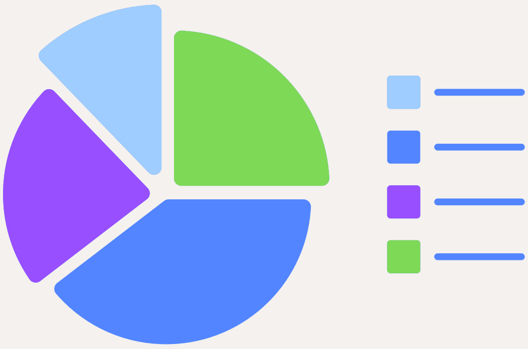
iamsujal456@gmail.com



Data Visualization Tools using Matplotlib

Sujal Gupta

Pie Charts

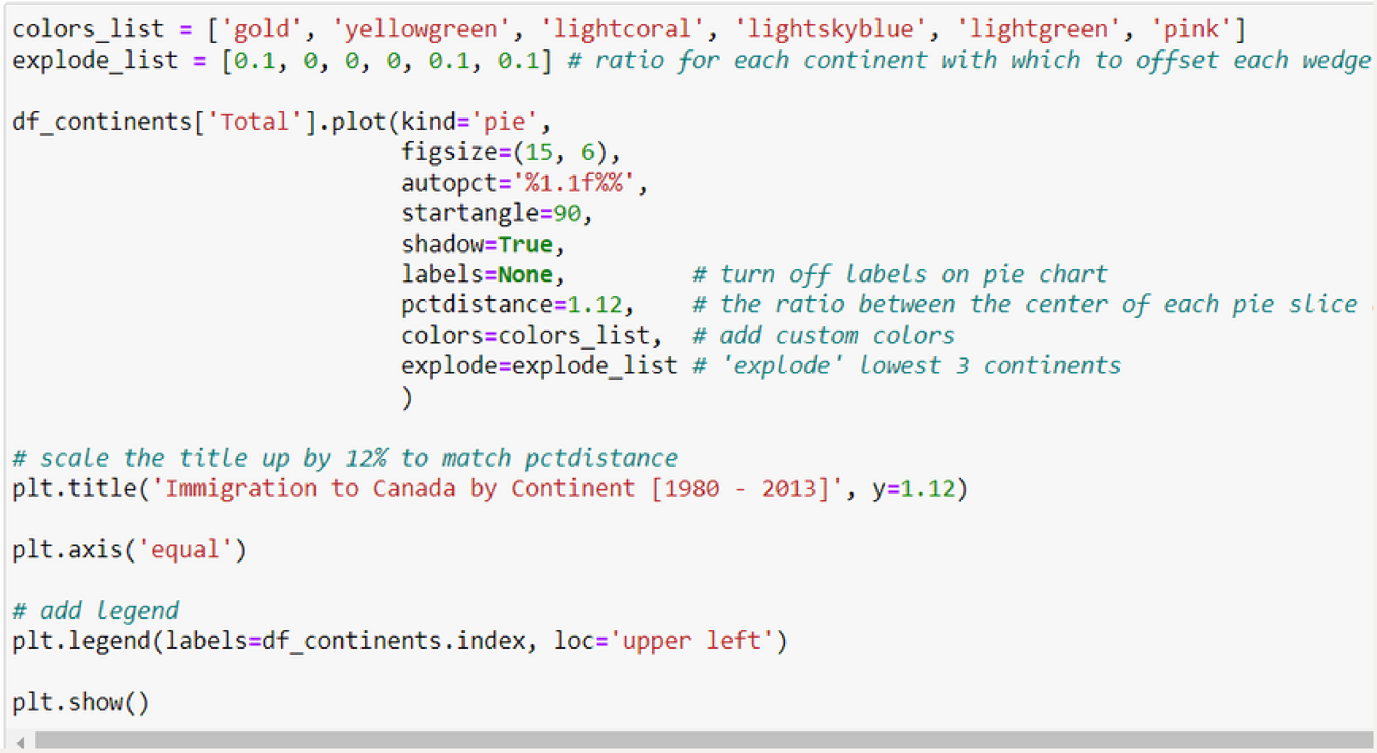
A pie chart is a circular graphic that displays numeric proportions by dividing a circle (or pie) into proportional slices.

We can create pie charts in Matplotlib by passing in the kind=pie keyword.

