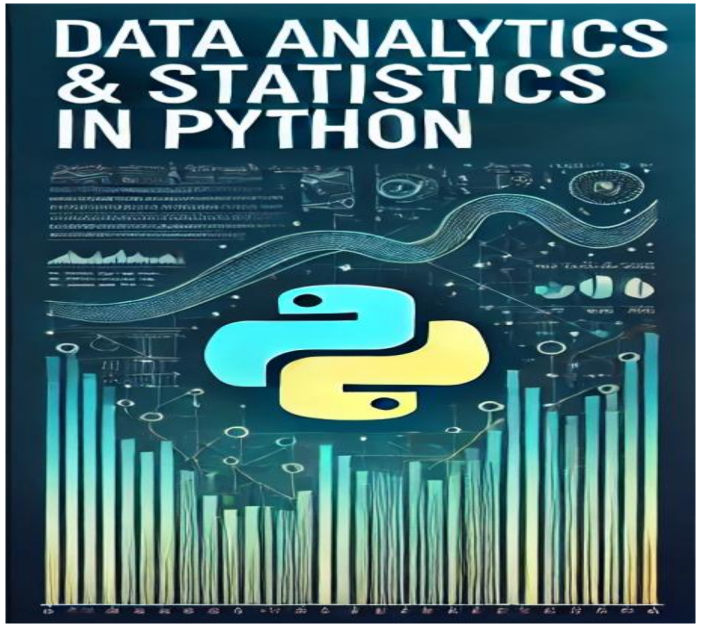
Data Analytics & Statistics in Python
Good & Bad Visualisation



Learning data-driven decision-making with Python

Instructor: Hamed Ahmadinia, Ph.D.

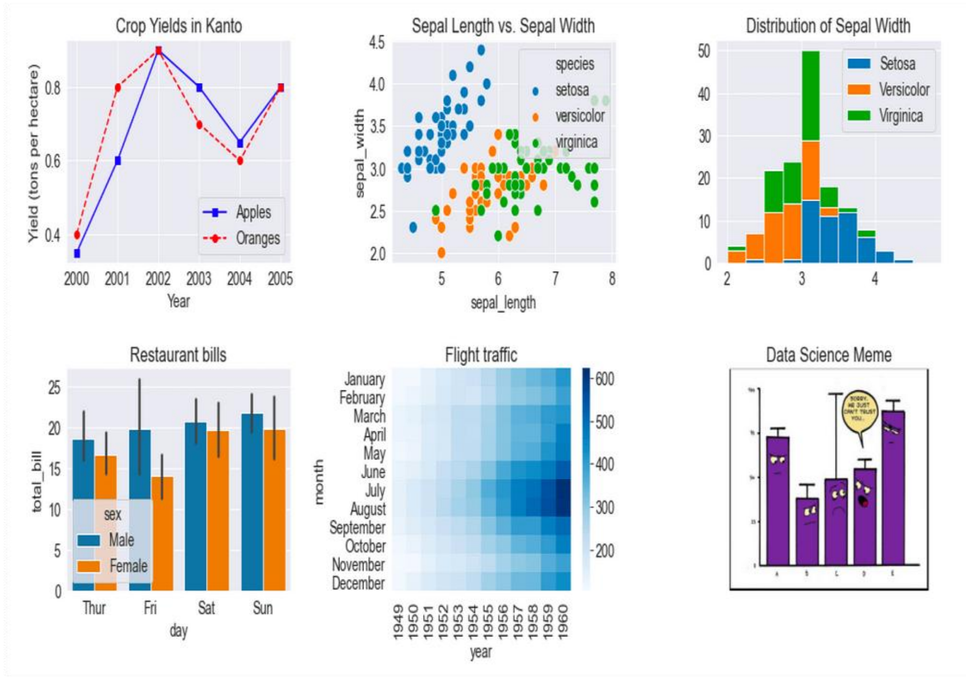
Email: hamed.ahmadinia@metropolia.fi

Data Visualisation in Python

Definition: graphical representation of information using visual elements (charts, graphs, maps).

