR"],data["China"],color='r',bottom=data["United Kingdom"])
ax.bar(data["YEAR"],data["India"],color='g',bottom=data["United Kingdom"]+data["China"]

plt.show()
```

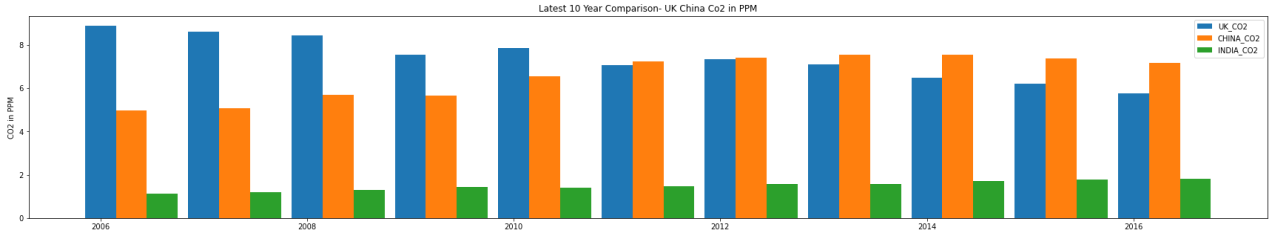


↥ back to top

14 Bar charts with width option

In [32]:

```
# Figure size
plt.figure(figsize=(30,5))

# Width of a bar
width = 0.3

# Plotting latest 10 year trend
plt.bar(data["YEAR"][-11:], data["United Kingdom"][-11:] , width, label='UK_CO2')
plt.bar(data["YEAR"][-11:] + width, data["China"][-11:], width, label='CHINA_CO2')
plt.bar(data["YEAR"][-11:] + 2*width , data["India"][-11:], width, label='INDIA_CO2')

plt.ylabel('CO2 in PPM')
plt.title('Latest 10 Year Comparison- UK China Co2 in PPM')

# Finding the best position for legends and putting it
plt.legend(loc='best')
plt.show()
```



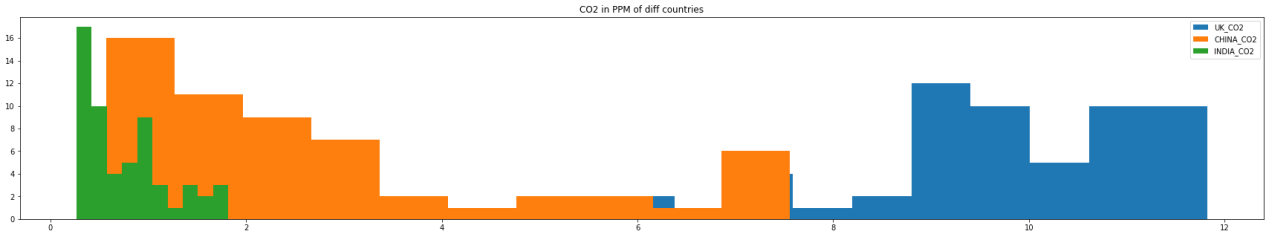
↥ back to top

15 Histogram

In [33]:

```
# Histogram charts

# Figure size
plt.figure(figsize=(30,5))

plt.hist(data["United Kingdom"],  label='UK_CO2')
```

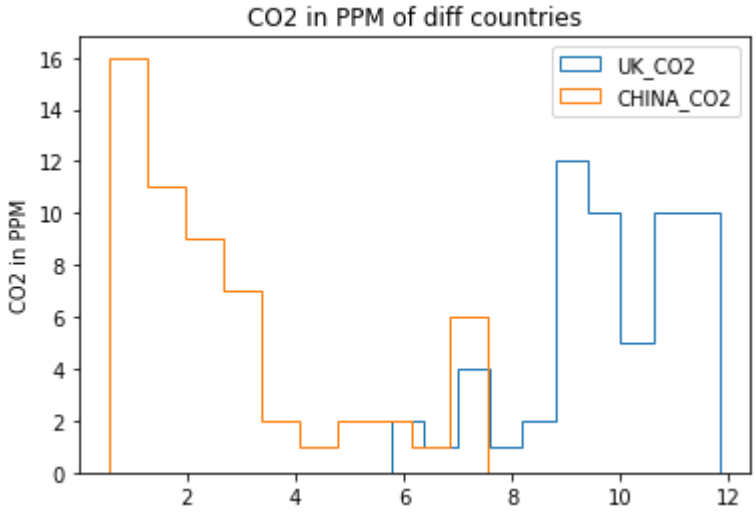
```
plt.hist(data["China"],  label='CHINA_CO2')
plt.hist(data["India"],  label='INDIA_CO2')

plt.title('CO2 in PPM of diff countries')

