anscombe_eda

October 8, 2025

Anscombe's Quartet: Exploratory Data Analysis

Rayan Bashir

October 8, 2025

1 Important Links

Github Repository

Interactive Plot

2 Abstract

This project explores Anscombe's Quartet, a set of four datasets that have nearly identical summary statistics but very different distributions when visualized. The goal was to demonstrate the importance of graphical analysis in data science and statistics. Using Python libraries such as Pandas, Seaborn, Plotly, and Altair, each dataset was analyzed to compute statistical measures including mean, variance, correlation, and regression coefficients. Scatter plots and regression lines were created to visualize how the datasets differ despite having similar numerical summaries. The results shows that relying only on statistics can be misleading and highlights the importance of visualization for correctly interpreting data patterns and relationships.

3 Introduction

Anscombe's Quartet was created by statistician Francis Anscombe in 1973 to illustrate the importance of looking at data visually instead of relying only on summary statistics. Each of the four datasets has the same mean, variance, correlation, and regression line, yet they have distinct distributions and patterns when plotted. The purpose of exploratory data analysis (EDA) is to understand the structure and characteristics of data through both statistical measures and visualizations. By examining Anscombe's Quartet, this project demonstrates how visualization helps identify trends, outliers, and relationships that would otherwise be hidden by summary statistics alone.

4 Methods

Calculated mean, variance, and standard deviation for both x and y values of each dataset

Computed covariance and correlation between x and y

Performed linear regression to find slope, intercept, and R^2

Created scatter plots and box plots to visualize relationships

5 Requirements

[1]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from scipy import stats
import pandas as pd
import statsmodels.api as sm
import plotly.express as px
```

```
sns.set(style="white")
```

6 Load data

[2]:
```
colours = ["orange", "blue", "green", "black"]

anscombe = pd.read_csv("anscombe.csv")

anscombe_melted = anscombe.melt(
    id_vars=["x123","x4"],
    value_vars=["y1","y2","y3","y4"],
    var_name="dataset",
    value_name="y"
)
anscombe_melted["dataset"] = anscombe_melted["dataset"].replace({
    "y1": "I",
    "y2": "II",
    "y3": "III",
    "y4": "IV"
})

anscombe_melted["x"] = anscombe_melted.apply(lambda row: row["x123"] if␣
 ↪row["dataset"] in ["I","II","III"] else row["x4"], axis=1)

anscombe_melted = anscombe_melted.drop(columns=["x123","x4"])
print(anscombe_melted)
```

```
   dataset      y     x
0        I   8.04  10.0
1        I   6.95   8.0
2        I   7.58  13.0
3        I   8.81   9.0
4        I   8.33  11.0
5        I   9.96  14.0
6        I   7.24   6.0
7        I   4.26   4.0
8        I  10.84  12.0
9        I   4.82   7.0
10       I   5.68   5.0
11      II   9.14  10.0
12      II   8.14   8.0
13      II   8.74  13.0
14      II   8.77   9.0
15      II   9.26  11.0
16      II   8.10  14.0
17      II   6.13   6.0
18      II   3.10   4.0
```

