# Exploratory Data Analysis (EDA) Cheat Sheet

## 1. Understanding the Data

This section helps you get acquainted with your dataset. You'll load your data into a DataFrame using Pandas, then use various methods to understand its structure and contents:

- pd.read_csv('your_dataset.csv'): Reads a CSV file into a Pandas DataFrame.
- df.head(): Displays the first few rows of the DataFrame.
- df.info(): Provides information about the DataFrame, including the data types and number of non-null values in each column.
- df.describe(): Generates descriptive statistics that summarize the central tendency, dispersion, and shape of the dataset's distribution.

### Quick Overview

In [ ]:
```python
import pandas as pd

# Load data into DataFrame
df = pd.read_csv('your_dataset.csv')

# Display first few rows
print(df.head())

# Summary information
print(df.info())

# Summary statistics
print(df.describe())
```

## 2. Handling Missing Data

### Identifying Missing Data

Missing data can skew analysis results, so it's essential to identify and handle it appropriately:

- df.isna().sum(): Counts the number of missing values in each column of the DataFrame.
- df.fillna(df.mean(), inplace=True): Fills missing values with the mean of each column. This is just one method; you can also choose to drop rows or columns with missing values.

In [ ]:
```python
# Check for missing values
print(df.isna().sum())
```

```
# Handle missing values
# Example: Fill missing values with mean
df.fillna(df.mean(), inplace=True)
```

## 3. Data Cleaning

### Addressing Inconsistencies

In [ ]:
```
# Clean data (address inconsistencies, outliers, errors)
# Example: Removing outliers
df = df[(df['column'] > lower_bound) & (df['column'] < upper_bound)]
```

## 4. Visualizing Data Distributions

Visualizations are powerful tools for understanding the distribution of data:

- Histograms: Display the frequency distribution of a numerical variable using bars.
- Density Plots: Show the distribution of data over a continuous interval.
- Box Plots: Provide a visual summary of the central tendency, dispersion, and skewness of numerical data, including outliers.

### Histograms

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

# Histogram
plt.hist(df['numerical_column'], bins=10)
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram of Numerical Column')
plt.show()
```

### Density Plots

In [ ]:
```
# Density plot
sns.kdeplot(df['numerical_column'], shade=True)
plt.xlabel('Value')
plt.ylabel('Density')
plt.title('Density Plot of Numerical Column')
plt.show()
```

### Box Plots

In [ ]:
```
# Box plot
sns.boxplot(x='category_column', y='numerical_column', data=df)
plt.xlabel('Category')
plt.ylabel('Value')
plt.title('Box Plot of Numerical Column Across Categories')
plt.show()
```

## 5. Analyzing Relationships

Understanding relationships between variables is crucial in data analysis:

- Scatter Plots: Visualize the relationship between two numerical variables by plotting points on a graph.
- Pair Plots: Show pairwise relationships in a dataset, including scatter plots for numerical variables and histograms for univariate distributions.
- Correlation Matrix: Displays the correlation coefficients between numerical variables as a heatmap, providing insights into their relationships.

### Scatter Plots

In [ ]:
```
# Scatter plot
plt.scatter(df['numerical_column1'], df['numerical_column2'])
plt.xlabel('Numerical Column 1')
plt.ylabel('Numerical Column 2')
plt.title('Scatter Plot of Numerical Columns')
plt.show()
```

### Pair Plots

In [ ]:
```
# Pair plot
sns.pairplot(df)
plt.title('Pair Plot of Numerical Columns')
plt.show()
```

### Correlation Matrix

In [ ]:
```
# Correlation matrix
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

## 6. Analyzing Categorical Variables

For categorical data, different types of visualizations and analyses are required:

- Bar Plots: Display the average value of a numerical variable for each category of a categorical variable.
- Count Plots: Show the frequency of each category in a categorical variable.

- Frequency Tables: Summarize the frequency of each category in a categorical variable in tabular form.

## Bar Plots

In [ ]:
```
# Bar plot
sns.barplot(x='category_column', y='numerical_column', data=df)
plt.xlabel('Category')
plt.ylabel('Average Value')
plt.title('Bar Plot of Numerical Column Across Categories')
plt.show()
```

## Count Plots

In [ ]:
```
# Count plot
sns.countplot(x='category_column', data=df)
plt.xlabel('Category')
plt.ylabel('Count')
plt.title('Count Plot of Categories')
plt.show()
```

## Frequency Tables

In [ ]:
```
# Frequency table
freq_table = pd.crosstab(index=df['category_column'], columns='count')
print(freq_table)
```