# Finding the best position for legends and putting it
plt.legend(loc='best')
plt.show()
```



In [34]:

```
# Histogram chart with histtype = Step

plt.figure(figsize=(30,5))
fig,ax = plt.subplots()

ax.hist(data["United Kingdom"],  label='UK_CO2',histtype="step")
ax.hist(data["China"],  label='CHINA_CO2',histtype="step")

ax.set_ylabel('CO2 in PPM')
plt.title('CO2 in PPM of diff countries')

# Finding the best position for legends and putting it
plt.legend(loc='best')
plt.show()
```

```
<Figure size 2160x360 with 0 Axes>
```



↥ back to top

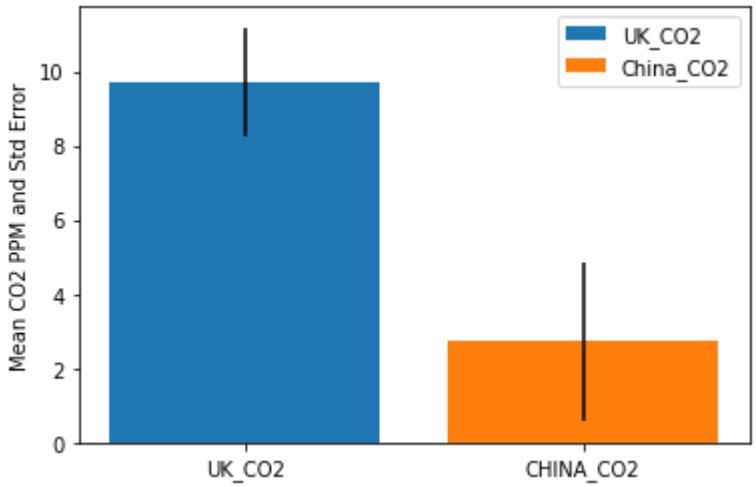
16 Statistical Plotting

16.1 Error Bars

In [35]:

```
# Figure size
plt.figure(figsize=(30,5))
fig,ax = plt.subplots()

ax.bar('UK_CO2',data["United Kingdom"].mean(), yerr=data["United Kingdom"].std(),label=
ax.bar('CHINA_CO2',data["China"].mean(), yerr=data["China"].std(),label='China_CO2')
ax.set_ylabel("Mean CO2 PPM and Std Error")

# Finding the best position for legends and putting it
plt.legend(loc='best')
plt.show()
```

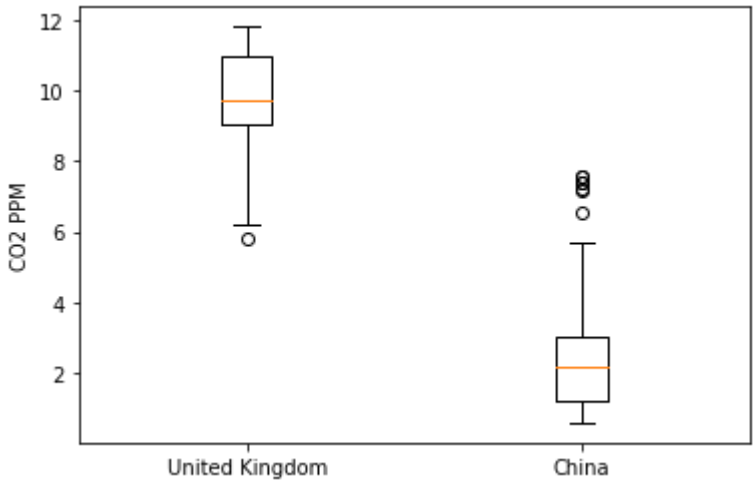
```
<Figure size 2160x360 with 0 Axes>
```



16.2 Box Plots

In [36]:

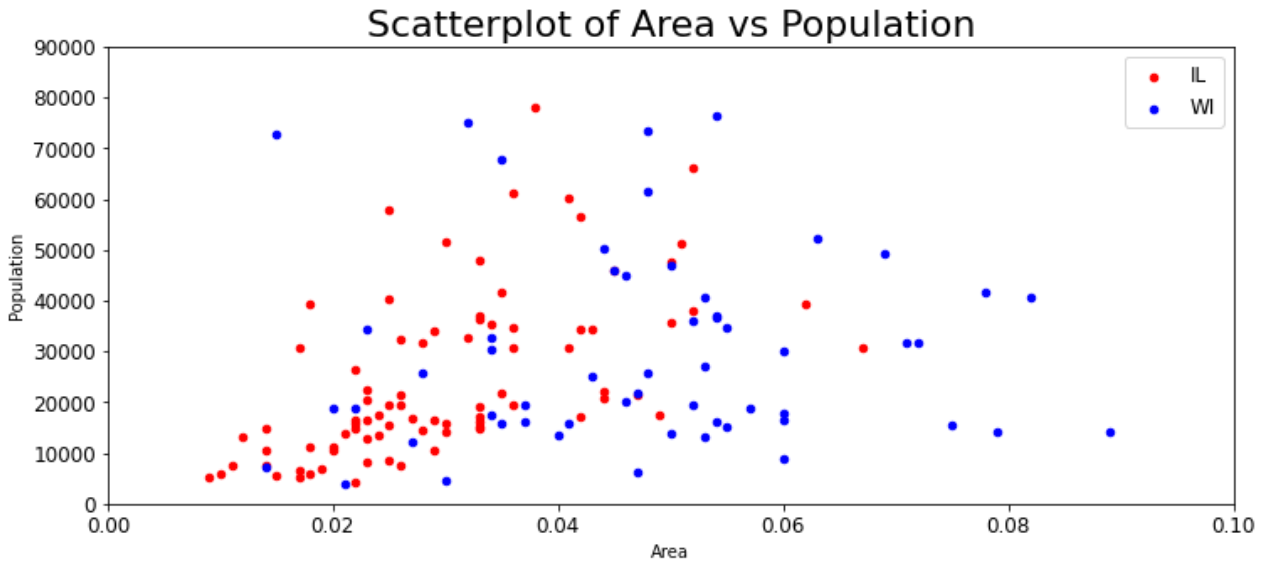
```
# box plot for UK and China CO2
fig,ax = plt.subplots()
ax.boxplot([data["United Kingdom"],data["China"]])
ax.set_xticklabels(["United Kingdom","China"])  # Diff from set_xlabel which labels x a
ax.set_ylabel("CO2 PPM")
plt.show()
```



16.3 Scatter Plot

In [37]:

```
# Import dataset
midwest = pd.read_csv(input_files+"midwest_filter.csv")

plt.figure(figsize=(12, 5))

# Draw Plot
plt.scatter('area', 'poptotal', data=midwest[midwest["state"].isin(["IL"])], s=20, labe
plt.scatter('area', 'poptotal', data=midwest[midwest["state"].isin(["WI"])], s=20, labe

# Decorations
plt.gca().set(xlim=(0.0, 0.1), ylim=(0, 90000),
              xlabel='Area', ylabel='Population')
plt.xticks(fontsize=12); plt.yticks(fontsize=12)
plt.title("Scatterplot of Area vs Population", fontsize=22)
plt.legend(fontsize=12)
plt.show()
```

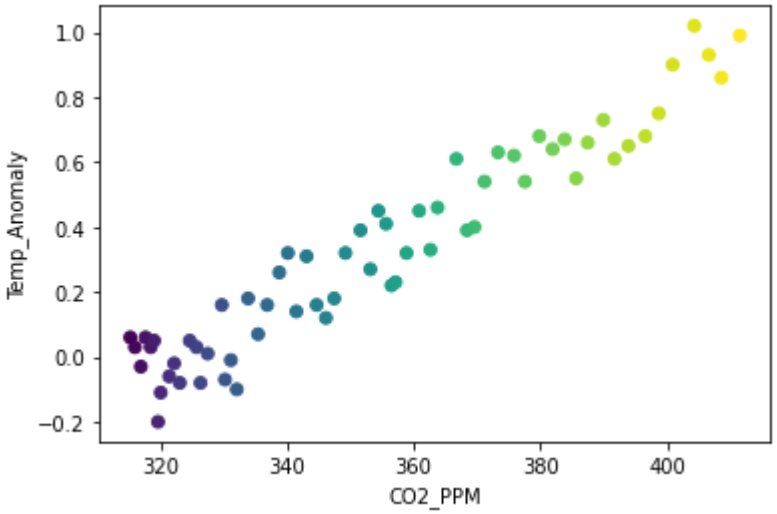


↥ back to top

17 Scatter plot for continuous levels like time axis

In [38]:

```
global_co2_temp_diff = pd.read_csv(input_files+"GLOBAL_CO2_PPM_TEMP_ANOMALY.csv",index_

fig,ax = plt.subplots()
ax.scatter(global_co2_temp_diff["CO2_PPM"],global_co2_temp_diff["Temp_Anomaly"],c=globa
ax.set_xlabel("CO2_PPM")
ax.set_ylabel("Temp_Anomaly")
plt.show()
```



↥ back to top

18 Choosing a style

In [39]:
```
plt.style.use("default")
#plt.style.use("ggplot")
#plt.style.use("seaborn")
#plt.style.use("tableau-colorblind10")
```

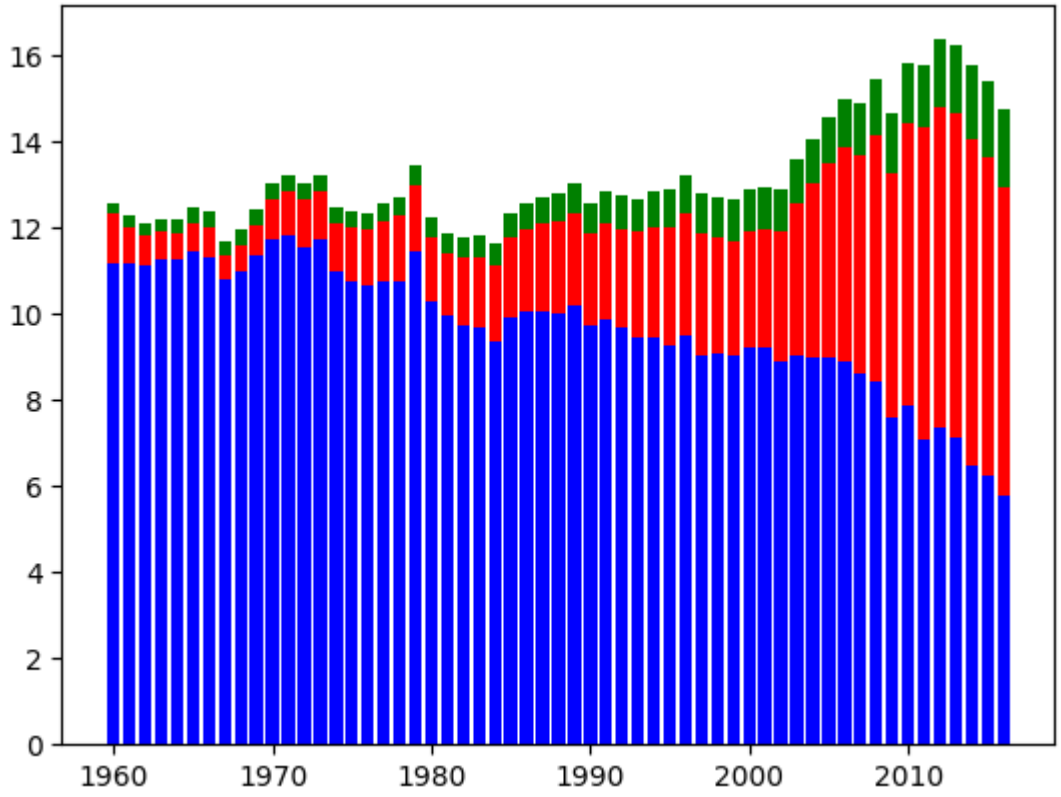
Read more:

- https://matplotlib.org/gallery/style_sheets/style_sheets_reference.html

↥ back to top

19 Saving your visualizations

In [40]:
```
# Stacked bar charts
data = CO2_PPM_COUNTRIES[["YEAR","United Kingdom","China","India"]]
fig,ax = plt.subplots()
ax.bar(data["YEAR"],data["United Kingdom"],color='b')
ax.bar(data["YEAR"],data["China"],color='r',bottom=data["United Kingdom"])
ax.bar(data["YEAR"],data["India"],color='g',bottom=data["United Kingdom"]+data["China"]

plt.show()

fig.savefig("Visualizations1.png",quality=50)  #.jpg,svg etc  ,  quality can be between
fig.savefig("Visualizations2.jpg",dpi=300)     # dpi can also control quality
fig.set_size_inches([5,3])                    # Size or aspect ratio of the file
```