```
19       II   9.13  12.0
20       II   7.26   7.0
21       II   4.74   5.0
22      III   7.46  10.0
23      III   6.77   8.0
24      III  12.74  13.0
25      III   7.11   9.0
26      III   7.81  11.0
27      III   8.84  14.0
28      III   6.08   6.0
29      III   5.39   4.0
30      III   8.15  12.0
31      III   6.42   7.0
32      III   5.73   5.0
33       IV   6.58   8.0
34       IV   5.76   8.0
35       IV   7.71   8.0
36       IV   8.84   8.0
37       IV   8.47   8.0
38       IV   7.04   8.0
39       IV   5.25   8.0
40       IV  12.50  19.0
41       IV   5.56   8.0
42       IV   7.91   8.0
43       IV   6.89   8.0
```

7 Summary Stats

Formulas

Mean (x): $\bar{y}x = (\Sigma x\ ) / n$ The average of all x-values.

Mean (y): $\bar{y}y = (\Sigma y\ ) / n$ The average of all y-values.

Variance (x): $s\ ^2 = \Sigma(x\ - \bar{y}x)^2 / (n - 1)$ Measures how spread out the x-values are.

Variance (y): $s\ ^2 = \Sigma(y\ - \bar{y}y)^2 / (n - 1)$ Measures how spread out the y-values are.