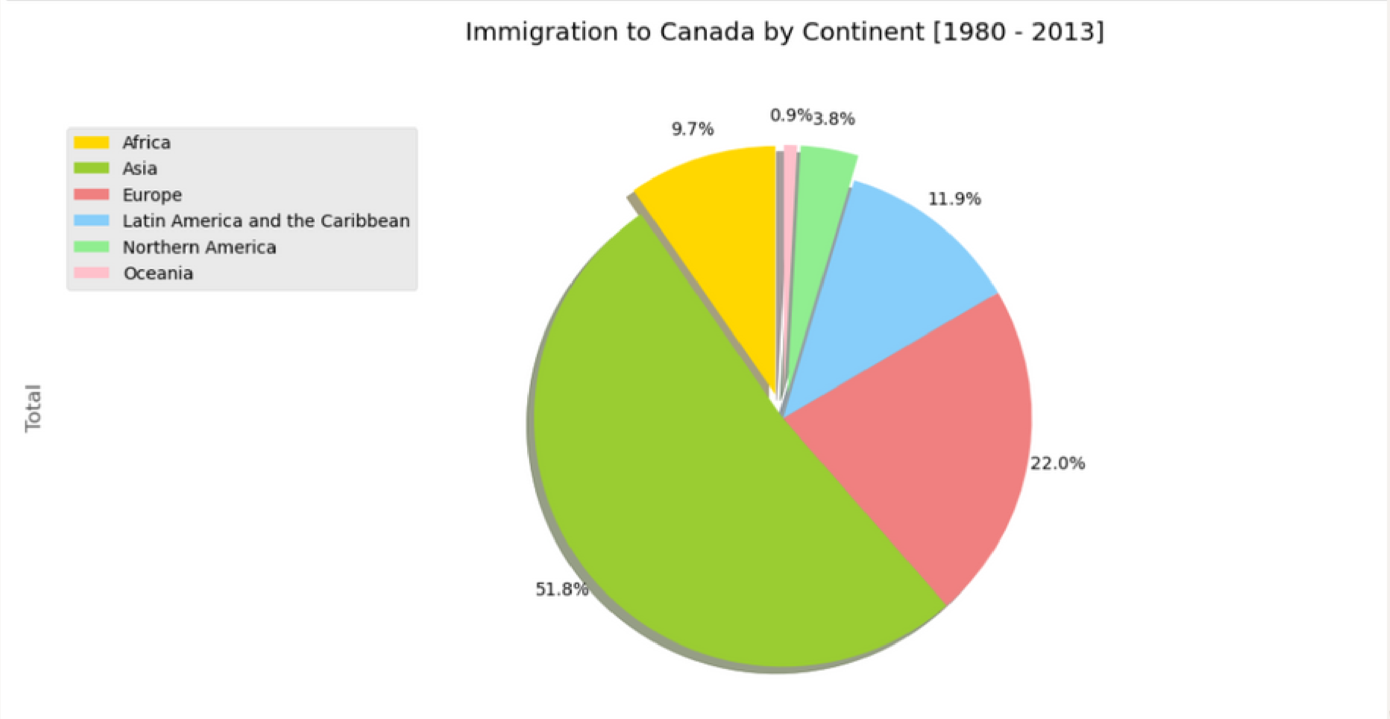
Several suggestions on how to use pie charts effectively:

- For simple data sets, use pie charts. Pie charts are best suited for data sets with a limited number of categories (typically no more than six).
- Make use of important colors. The colors you choose for your pie chart should be significant and easy to identify.
- Make use of a legend. If you have more than three categories, use a legend to identify the colors in each.
- Keep the pie chart as simple as possible. On your pie chart, avoid using too many labels or other features.
- Pie charts can be compared side by side. When comparing two or more pie charts, it's useful to show them side by side so they may be easily compared.

Sujal Gupta



```python
colors_list = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue', 'lightgreen', 'pink']
explode_list = [0.1, 0, 0, 0, 0.1, 0.1] # ratio for each continent with which to offset each wedge

df_continents['Total'].plot(kind='pie',
                            figsize=(15, 6),
                            autopct='%1.1f%%',
                            startangle=90,
                            shadow=True,
                            labels=None,         # turn off labels on pie chart
                            pctdistance=1.12,    # the ratio between the center of each pie slice
                            colors=colors_list,  # add custom colors
                            explode=explode_list # 'explode' lowest 3 continents
                            )

# scale the title up by 12% to match pctdistance
plt.title('Immigration to Canada by Continent [1980 - 2013]', y=1.12)

plt.axis('equal')

# add legend
plt.legend(labels=df_continents.index, loc='upper left')

plt.show()
```



Sujal Gupta

Box Plot

A box plot is a way of statistically representing the distribution of the data through five main dimensions:

- **Minimum:** The smallest number in the dataset excluding the outliers.
- **First quartile:** Middle number between the minimum and the median.
- **Second quartile (Median):** Middle number of the (sorted) dataset.
- **Third quartile:** Middle number between median and maximum.
- **Maximum:** The largest number in the dataset excluding the outliers.