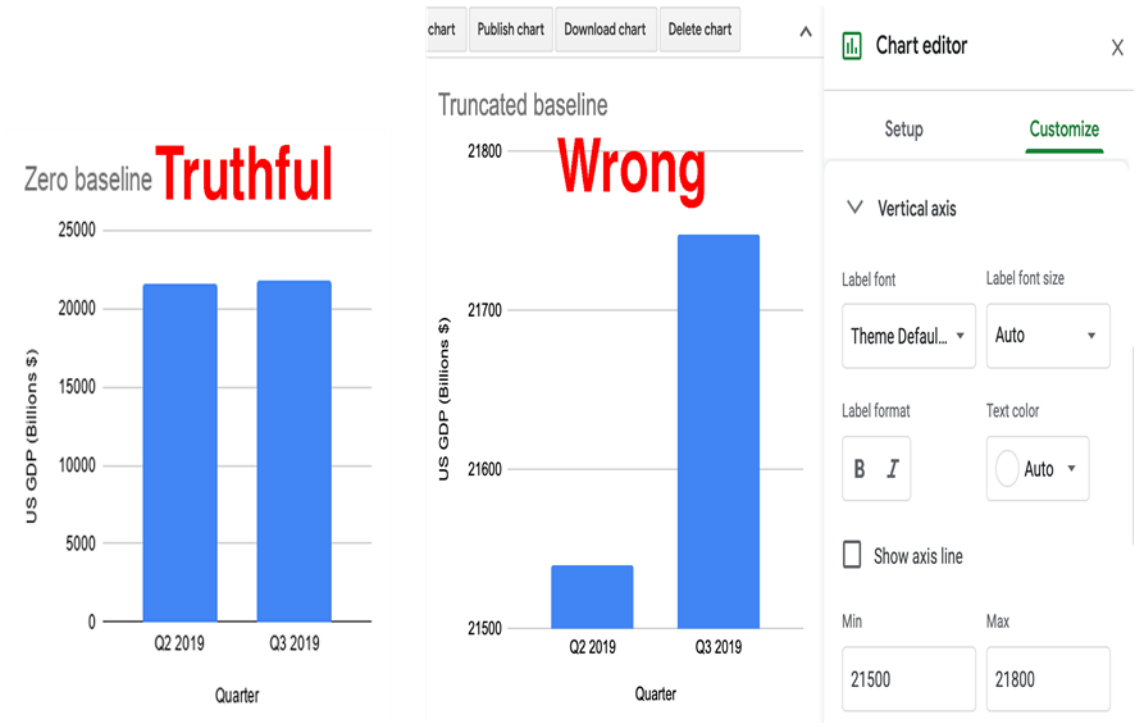
Importance: Helps quickly and clearly understand complex data.

Python Visualization Libraries: Matplotlib, Seaborn, Plotly, Pandas.



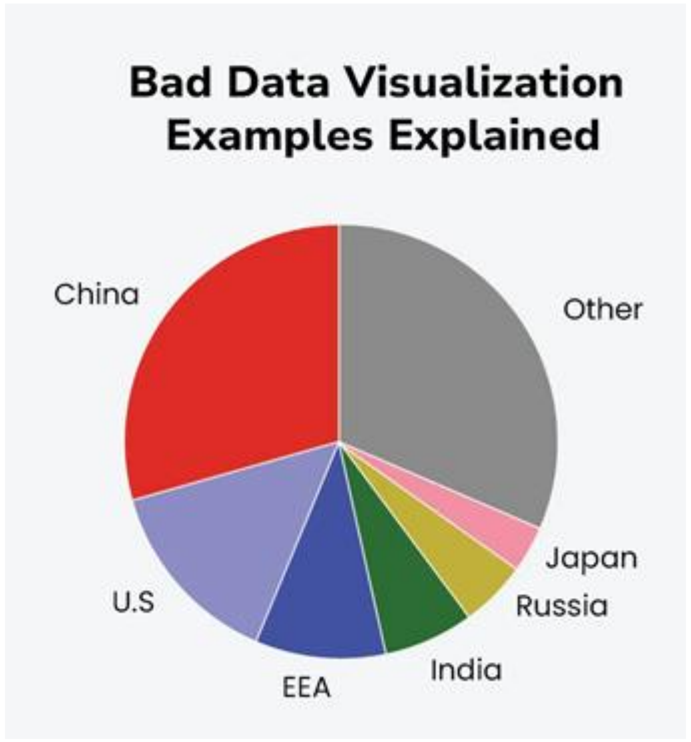
Characteristics of Good Data Visualisation

- Accurate representation
- Clear labeling and titles
- Appropriate scales and axes
- Effective color usage
- Minimal clutter
- Suitable chart type selection

