ChatGPT

Here's a **cheat sheet** summarizing key functions and useful features for **NumPy**, **Pandas**, **Matplotlib**, **Seaborn**, **Plotly**, and **Bokeh**. These are essential for quick reference and working efficiently with these libraries.

NumPy Cheat Sheet

1. Creating Arrays

- `np.array(list)`: Convert Python list into a NumPy array.

```python
arr = np.array([1, 2, 3])
```

- `np.zeros(shape)`: Create an array filled with zeros.

```python
arr = np.zeros((2, 3))
```

- `np.ones(shape)`: Create an array filled with ones.

```python
arr = np.ones((2, 3))
```

- `np.arange(start, stop, step)`: Create an array with evenly spaced values.

```python
arr = np.arange(0, 10, 2)
```

2. Array Operations

- `np.add(a, b)`, `a + b`: Element-wise addition of arrays.
- `np.subtract(a, b)`, `a - b`: Element-wise subtraction.
- `np.multiply(a, b)`, `a * b`: Element-wise multiplication.

- `np.dot(a, b)`: Matrix multiplication.

```python
result = np.dot(a, b)
```

3. Array Manipulation

- `arr.shape`: Get the shape of an array.

```python
arr.shape  # (2, 3)
```

- `arr.reshape(new_shape)`: Reshape the array.

```python
arr = arr.reshape(3, 2)
```

- `arr.flatten()`: Flatten a multi-dimensional array into one dimension.

4. Statistical Operations

- `np.mean(arr)`: Mean of the array.
- `np.median(arr)`: Median of the array.
- `np.std(arr)`: Standard deviation of the array.
- `np.var(arr)`: Variance of the array.
- `np.min(arr)`, `np.max(arr)`: Minimum/Maximum value of the array.

5. Random Numbers

- `np.random.rand(n)`: Generate an array of random numbers between 0 and 1.
- `np.random.randn(n)`: Generate an array of random numbers from a normal distribution.

Pandas Cheat Sheet

1. Creating DataFrames

- `pd.DataFrame(data)`: Create a DataFrame from a dictionary, list, or NumPy array.

```python
df = pd.DataFrame({'A': [1, 2], 'B': [3, 4]})
```

2. Data Selection

- `df['col_name']`: Select a column.

```python
df['A']
```

- `df.loc[row_index]`: Select rows by label.

```python
df.loc[0]  # First row
```

- `df.iloc[row_index]`: Select rows by position.

```python
df.iloc[0]  # First row
```

3. Data Aggregation

- `df.groupby('col_name')`: Group by a column.

```python
grouped = df.groupby('col_name').mean()
```

- `df.mean()`, `df.sum()`, `df.count()`: Basic aggregation functions.

4. Handling Missing Data

- `df.isna()`: Detect missing values.
- `df.dropna()`: Drop rows with missing values.
- `df.fillna(value)`: Replace missing values with a specific value.

```python

```

```
df.fillna(0)
```

5. Merging and Joining

- `pd.merge(df1, df2, on='key')`: Merge two DataFrames on a key.

```python
merged_df = pd.merge(df1, df2, on='key')
```

6. Sorting

- `df.sort_values('col_name')`: Sort DataFrame by column.

```python
df.sort_values('Age')
```

Matplotlib Cheat Sheet

1. Basic Plotting

- `plt.plot(x, y)`: Basic line plot.

```python
plt.plot(x, y)
plt.show()
```

- `plt.scatter(x, y)`: Scatter plot.

```python
plt.scatter(x, y)
```

2. Customization

- `plt.title('Title')`: Add title.
- `plt.xlabel('X-axis label')`, `plt.ylabel('Y-axis label')`: Add axis labels.

- `plt.grid(True)`: Show grid lines.

3. Subplots

- `plt.subplots(nrows, ncols)`: Create multiple plots.

```python
fig, ax = plt.subplots(2, 2)
```

4. Saving Plots

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

```python
plt.savefig('plot.png')
```

Seaborn Cheat Sheet

1. Basic Plots

- `sns.lineplot(x, y)`: Line plot.
- `sns.barplot(x, y)`: Bar plot.
- `sns.boxplot(x, y)`: Box plot.
- `sns.histplot(data)`: Histogram.

2. Advanced Plots

- `sns.pairplot(df)`: Pairwise relationships plot.

```python
sns.pairplot(df)
```

- `sns.heatmap(data)`: Heatmap for correlation.

```python
sns.heatmap(df.corr(), annot=True)
```

3. Categorical Data

- `sns.catplot(x, y, data=df, kind='bar')`: Categorical plot (bar, box, etc.).

```python
sns.catplot(x="day", y="total_bill", data=tips, kind="box")
```

4. Styling

- `sns.set_style('whitegrid')`: Set plot style.
- `sns.set_palette('deep')`: Set color palette.

Plotly Cheat Sheet

1. Basic Plots

- `px.scatter(df, x, y)`: Scatter plot.

```python
fig = px.scatter(df, x="gdpPercap", y="lifeExp")
fig.show()
```

- `px.bar(df, x, y)`: Bar plot.

```python
fig = px.bar(df, x="continent", y="pop")
```

2. 3D Plots

- `px.scatter_3d(df, x, y, z)`: 3D scatter plot.

```python
fig = px.scatter_3d(df, x="gdpPercap", y="lifeExp", z="pop")
fig.show()
```

3. Subplots

- `make_subplots()`: Create subplots with multiple charts.

```python
from plotly.subplots import make_subplots
fig = make_subplots(rows=1, cols=2)
```

4. Interactivity

- `fig.update_layout()`: Update layout properties like titles, axes, etc.

```python
fig.update_layout(title="GDP vs Life Expectancy")
```

Bokeh Cheat Sheet

1. Basic Plot

- `figure()`: Create a new plot.

```python
from bokeh.plotting import figure, show
plot = figure(title="Simple Plot", x_axis_label='X', y_axis_label='Y')
plot.line([1, 2, 3, 4], [4, 6, 2, 3], legend_label="Line", line_width=2)
show(plot)
```

2. Widgets

- `Slider(start, end, value)`: Create a slider.

```python
from bokeh.models import Slider
slider = Slider(start=0, end=10, value=1, step=0.1, title="Value")
```

- `TextInput(value)`: Create a text input widget.

```python
```

```python
from bokeh.models import TextInput
text_input = TextInput(value="default text")
```

3. Layouts

- `column()`, `row()`: Arrange plots and widgets vertically or horizontally.

```python
from bokeh.layouts import column
layout = column(slider, plot)
show(layout)
```

4. Interactive Plot

- `update()`: Dynamically update plot based on widget interaction.

Summary of Key Functions

- **NumPy**: Array creation, vectorized operations, random number generation.
- **Pandas**: DataFrame manipulation, grouping, aggregation, merging, and handling missing data.
- **Matplotlib**: Static plots, customization, subplots, and saving plots.
- **Seaborn**: Statistical plots, customization, and categorical plots.
- **Plotly**: Interactive plots, 3D plots, and subplots.
- **Bokeh**: Interactive plots, layouts, and widgets.

This cheat sheet covers key functionalities and provides examples for quick reference. You can dive deeper into each function by referring to the official documentation of the libraries.