To make a boxplot, we can use the kind=box in the plot method invoked on a pandas series or data frame.

Sujal Gupta

Some tips on how to use box plots effectively:

- Check to see if the data you're plotting is continuous. Box plots are not suitable for discrete data such as category or count data.
- Make sure to carefully label the axes of your box plot. This will assist your audience in comprehending what the data signifies.
- For box plots, use consistent colors and symbols. This will make comparing different box plots easier.
- Consider including a legend to explain the symbols in your box plots. This is especially significant if you're working with several symbols in a single plot.

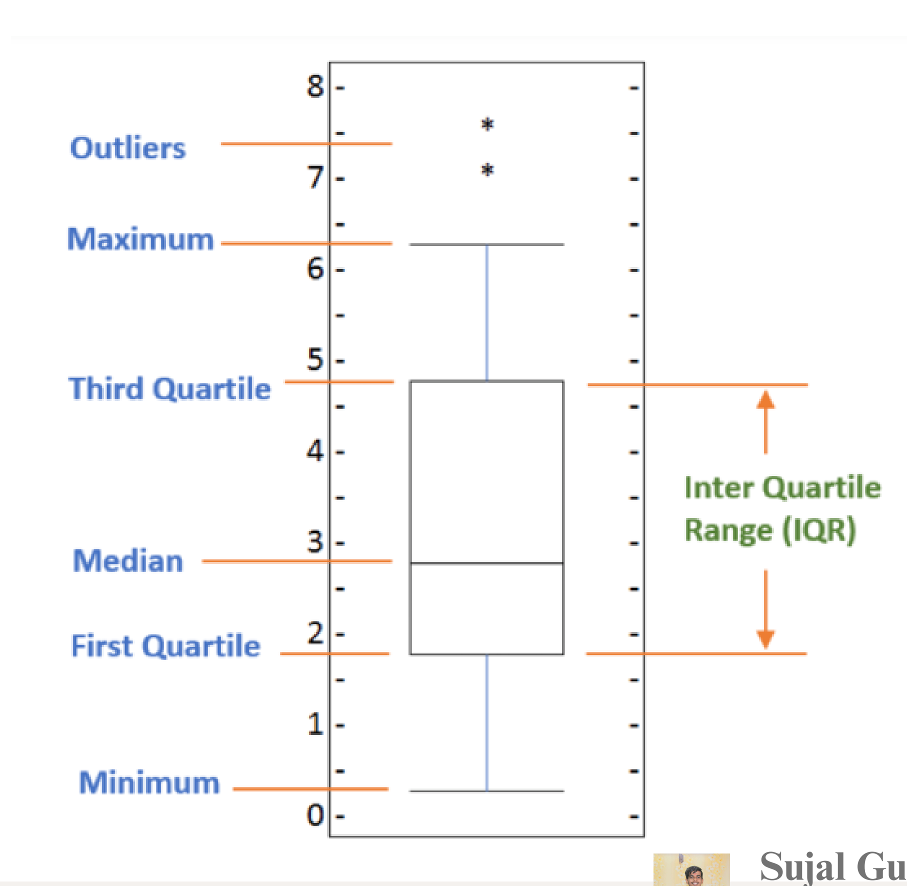
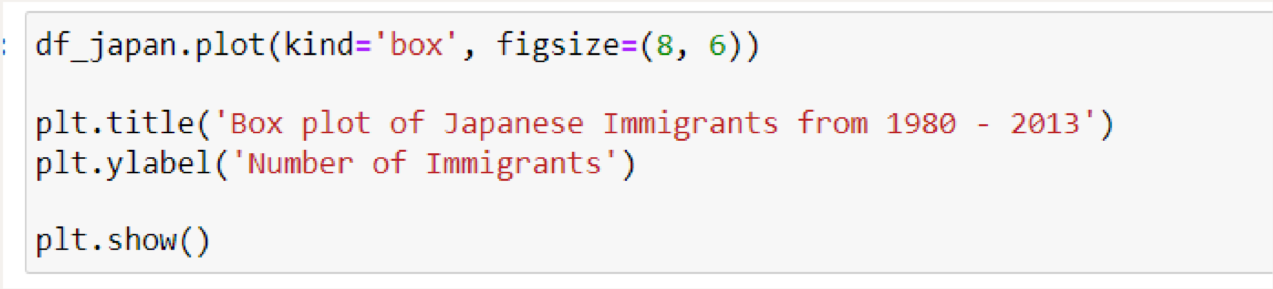
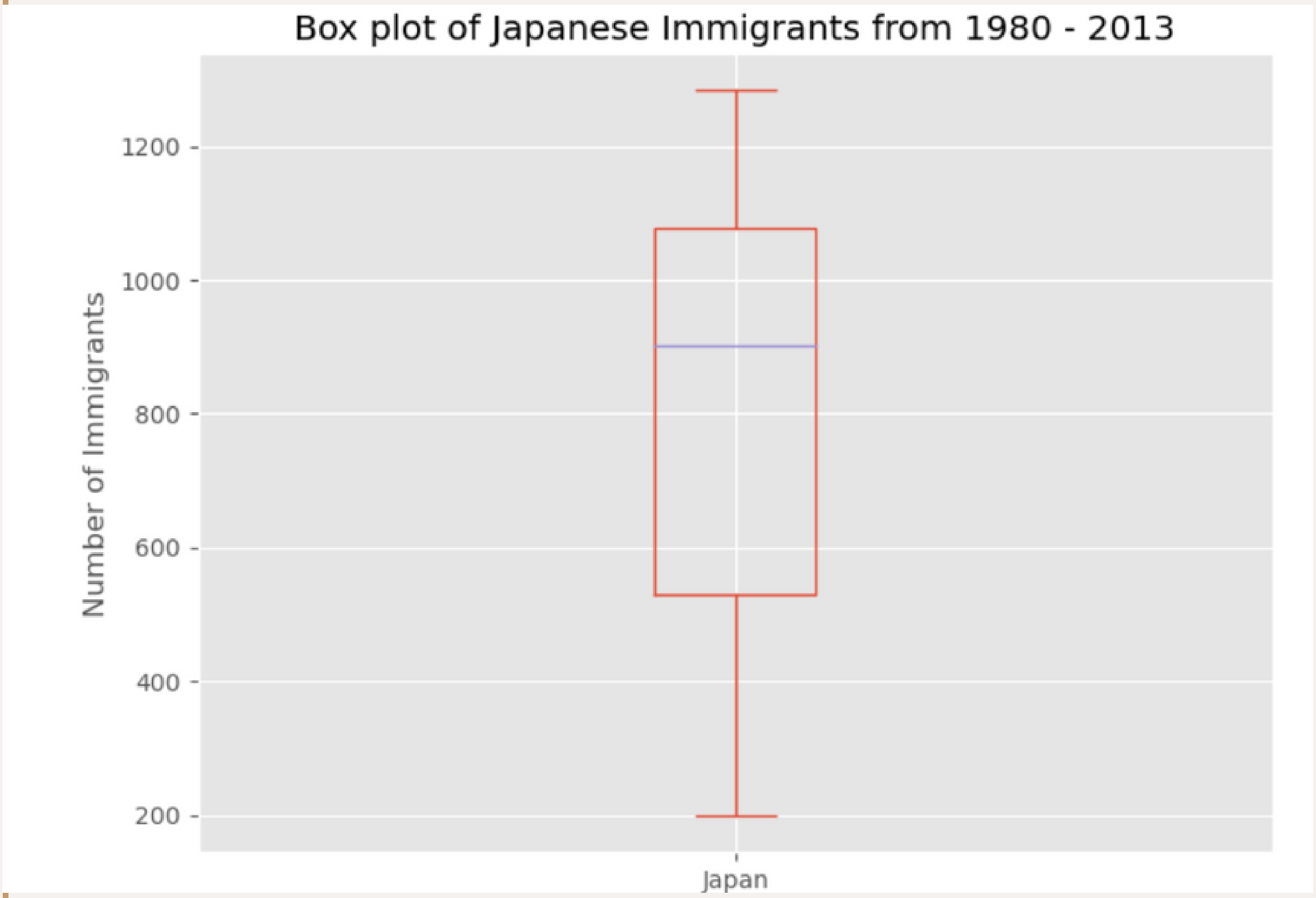


fig: Box Plots

Sujal Gupta



```
df_japan.plot(kind='box', figsize=(8, 6))

plt.title('Box plot of Japanese Immigrants from 1980 - 2013')
plt.ylabel('Number of Immigrants')

plt.show()
```