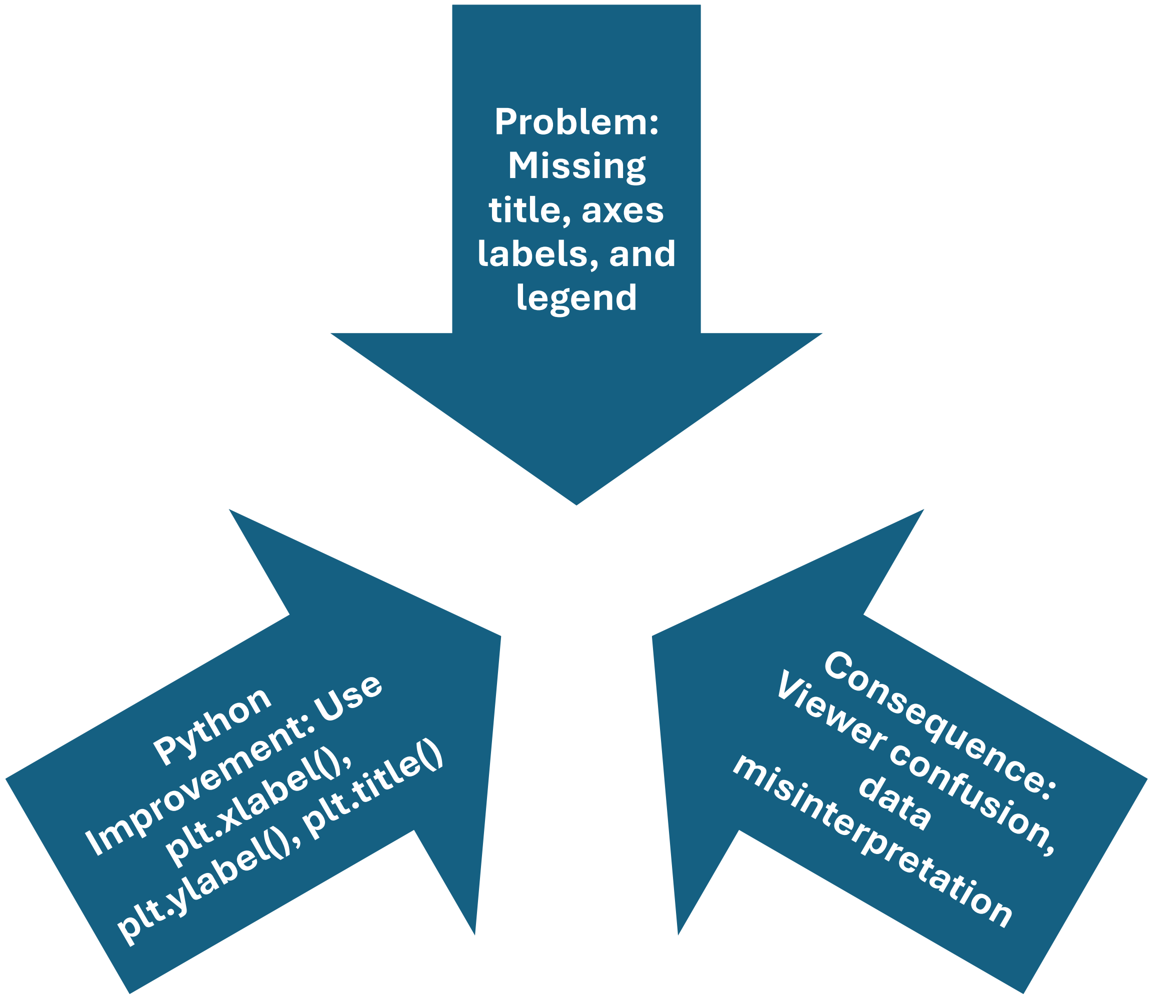


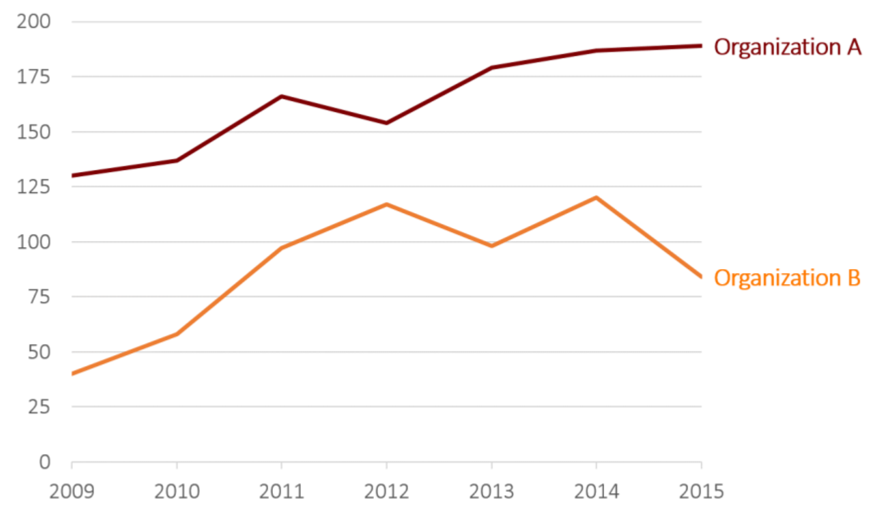
Common Pitfalls in Data Visualisation

- Poor labeling
- Misleading scales
- Cluttered visuals
- Inappropriate color choices
- Unnecessary effects
- Wrong chart type
- Ignoring data order
- Overplotting



Example: Poor Labeling


Problem: Missing title, axes labels, and legend