Standard Deviation (x): $s\ = \sqrt{}s\ ^2$ The typical distance of x-values from their mean.

Standard Deviation (y): $s\ = \sqrt{}s\ ^2$ The typical distance of y-values from their mean.

Covariance: $\mathrm{cov}(x, y) = \Sigma(x\ - \bar{y}x)(y\ - \bar{y}y) / (n - 1)$ Shows how x and y vary together (positive = move together, negative = move oppositely).

Correlation (r): $r = \mathrm{cov}(x, y) / (s\ \cdot s\ )$ Describes the strength and direction of the linear relationship between x and y.

Slope (m): $m = \mathrm{cov}(x, y) / s\ ^2$ Represents how much y changes for each unit increase in x (from linear regression).

Intercept (b): $b = \bar{y}y - m \cdot \bar{y}x$ The predicted value of y when $x = 0$.

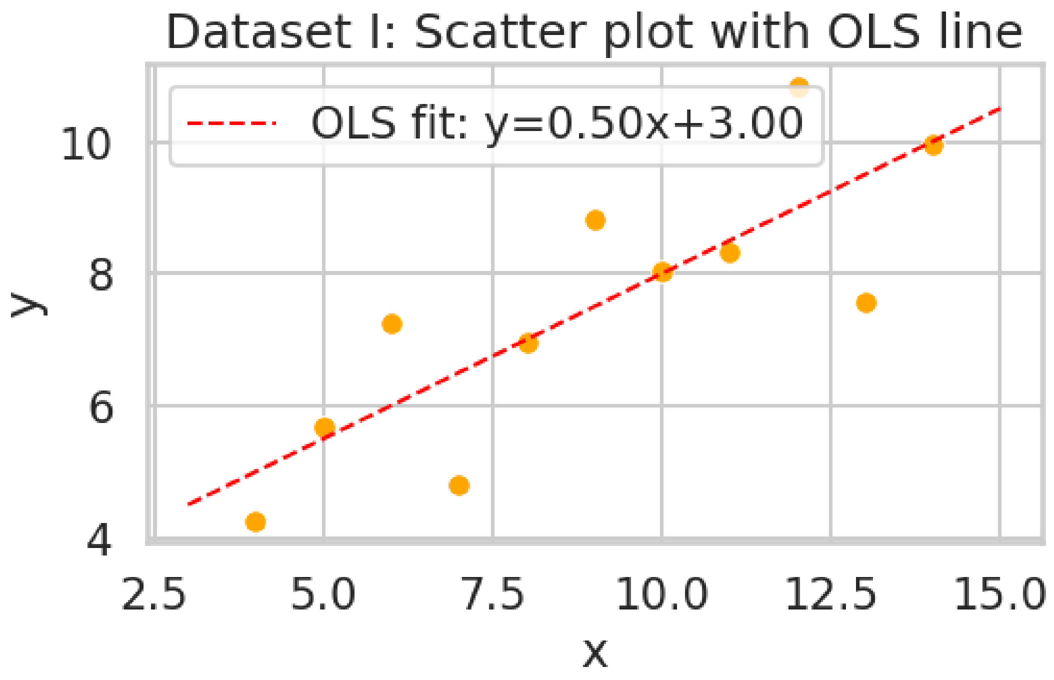
Coefficient of Determination (R^2): $R^2 = r^2$ Represents the proportion of the variance in y explained by x.

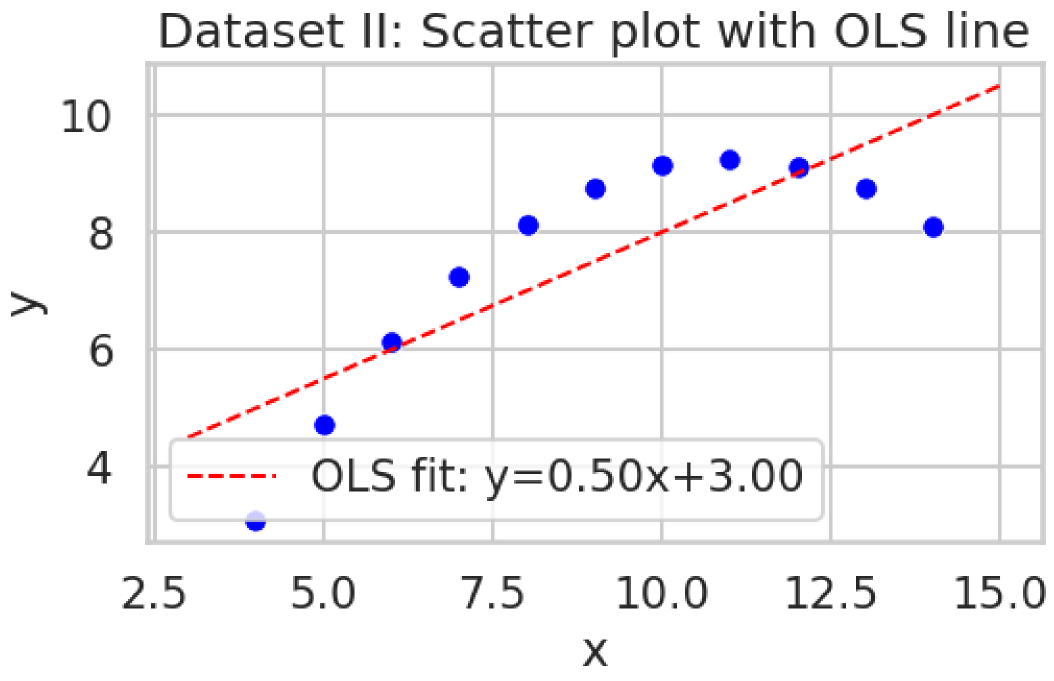
[3]:
```
def dataset_stats(df, name):
    x = df["x"]
    y = df["y"]