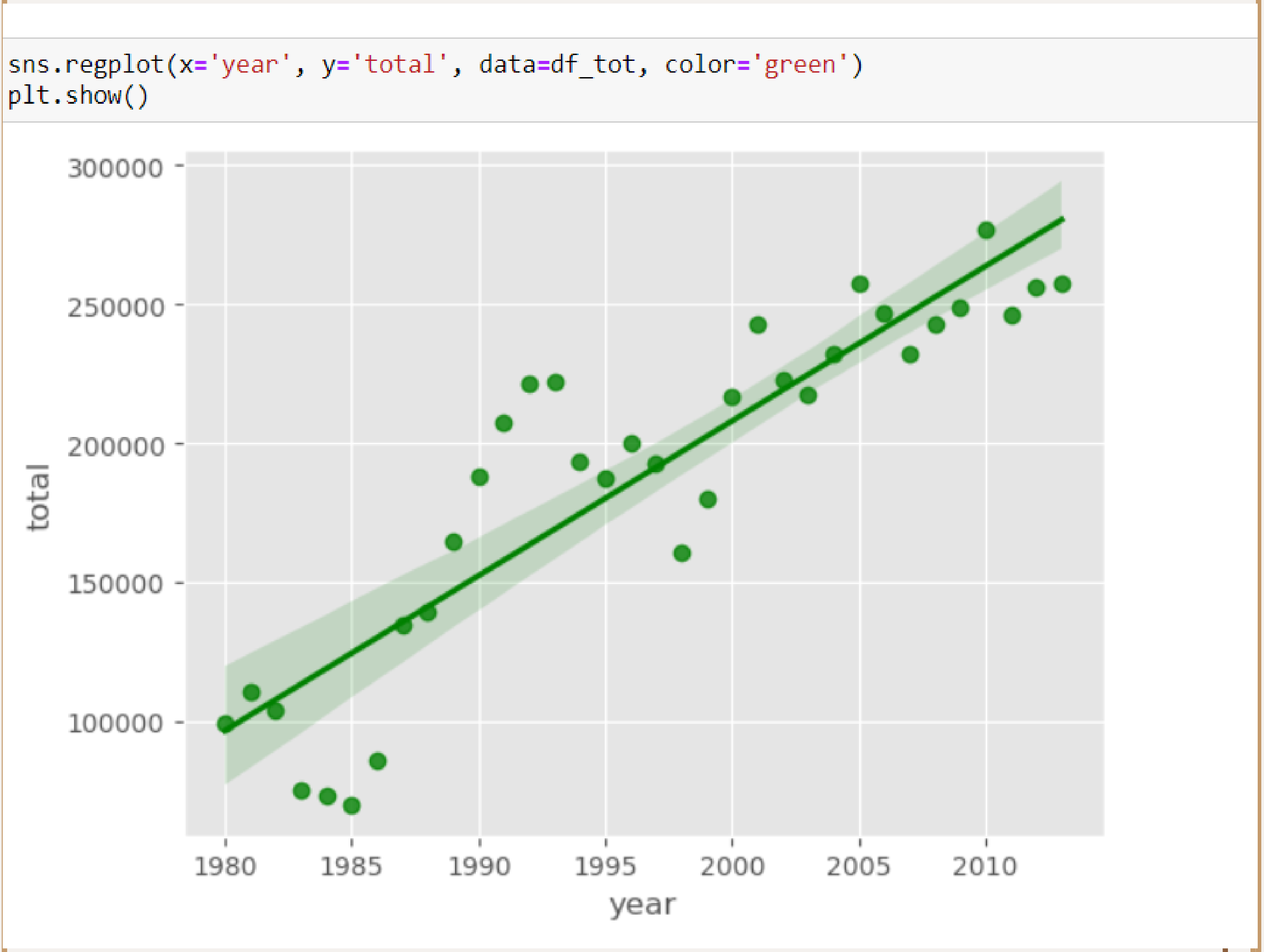


We can immediately make a few key observations from the plot above:

- The **minimum** number of immigrants is around 200 (min), the **maximum** number is around 1300 (max), and the **median** number of immigrants is around 900 (median).
- 25% of the years for the period 1980 - 2013 had an annual immigrant **count** of ~500 or fewer (**First quartile**).
- 75% of the years for the period 1980 - 2013 had an annual immigrant count of ~1100 or fewer (**Third quartile**).