Python Improvement: Use plt.xlabel(), plt.ylabel(), plt.title()
Consequence: Viewer confusion, data misinterpretation

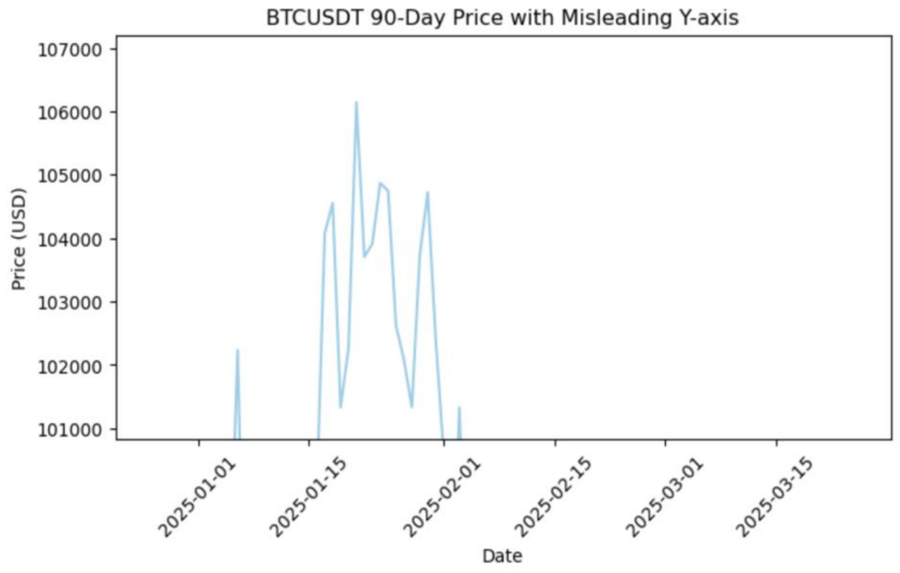


Example: Misleading Scales

Problem: Distorting data perception by inappropriate zoom

Consequence: Exaggerated differences

Python Solution: Proper scaling with plt.xlim(), plt.ylim()