    mean_x = np.mean(x)
    mean_y = np.mean(y)
    var_x = np.var(x, ddof=1)
    var_y = np.var(y, ddof=1)
    std_x = np.std(x, ddof=1)
    std_y = np.std(y, ddof=1)
    cov_xy = np.cov(x, y, ddof=1)[0, 1]
    corr = np.corrcoef(x, y)[0, 1]

    slope, intercept, r_value, p_value, std_err = stats.linregress(x, y)
    r_squared = r_value**2

    return {
        "Dataset": name,
        "Mean x": round(mean_x, 2),
        "Mean y": round(mean_y, 2),
        "Var x": round(var_x, 2),
        "Var y": round(var_y, 2),
        "Std x": round(std_x, 2),
        "Std y": round(std_y, 2),
        "Cov xy": round(cov_xy, 2),
        "Correlation": round(corr, 3),
        "Slope": round(slope, 3),
        "Intercept": round(intercept, 3),
        "R²": round(r_squared, 3)
    }

results = []
for dataset_name, group in anscombe_melted.groupby("dataset"):
    results.append(dataset_stats(group, dataset_name))

stats_df = pd.DataFrame(results)

print(stats_df.set_index("Dataset").T)
```

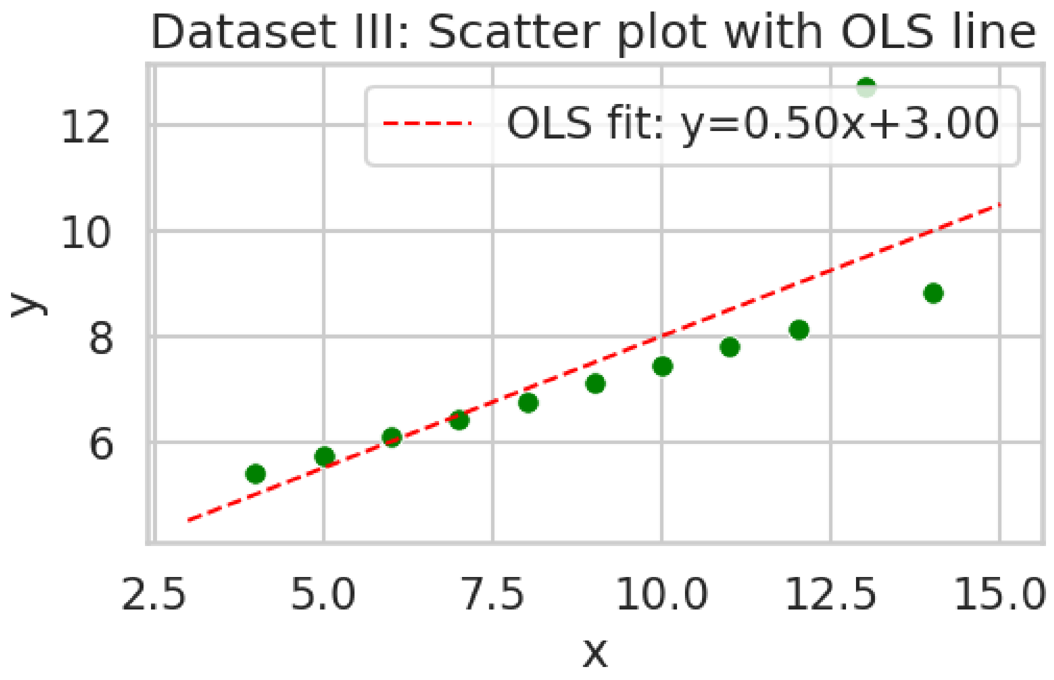
```
Dataset           I       II      III       IV
Mean x        9.000    9.000    9.000    9.000
Mean y        7.500    7.500    7.500    7.500
Var x        11.000   11.000   11.000   11.000
Var y         4.130    4.130    4.120    4.120
Std x         3.320    3.320    3.320    3.320
Std y          2.030    2.030    2.030    2.030
Cov xy         5.500    5.500    5.500    5.500
Correlation    0.816    0.816    0.816    0.817
Slope          0.500    0.500    0.500    0.500
Intercept      3.000    3.001    3.002    3.002
R²             0.667    0.666    0.666    0.667
```

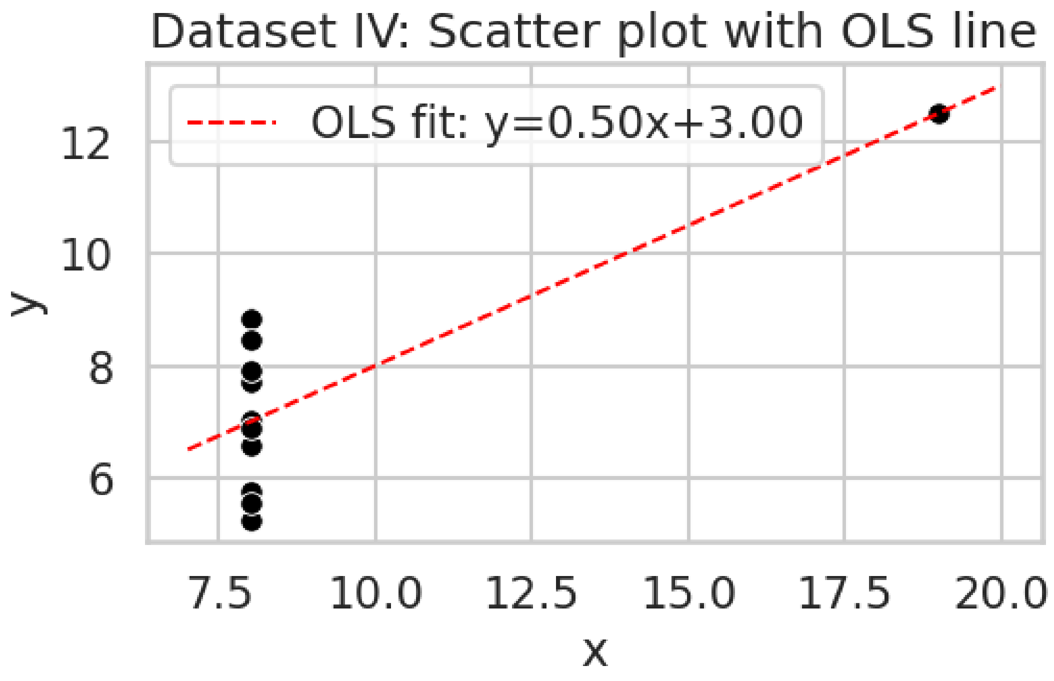
8 Scatter + regression line (seaborn) - one figure per dataset

[4]:
```
sns.set(style="whitegrid", context="talk")
for (name, group), colour in zip(anscombe_melted.groupby('dataset'), colours):
    plt.figure(figsize=(6,4))
    ax = sns.scatterplot(data=group, x='x', y='y', s=70, color=colour)

