Here’s a list of important functions for **Seaborn** and **Matplotlib**, covering basic to advanced functionalities for creating visualizations.

**Seaborn Functions**

**1. Basic Plotting**

1. **sns.set()** – Set the default aesthetic parameters.
2. **sns.set\_style()** – Set the style of the plot (e.g., "white", "darkgrid").
3. **sns.lineplot()** – Plot a line chart.
4. **sns.scatterplot()** – Plot a scatter plot.
5. **sns.barplot()** – Plot a bar plot with categories and numerical variables.
6. **sns.histplot()** – Plot a histogram for univariate or bivariate data.
7. **sns.kdeplot()** – Plot a Kernel Density Estimate (KDE) for continuous data.
8. **sns.boxplot()** – Plot a boxplot for visualizing distributions and outliers.
9. **sns.violinplot()** – Plot a violin plot to show data distribution with KDE.
10. **sns.stripplot()** – Plot a scatter plot on a categorical axis.
11. **sns.swarmplot()** – Plot a scatter plot on a categorical axis with no overlap of points.
12. **sns.countplot()** – Plot the count of categorical data.
13. **sns.pairplot()** – Plot pairwise relationships in a dataset.
14. **sns.jointplot()** – Plot a bivariate plot with marginal distributions.
15. **sns.lmplot()** – Plot a linear regression model between two variables.

**2. Advanced Plotting**

1. **sns.heatmap()** – Plot a heatmap for matrix-style data.
2. **sns.clustermap()** – Plot a heatmap with hierarchical clustering.
3. **sns.pairplot()** – Plot pairwise relationships in a dataset with multiple types of plots.
4. **sns.catplot()** – Plot a categorical plot that allows multiple types (bar, violin, box, etc.).
5. **sns.regplot()** – Plot a scatter plot with a regression line.
6. **sns.factorplot()** – Plot relationships between a categorical and numerical variable (deprecated, replaced by catplot()).
7. **sns.jointplot()** – Plot a scatter plot along with histograms for the two variables.
8. **sns.despine()** – Remove or modify the spines (borders) around the plots.
9. **sns.violinplot()** – Visualize the distribution of data across multiple levels.
10. **sns.boxenplot()** – Plot an enhanced box plot for larger datasets.

**3. Data Manipulation**

1. **sns.load\_dataset()** – Load example datasets provided by Seaborn.
2. **sns.relplot()** – Plot a relationship between variables, allowing for multiple subplots (lineplot, scatterplot).
3. **sns.pairgrid()** – Create subplots to show pairwise relationships for multiple variables.
4. **sns.FacetGrid()** – Create a grid for plotting multiple plots based on categorical data.
5. **sns.cubehelix\_palette()** – Create a sequential colormap for plotting.
6. **sns.color\_palette()** – Set custom color palettes for the plots.
7. **sns.set\_palette()** – Change the current color palette for visualizations.

--------------------------------------------------------------------------------------------------------------------------------------

**Matplotlib Functions**

**1. Basic Plotting**

1. **plt.plot()** – Plot a line or set of points.
2. **plt.scatter()** – Plot a scatter plot of x and y points.
3. **plt.bar()** – Plot a vertical bar chart.
4. **plt.barh()** – Plot a horizontal bar chart.
5. **plt.hist()** – Plot a histogram for continuous data.
6. **plt.pie()** – Plot a pie chart.
7. **plt.boxplot()** – Plot a box plot for data distributions.
8. **plt.violinplot()** – Plot a violin plot for data distribution with KDE.
9. **plt.stem()** – Plot a stem plot.
10. **plt.errorbar()** – Plot an error bar plot.

**2. Customization**

1. **plt.title()** – Set the title of the plot.
2. **plt.xlabel()** – Set the label for the x-axis.
3. **plt.ylabel()** – Set the label for the y-axis.
4. **plt.legend()** – Add a legend to the plot.
5. **plt.grid()** – Add a grid to the plot.
6. **plt.xlim()** – Set the limits for the x-axis.
7. **plt.ylim()** – Set the limits for the y-axis.
8. **plt.xticks()** – Set the tick marks and labels for the x-axis.
9. **plt.yticks()** – Set the tick marks and labels for the y-axis.
10. **plt.subplots()** – Create a figure and a set of subplots.
11. **plt.subplot()** – Create a subplot within a figure.
12. **plt.gca()** – Get the current axis of the plot.
13. **plt.gcf()** – Get the current figure.
14. **plt.savefig()** – Save the current figure to a file.
15. **plt.tight\_layout()** – Automatically adjust subplot parameters for a tidy layout.
16. **plt.show()** – Display the plot.

**3. Advanced Plotting**

1. **plt.fill\_between()** – Fill the area between two horizontal curves.
2. **plt.contour()** – Plot contour lines for 2D arrays.
3. **plt.contourf()** – Plot filled contour areas for 2D arrays.
4. **plt.imshow()** – Display an image or a heatmap.
5. **plt.matshow()** – Visualize matrix data with colors.
6. **plt.pcolor()** – Create a pseudocolor plot with a non-regular rectangular grid.
7. **plt.hexbin()** – Plot a hexagonal binning plot for 2D data points.
8. **plt.quiver()** – Plot a 2D field of arrows.
9. **plt.streamplot()** – Plot a streamlines of a vector flow.
10. **plt.eventplot()** – Plot identical vertical or horizontal lines for events.

**4. Working with Figures and Subplots**

1. **plt.figure()** – Create a new figure.
2. **plt.add\_subplot()** – Add an axis to the figure at a specific position.
3. **plt.subplots\_adjust()** – Adjust the spacing of subplots.
4. **plt.text()** – Add text annotations to the plot.
5. **plt.annotate()** – Annotate a point in the plot with an arrow.
6. **plt.colorbar()** – Add a colorbar to the plot.
7. **plt.clf()** – Clear the current figure.
8. **plt.close()** – Close a figure window.