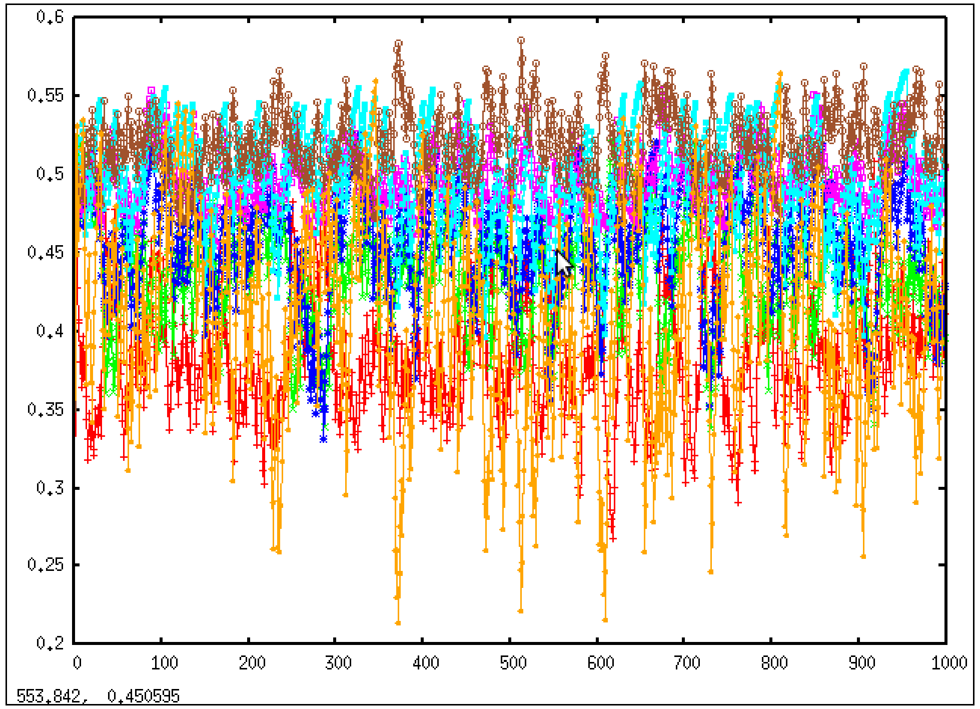


Example: Cluttered Visualisations

Problem: Overcrowded charts

Consequence: Difficulty identifying key data insights

Python Solution: Simplify visual elements, utilize sns.despine() for clarity

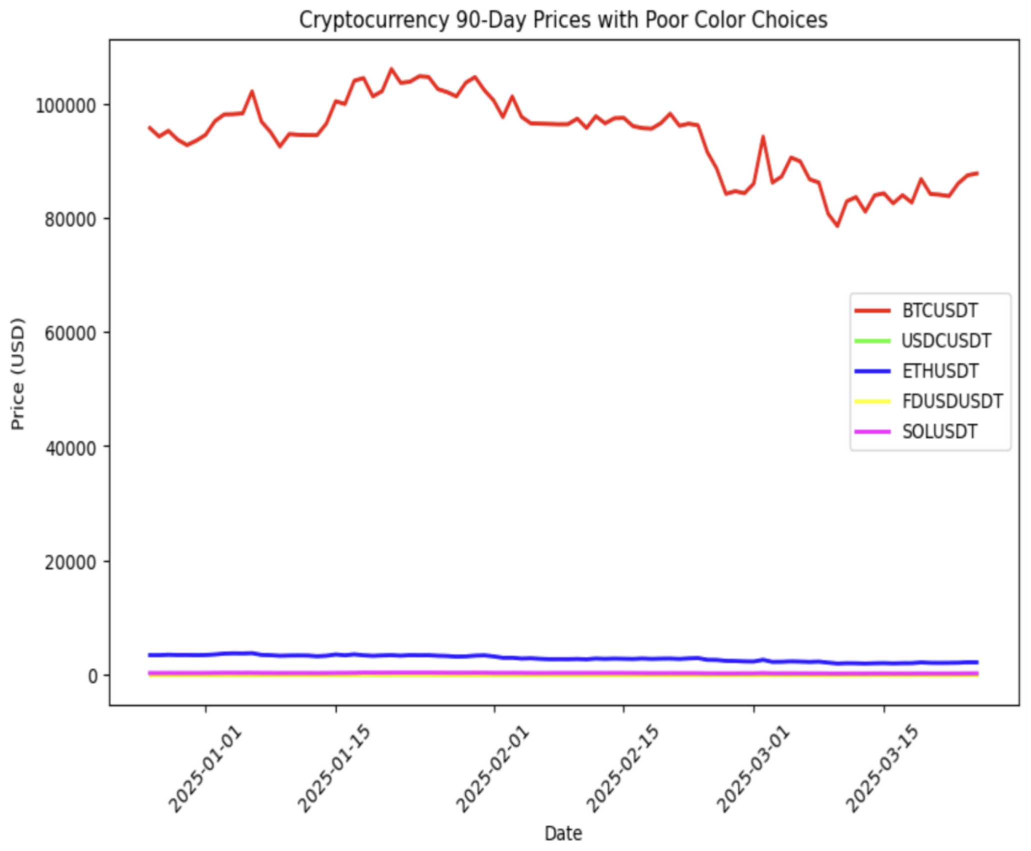


Example: Bad Color Choices

Problem: Poor contrast, inaccessible color choices

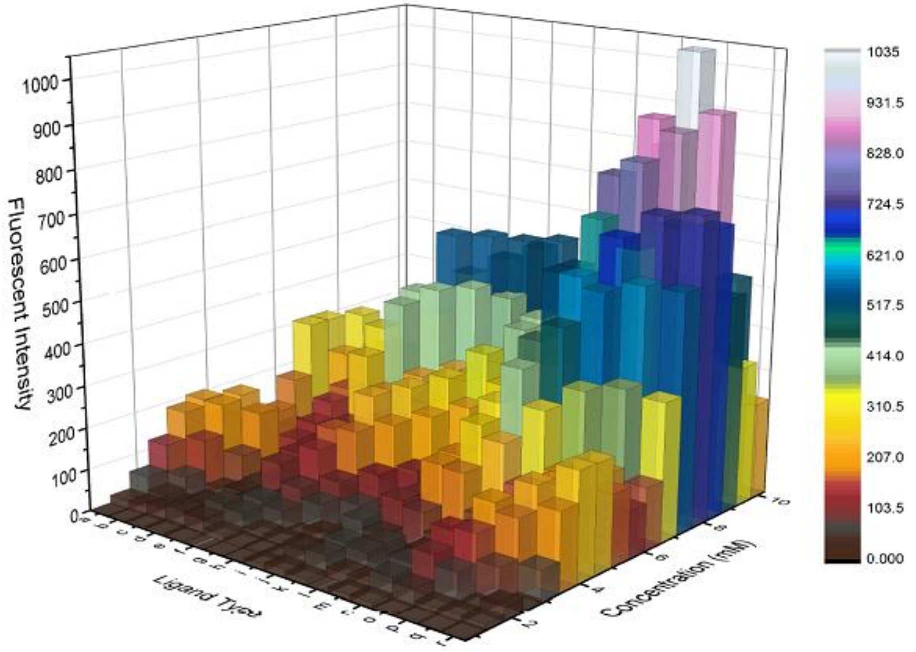