Regression Plots

A regression plot is a type of scatter plot that shows the relationship between two variables. The independent variable is plotted on the x-axis, and the dependent variable is plotted on the y-axis. A regression line is then fit to the data, which shows the best-fit line for the data.

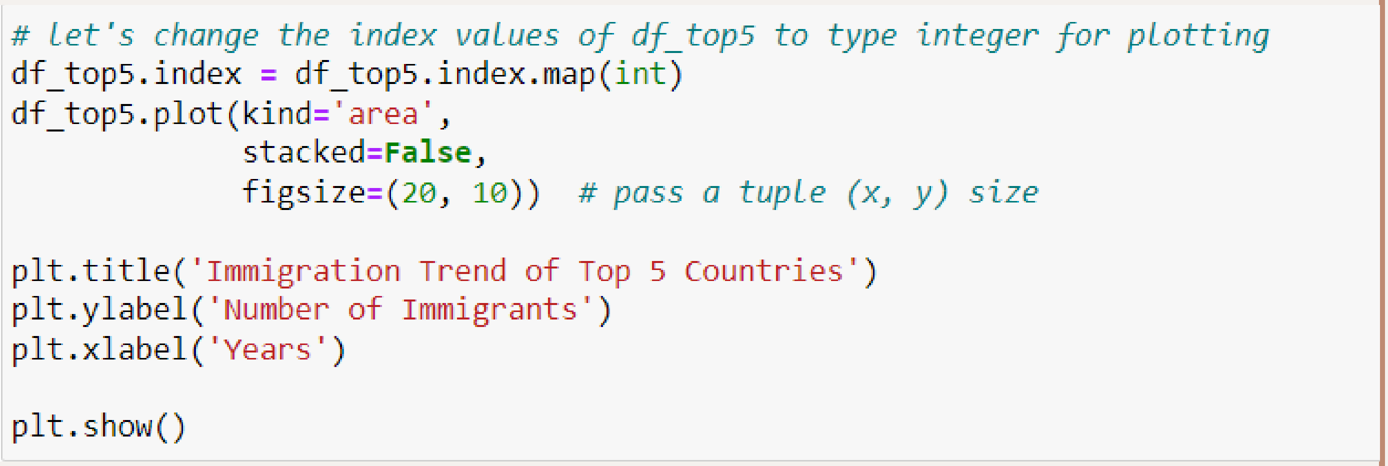


```
sns.regplot(x='year', y='total', data=df_tot, color='green')
plt.show()
```

