Data Visualization in Python

What is Data Visualization?

- Data visualization is the process of using charts, graphs, diagrams, and maps to visually represent data and information.
- It is used to explore, analyze, and communicate data insights in a simple, understandable, and visually appealing way.
- Data visualization is widely used in various fields such as business, healthcare, education, government, and science, among others.
- It helps in identifying patterns, trends, relationships, and outliers in data that may be difficult to discern from tables or spreadsheets.
- Different types of data visualizations include bar charts, line charts, scatter plots, histograms, heat maps, tree maps, and network diagrams.
- Choosing the right visualization for your data can greatly enhance the clarity and effectiveness of your analysis.

Before we get Started:

let's first install the necessary libraries (if required)

In [1]:
```
#!pip install matplotlib
#!pip install seaborn
#!pip install pandas
```

Step 1: Import the necessary libraries

First, we need to import the necessary libraries into our Python script.

In [2]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
```

Step 2: Load the data

Next, we need to load the data that we want to visualize. For this tutorial, we will be using the Iris dataset, which is a popular dataset for data visualization and machine learning.

In [3]:
```
iris = sns.load_dataset('iris') # Loading the dataset
df = pd.DataFrame(iris) # Storing the dataframe into Pandas
```

In [4]:
```
df.head()
```

Out[4]:

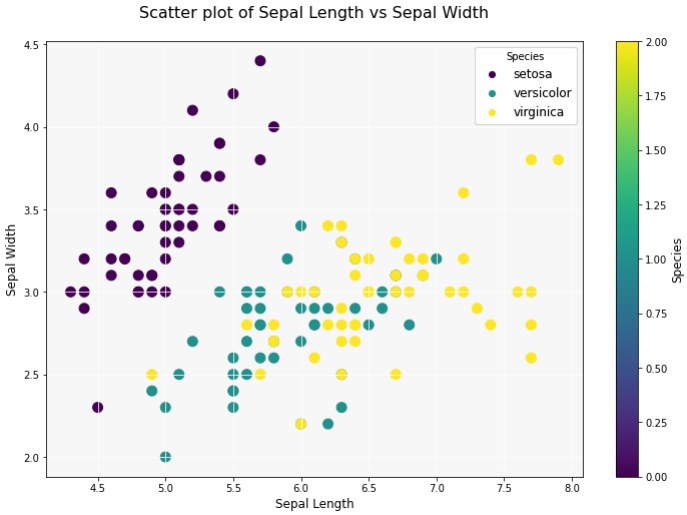
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [5]:
```
df['species'] = df['species'].astype('category')
```

Step 3: Create a basic plot with customization

Now that we have our data loaded, we can create a basic plot to visualize it. In this example, we will create a scatter plot to visualize the relationship between the sepal length and sepal width of the iris flowers. Add the following code to your Python script:

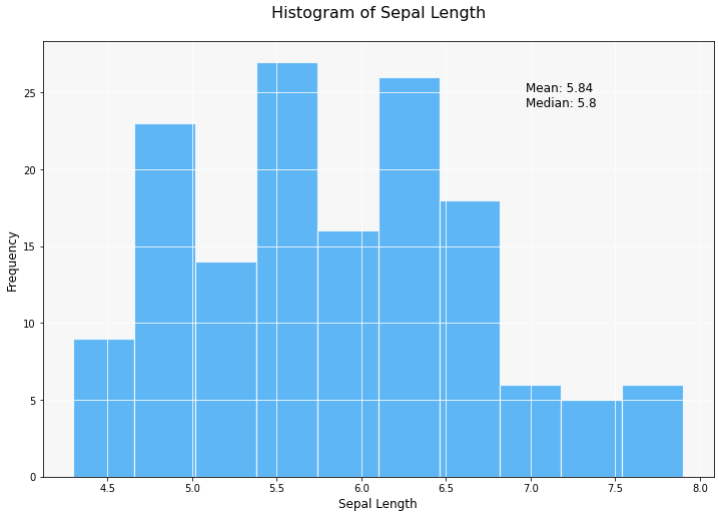
In [6]:
```
# Create scatter plot
fig, ax = plt.subplots(figsize=(12, 8))
scatter = ax.scatter(df['sepal_length'], df['sepal_width'], c=df['species'].cat.codes, cmap='viridis', s=100)

# Add labels and title
ax.set_xlabel('Sepal Length', fontsize=12)
ax.set_ylabel('Sepal Width', fontsize=12)
ax.set_title('Scatter plot of Sepal Length vs Sepal Width\n', fontsize=16)