Consequence: Reduced readability, confusion for color-impaired viewers

Python Solution: Accessible palettes, e.g., sns.color_palette("colorblind")



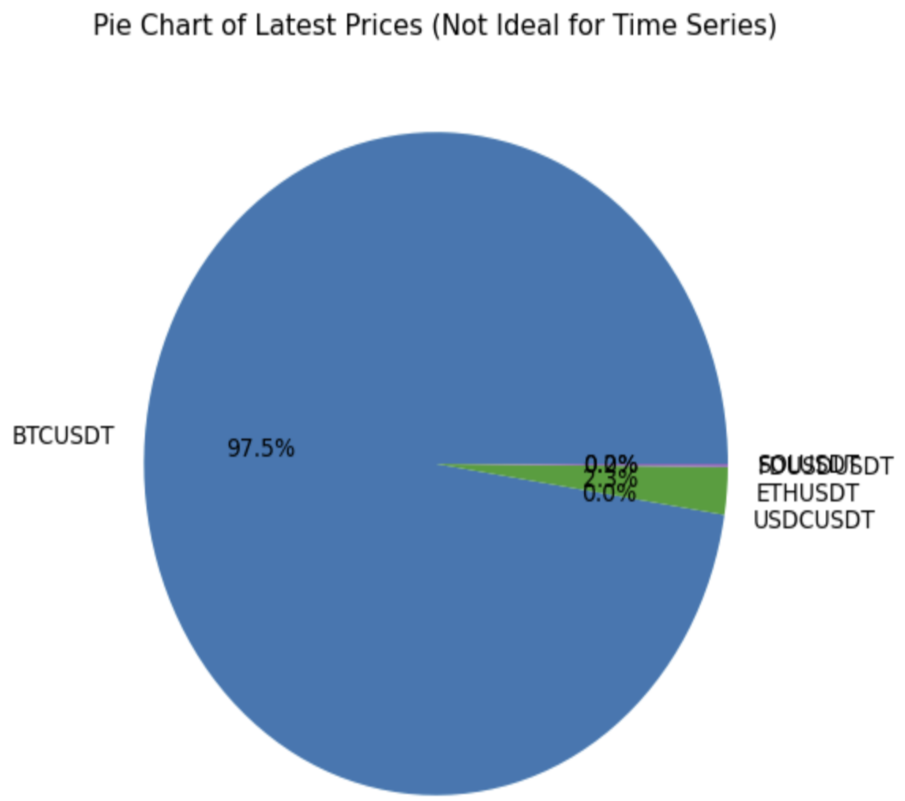
Example: Unnecessary Effects

- Problem: Unneeded 3D visuals
- Consequence: Distraction from core data insights
- Recommendation: Use effects sparingly and purposefully