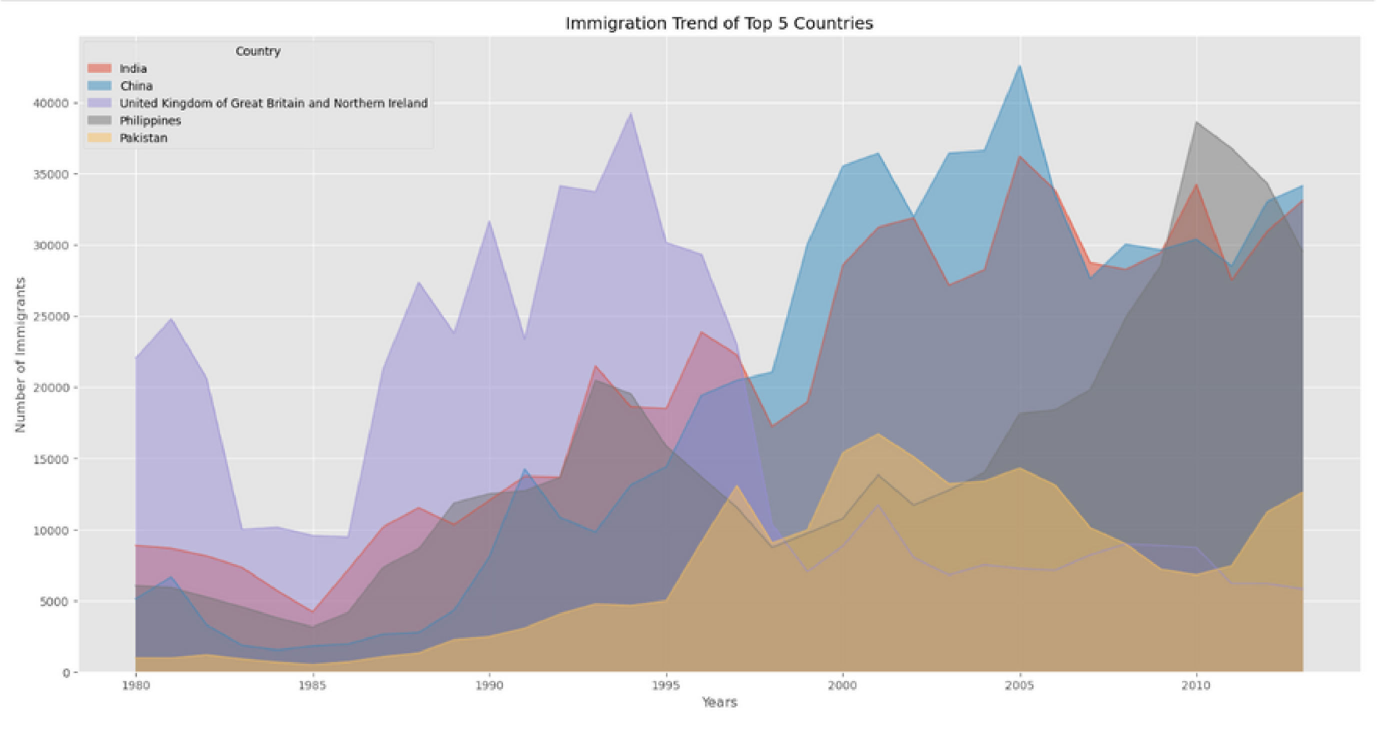
Sujal Gupta

Area plots

An area plot is a sort of chart that shows the evolution of quantitative values over time. It combines a line chart and a bar chart, with the region below the line filled in with a solid color. They can also be used to compare multiple types of data, such as sales of various goods or team performance.



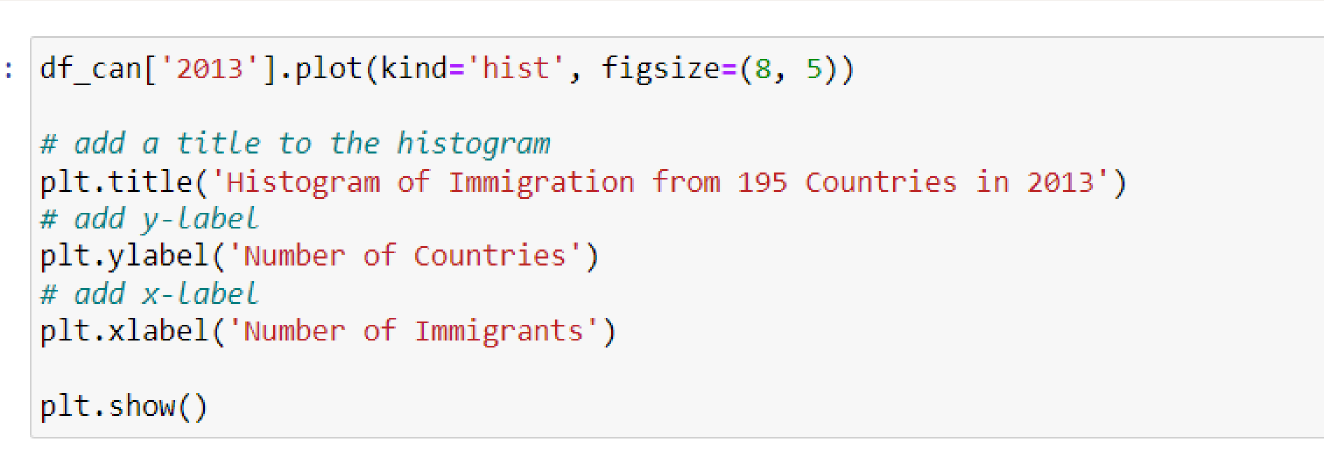
```
# let's change the index values of df_top5 to type integer for plotting
df_top5.index = df_top5.index.map(int)
df_top5.plot(kind='area',
             stacked=False,
             figsize=(20, 10))  # pass a tuple (x, y) size

plt.title('Immigration Trend of Top 5 Countries')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')

plt.show()
```