# Add colorbar legend
cbar = fig.colorbar(scatter)
cbar.set_label('Species', fontsize=12)

# Set tick labels font size
ax.tick_params(axis='both', labelsize=10)

# Set plot background color and grid
ax.set_facecolor('#f7f7f7')
ax.grid(color='white')

# Create dictionary to map codes to species names
species_dict = {0: 'setosa', 1: 'versicolor', 2: 'virginica'}

# Add legend with species names
legend_handles, _ = scatter.legend_elements()
legend_labels = [species_dict[code] for code in df['species'].cat.codes.unique()]
legend = ax.legend(legend_handles, legend_labels, title='Species', fontsize=12)
ax.add_artist(legend)

# Show plot
plt.show()
```



Step 4: Create a histogram with customization

Another common type of plot is a histogram, which shows the distribution of a single variable. Here's an example of how to create a histogram of the sepal length:

In [7]:
```
# Create histogram
fig, ax = plt.subplots(figsize=(12, 8))
hist = ax.hist(df['sepal_length'], bins=10, edgecolor='white', linewidth=1.2, color='#38A6F5', alpha=0.8)

# Add labels and title
ax.set_xlabel('Sepal Length', fontsize=12)
ax.set_ylabel('Frequency', fontsize=12)
ax.set_title('Histogram of Sepal Length\n', fontsize=16)

# Set tick labels font size
ax.tick_params(axis='both', labelsize=10)

# Set plot background color and grid
ax.set_facecolor('#f7f7f7')
ax.grid(color='white')

# Add text annotation for mean and median
mean = round(df['sepal_length'].mean(), 2)
median = round(df['sepal_length'].median(), 2)
ax.text(0.72, 0.85, f'Mean: {mean}\nMedian: {median}', transform=ax.transAxes, fontsize=12)

# Show plot
plt.show()
```

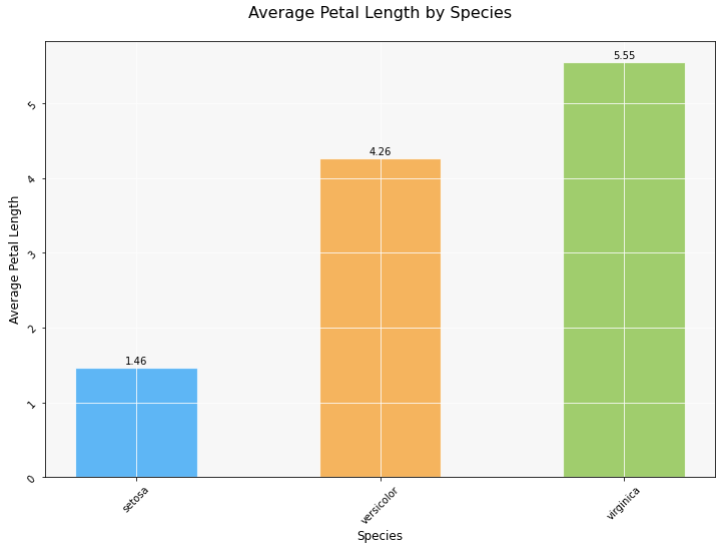


Step 5: Create a bar chart

A bar chart is used to compare different categories or groups. Here's an example of how to create a bar chart of the average petal length for each species of iris:

In [8]:
```
# Calculate mean petal length for each species
mean_petal_length = df.groupby('species')['petal_length'].mean()

# Create bar chart
fig, ax = plt.subplots(figsize=(12, 8))
bar = ax.bar(mean_petal_length.index, mean_petal_length.values, width=0.5,
             edgecolor='white', linewidth=1.2, color=['#38A6F5', '#F5A338', '#8BC34A'], alpha=0.8)

# Add labels and title
ax.set_xlabel('Species', fontsize=12)
ax.set_ylabel('Average Petal Length', fontsize=12)
ax.set_title('Average Petal Length by Species\n', fontsize=16)