Example: Wrong Chart Type

- Problem: Misrepresentation (e.g., pie chart for time series)
- Consequence: Misinterpretation of data
- Correct Approach: Select chart types matching data nature (line charts for trends)

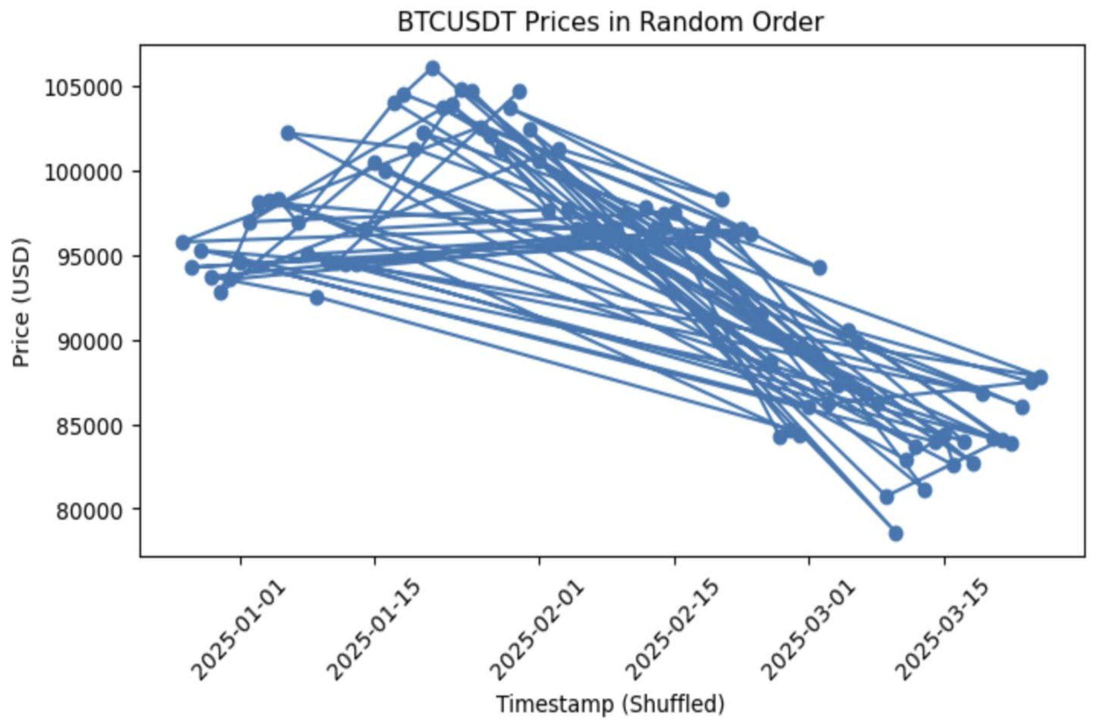


Example: Ignoring Data Order

Problem: Random data order obscuring trends

Consequence: Loss of meaningful patterns

Python Solution: Ensure ordered data presentation using df.sort_values()