    X = sm.add_constant(group['x'])
    model = sm.OLS(group['y'], X).fit()
    xs = np.linspace(group['x'].min() - 1, group['x'].max() + 1, 100)
    ys = model.params['const'] + model.params['x'] * xs
    plt.plot(xs, ys, linestyle='--', linewidth=1.5, color="red",
             label=f"OLS fit: y={model.params['x']:.2f}x+{model.params['const']:
 ↪.2f}")
    plt.title(f"Dataset {name}: Scatter plot with OLS line")
    plt.xlabel("x")
    plt.ylabel("y")
    plt.legend()
    plt.tight_layout()
    plt.savefig(f"scatter_reg_{name}.png", dpi=200)
```


Dataset I: Scatter plot with OLS line
OLS fit: y=0.50x+3.00
y
x
10
8
6
4
2.5
5.0
7.5
10.0
12.5
15.0

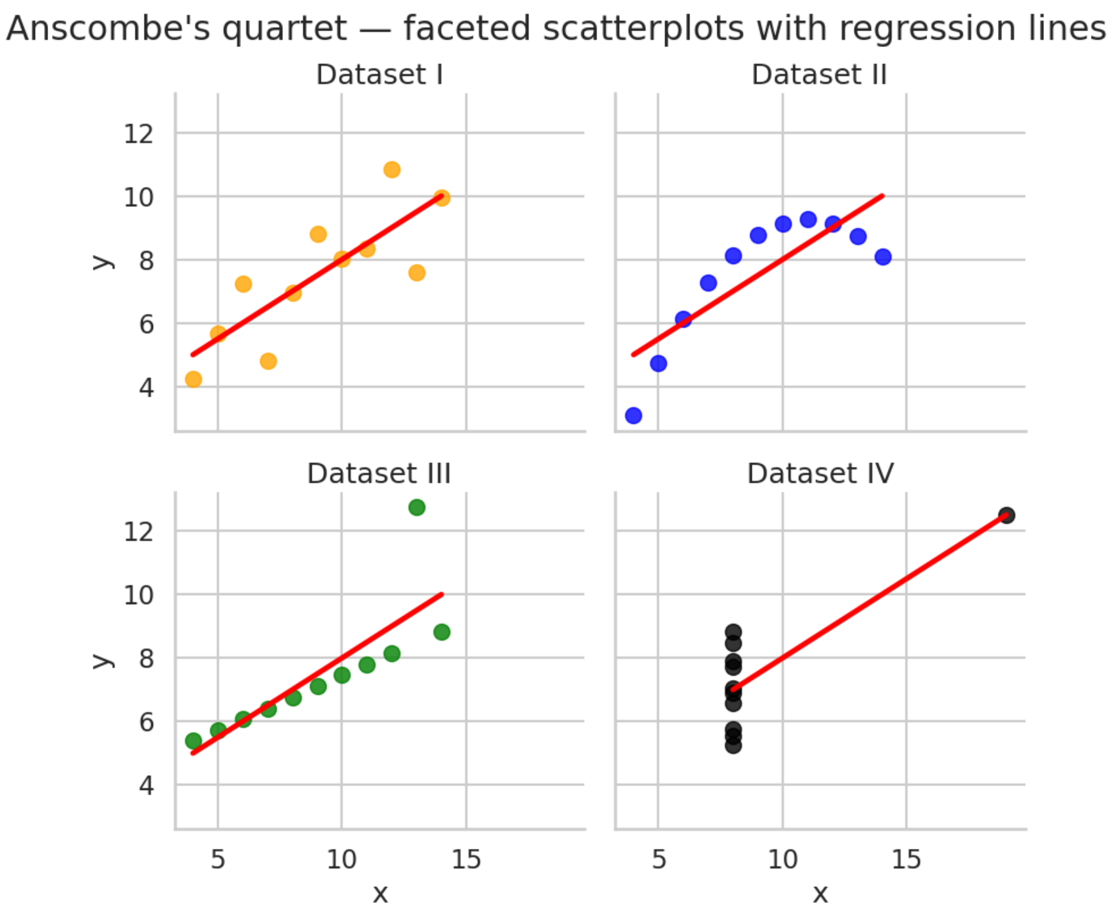

Dataset II: Scatter plot with OLS line
OLS fit: y=0.50x+3.00
y
x
10
8
6
4
2.5
5.0
7.5
10.0
12.5
15.0


Dataset III: Scatter plot with OLS line
OLS fit: y=0.50x+3.00
y
x
2.5
5.0
7.5
10.0
12.5
15.0
6
8
10
12


Dataset IV: Scatter plot with OLS line
OLS fit: y=0.50x+3.00
y
x
7.5
10.0
12.5
15.0
17.5
20.0
6
8
10
12

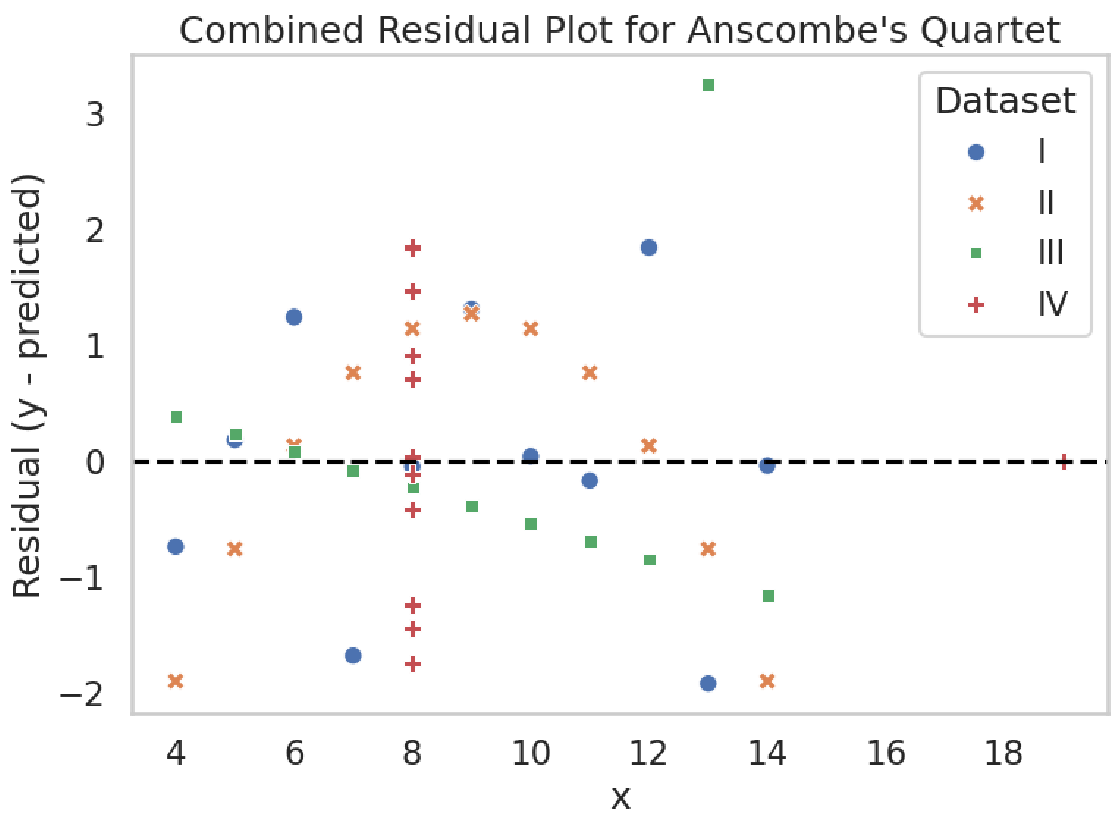
9 Combined visualization (faceted) using seaborn

[5]:
```
g = sns.lmplot(
    data=anscombe_melted,
    x="x",
    y="y",
    col="dataset",
    col_wrap=2,
    ci=None,
    height=4,
    aspect=1.1,
    scatter_kws={"s":100}
)

