#_ important Seaborn Library Operations [+100]

Basic Plots:

- sns.lineplot(x, y): Plot data and a linear regression model fit.
- sns.scatterplot(x, y): Draw a scatter plot with possibility of several semantic groupings.
- sns.distplot(a): Flexibly plot a univariate distribution of observations.
- sns.barplot(x, y): Show point estimates and confidence intervals as rectangular bars.
- sns.countplot(x): Show the counts of observations in each categorical bin using bars.
- sns.boxplot(x, y): Draw a box plot to show distributions with respect to categories.
- sns.violinplot(x, y): Draw a combination of boxplot and kernel density estimate.
- sns.stripplot(x, y): Draw a scatterplot where one variable is categorical.
- sns.swarmplot(x, y): Draw a categorical scatterplot with non-overlapping points.
- sns.heatmap(data): Plot rectangular data as a color-encoded matrix.

Advanced Plots:

- sns.jointplot(x, y): Draw a plot of two variables with bivariate and univariate graphs.
- sns.pairplot(data): Plot pairwise relationships in a dataset.
- sns.lmplot(x, y, data): Plot data and regression model fits across a FacetGrid.
- sns.kdeplot(data): Fit and plot a univariate or bivariate kernel density estimate.
- sns.regplot(x, y): Plot data and a linear regression model fit.
- sns.residplot(x, y): Plot the residuals of a linear regression.

By: Waleed Mousa

- sns.relplot(x, y, kind): Figure-level interface for drawing relational plots onto a FacetGrid.
- sns.catplot(x, y, kind): Figure-level interface for drawing categorical plots onto a FacetGrid.
- sns.factorplot(x, y, kind): Deprecated, use sns.catplot() instead.
- sns.clustermap(data): Plot a matrix dataset as a hierarchically-clustered heatmap.

Styling **and** Color:

- sns.set_style(style): Set the aesthetic style of the plots.
- sns.set_context(context): Set the context parameters for plotting.
- sns.despine(): Remove the top and right spines from plot(s).
- sns.set_palette(palette): Set the color palette for all plots.
- sns.color_palette(palette): Return a color palette.
- sns.palplot(palette): Plot the values in a color palette as a horizontal array.

Statistical Estimation **within** Categories:

- sns.pointplot(x, y): Show point estimates and confidence intervals using scatter plot glyphs.
- sns.lmplot(x, y, hue): Plot data and regression model fits across a FacetGrid, colored by hue.
- sns.regplot(x, y, logistic=True): Plot data with a logistic regression model fit.

Multivariate Distributions:

- sns.jointplot(x, y, kind="kde"): Draw a plot of two variables with bivariate and univariate KDE graphs.
- sns.pairplot(data, hue): Plot pairwise relationships colored by a variable.
- sns.kdeplot(data, data2): Bivariate kernel density estimate.

Grids:

- `sns.FacetGrid(data, col, row)`: Multi-plot grid for plotting conditional relationships.
- `sns.PairGrid(data)`: Subplot grid for plotting pairwise relationships.
- `sns.JointGrid(x, y, data)`: Grid for drawing joint plots with univariate and bivariate visualizations.

Matrix Plots:

- `sns.clustermap(data, method)`: Hierarchically clustered heatmap.
- `sns.heatmap(data, annot=True)`: Heatmap with annotation cells.

Time Series:

- `sns.lineplot(x="time", y="signal", data=data)`: Time series line plot.
- `sns.tsplot(data)`: Deprecated, use `sns.lineplot()` instead.

Themes:

- `sns.set_theme(style, palette)`: Set the theme for seaborn plots.
- `sns.axes_style(style)`: Return a parameter dict for the aesthetic style of the plots.

Seaborn Datasets:

- `sns.load_dataset(name)`: Load a dataset from the online repository (requires internet).

Plotting with Parameters:

- `sns.lineplot(x, y, hue)`: Line plot with color by another variable.
- `sns.barplot(x, y, hue)`: Bar plot with color by another variable.
- `sns.countplot(x, data, palette)`: Count plot with specified palette.

