DALUBHASAAN NG LUNGSOD NG LUCENA
(formerly City College of Lucena)
City Hall Annex, Isabang, Lucena City
Telephone No. (042) 797-1671

CC104 - Data Structures and Algorithm
AY 2023 - 2024 | First Semester

Data Visualization

Data Visualization is the process of translating data and information into a visual context, making it easier for us to understand complex data sets. It involves using visual elements like charts, graphs, and maps.

Imagine you have a spreadsheet full of numbers. It might be hard to see any patterns or trends just by looking at the raw data. But if you take that same data and create a bar graph or a pie chart, patterns and trends become immediately apparent. That's the power of data visualization!
Here's why it's important:

1. Understand Trends: Visualizing data allows us to spot trends, patterns, and outliers that might not be obvious in raw, numerical data.
2. Simplify Complex Data: Large amounts of data can be difficult to understand. Visualization simplifies this and presents it in a way that is easy to interpret.
3. Aid in Decision Making: By understanding trends and patterns, businesses and individuals can make data-driven decisions.
4. Tell a Story: Data visualization can be used to tell a story about the data, making it a powerful tool for presentations or reports.

In essence, data visualization is a powerful tool that can help anyone, not just data scientists, to understand data better and make informed decisions. It's like a bridge that helps us cross from raw numbers to meaningful insights. I hope this helps! 😊

GUIDE TO VISUALIZATIONS

1. Histograms:

- Best Used For: Understanding the distribution of a single variable. Histograms show the frequency or probability distribution of a continuous variable.

2. Line Plot:

- Best Used For: Showing the relationship between two continuous variables, often used to represent trends over time.

3. Distplot (Deprecated in recent Seaborn versions; consider histplot and kdeplot):

- Best Used For: Combining a histogram and a kernel density estimate (KDE) plot to visualize the distribution of a single variable.

4. Violin Plots:

- Best Used For: Displaying the distribution and probability density of a continuous variable across different categories. It combines aspects of box plots and kernel density plots.

5. Boxplots:

- Best Used For: Summarizing the distribution of a continuous variable, particularly in comparing distributions across different categories. It provides information about the median, quartiles, and potential outliers.

6. KDE Plot:

- Best Used For: Visualizing the probability density function of a continuous variable. It can provide a smoother representation of the distribution compared to histograms.

7. FacetGrid (for creating subplots):

- Best Used For: Creating a grid of subplots based on categorical variables, allowing you to visualize relationships between multiple variables simultaneously.

8. Heatmap:

- Best Used For: Displaying a matrix of values using colors. Heatmaps are often used to show correlations in a dataset or to represent the magnitude of values in a 2D space.

It's important to choose the right visualization tool based on the nature of your data and the specific insights you want to gain. For example, if you want to understand the distribution of a single variable, a histogram or KDE plot might be appropriate. If you're comparing distributions across different categories, boxplots or violin plots might be more suitable.

Always consider the context of your analysis and the story you want to tell with the visualizations when selecting the appropriate tool. Additionally, experimenting with different visualization types can help you find the most effective way to convey your data's insights.

PYTHON COMMAND

1. Matplotlib:

Histogram:

```python
import matplotlib.pyplot as plt
data = [1, 2, 2, 3, 3, 3, 4, 4, 5]

plt.hist(data, bins=5, color='purple', edgecolor='black')
plt.xlabel('Values')
plt.ylabel('Frequency')
plt.title('Histogram')
```

Line Plot:

```python
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]

plt.plot(x, y, label='Line Plot')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Simple Line Plot')
plt.legend()
plt.show()
```

2. Seaborn:

Histogram:

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
sns.histplot(tips['total_bill'], bins=30, kde=True, color='skyblue')
plt.title('Histogram with Seaborn')
plt.show()
```

Line Plot:

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
sns.lineplot(x='total_bill', y='tip', data=tips, hue='sex', ci=None)
plt.title('Line Plot with Seaborn')
plt.show()
```

Distplot (deprecated in recent Seaborn versions, consider histplot and kdeplot):

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
# Deprecated: sns.distplot(tips['total_bill'], bins=30, color='skyblue')
sns.histplot(tips['total_bill'], bins=30, kde=True, color='skyblue')
plt.title('Histogram with Seaborn')
plt.show()
```

Violin Plots:

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
```

```python
sns.violinplot(x='day', y='total_bill', data=tips, palette='pastel')
plt.title('Violin Plot with Seaborn')
plt.show()
```

Boxplots:

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
sns.boxplot(x='day', y='total_bill', data=tips, palette='pastel')
plt.title('Box Plot with Seaborn')
plt.show()
```

KDE Plot:

```python
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset('tips')
sns.kdeplot(tips['total_bill'], fill=True, color='skyblue')
plt.title('KDE Plot with Seaborn')
plt.show()
```

These are basic examples, and each of these functions has additional parameters that allow you to customize the appearance of the plots. You can refer to the official documentation for Matplotlib and Seaborn for a comprehensive list of parameters and their descriptions:

References

- Matplotlib Documentation
- Seaborn Documentation