titles = ["Dataset I", "Dataset II", "Dataset III", "Dataset IV"]

for ax, title, colour in zip(g.axes, titles, colours):
    ax.set_title(title)

    for collection in ax.collections:
        collection.set_facecolor(colour)
        collection.set_edgecolor(colour)

    ax.lines[0].set_color("red")

plt.suptitle("Anscombe's quartet - faceted scatterplots with regression lines",␣
 ↪y=1.02)
g.savefig(f"resid_faceted.png", dpi=200)
```



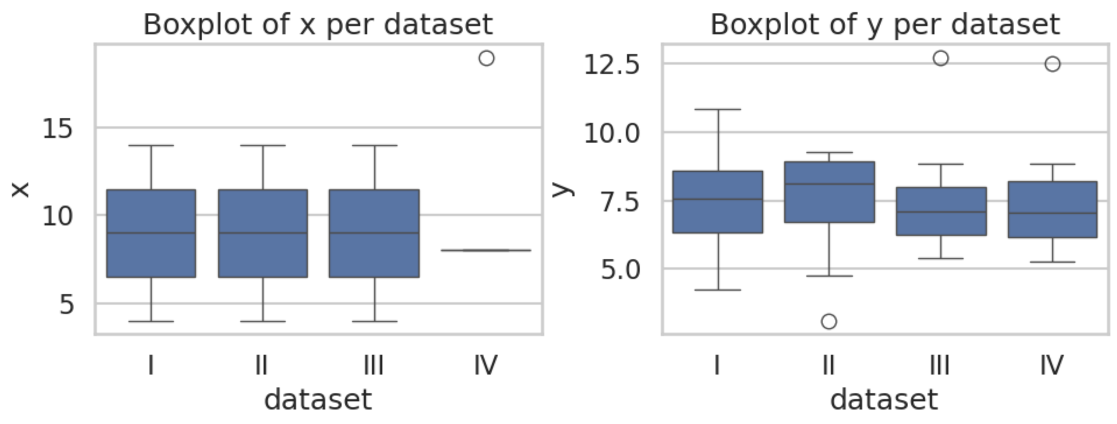
10 Residual plots

[6]:
```python
datasets = ['I', 'II', 'III', 'IV']

residual_data = []

for dataset in datasets:
    subset = anscombe_melted[anscombe_melted["dataset"] == dataset]
    x = subset['x']
    y = subset['y']

    coeffs = np.polyfit(x, y, 1)
    y_pred = np.polyval(coeffs, x)

