Report

**Describing every class:**

This code defines three classes: `DataProcessor`, `DataAnalyzer`, and `DataVisualizer`.

* The `DataProcessor` class has two methods: `\_\_init\_\_` and `load\_data`. The `\_\_init\_\_` method initializes the class instance with the file path of the data to be processed. The `load\_data` method reads the data from the file and returns it as a Pandas DataFrame. The `clean\_data` method takes a DataFrame as input, drops rows with missing data in the "Category" column, fills missing values in the "Rating" column with 0, and drops the "Product Link" column. The cleaned DataFrame is returned.
* The `DataAnalyzer` class has four methods: `get\_average\_of\_column`, `get\_distribution\_of\_column`, `get\_median\_of\_column`, and `get\_mod\_of\_column`. These methods take a column of data as input and return the column's average, standard deviation, median, and mode, respectively.
* The `DataVisualizer` class has four methods: `plot\_line\_chart`, `plot\_distribution`, `plot\_pie`, and `plot\_scatter`. These methods each create a different data visualization type and return the resulting plot. The `plot\_line\_chart` method plots a line chart of the "No of Seller" vs "Price" for each category of products. The `plot\_distribution` method plots a distribution plot of the "Quantity" column. The `plot\_pie` method plots a pie chart showing the percentage of products in each category. The `plot\_scatter` method plots a scatter plot of "Price" vs "Quantity" for each category, with the size of the points indicating the product rank.

The last part of the code creates an instance of the `DataProcessor` class, loads the data, and cleans it. Then it creates an instance of the `DataAnalyzer` class and uses its methods to analyze the data. Finally, it creates an instance of the `DataVisualizer` class and uses its methods to create different visualizations of the data.

**What Functionalities were used:**

The code uses several functionalities to process, analyze, and visualize data. Here are the main ones:

1. `numpy` (imported as `np`) is used to perform mathematical operations on arrays and calculate mean and standard deviation.

2. `pandas` (imported as `pd`) is used to read, manipulate, and clean data in the form of DataFrames.

3. `seaborn` (imported as `sns`) is used to create visualizations of the data, such as line charts, scatter plots, and distribution plots.

4. `matplotlib.pyplot` (imported as `plt`) is used to create pie charts of the data.

* The `DataProcessor` class is used to load and clean the data using Pandas methods.
* The `DataAnalyzer` class is used to perform statistical analysis on the data, such as calculating the mean, standard deviation, median, and mode of columns using NumPy methods.
* The `DataVisualizer` class is used to create different types of visualizations of the data using Seaborn and Matplotlib. It can create line charts, distribution plots, pie charts, and scatter plots of the data.

**Data Cleaning:**

The data was cleaned using the `clean\_data` method in the `DataProcessor` class. Here are the steps that were taken:

1. The method drops any rows that have missing values in the "Category" column using the `dropna` method with the `subset` parameter set to "Category".

2. The method fills any missing values in the "Rating" column with a value of 0 using the `fillna` method with the `inplace` parameter set to True.

3. The method drops the "Product Link" column using the `drop` method with the `axis` parameter set to 1 (indicating that the column should be dropped).

After these steps are taken, the cleaned data is returned as a Pandas DataFrame.

Here is the code for the `clean\_data` method:

```

def clean\_data(self, data):

data = data.dropna(subset=["Category"])

data["Rating"].fillna(0, inplace=True)

data.drop("Product Link", axis=1, inplace=True)

return data

```

Note that the `inplace=True` parameter is used in the `fillna` and `drop` methods to modify the DataFrame in place rather than returning a new copy.

**Charts Description:**

The code creates four different visualizations of the data using Seaborn and Matplotlib.

1. `plot\_line\_chart`: This visualization is a line chart that shows the relationship between "No of Seller" and "Price" for each category of products. The x-axis represents the number of sellers, the y-axis represents the price, and each category is represented by a different line color.

2. `plot\_distribution`: This visualization is a distribution plot that shows the distribution of values in the "Quantity" column. The x-axis represents the quantity, and the y-axis represents the frequency of each value. The plot also includes a kernel density estimate (KDE) that shows the shape of the distribution.

3. `plot\_pie`: This visualization is a pie chart that shows the percentage of products in each category. The chart includes three categories: "Electronics", "Clothing Shoes & Jewelry", and "Gift Cards". Each category is represented by a different color, and the percentage of products in each category is displayed in the corresponding slice of the chart.

4. `plot\_scatter`: This visualization is a scatter plot that shows the relationship between "Price" and "Quantity" for each category of products. The x-axis represents the price, the y-axis represents the quantity, and each category is represented by a different color. The size of the points indicates the product rank, with larger points indicating higher ranks.

Overall, these visualizations provide insights into the different aspects of the data, such as the relationship between price and quantity, the distribution of quantities, and the percentage of products in each category.

**Charts Analysis:**

1. `plot\_line\_chart`: This chart shows the relationship between the number of sellers and the price of products in each category. The chart indicates that there is a positive correlation between the number of sellers and the price of products in all categories. As the number of sellers increases, the price of products tends to increase. The chart also shows that products in the "Electronics" category tend to have higher prices than products in the other categories.

2. `plot\_distribution`: This chart shows the distribution of values in the "Quantity" column. The chart indicates that the most common quantity of products sold is around 20, with a long tail of less common quantities. The KDE plot indicates that the distribution is somewhat skewed to the right, with a longer tail on the right side of the chart.

3. `plot\_pie`: This chart shows the percentage of products in each category. The chart indicates that the "Electronics" category has the highest percentage of products, followed by the "Clothing Shoes & Jewelry" category. The "Gift Cards" category has the lowest percentage of products.

4. `plot\_scatter`: This chart shows the relationship between the price and quantity of products in each category. The chart indicates that there is a positive correlation between price and quantity in all categories. As the price of products increases, the quantity tends to increase as well. The chart also shows that products in the "Electronics" category tend to have higher prices and quantities than products in the other categories. Finally, the chart shows that higher-ranked products tend to have higher prices and quantities than lower-ranked products.