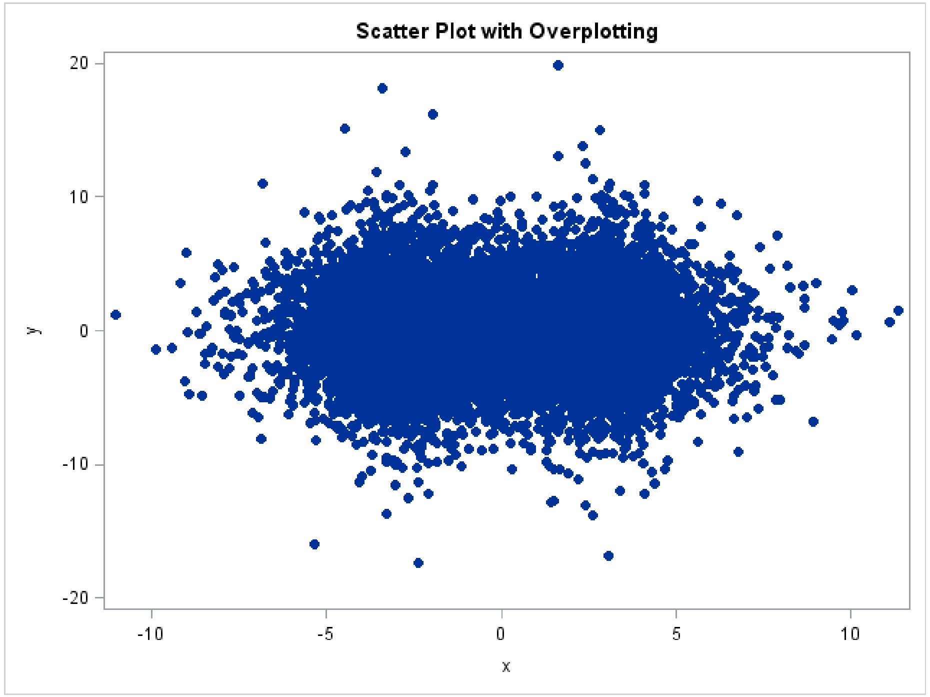


Example: Overplotting Issues

Problem: Data points overlapping excessively

Consequence: Hidden insights, unclear visualization

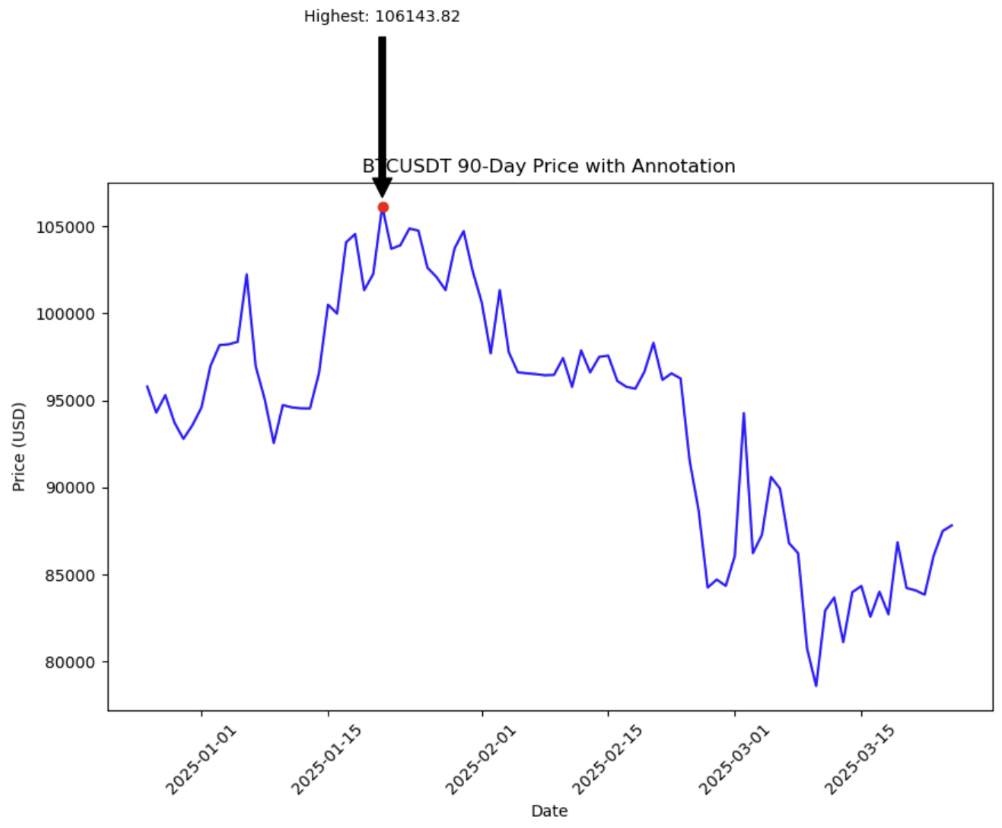
Solutions: Transparency (alpha), jittering, hexbin plots (plt.hexbin()), or density plots (sns.kdeplot())



Enhanced Visualisation with Annotations

- Benefit: Highlighting key insights, improving viewer engagement

- Example: Annotating highest price point clearly using plt.annotate()



Conclusion

- Effective data visualisation aids:
  1. clear communication,
  2. supports informed decision-making,
  3. and ensures professionalism in data analytics.