    residuals = y - y_pred

    residual_data.append(pd.DataFrame({
        'x': x,
        'residual': residuals,
        'dataset': dataset
    }))

residuals_anscombe = pd.concat(residual_data)

plt.figure(figsize=(8, 6))
sns.scatterplot(data=residuals_anscombe, x='x', y='residual', hue='dataset',␣
 ↪style='dataset', s=80)

plt.axhline(0, color='black', linestyle='--')

plt.title("Combined Residual Plot for Anscombe's Quartet")
plt.xlabel("x")
plt.ylabel("Residual (y - predicted)")
plt.legend(title="Dataset")
plt.grid(False)
plt.tight_layout()
plt.show()
```



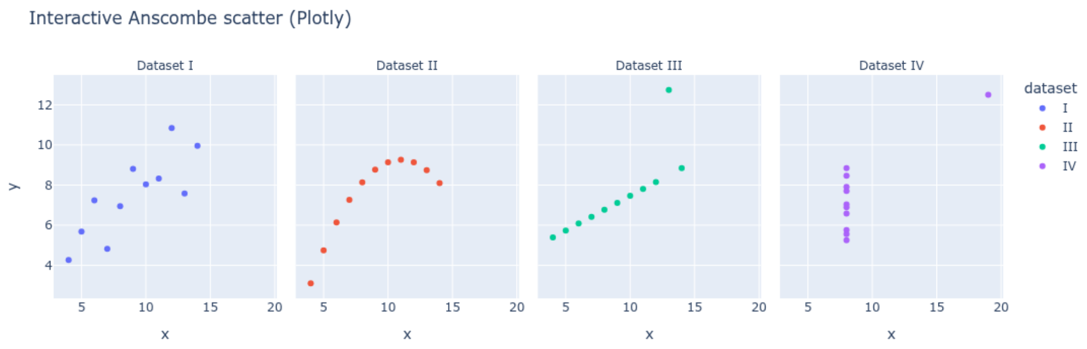
11 Distribution comparisons (box + violin)

[7]:
```
plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
sns.boxplot(data=anscombe_melted, x='dataset', y='x')
plt.title("Boxplot of x per dataset")
plt.subplot(1,2,2)
sns.boxplot(data=anscombe_melted, x='dataset', y='y')
plt.title("Boxplot of y per dataset")
plt.tight_layout()
plt.savefig("boxplots_xy.png", dpi=200)
```



12 Interactive scatter with plotly

[8]:
```
fig = px.scatter(
    anscombe_melted,
    x='x', y='y',
    color='dataset',
    facet_col='dataset',
    title="Interactive Anscombe scatter (Plotly)"
)
fig.update_layout(height=400, width=1000)

new_titles = {"I": "Dataset I", "II": "Dataset II", "III": "Dataset III", "IV":␣
 ↪"Dataset IV"}
fig.for_each_annotation(lambda a: a.update(text=new_titles[a.text.
 ↪split("=")[-1].strip()]))

fig.show()
fig.write_html("anscombe_plotly.html")
```



13 Interpretation

Statistics like mean, variance, and correlation provide a summary of the data but don't capture relationships, trends, or patterns fully. For example, two datasets could have the same mean and variance but very different distributions or correlations. Visualizations help reveal these patterns, making the conclusions more reliable.

14 Collaboration Notes

Kaysan: Residual plot code

15 Conclusion & Next Steps

The analysis highlights trends and relationships in the dataset, but further investigation with additional data or advanced models could provide deeper insights. Next steps include exploring outliers, testing assumptions for regression, and extending the analysis to related datasets.

16 Export as PDF code

```
[11]: !jupyter nbconvert --to pdf anscombe_eda.ipynb --output anscombe_report.pdf
```

```
[NbConvertApp] Converting notebook anscombe_eda.ipynb to pdf
[NbConvertApp] Support files will be in anscombe_report_files/
[NbConvertApp] Making directory ./anscombe_report_files
[NbConvertApp] Writing 53517 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 298123 bytes to anscombe_report.pdf
```

```
[ ]:
```