Customizing Plots:

- `sns.set(rc={"figure.figsize":(x, y)})`: Change the size of the figure.

- `sns.plt.title("Title")`: Add a title to a plot.
- `sns.plt.xlabel("Label")`: Add a label to the x-axis.
- `sns.plt.ylabel("Label")`: Add a label to the y-axis.
- `sns.plt.xlim(min, max)`: Set the x-axis limits.
- `sns.plt.ylim(min, max)`: Set the y-axis limits.

```
Distribution Plots:
```

- `sns.histplot(data, bins)`: Histogram with specified number of bins.
- `sns.ecdfplot(data)`: Empirical Cumulative Distribution Function plot.
- `sns.rugplot(data)`: Draw a rugplot on the support axis.

```
Categorical Plots:
```

- `sns.catplot(x, y, data, kind)`: Categorical plot with kind: "point", "bar", "count", "box", "violin", "strip".
- `sns.boxenplot(x, y, data)`: Enhanced box plot for larger datasets.

```
Pairwise Relationships:
```

- `sns.pairplot(data, vars, hue, palette)`: Grid of pairwise relationships colored by a variable.

```
Plotting with Pandas:
```

- `sns.barplot(x="column1", y="column2", data=df)`: Bar plot using Pandas dataframes.
- `sns.lineplot(data=df[["col1", "col2", "col3"]])`: Line plot with multiple columns from a dataframe.

```
Saving Plots:
```

- `plt.savefig("filename.png")`: Save the current plot to a file.

```
Style and Palette Control:
```

- `sns.set_palette("husl", 3)`: Set the color palette to a specific number of colors.

- `sns.color_palette("rocket", as_cmap=True)`: Use a sequential colormap.

Subplots and Multiple Axes:

- `f, axes = plt.subplots(2, 2, figsize=(7, 7))`: Creating a subplot with 2x2 axes.
- `sns.despine(left=True)`: Remove the left spine from all subplots.

Combining Plots:

- `sns.violinplot(x, y, data, ax=axes[0, 0])`: Add a plot to a specific axis.

Advanced Customization:

- `sns.set_context("talk")`: Set context to "talk" for larger elements.
- `sns.set_style("whitegrid")`: Set background to white grid.
- `sns.despine(offset=10, trim=True)`: Offset and trim the spines.

Statistical Estimation within Categories:

- `sns.barplot(x="day", y="total_bill", data=tips, estimator=median)`: Estimator to change the aggregation function.

Plot Aesthetics:

- `sns.axes_style("darkgrid")`: Use a dark grid background.
- `sns.set_context("paper", font_scale=2)`: Set context to "paper" and increase font scale.

Plotting Wide-form Data:

- `sns.lineplot(data=df)`: Line plot with all columns in DataFrame.

Facet Grids and Categorical Data:

- `g = sns.FacetGrid(tips, col="time", row="smoker")`: Create a facet grid.
- `g.map(sns.histplot, "total_bill")`: Map a plot type to a facet grid.

Customizing with Matplotlib:

- `plt.title("Custom Title")`: Set the title using Matplotlib.
- `plt.ylim(0, 100)`: Set the y-axis limits using Matplotlib.
- `plt.xticks(rotation=45)`: Rotate x-axis labels.

Advanced Pairplot Customization:

- `sns.pairplot(data, diag_kind="kde", markers="+")`: Advanced customization of pairplot.
- `sns.pairplot(data, hue="species", palette="husl")`: Pairplot with hue and custom palette.

Correlation Heatmaps:

- `sns.heatmap(data.corr(), annot=True)`: Heatmap of the correlation matrix with annotations.

Seaborn Themes:

- `sns.set_theme(style="darkgrid", palette="pastel")`: Set the overall theme for Seaborn plots.