Sujal Gupta

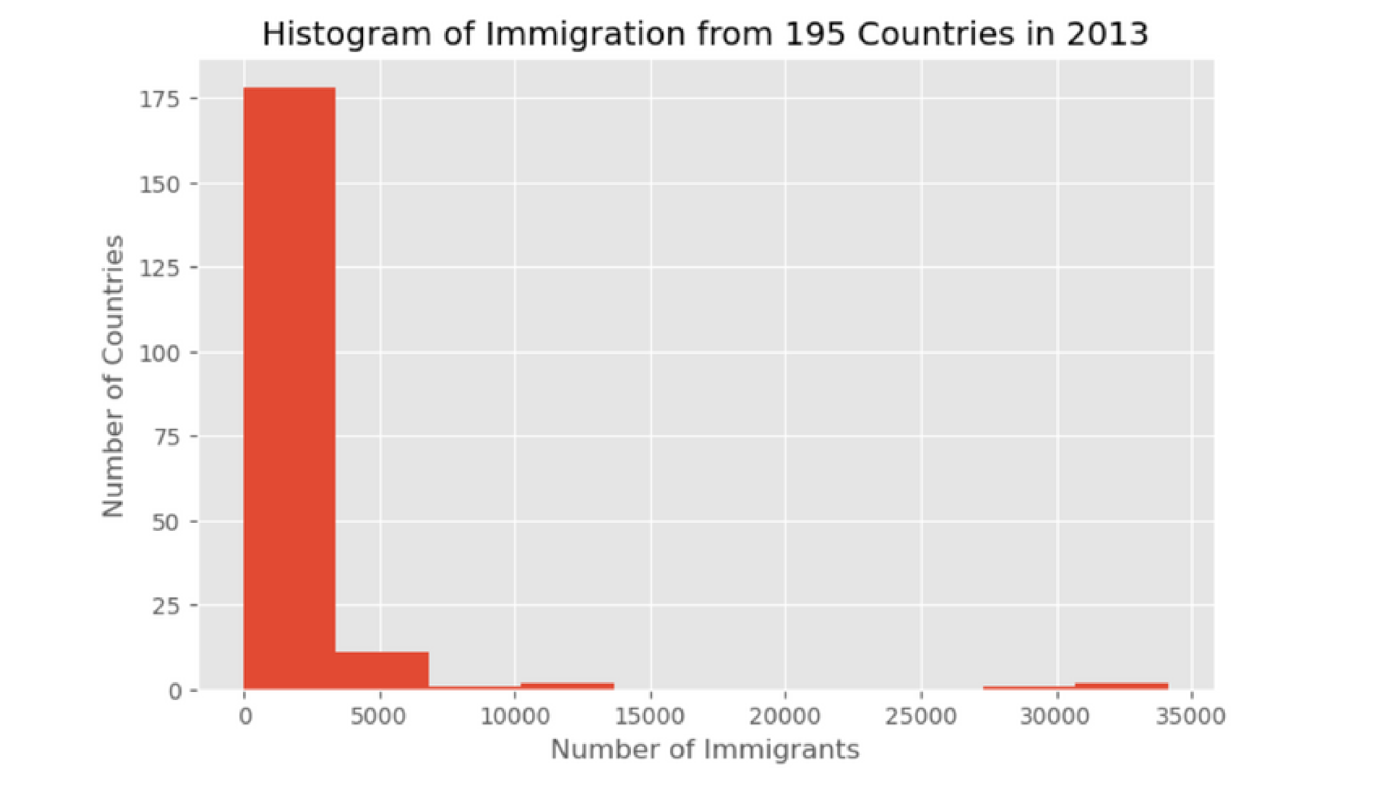
Histogram

A histogram is a way of representing the frequency distribution of a numeric dataset. The way it works is it partitions the x-axis into bins, assigns each data point in our dataset to a bin, and then counts the number of data points that have been assigned to each bin. So the y-axis is the frequency or the number of data points in each bin. Note that we can change the bin size and usually one needs to tweak it so that the distribution is displayed nicely



```
df_can['2013'].plot(kind='hist', figsize=(8, 5))

# add a title to the histogram
plt.title('Histogram of Immigration from 195 Countries in 2013')
# add y-label
plt.ylabel('Number of Countries')
# add x-label
plt.xlabel('Number of Immigrants')

plt.show()
```

Sujal Gupta



In the above plot, the x-axis represents the population range of immigrants in intervals of 3412.9. The y-axis represents the number of countries that contributed to the aforementioned population.

Sujal Gupta

Thank You!!!

Sujal Gupta

in