# Set tick labels font size and angle
ax.tick_params(axis='both', labelsize=10, rotation=45)

# Set plot background color and grid
ax.set_facecolor('#f7f7f7')
ax.grid(color='white')
```

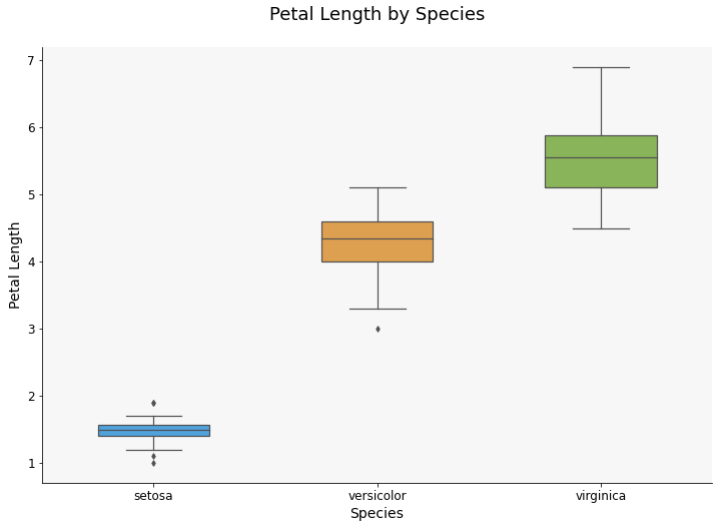
```
# Add value labels to bars
for i in range(len(bar)):
    value = round(mean_petal_length.values[i], 2)
    ax.text(i, bar[i].get_height() + 0.05, value, ha='center', fontsize=10)

# Show plot
plt.show()
```



Step 6: Create a box plot

A box plot is used to show the distribution of a variable across different categories or groups. Here's an example of how to create a box plot of the petal length for each species of iris:

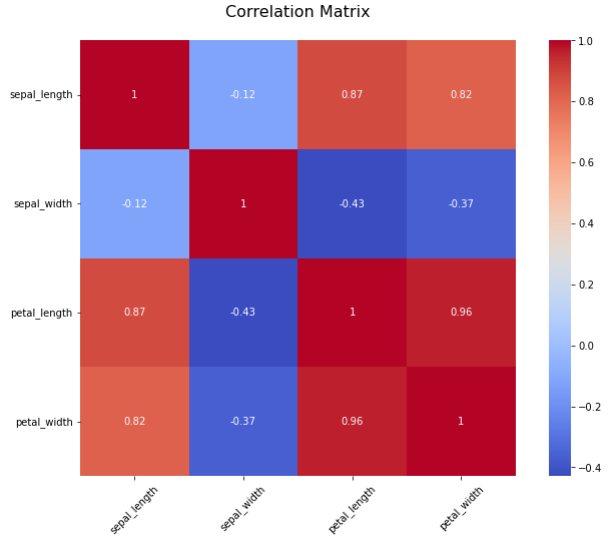
In [9]:
```
# Create box plot
fig, ax = plt.subplots(figsize=(12, 8))
box = sns.boxplot(x='species', y='petal_length', data=df, width=0.5, linewidth=1.2, fliersize=4, notch=False, palette=['#38A6F5', '#F5A338', '#8BC34A'])

# Add labels and title
ax.set_xlabel('Species', fontsize=14)
ax.set_ylabel('Petal Length', fontsize=14)
ax.set_title('Petal Length by Species\n', fontsize=18)

# Set tick labels font size and angle
ax.tick_params(axis='both', labelsize=12, rotation=0)

# Set plot background color
ax.set_facecolor('#f7f7f7')

# Remove top and right spines
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

# Set font size for x and y axis labels
ax.xaxis.label.set_size(14)
ax.yaxis.label.set_size(14)

# Add value labels to outliers
for i in range(len(box.artists)):
    outliers = box.artists[i].get_children()[1:-1]
    if outliers:
        for j in range(len(outliers)):
            value = round(float(outliers[j].get_ydata()[0]), 2)
            ax.text(i + 0.15, outliers[j].get_ydata()[0] + 0.1, value, ha='center', fontsize=10)

# Show plot
plt.show()
```



Step 8: Create a heat map

A heat map is used to show the relationship between two variables using colors. Here's an example of how to create a heat map of the correlation between the variables in the iris dataset:

In [10]:
```
# Calculate correlation matrix
corr = df.corr()

# Create heat map
fig, ax = plt.subplots(figsize=(12, 8))
hm = sns.heatmap(corr, annot=True, cmap='coolwarm', square=True, ax=ax)

# Customize heat map
hm.set_xticklabels(hm.get_xticklabels(), rotation=45, fontsize=10)
hm.set_yticklabels(hm.get_yticklabels(), rotation=0, fontsize=10)
hm.set_title('Correlation Matrix\n', fontsize=16)

# Show plot
plt.show()
```



Conclusion

- In this tutorial, we covered the basics of data visualization in Python using the Matplotlib and Seaborn libraries.
- We showed how to create scatter plots, histograms, bar charts, box plots, and heat maps using Python code.
- By following these steps, you should now have a good understanding of how to create various types of plots in Python for data analysis and visualization.