Kernel Density Estimation:

- `sns.kdeplot(data)`: Plot univariate or bivariate distributions using kernel density estimation.
- `sns.kdeplot(x, y, shade=True)`: Create a two-dimensional, shaded KDE plot.

Combining Plots:

- `sns.jointplot(x="x", y="y", data=data, kind="kde")`: Draw a plot of two variables with bivariate and univariate graphs.

- `sns.pairplot(data, hue="hue_var")`: Plot pairwise relationships in a dataset with hue differentiation.

Statistical Annotations:

- `sns.boxplot(x="x", y="y", data=data, showmeans=True)`: Add statistical annotations to boxplot.

Conditional Plots with FacetGrid:

- `g = sns.FacetGrid(data, col="col_var")`: Create a FacetGrid for conditional plots.
- `g.map(sns.histplot, "hist_var")`: Map a histogram to each facet.

Plotting Categorical Data:

- `sns.countplot(x="x", data=data)`: Show the counts of observations in each categorical bin.
- `sns.catplot(x="x", y="y", data=data, kind="violin")`: Use CatPlot to draw a categorical violin plot.

Plotting with Pandas DataFrames:

- `sns.lineplot(data=df, x="x", y="y")`: Line plot using a DataFrame.
- `sns.barplot(data=df)`: Bar plot using a DataFrame.

Working with Palettes:

- `sns.choose_colorbrewer_palette(data_type)`: Interactive palette selection tool.
- `sns.diverging_palette(h_neg, h_pos, s=75, l=50)`: Create a diverging color palette.

Advanced Customization:

- `sns.set_context("notebook", font_scale=1.5, rc={"lines.linewidth": 2.5})`: Advanced context settings for larger font and line widths.

- sns.lmplot(x="x", y="y", data=data, logistic=True, truncate=False): Customizing regression plots with logistic regression and no truncation.

Time Series Data:

- sns.lineplot(x="time", y="value", hue="region", style="event", data=data): Line plot with time series data with hue and style differentiation.

Distribution Comparisons:

- sns.kdeplot(data=data, x="value", hue="metric", multiple="fill"): Compare distributions with hue differentiation and filled KDE plots.
- sns.ecdfplot(data=data, x="value", hue="group", stat="count"): Empirical cumulative distribution plot with count normalization.

PairGrid Customization:

- g = sns.PairGrid(data, hue="hue_var", palette="Set2", diag_sharey=False): Customized PairGrid with hue and palette settings, and non-shared y-axes in diagonals.
- g.map_upper(sns.scatterplot): Map a scatter plot to the upper triangle of the PairGrid.
- g.map_lower(sns.kdeplot): Map a KDE plot to the lower triangle of the PairGrid.
- g.map_diag(sns.histplot): Map histograms to the diagonal of the PairGrid.
- g.add_legend(): Add a legend to a PairGrid.

Multi-Plot Grids:

- g = sns.FacetGrid(data, col="col_var", row="row_var", margin_titles=True): Create a FacetGrid with row and column variables and margin titles.
- g.map_dataframe(sns.scatterplot, x="x", y="y"): Map a DataFrame plot function to a FacetGrid.

- `g.set_axis_labels("X Axis", "Y Axis")`: Set axis labels for FacetGrid.
- `g.set_titles(col_template="{col_name} column", row_template="{row_name} row")`: Set titles with templates for a FacetGrid.

Themes and Styles:

- `sns.set_style("whitegrid")`: Set the Seaborn plot style to whitegrid for all plots.
- `sns.set_palette("Accent")`: Set the color palette to "Accent" for all plots.
- `sns.despine(trim=True)`: Remove the top and right spines and trim them.

Saving Figures:

- `plt.savefig("output.png")`: Save the current figure to a file named "output.png".

Interactive Visualizations:

- `sns.pairplot(data, kind="reg", diag_kind="kde", plot_kws=dict(scatter_kws=dict(s=50, edgecolor="b", alpha=0.7)))`: Pairplot with regression in pairwise relationships and KDE in the diagonal, including custom plot settings.