```
C:\Users\Prateek\AppData\Local\Temp/ipykernel_14328/520862575.py:10: MatplotlibDeprecati
onWarning: savefig() got unexpected keyword argument "quality" which is no longer suppor
ted as of 3.3 and will become an error two minor releases later
  fig.savefig("Visualizations1.png",quality=50)  #.jpg,svg etc  ,  quality can be betwee
n 0 to 100
```

In [41]:
```
summer_2016_medals = pd.read_csv(input_files+"Olympic_Medals_Rio.csv")
```

In [42]:
```
summer_2016_medals.head(5)
```

Out[42]:

	id	name	nationality	sex	dob	height	weight	sport	gold	silver	bronze
0	736041664	A Jesus Garcia	ESP	male	10/17/69	1.72	64.0	athletics	0	0	0
1	532037425	A Lam Shin	KOR	female	9/23/86	1.68	56.0	fencing	0	0	0
2	435962603	Aaron Brown	CAN	male	5/27/92	1.98	79.0	athletics	0	0	1
3	521041435	Aaron Cook	MDA	male	1/2/91	1.83	80.0	taekwondo	0	0	0
4	33922579	Aaron Gate	NZL	male	11/26/90	1.81	71.0	cycling	0	0	0

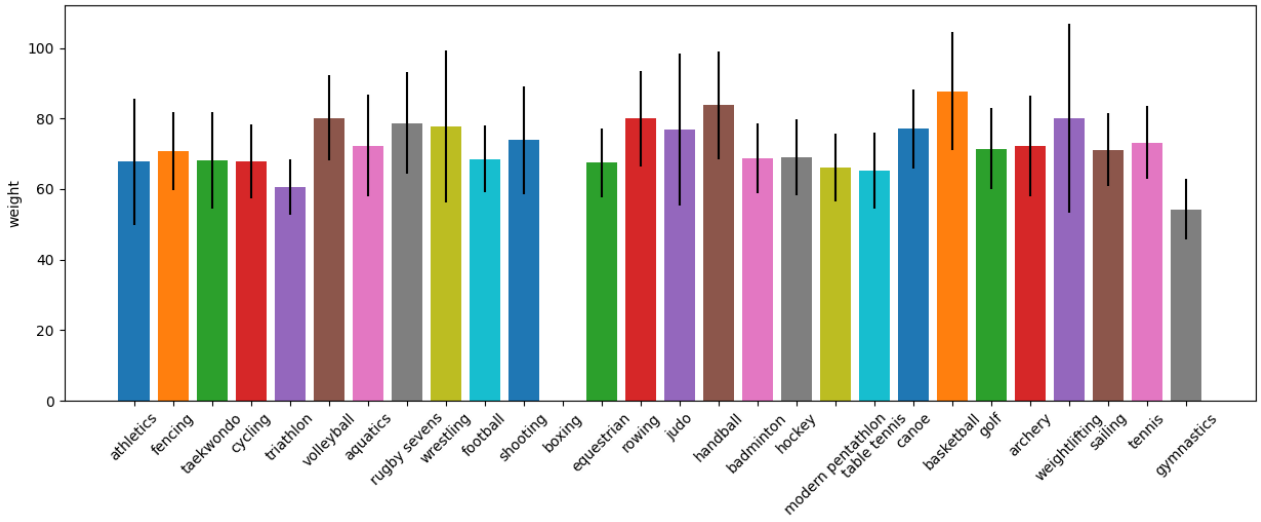
↥ back to top

20 Automating plots for large data frame

In [43]:

```
# Extract the "Sport" column
sports_column = summer_2016_medals["sport"]
# Find the unique values of the "Sport" column
sports = sports_column.unique()

fig, ax = plt.subplots(figsize=(15, 5))
# Loop over the different sports branches
for sport in sports:
  # Extract the rows only for this sport
  sport_df = summer_2016_medals[summer_2016_medals["sport"] == sport]
  # Add a bar for the "Weight" mean with std y error bar
  ax.bar(sport, sport_df["weight"].mean(), yerr=sport_df["weight"].std())

ax.set_ylabel("weight")
ax.set_xticklabels(sports, rotation=45,fontsize=10)

plt.show()
```

```
C:\Users\Prateek\AppData\Local\Programs\Python\Python39\lib\site-packages\matplotlib\axes\_base.py:2283: UserWarning: Warning: converting a masked element to nan.
  xys = np.asarray(xys)
C:\Users\Prateek\AppData\Local\Temp/ipykernel_14328/2152315007.py:15: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_xticklabels(sports, rotation=45,fontsize=10)
```



↥ back to top

21 Matplotlb CheatSheet

- https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Python_Matplotlib_Cheat_Sheet.pdf

22 Learn more

1. Gallery Offers many examples along with matplotlib codes https://matplotlib.org/gallery.html

2. Plotting data in 3D https://matplotlib.org/mpl_toolkits/mplot3d/tutorial.html

3. Visualizing data from images https://matplotlib.org/users/image_tutorial.html

4. Animations https://matplotlib.org/api/animation_api.html

5. Matplotlib for Geospatial data https://scitools.org.uk/cartopy/docs/latest

↥ back to top

Great Job!
